based interface facilitates navigation. The user can select next and previous buttons, move a slider to jump ahead and behind, and select specific icons. All icons are defined on the process screen, and the user can jump to the process screen multiple ways. The pointer can be positioned on an icon to cause the definition of that icon to appear. There are many ways for the user to jump around as well as to be guided through a process, unlike web interfaces which are random interfaces that are not for designed for a particular process, such as found in the known Radiotherapy Department Toolkit. A selected process of the computer aided planning tool guides the user through steps of that particular process as a part of process.

**[0042]** The process-based interface of the process screen shown in Fig. 2 enables the user to build a process based on a set of menu commands. The process-based interface of the computer aided planning tool enables the user to focus on a selected process to plan and implement a project. In contrast, the known Radiotherapy Department Toolkit lacks the guided, directed, and customizable tour of screens, for example, approximately 45 to 60 screens, of which only a portion, for example, 20 screens, are pertinent to any particular user, that is provided by the computer aided planning tool.

**[0043]** The processes which are listed on the process screen shown in Fig. 2 are user definable so that the processes can be configured on a profession-specific basis, such as processes for planning an entire healthcare facility. Furthermore, a hospital administrator is likely to utilize different processes that an architect or a physicist. Consequently, the process-based interface can be redesigned. The resulting process screen is also a file which can be rendered unmodifiable, display-only information. This provides a flexible process-based platform for the computer aided planning tool.

**[0044]** The process screen shown in Fig. 2 enables a user to position the mouse pointer to highlight one of the identified processes under "Select a process..." on the right hand side of the screen and view a description of that process below. Moreover, the features of the highlighted process appear on the left hand side of the screen. In the exemplary implementation of the computer aided planning tool for planning a healthcare facility, the process screen can include the processes "Edit a department or room template," Explore Neoforma Healthcare," "Linac Shielding - New Construction," "Linac Shielding - Retrofit," "Plan a new room or department," and "Select a product," for example. The user can define other processes or delete one or more of the process selections listed on the process screen shown in Fig. 2 to provide a process-based interface that can be customized at the home page level.

**[0045]** The processes can vary in complexity. In this regard, "Plan a new room or department" is a comprehensive process tool for planning and implementing departments and individual rooms of a healthcare facility and "Linac Shielding - New Construction" is a specialized process tool for designing and implementing shielding for a radiotherapy room at a healthcare facility. On the other hand, "Select a product" is a streamlined process for identifying products for implementation of a component of a project.

**[0046]** Clicking mouse 18 on the highlighted process on which the pointer is positioned selects the identified process. As shown in Fig. 2, the highlighted process is "Explore Neoforma Healthcare." This can be designated as the default process and provides access to various screen galleries for the user to browse. Clicking mouse 18 causes "Explore Neoforma Healthcare" to be selected. This process enables the user to simply access and view screens contained in various screen galleries associated with the features on the left hand side of the process screen shown in Fig. 2. This process allows the user to navigate through screen galleries, rather than the screen galleries being tied to one of the other processes listed in the process screen and thus browse the program in overview fashion. The screen galleries comprise the following galleries.

**[0047]** A department galley enables a user to view the overall scope of one component of a planning project. The screens contained in the department gallery provide the user a bird's-eye view of an entire department, such as a radiotherapy department. For example, Fig. 3 illustrates such a department gallery screen accessed through the "Explore Neoforma Healthcare" process and displayed on monitor 14. The department gallery screen shown in Fig. 3 lists available room choices in the lower right portion of the screen which are linked to detailed descriptions and requirements of each room. The process from which the screen is entered, that is, "Explore Neoforma Healthcare," appears at the upper left hand corner of the screen shown in Fig. 3, while the screen being viewed by the user is identified in the lower right hand corner ("Department Gallery"). The department gallery enables easy, mouse-driven navigation from room to room within the department. The department gallery provides three-dimensional department visualization as shown in Fig. 3, as well as specific information regarding the location and relationship of rooms for design optimization and implementation.

**[0048]** As shown in the upper left portion of Fig. 3, the screen was submitted by a product vendor. The information contained in the screen is therefore identifiable with a specific source and is unmodifiable, display-only information.

**[0049]** A rooms gallery is enabled by the user selecting any room from the department gallery. The rooms gallery enables the user to access information stored in the relational database that provides an in-depth description of the equipment recommended for that room. The screens contained in the rooms gallery displays only product information relevant to the selected room, so that the product selection and evaluation process is focused. The screens contained in the rooms gal-



IPR2023-00330 Page 00383

lery also present product categories sorted in logical, functional, and room-determined groupings.

**[0050]** A products gallery is enabled when the user selects any product category either by clicking the mouse on the three-dimensional representation of the category or by choosing from the categories listing. The screens contained in the products gallery display listings of vendors and their products. The products gallery provides detailed descriptive information submitted by the vendor of the product for each of the listed products. The screens contained in the products gallery also enable the user to review product highlights and choose vendors for further evaluation. The products gallery provides all of the relevant information needed for full evaluation of a product family or specific product for implementation of a component of the project being planned and potential purchase by the user. The screens contained in the products gallery provide detailed descriptions, features, benefits, and specifications relating to the products. The screens contained in the products gallery also preferably provide precise product drawings, photos, and/or graphics so that the user can gain an accurate understanding of the product. The products gallery also preferably directs the user to the product vendor and provides contact information for the vendor by the user clicking mouse 18 on the vendor logo or name.

**[0051]** A company information gallery enables the vendor to promote its products to the user. The company information gallery contains screens that include corporate profile or background information, awarded certifications, and listings of the international markets in which the vendor is active. The company information gallery links to the products gallery screens for all products that the vendor has listed in the product catalog stored in the database. The screens contained in the company information gallery provide information regarding how the user can directly contact the vendor.

**[0052]** Unlike other processes listed on the process screen, the process invoked by selection of "Explore Neoforma Healthcare" is a substantially unguided process. This process also enables the user to browse tools available in connection with other listed processes, as well.

**[0053]** A streamlined process selectable from the process screen shown in Fig. 2 is "Select a product." The user first positions the pointer on "Select a product" to highlight that process, as shown in Fig. 4, and clicks mouse 18 to select that process. Selection of the product selection process causes the computer aided planning tool to access the company information gallery and display a screen, such as the product vendor screen shown in Fig. 5, which lists various products available from that vendor in the lower left portion of the screen. The user next performs the task of selecting a product by pointing to a desired product and clicking mouse 18. The computer aided planning tool user then accesses the product catalog in the database and accesses the product gallery linked to the screen in the company information gallery shown in Fig. 5. This causes a screen in the product gallery, such as the screen shown in Fig. 6, to be displayed from which the user can complete the task of implementing a component of the project, for example, designation of the treatment couch shown in Fig. 6 for a radiotherapy department of a healthcare facility.

**[0054]** "Linac Shielding - Retrofit" and "Linac Shielding - New Construction" are examples of more complicated processes selectable from the process screen shown in Fig. 2. The user first positions the pointer on "Linac Shielding - New Construction" to highlight that process, as shown in Fig. 7, and clicks mouse 18 to select that process. Alternatively, the user can position the pointer on "Linac Shielding - Retrofit" to highlight that process, as shown in Fig. 7, and clicks mouse 18 to select that process. There are slight differences between these two processes, but a majority of the screens utilized by these two processes are substantially similar. That is, the processes are both linked to shielding screens.

**[0055]** For example, selection of the linac shielding process for new construction causes the computer aided planning tool to access product information stored in the database for all products in the healthcare industry related to a linear accelerator room, bound intricately into the process of designing a linear accelerator room, and to jump immediately into a specialized tool for designing shielding in part based on access to applicable regulatory information contained in the library information stored in the database. The user can set up project information for the shielding project.

**[0056]** Additionally, the tool for designing shielding is linked to library information useful in the context of a shielding project, for example, information contained in the library stored on the database relating to similar projects implemented by others. That is, the user can access the room gallery and browse linear accelerator room information to obtain information within the process of designing shielding for a new construction project. This enables the user to explore a community-based and edited library of information, while utilizing the specialized shielding tool.

**[0057]** Library information stored in the database can also be accessed for used in calculations performed by the specialized shielding tool comprising the linac shielding process for new construction, such as regulations issued by governmental agencies or other organizations which have oversight responsibility for a project that involves shielding. For example, the screen shown in Fig. 9 lists common barriers found in a radiation therapy room, such as walls and doors, as well as links to regulations relating to dose limits that apply to the design of barriers. As shown in the upper left hand portion of the screen shown in Fig. 9, regulations promulgated by Taiwan can be linked to the barrier design for shielding in the "Linac Shielding - New Construction" process.



IPR2023-00330 Page 00384

**[0058]** The screen shown in Fig. 9 accessed through the linac shielding process for new construction is linked to regulatory information contained in the library information stored in the database. As described above, the particular screen shown in Fig. 9 is linked to regulations promulgated by Taiwan. This evidences the flexibility of the computer aided planning tool to plan projects, such as radiation therapy rooms, worldwide. The regulations that apply to a specific project being planned by the user can also be selected by the user. The user can compare one project design to another as a function of the applicable regulations in one part of world to another. For example, the applicable regulations can be regulations promulgated by the United States of America, instead of Taiwan, as shown in the radiation safety screen shown in Fig. 10. The screen shown in Fig. 10 displays the regulatory information with respect to radiation safety requirements in the United States contained in the library information stored in the database. The information contained in the library also contains information submitted by the community of entities which is supplied with the computer aided planning tool with respect to previous shielding designs, such as barrier walls, that the user can view for suggestions on how to create a wall for the project being planned.

**[0059]** As in the case of product information stored in the database, library information can also be unmodifiable, display-only information or alternatively can be customizable information which can be imported by the user utilizing a file manager program to be edited for use by the user. In this regard, a new file can be created through a files menu. The creator of the file can then determine whether the file is unmodifiable, display-only information or customizable information. In order to designate the file that is created as unmodifiable, display-only information, the entity that creates the file includes the log-in name and unique password assigned to that entity with the file. For example, the source of the regulatory information displayed in the screen shown in Fig. 10 is identified with the source of that information and is therefore unmodifiable, display-only information. By default, if no such log-in name or password is included, the information is customizable and can be imported by a user and modified for use. If the library information is designated as unmodifiable, display-only information, only the entity that created the file can access that file and update the information in the file. Therefore, the maintenance responsibility for information in the library that is designated unmodifiable, display-only information resides with the entity that created the file.

**[0060]** The screen for the specialized shielding tool which comprises the linac shielding process for new construction is shown in Fig. 11. The shielding tool can be utilized by the user to optimize the design of barriers to obtain cost savings through linkage of parameters of various materials that can be utilized for a shielding design project to the shielding tool for calculating material thicknesses for barriers, such as walls. As shown in Fig. 11, the shielding tool has not been enabled by the user to calculate shielding requirements, since the default is the "Calculation Off" button at the top of the table. The screen also illustrates that no specific vendor material has been selected for inclusion in the design for the shielding barrier being designed by the user in connection with the project being planned.

**[0061]** The links to materials from vendors, for example, specific components of material, such as modular walls, or shielding designs utilizing materials provided by various vendors in a design created by the user, are linked to the shielding tool for implementing a project. In this regard, the shielding tool is linked to a gallery of materials information stored in the database and linked to the shielding tool. The materials, such as lead and concrete blocks, can be generic and vendor-specific materials. The materials information includes parameters of various materials, which are used by the shielding tool to calculate required thicknesses of materials utilized in the design of shielding for a radiation therapy room. The screen shown in Fig. 12 indicates selection of a vendor-specific concrete block material by positioning the pointer on the top "Concrete (normal)" line in the table shown in Fig. 11 and scrolling utilizing the up/down arrows to a vendor-specific concrete block material from the product catalog stored in the database and clicking mouse 18 on that material. The selection is linked to the "Linac Wall Materials" gallery in the gallery of materials information. The user can access the vendor-specific information for the wall material by clicking on the selected material in the table shown in Fig. 12, which causes material information to be displayed. The information displayed in response to clicking on the material shown in the table in Fig. 12 is displayed in the screen shown in Fig. 13.

**[0062]** The shielding tool automatically imports the parameters of the material selected by the user for calculating the shielding thickness requirements for the selected materials. The calculations are initiated by the user selecting the "Auto" check box on the left hand side of the table shown in Fig. 12 and also selecting the "Calculation On" button above the center of the table. In the example calculation illustrated by the screen shown in Fig. 12, the thickness of lead is set by the user, and the shielding tool calculates the thickness of the selected concrete block material that is needed to satisfy the given dosage regulations. The shielding tool utilizes conventional shielding computations to determine thicknesses for materials and calculates thicknesses using the parameters of the selected material, as indicated by the partial product identification and density parameter which also appear in the screen shown in Fig. 13.

**[0063]** The user can also access specific product information that is used in radiation therapy rooms. The information accessible to the user includes information from the materials gallery. For example, the screen shown in Fig. 13 is linked to the shielding tool shown in Figs. 11 and 12. If the user had simply browsed the



IPR2023-00330 Page 00385

materials gallery for a material for designing a barrier wall and selected the vendor-specific wall material displayed in the screen shown in Fig. 13, the user can import the material parameters contained in the file for the material through the file manager by invoking "Insert Wall" which appears at the left central portion of the screen shown in Fig. 12 and thus select the vendor-specific wall material for calculating the thickness of that material for the shielding project being planned.

**[0064]** Also, as indicated above, the product gallery for radiation therapy rooms is accessible to the user through the linac shielding process for new construction. For example, vendor-specific equipment from the product gallery can be accessed through the "Linac Shielding - New Construction" process, as shown in Fig. 14. The screen shown in Fig. 14 displays product information with respect to vendor-specific equipment utilized in a radiation therapy room at a healthcare facility. Consequently, there is also a link to the company information gallery, as well, through the linac shielding process for new construction.

**[0065]** In summary, the linac shielding process for new construction enables the user to plan a design based on community provided information, regulations, generic and vendor-specific materials that can be utilized, and vendor-specific products. Then, the shielding tool calculates material thicknesses for barriers when "Auto" and "Calculation On" which appear in the screen shown in Fig. 12 are selected by the user using the parameters of the materials selected by the user. The user can then access the company information gallery for more information if the shielding design for the project being planned is satisfactory.

**[0066]** Another process selectable from the process screen shown in Fig. 2 is "Edit a department or room template." The user first positions the pointer on "Edit a department or room template" to highlight that process, as shown in Fig. 15 and then clicks mouse 18 to select that process. Selection of the department or room template editing process causes the computer aided planning tool to access the screens linked to department templates. For example, the department template screens include templates for different departments, such as different templates for a radiation therapy department at a healthcare facility. A screen which provides one exemplary department template for a radiation therapy department is shown in Fig. 16. Since this screen identifies the source of the displayed information in the upper right portion of the screen, the department template shown in Fig. 16 comprises unmodifiable, display-only information. In contrast, any department template which is modifiable, such as a department template in a file created by the user or customizable because the source of the information is not identified, can be imported by the user and modified to design a department relating to the project.

**[0067]** As shown in Fig. 16, the screen lists various rooms included in the radiation therapy department included in the particular department template that appears in the screen. The user can position the pointer on an individual room identified in the list of rooms and click mouse 18 to access the screen that corresponds to the selected room. Therefore, the user can navigate among rooms of the department shown in the department level screen shown in Fig. 16.

**[0068]** Finally, another process selectable from the process screen shown in Fig. 2 is "Plan a new room or department." The user first positions the pointer on "Plan a new room or department" to highlight that process, as shown in Fig. 17, and clicks mouse 18 to select that process. Selection of the new room or department planning process causes the computer aided planning tool to access the department planning template screens, such as the department planning template screen for a radiation therapy department shown in Fig. 18. The templates enable the user to view the designs and implementations of similar projects and learn from the prior experience of others who have planned those projects. These templates enable the user to design a department based on gleaning an understanding of information displayed in screens which consist of unmodifiable, display-only information indicated when the particular template is identified with the source that submitted the template or by selecting a screen that displays customizable information and importing that template so that the design embodied in the department template can be edited as needed to create a design for the department being planned. Therefore, the user can create a design or import a template if not designated unmodifiable, display-only information.

**[0069]** The rooms which comprise the department, such as the radiation therapy department profiled in the screen shown in Fig. 18, are linked to the department planning template. The rooms templates linked to the department planning template can be selected by the user locating the pointer to one or more rooms listed in the department planning template and clicking mouse 18 on the individual rooms. One of the room planning templates accessible upon selection of the "Administration Office" from the department planning template shown in Fig. 18 is illustrated by the room planning template for the "Administration Office" shown in Fig. 19.

**[0070]** The department and room planning templates which are accessed by the "Plan a new room or department process" can be created by any entity which is supplied with the computer aided planning tool. The entity can create a file that becomes a department and/or room template and designate those/that template unmodifiable, display-only information by including the log-in name and unique password of that entity or customizable information by not including the entity log-in name and unique password. If the source of the template is identified, the responsibility for maintaining the template resides with the identified source, which is the only entity that has access to the information through log-in name and unique password to update or other-



IPR2023-00330 Page 00386

wise modify the information. For example, the department planning and room planning templates shown in Figs. 18 and 19 are customizable department and room information which can be imported by the user and edited to create a design for a project being planned by the user. Templates can be individually or collectively incorporated into the project being planned by the user.

**[0071]** As illustrated by the screen shown in Fig. 19, the room planning template can be further linked to product information in the products gallery. The user can access the products gallery to complete the process of planning a project by performing the task of implementing the components of the project, such as selecting the equipment, furniture, and accessories, such as interior decorating appointments, based on positioning the pointer on the products listed in the room planning template listed in the screen shown in Fig. 19 and double clicking on the products.

**[0072]** If department planning and/or room planning templates are unmodifiable, display-only information identified with the entity which is the source of that information or vendor-specific products are linked to the products listed in the room planning templates for implementing components of the room, a link to the company information gallery exists. Therefore, the user can access information for contacting the identified entity or vendor, for example, the web and e-mail addresses, the telephone and facsimile transmission numbers, and street addresses, as well as the company profile, of the identified entity or vendor.

**[0073]** The user can use an accessed web site address of a vendor, for example, to contact that vendor. The computer aided planning tool enables the user to connect to the web site of a vendor by positioning the pointer on the name or logo of the vendor in the screen which profiles that company in the company information gallery. For example, the vendor identified in the screen shown in Fig. 5 can be accessed through any product screen or unmodifiable, display-only information sourced by that vendor, and the user can position the pointer on the company name or logo and connect to the home page at the web site of that particular company by clicking mouse 18 from that screen. On the other hand, if the user positions the pointer on the company name or logo in a screen in the products gallery which includes a specific product reference, for example, on the name of the company identified in Fig. 13, and clicks mouse 18, the user is connected to the relevant page at the vendor web site that contains information pertinent to the specific product. Therefore, the applicable web site page is directly accessible from the screen in the products gallery relating to that product, which obviates the need for the user to browse the web site of the vendor for specific product information.

**[0074]** Once the user has planned the design for a project, including a department and room design, the user can utilize the computer aided planning tool to perform the task of actually planning the implementation of the various components of the project being planned, for example, the selection of products for implementation of the components incorporated into the design for the project. For example, the user can select a customizable template for a department for a healthcare facility and select that template as the department planning template for the project being planned, as illustrated in the screen shown in Fig. 20. The user can then access the room planning template linked to that department planning template, as shown in Fig. 21. The user can then select products to implement the components of the room. In this regard, the user can browse the products gallery, and browsed product information can be actively added to the project being planned, as indicated by the selection of a vendor-specific product in the first line of the table which appears in the screen shown in Fig. 21. In order to complete the task of implementing the components of a project being planned, as well as determine the cost of the project, the user may need additional information from various vendors of products. This task can be facilitated by two-way connectivity to the various vendors by user broadcasted e-mail.

**[0075]** After the user has completed the task of assembling a table of vendor-specific products for implementing the components of a project being planned, the user decides what, if any, additional product information is needed. The user then generates pre-formatted e-mail messages, for example, "I am [the user profiled under the user options]. I am requesting [check box(es)] about the [product from the products listed in the table shown in Fig. 21]." The check boxes appear in the lower left hand corner of the screen shown in Fig. 21. These check boxes include "Information," "Quote," and "References." The user can also elect to add specific comments or requests which are appended to the pre-formatted e-mail message, for example, "Please quote in Yen." The additional information sought by the user positioning the pointer to one or more check boxes and clicking mouse 18 is then assembled into a pre-formatted e-mail message to the vendor of each product which appears in the table in the screen shown in Fig. 21.

**[0076]** The table which appears in the screen shown in Fig. 21 is linked to the company information gallery. Therefore, the e-mail addresses of the vendors which have e-mail addresses are accessed and assembled into the respective pre-formatted e-mail messages. If the vendor does not have an e-mail address, the request is e-mailed to the supplier of the computer aided planning tool, which in turn generates a facsimile transmission incorporating the request to the vendor.

**[0077]** All of the e-mail requests that are to be broadcast are routed to an e-mail queue, which is accessible from the project being planned and mailed as part of the process. The e-mail requests are sent through a standard e-mail connection over the Internet.

**[0078]** Each pre-formatted e-mail message also con-



IPR2023-00330 Page 00387

tains "This is from [a user]. Please reply by filling in between the brackets with the requested information." The vendor then replies by entering the requested information between the brackets and responding to the user by way of an e-mail reply over the Internet.

**[0079]** When e-mail is checked by the user, the user identifies pertinent e-mail which is generically titled "Response to Healthcare Inquiry," and the user saves this e-mail to the e-mail queue. Every time that the computer aided planning tool is opened, the e-mail queue is checked. Replies from vendors that have been transferred to the e-mail queue are parsed, and the additional information that was requested by the user is converted to vendor responses. The information is also entered into the line of the table which appears in the screen shown in Fig. 21 for the particular product to which the additional information relates. As shown in the screen which appears in Fig. 21, an indication is displayed whether or not the vendor has responded. Therefore, status reports can be generated by the user. The information listed in the table which appears in the screen shown in Fig. 21 is also preferable directly linked to a spreadsheet application program for calculating the cost of the project being planned. Once the table which appears in the screen shown in Fig. 21 is complete, an equipment list can be generated for implementing the components of the project being planned. In another embodiment, compatibility templates can be utilized to determine which products are compatible with other specific products based on the information linked to the table which appears in the screen shown in Fig. 21.

**[0080]** The computer aided planning tool in accordance with the invention provides a platform having a process-based structure unlike known CAD productivity tools or the Radiotherapy Department Toolkit. The process-based structure comprises various planning templates linked to information stored in a relational database related to the task performed utilizing that particular template. The computer aided planning tool enables users to navigate smoothly back and forth from information available in various information galleries to process tools utilized to design the components of the project. The database comprises a product catalog backbone and a library of additional information for targeted access. Unlike known productivity tools, the information is unmodifiable or customizable by designation of the entity which submits that information. For example, product vendors can designate their information unmodifiable and distribute their own files of information to users directly. Database information can be supplemented by enabling users to connect to web sites over the Internet, and unlike the known prior art, a particular page at a web site may be accessed based on information selected during utilization of a particular planning template, rather than requiring users to browse the web site for the relevant page. Finally, automated two-way e-mail over the Internet is provided to facilitate the acquisition of supplemental information which is then accessible by users and can be automatically imported for use in the planning templates, including use in applications programs, such as spreadsheets for calculating costs for the project.

**[0081]** It will be understood and appreciated that the embodiments of the present invention described above are susceptible to various modifications, changes, and adaptations. For example, the computer aided planning tool in accordance with the invention can also be provided for planning projects in the hospitality (e.g., hotel, restaurant, entertainment), education, retail, and service industry (hair salons, dentists, etc.). All is intended to be comprehended within the meaning and range of equivalents of the appended claims.

## Claims

1. A computer-implemented tool for planning a project comprising:

   means for providing at least one planning process tool for planning a given type of project comprising a plurality of planning templates, each planning template for planning a component of the given type of project;
   means connected to the means for providing the planning process tool for providing a database of stored information related to the given type of project;
   means connected to the means for providing the planning process tool for displaying the planning templates for enabling selection of a planning template;
   means for selecting the planning template;
   means responsive to selection of the planning template for displaying at least one planning task for implementing the component of the given type of project;
   means for selecting the planning task;
   means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected planning task, whereby the planning template is linked to information stored in the database that is related to planning the component of the given type of project;
   means connected to the planning template for selecting access to information stored in the database for implementing the component: and
   means responsive to information accessed from the database for displaying the accessed information.

2. A computer-implemented planning tool comprising:

   means incorporated into a computer and executed by the computer to provide a knowledge-based tool for performing at least one planning



IPR2023-00330 Page 00388

process for planning a given type of project, the tool comprising at least one planning template for planning a component of the given type of project;

means connected to the computer for providing a database of stored information related to the given type of project, the database comprising a library of published information, the published information comprising unmodifiable, display-only information and customizable information, the unmodifiable, display-only information being associated with an identifiable source of the published information and the customizable information being associated with an anonymous information source;

means connected to the computer for displaying the planning template for enabling selection of the planning template;

means for selecting the planning template;

means responsive to selection of the planning template for displaying at least one planning task for implementing the component of the given type of project;

means connected to the database for enabling access to the library in the database based on the selected planning task, whereby the planning template is linked to the library stored in the database related to planning the component of the given type of project;

means connected to the planning template for selecting access to the library stored in the database for providing information from the library stored in the database relating to the component; and

means responsive to information from the library accessed from the database for displaying the accessed information.

**3.** A computer-implemented planning tool comprising:

means incorporated into a computer and executed by the computer to provide a knowledge-based tool for performing at least one planning process for planning a given type of project, the tool comprising at least one planning template for planning a component of the given type of project;

means connected to the means for providing the planning process tool for providing a database of stored information related to the given type of project;

means connected to the computer for displaying the planning template for enabling selection of the planning template;

means for selecting the planning template;

means responsive to selection of the planning template for displaying at least one planning task for implementing the component of the given type of project;

means for selecting the planning task;

means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected planning task, whereby the planning template is linked to information stored in the database that is related to planning the component of the given type of project;

means connected to the planning template for selecting access to information stored in the database for implementing the component; and

means responsive to information accessed from the database for displaying the accessed information.

**4.** A computer-implement tool comprising:

means incorporated into a computer and executed by the computer to provide a knowledge-based tool for a selection process for a given type of project for selecting a component of the given type of project;

means connected to the means for providing the process tool for providing a database of stored information related to the given type of project;

means responsive to selection of the process for displaying at least one task for implementing the component of the given type of project;

means for selecting the task;

means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected task, whereby the process is linked to information stored in the database that is related to the component of the given type of project;

means responsive to information accessed from the database for displaying the accessed information;

means external to the process tool and database, the external means comprising means for maintaining the information stored in the database for implementing the component;

an Internet connection between the process tool and the external means;

means connected to the process tool for sending a request for component implementing information to a multiple page web site of the external means over the Internet; and

means connected to the external means and responsive to the request for accessing a particular page of pertinent information relating to the component implementing information and communicating the pertinent information on the particular page of the web site to the process tool over the Internet.



IPR2023-00330 Page 00389

5. A computer-implemented planning tool comprising:

means incorporated into a computer and executed by the computer to provide a knowledge-based tool for performing at least one planning process for planning a given type of project, the tool comprising at least one planning template for planning a component of the given type of project;
means connected to the means for providing the planning process tool for providing a database of stored information related to the given type of project;
means connected to the computer for displaying the planning template for enabling selection of the planning template;
means for selecting the planning template;
means responsive to selection of the planning template for displaying at least one planning task for implementing the component of the given type of project;
means for selecting the planning task;
means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected planning task, whereby the planning template is linked to information stored in the database that is related to planning the component of the given type of project;
means connected to the planning template for selecting access to information stored in the database for implementing the component;
means responsive to information accessed from the database for displaying the accessed information;
means external to the planning process tool and database, the external means comprising means for maintaining the information stored in the database for implementing the component;
an Internet connection between the planning process tool and the external means;
means connected to the planning process tool for sending a request for component implementing information to a web site of the external means over the Internet; and
means connected to the external means and responsive to the request for accessing a particular page of pertinent information relating to the component implementing information and communicating the pertinent information on the particular page of the web site to the process tool over the Internet.

6. A computer-implemented tool for planning a project comprising:

means for providing at least one planning process tool for planning a given type of project comprising at least one planning template for planning a component of the given type of project;
means connected to the means for providing the planning process tool for providing a database of stored information related to the given type of project;
means connected to the means for providing the planning process tool for displaying the planning template for enabling selection of the planning template;
means for selecting the planning template;
means responsive to selection of the planning template for displaying at least one planning task for implementing the component of the given type of project;
means for selecting the planning task;
means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected planning task, whereby the planning template is linked to information stored in the database that is related to planning the component of the given type of project;
means connected to the planning template for selecting access to information stored in the database for implementing the component;
means responsive to information accessed from the database for displaying the accessed information and enabling selection of at least one query regarding the displayed accessed information;
means for selecting the query;
means responsive to selecting the query for formatting a request based on the selected query;
means responsive to the request for communicating the request to means external to the planning process tool and database, the external means comprising means responsive to the request for processing the communicated request comprising at least additional information representative of a response to the query and means for communicating the response to the request to the planning process tool; and
means responsive to the response to the communicated request for displaying the additional information related to implementation of the component.

7. A computer-implemented planning tool as defined in one of the claims 1 to 6 wherein the database comprises product information, whereby the database comprises a product catalog.

8. A computer-implemented planning tool as defined in one of the claims 1 to 7 wherein the given type of project comprises a facility comprising rooms and



IPR2023-00330 Page 00390

the planning template comprises means for configuring at least one room design.

**9.** A computer-implemented planning tool as defined in claim 8 wherein the database comprises product information, whereby the database comprises a product catalog, and wherein the component comprises an item selected from among the group of items consisting of equipment, furniture, and accessories.

**10.** A computer-implemented planning tool as defined in claim 9 wherein the facility is a healthcare facility and wherein the equipment comprises medical equipment.

**11.** A computer-implemented planning tool as defined in one of the claims 1 to 10 wherein the database comprises a library of published information, the published information comprising unmodifiable, display-only information and customizable information, the unmodifiable, display-only information being associated with an identifiable source of the published information and the customizable information being associated with an anonymous information source and wherein the means connected to the database for enabling access to at least a portion of the stored information in the database based on the selected planning task enables access to the library in the database based on the selected planning task, whereby the planning template is linked to the library stored in the database related to planning the component of the given type of project, and further comprising:

means connected to the planning template for selecting access to the library stored in the database for providing information from the library stored in the database relating to the component; and

means responsive to information from the library accessed from the database for displaying the accessed information.

**12.** A computer-implemented planning tool as defined in claim 11, further comprising means connected to the means responsive to information from the library accessed from the database for displaying the accessed information for importing only customizable information from the accessed library information for editing and saving in an edited file.

**13.** A computer-implemented planning tool as defined in claim 11 wherein the unmodifiable, display-only information corresponds to library information submitted by the identifiable source together with a password assigned to the identifiable source and the unmodifiable, display-only information is maintained by the identifiable source.

**14.** A computer-implemented planning tool as defined in claim 13, further comprising an Internet connection between the planning process tool and the identifiable and anonymous sources, and wherein library information is accessible from the identifiable and anonymous sources by the planning process tool over the Internet.

**15.** A computer-implemented planning tool as defined in one of the claims 1 to 14, further comprising means for calculating a quantity of the implementing component.

**16.** A computer-implemented planning tool as defined in one of the claims 1 to 15, further comprising means for calculating a cost for the quantity of the implementing component.

**17.** A computer-implemented planning tool as defined in one of the claims 1 to 11, further comprising means for calculating a quantity of the implementing component.

**18.** A computer-implemented planning tool as defined in claim 3, further comprising means external to the planning process tool and database and an Internet connection between the planning process tool and the external means, the external means comprising means for maintaining the information stored in the database for implementing the component and submitting the maintenance information over the Internet together with a password assigned to the external means.

**19.** A computer-implemented tool as defined in claim 4 or 5, further comprising means connected to the external means for maintaining the pertinent information relating to the component implementing information.

**20.** A computer-implemented planning tool as defined in claim 6 wherein the means responsive to information accessed from the database for displaying the accessed information and enabling selection of at least one query regarding the displayed accessed information comprises means for enabling selection of a query selected from among the group of queries consisting of information and quotation and the means responsive to selecting the query for formatting a request based on the selected query comprises means for assembling an e-mail message comprising the selected query and header data containing one of the e-mail address corresponding to the external means and a supplier of the planning process tool and a return e-mail address corresponding to the address of the plan-



IPR2023-00330 Page 00391

ning process tool and the means responsive to the request for communicating the request to means external to the planning process tool and database comprises means connectable to the Internet for transmitting the e-mail message to one of the external means and the supplier of the planning process tool.

**21.** A computer-implemented planning tool as defined in claim 20 wherein the means responsive to information accessed from the database for displaying the accessed information and enabling selection of at least one query regarding the displayed accessed information comprises displaying a respective check box for enabling selection of a query selected from among the group of queries consisting of information and quotation and wherein information is requested as a default for failure of selection of a check box.

**22.** A computer-implemented planning tool as defined in claim 20 wherein the e-mail address of the request corresponds to the external means and the request is transmitted to the external means and wherein the means responsive to the request for processing the communicated request comprising at least additional information representative of a response to the query comprises means for assembling an e-mail message comprising the additional information and the return e-mail address corresponding to the address of the planning process tool and the means for communicating the response to the request to the planning process tool comprises means connectable to the Internet for transmitting the e-mail message response to the planning process tool.

**23.** A computer-implemented planning tool as defined in claim 22, further comprising:

means connected to the planning process tool responsive to the additional information for storing the additional information in an e-mail queue and wherein the means connected to the planning template for selecting access to information stored in the database for implementing the component also enables access to the additional information stored in the e-mail queue for implementing the component and the means responsive to information accessed from the database for displaying the accessed information also is responsive to the additional information accessed from the e-mail queue for displaying the accessed additional information.

**24.** A computer-implemented planning tool as defined in claim 20 wherein the e-mail address of the request corresponds to the supplier of the planning process tool and the request is transmitted to the supplier, and further comprising means at the address of the supplier for responding to the request by converting the message to a facsimile transmission and sending the facsimile transmission to means for receiving a facsimile transmission at a telephone number corresponding to the external means.

**25.** A computer-implemented planning tool as defined in claim 6, further comprising means connected to the planning process tool for enabling incorporation of additional customizable information into the query for communication to the external means.



IPR2023-00330 Page 00392

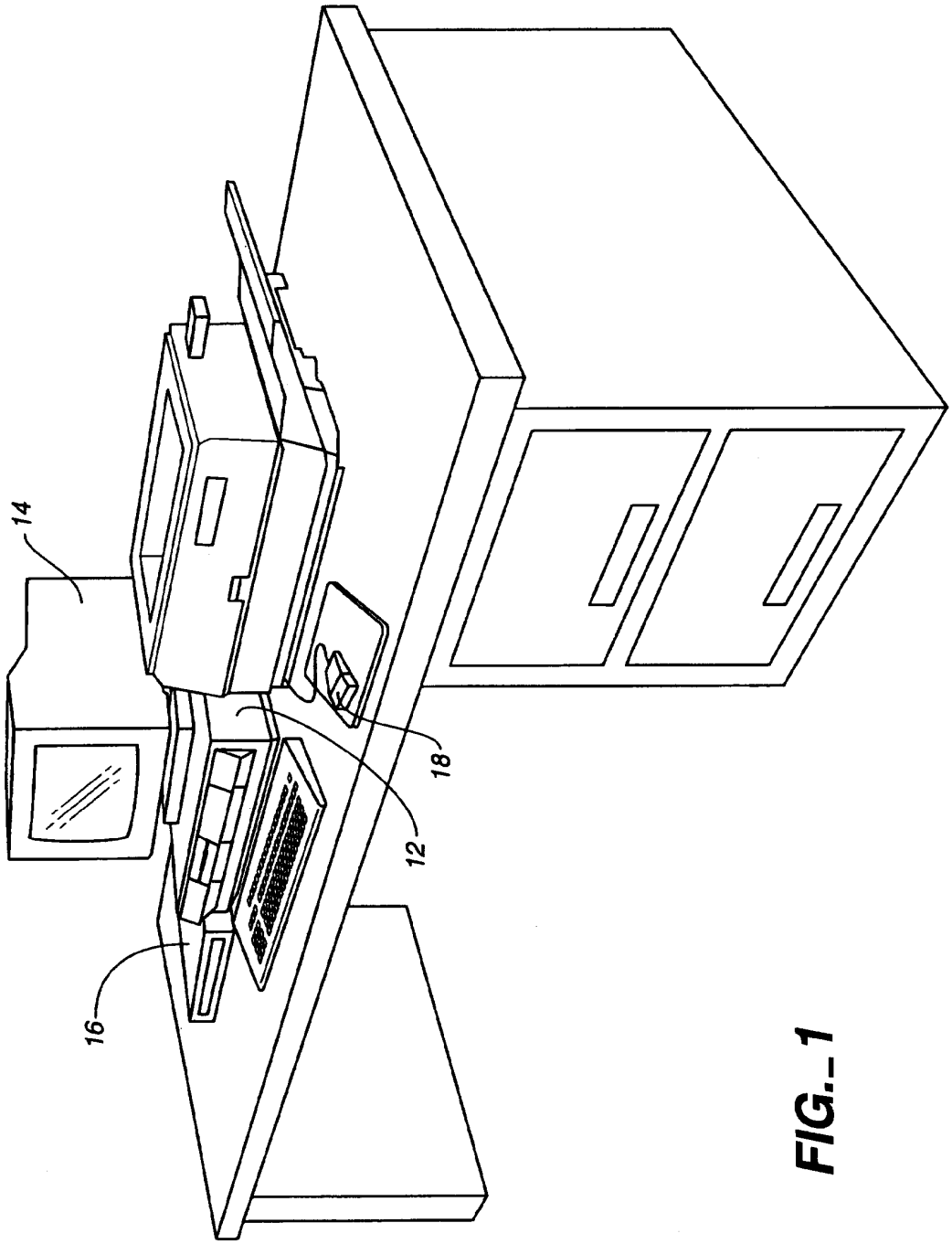



FIG._1



IPR2023-00330 Page 00393

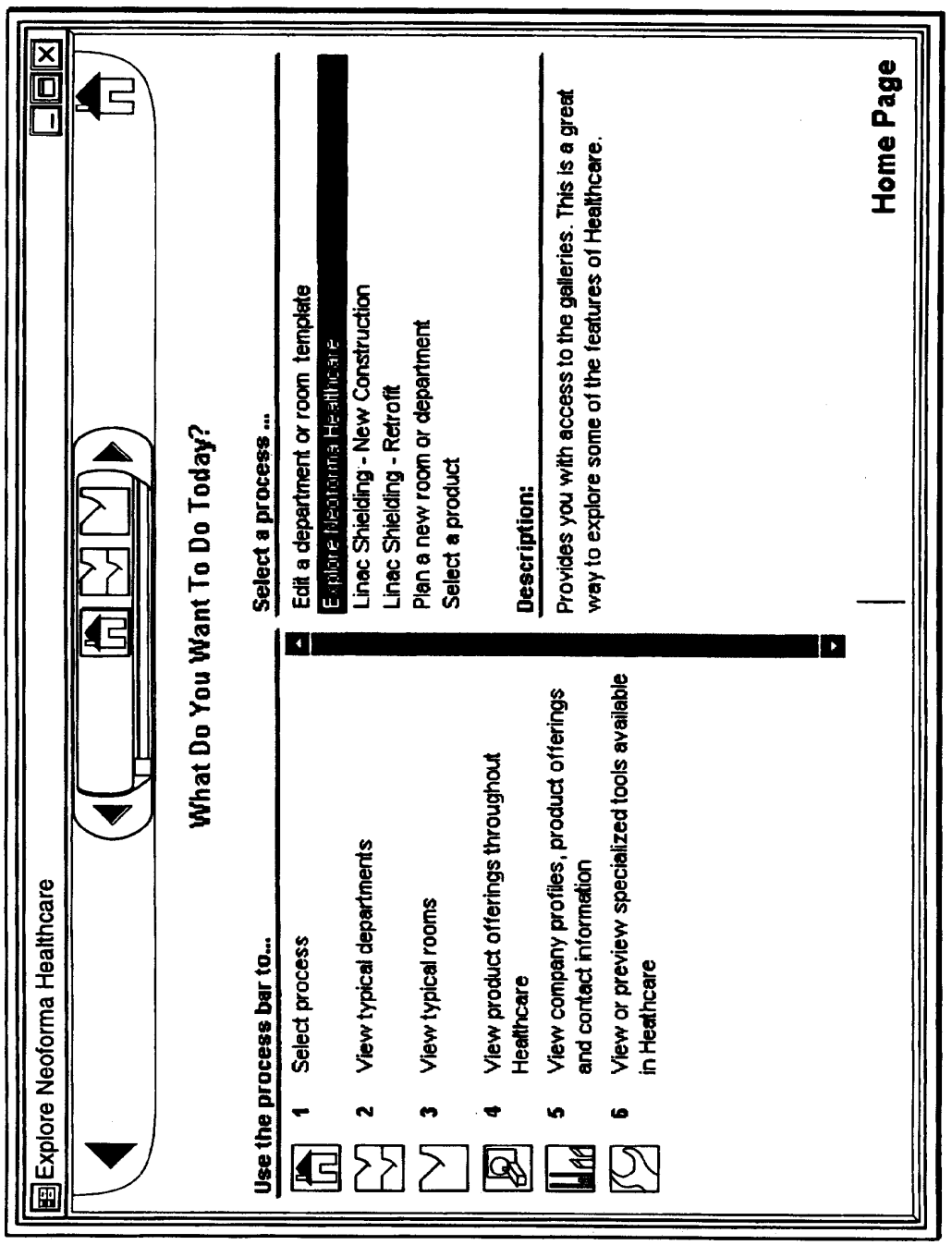



FIG._2



IPR2023-00330 Page 00394

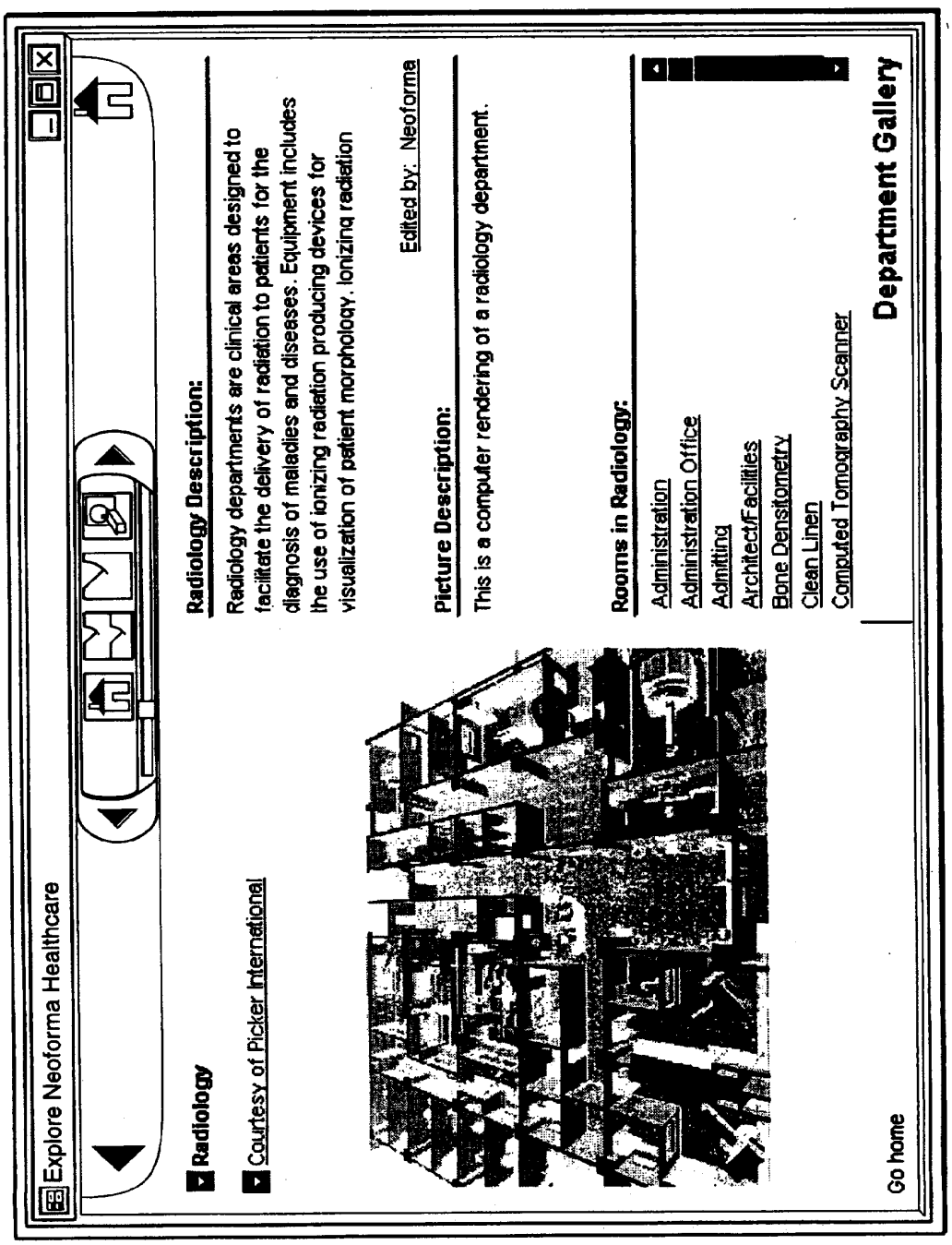



FIG._3



IPR2023-00330 Page 00395

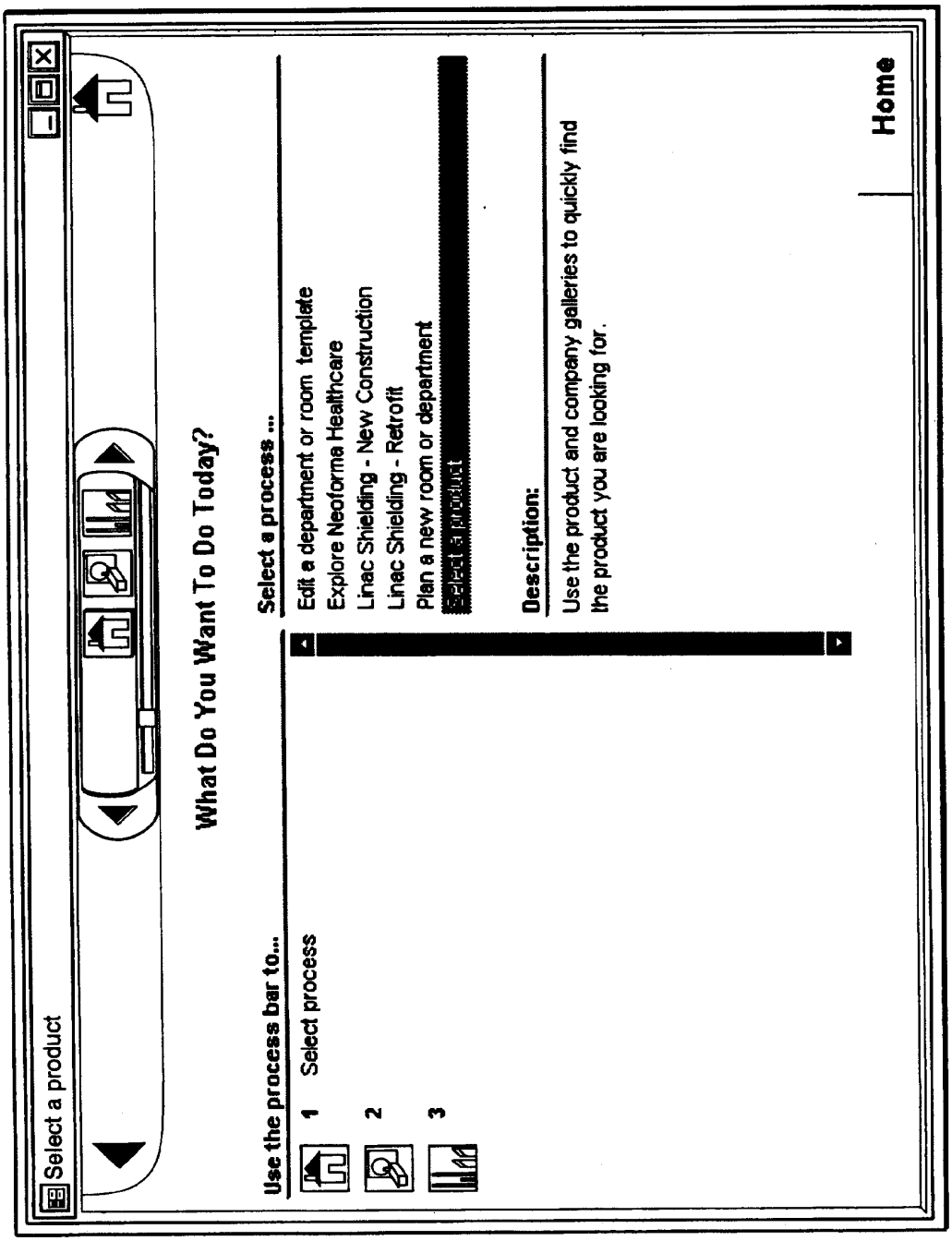



FIG._4



IPR2023-00330 Page 00396

FIG._5



IPR2023-00330 Page 00397

FIG._6



IPR2023-00330 Page 00398

FIG._7



IPR2023-00330 Page 00399

FIG._8



IPR2023-00330 Page 00400

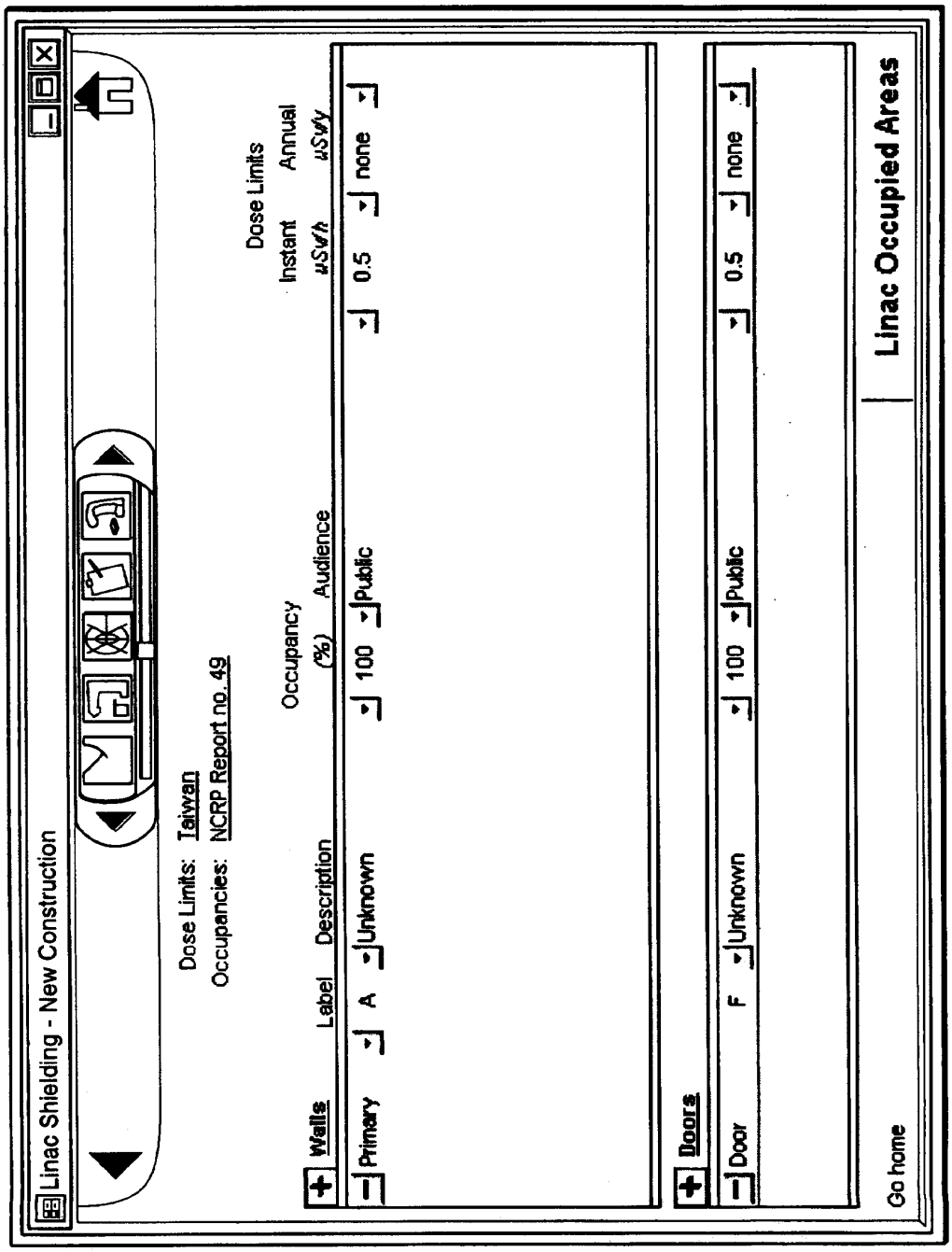



FIG._9



IPR2023-00330 Page 00401

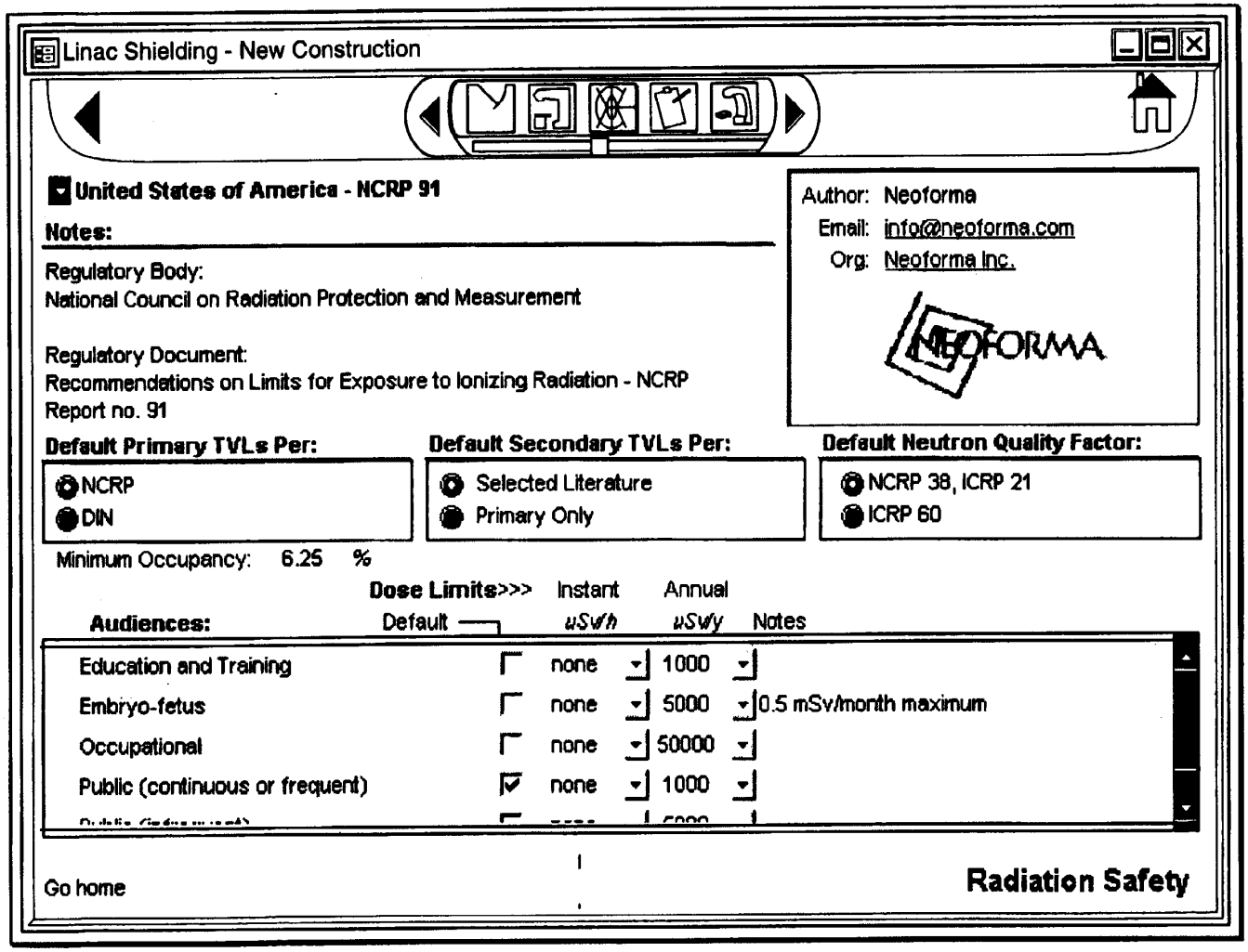



FIG._10



EP 0 895 171 A2

IPR2023-00330 Page 00402

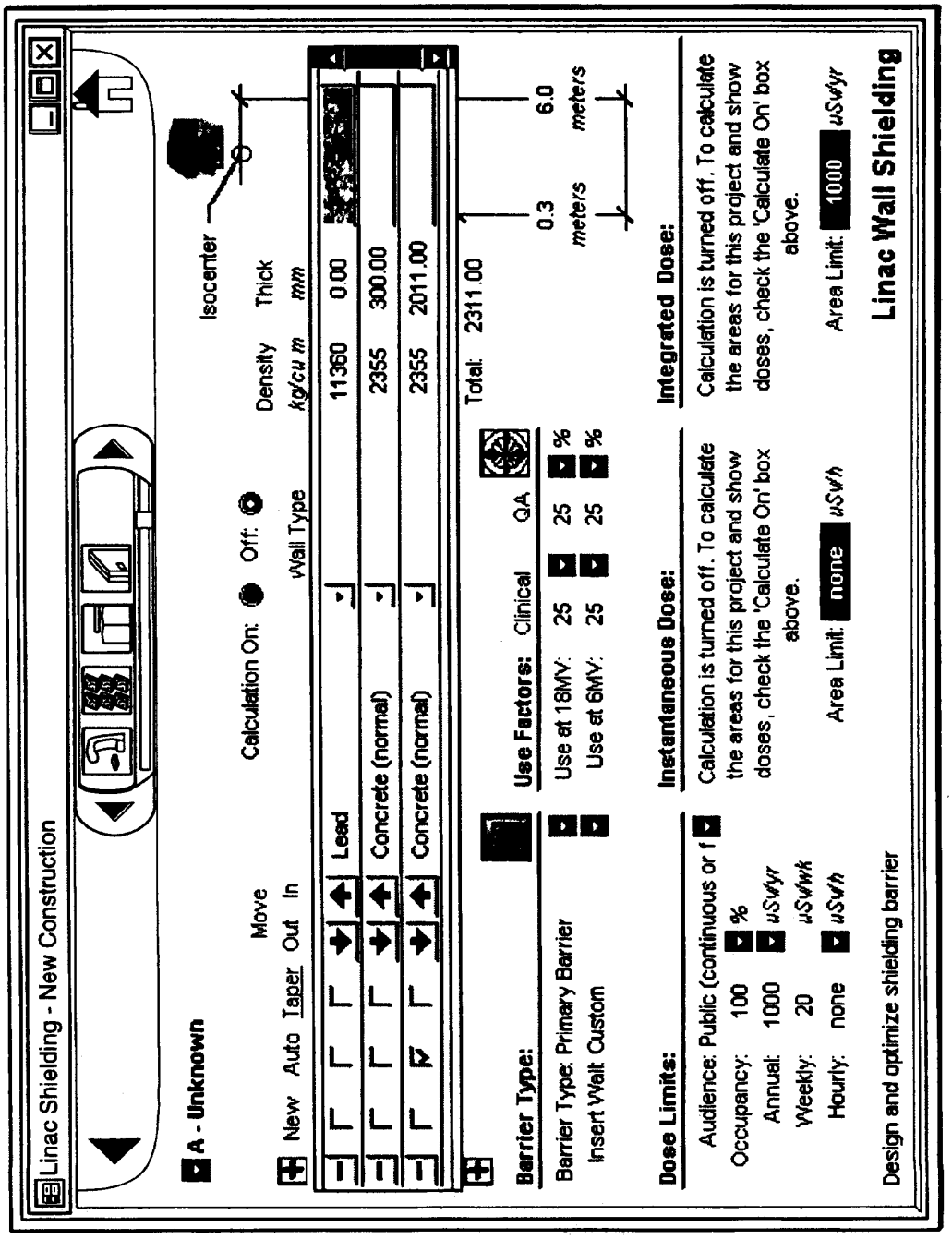



FIG._11



IPR2023-00330 Page 00403

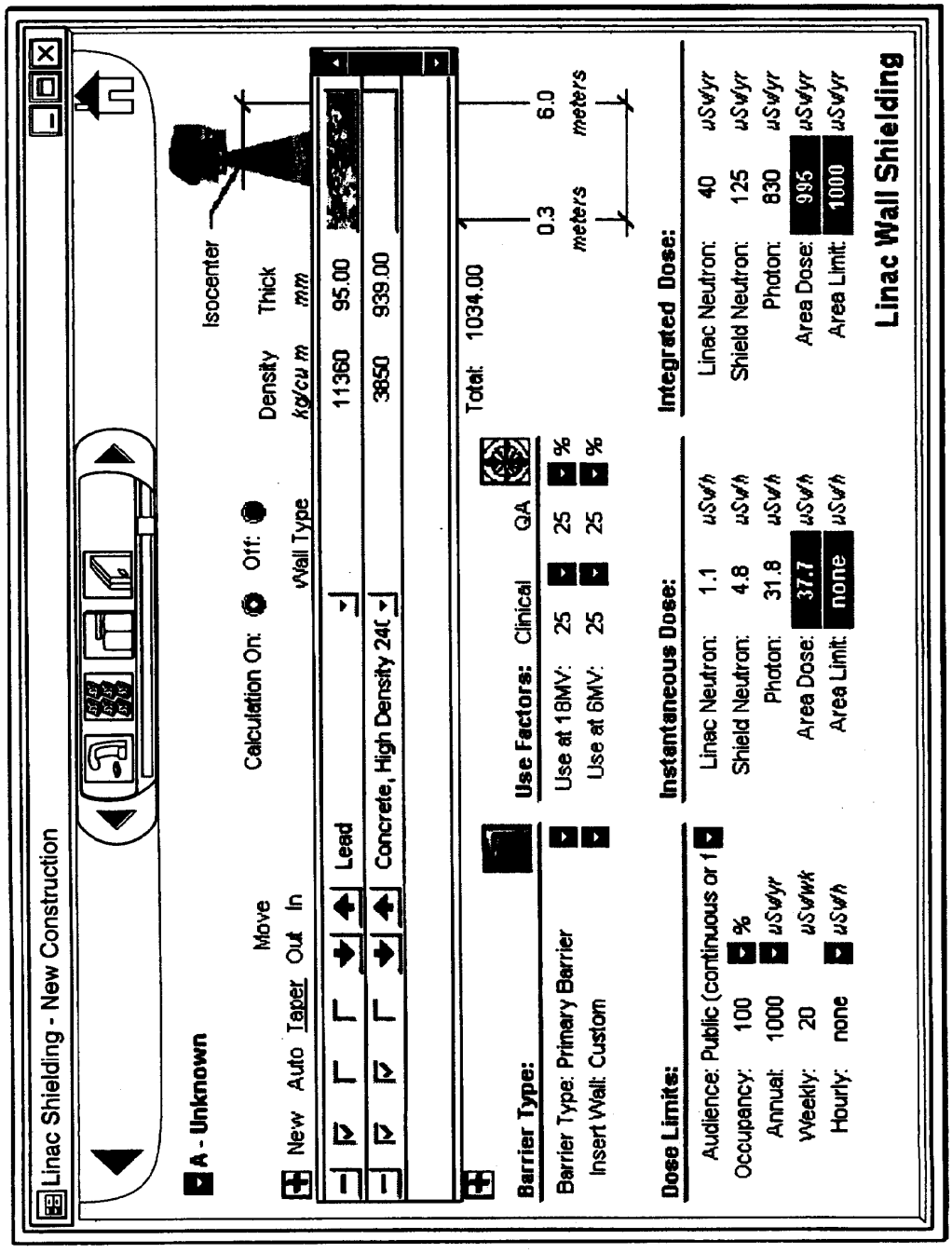



FIG._12



IPR2023-00330 Page 00404

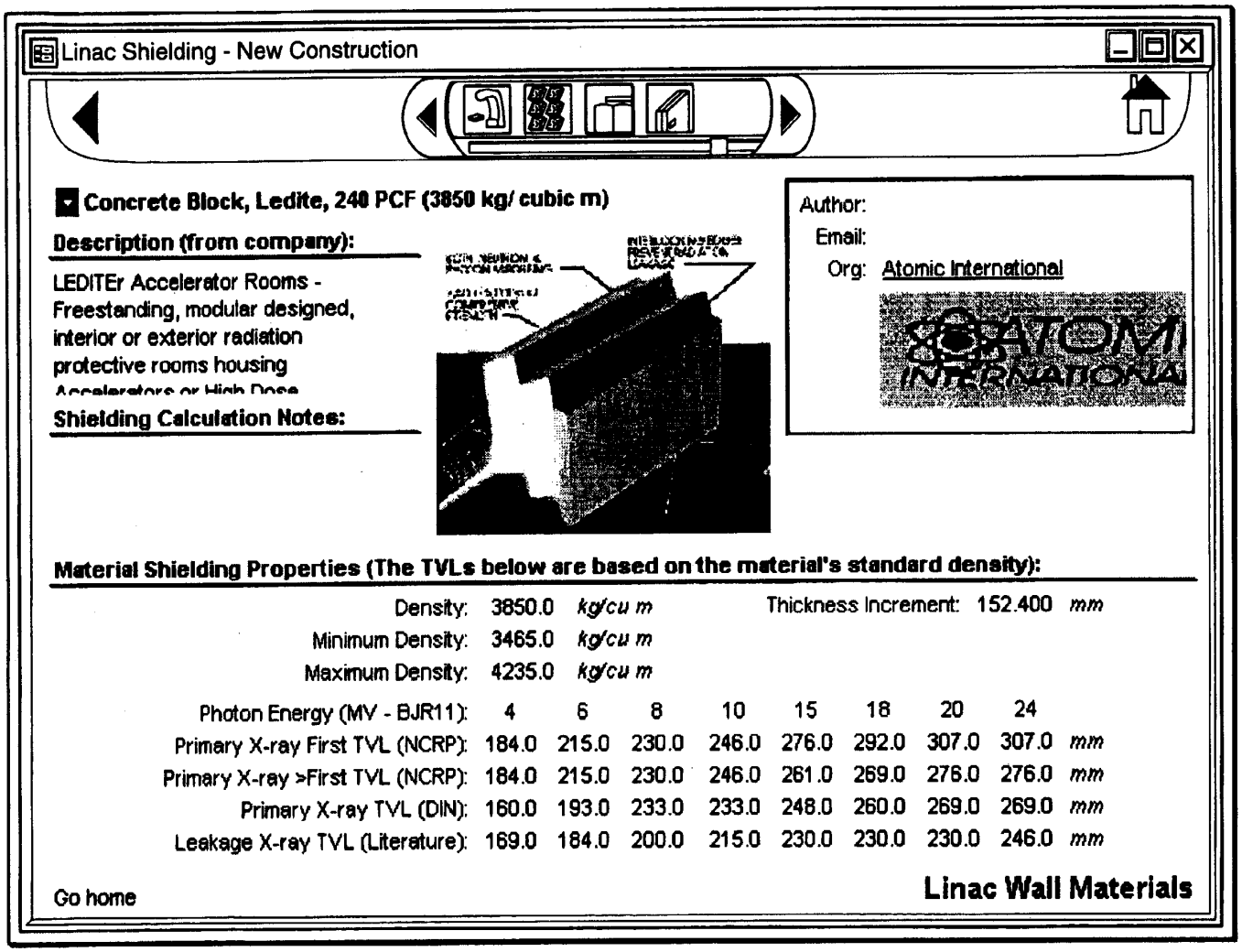



FIG._13



IPR2023-00330 Page 00405

FIG._14



IPR2023-00330 Page 00406

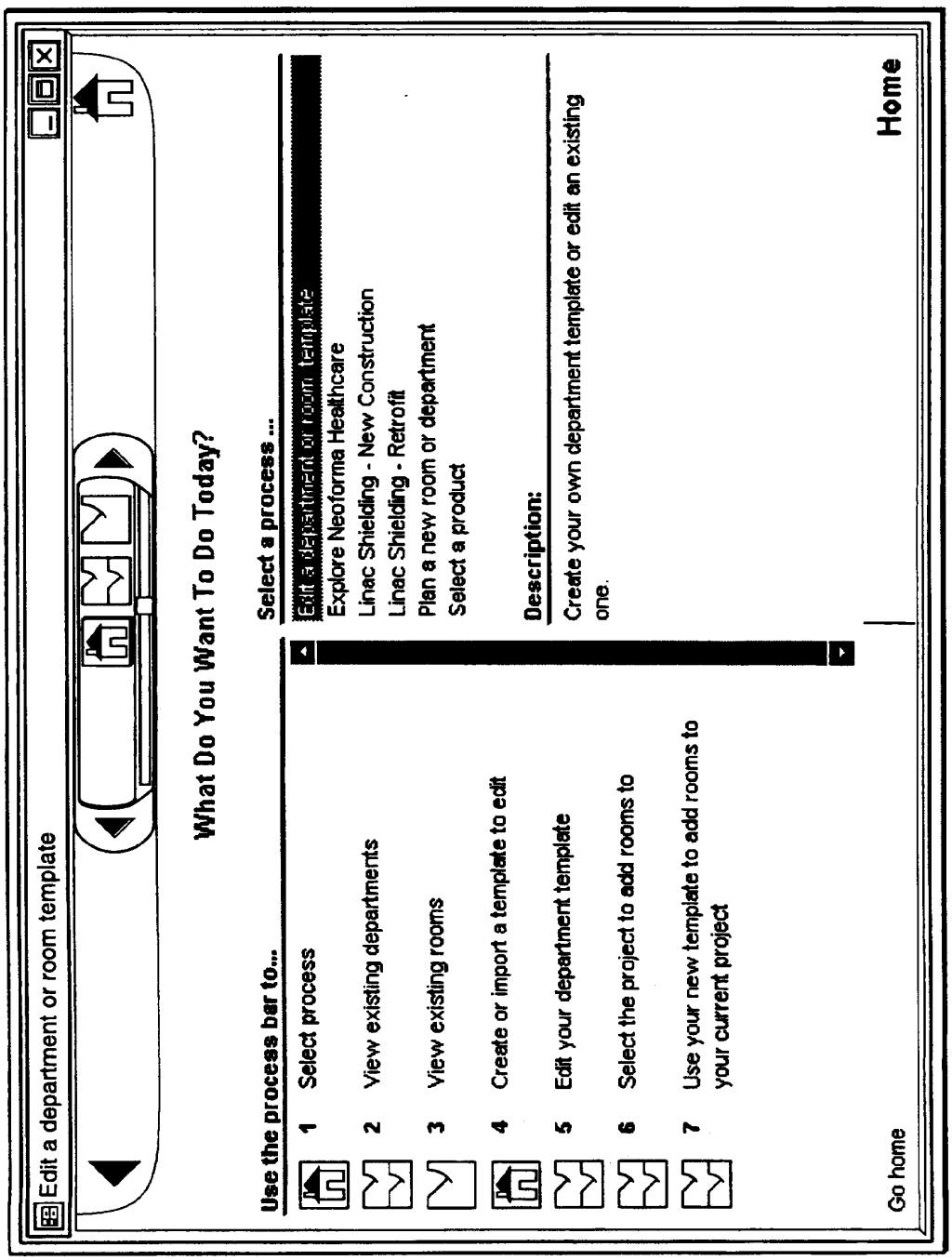



FIG._15



IPR2023-00330 Page 00407

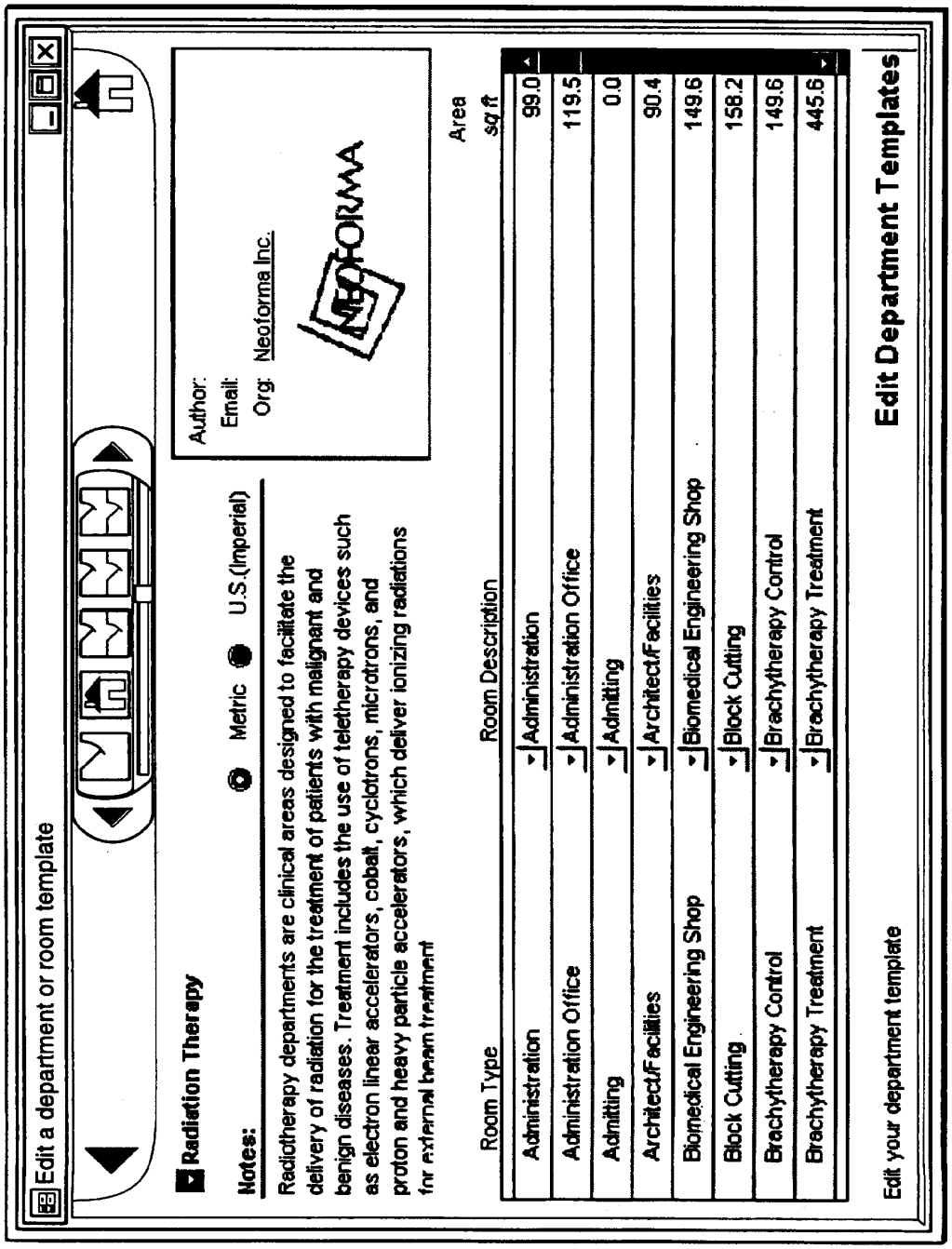



FIG._16



IPR2023-00330 Page 00408

FIG._17



IPR2023-00330 Page 00409

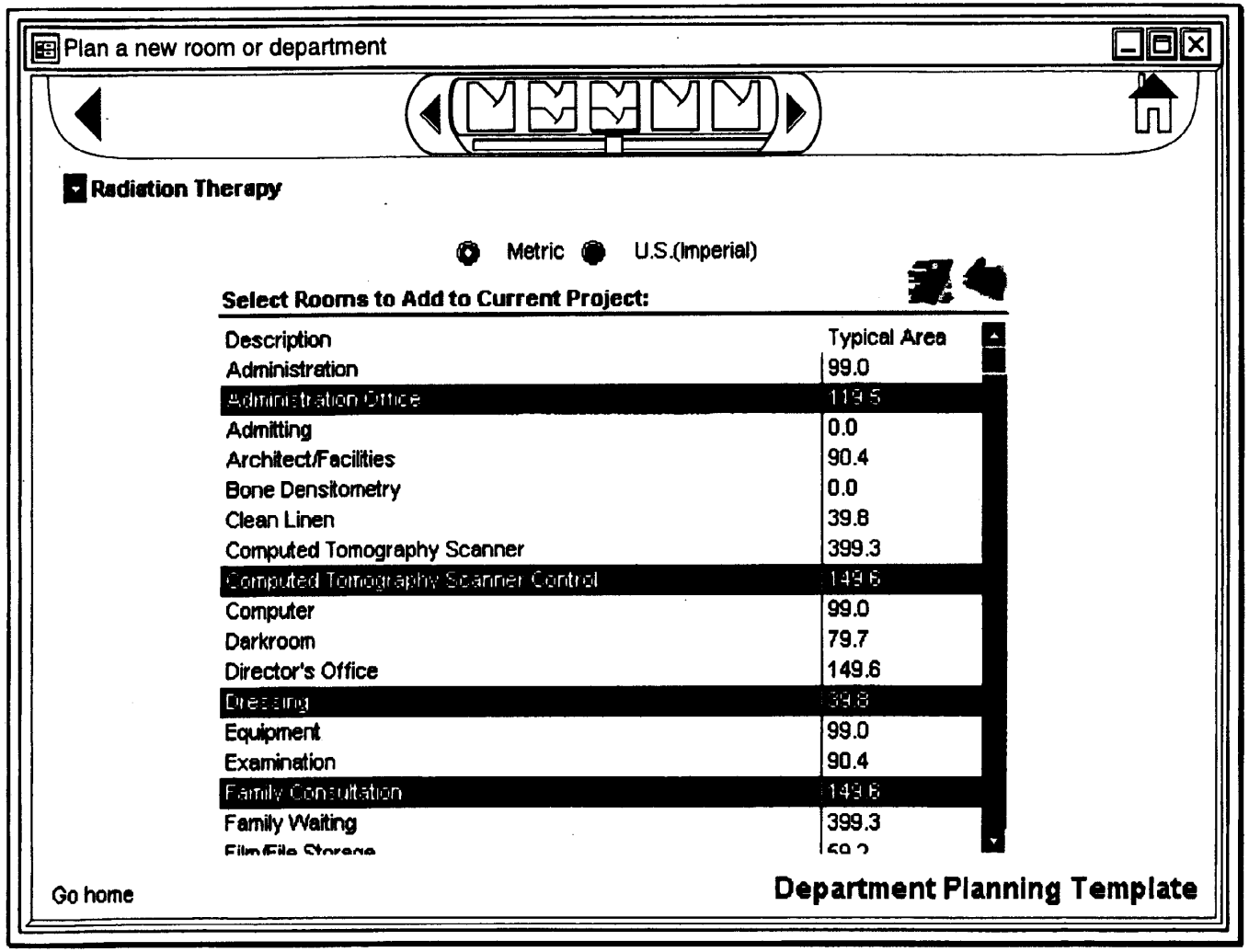



FIG._18

EP 0 895 171 A2

IPR2023-00330 Page 00410

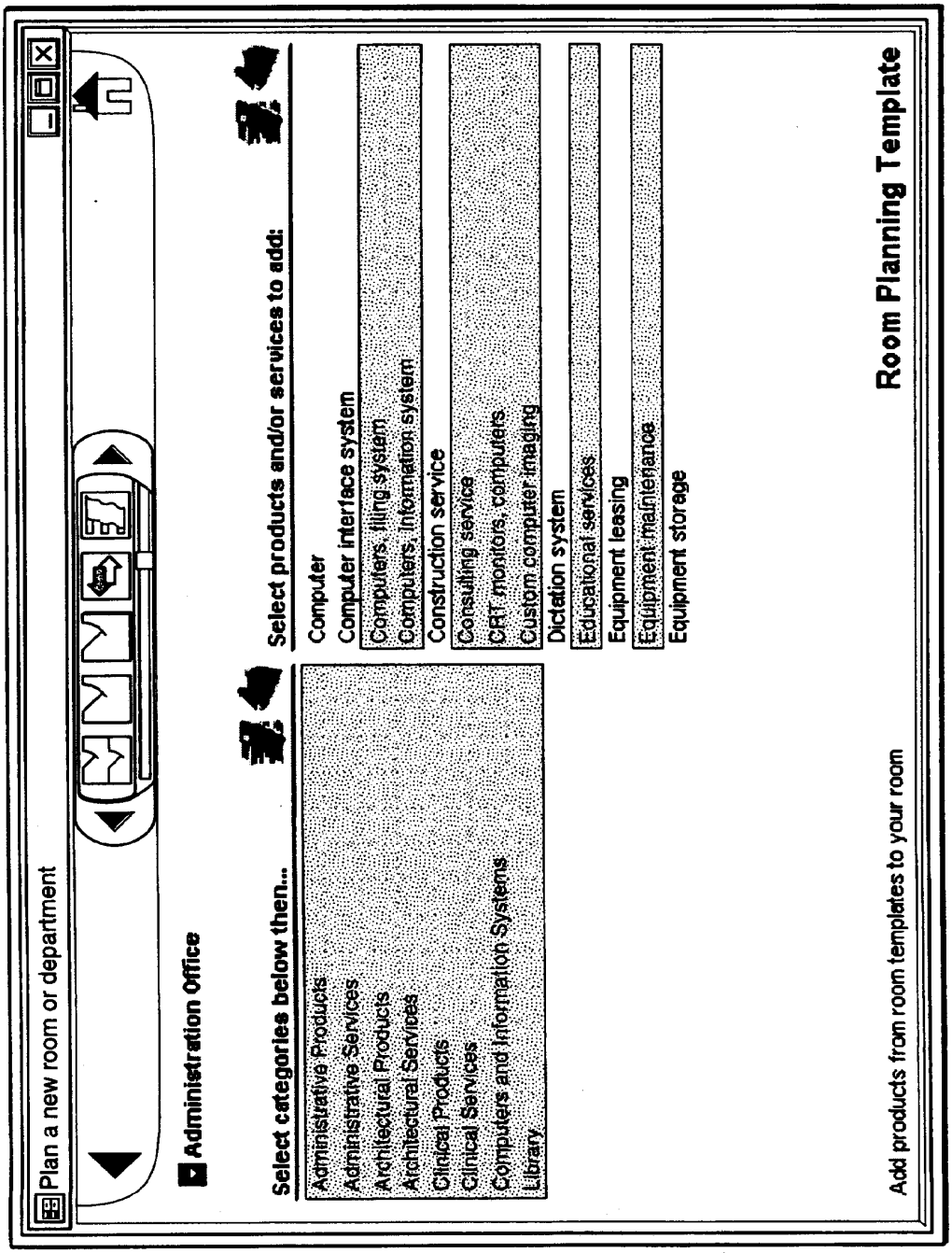



FIG._19



IPR2023-00330 Page 00411

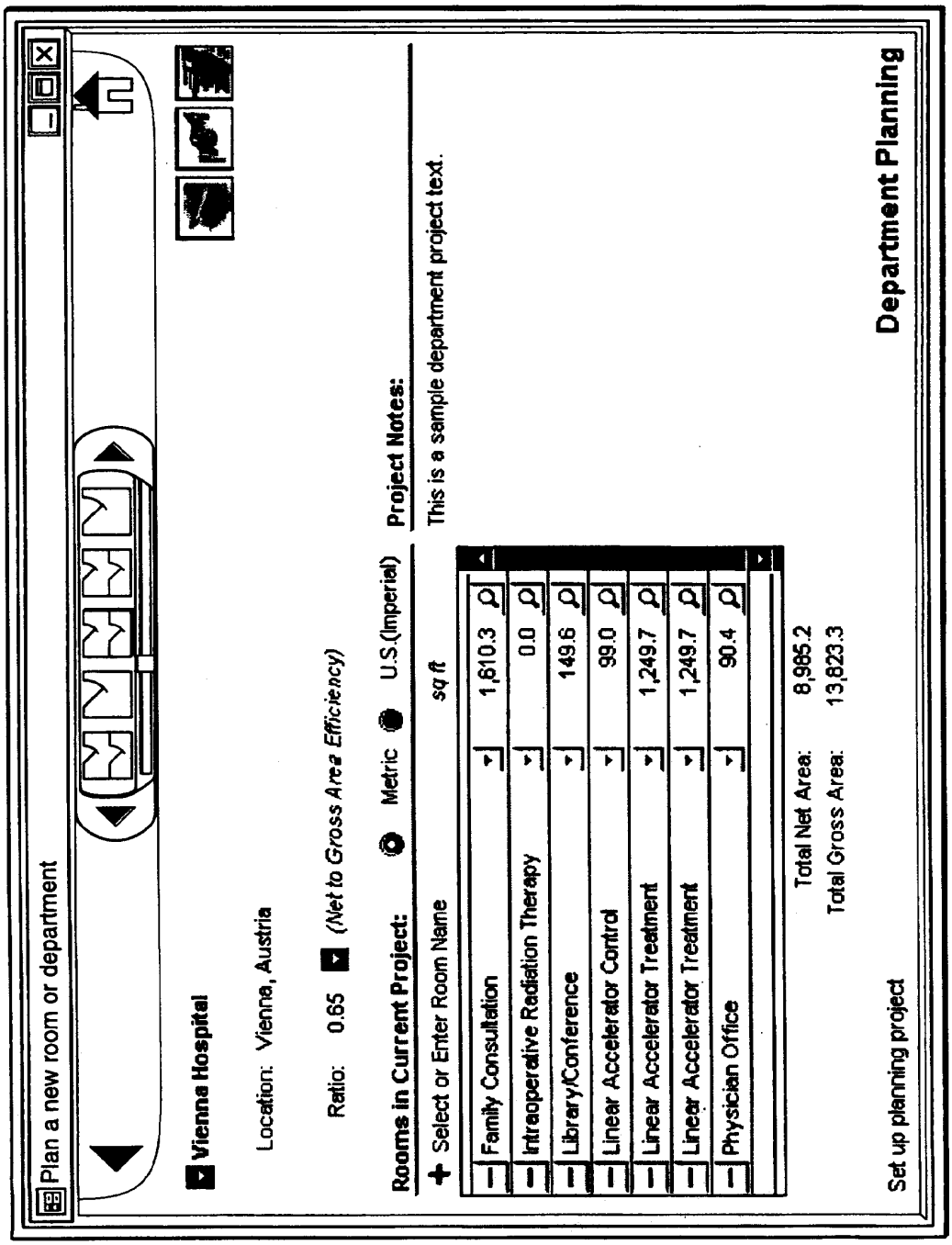



FIG._20



IPR2023-00330 Page 00412

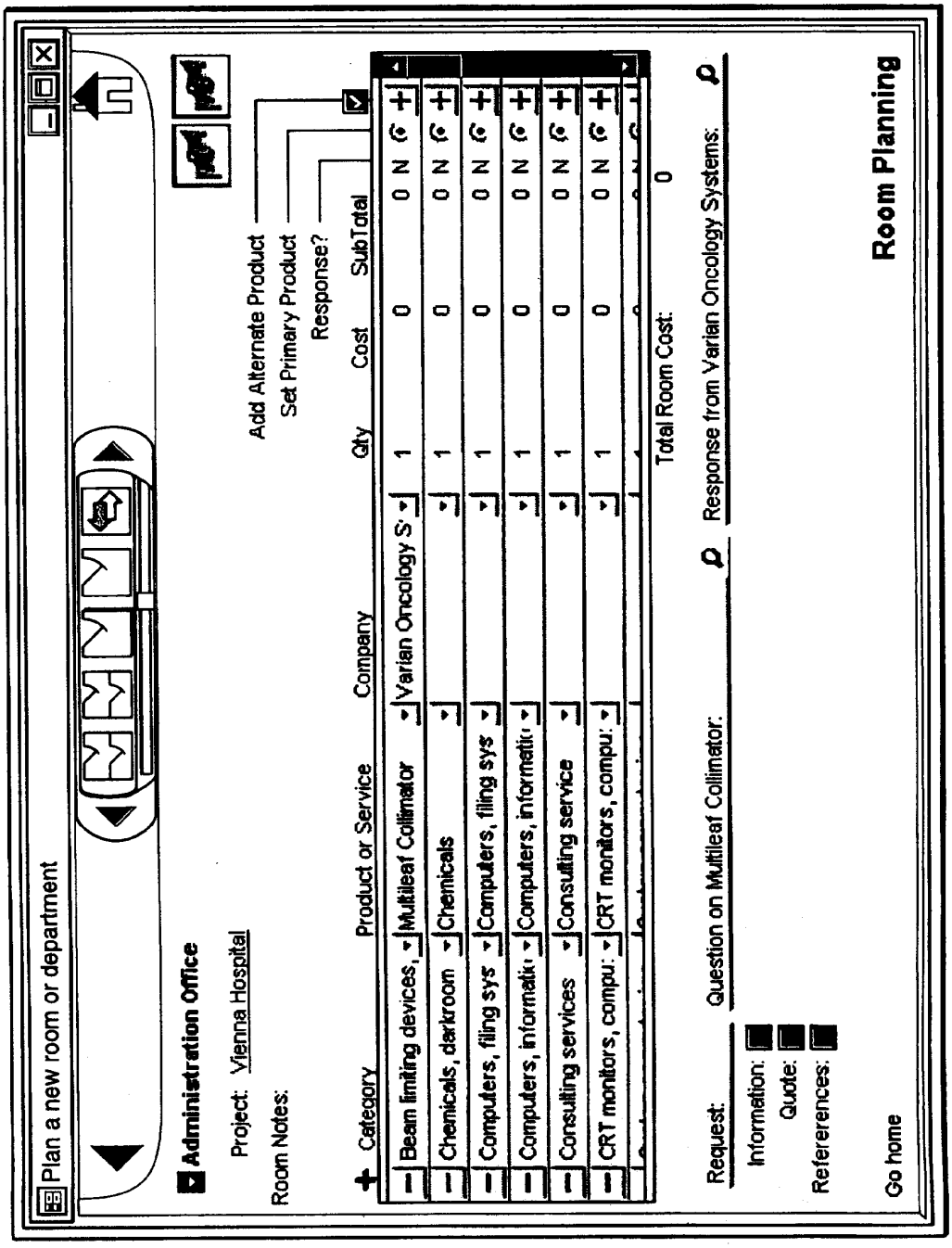



FIG._21



IPR2023-00330 Page 00413

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 926 607 A2

# (12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
**30.06.1999 Bulletin 1999/26**

(51) Int. Cl.[6]: **G06F 17/30**

(21) Application number: **98124276.1**

(22) Date of filing: **18.12.1998**

(84) Designated Contracting States:
**AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE**
Designated Extension States:
**AL LT LV MK RO SI**

(30) Priority: **23.12.1997 US 997482**
**23.12.1997 US 997705**

(71) Applicant: **Ricoh Company**
**Tokyo 143-8555 (JP)**

(72) Inventors:
- **Motoyama, Tetsuro,**
  **c/o RICOH CORPORATION,**
  **Systems**
  **San Jose, CA 95134-2088 (US)**
- **Fong, Avery,**
  **c/o RICOH CORPORATION,**
  **Systems**
  **San Jose, CA 95134-2088 (US)**
- **Bhatnagar,Anurag,**
  **c/o RICOH CORPORATION,**
  **Systems**
  **San Jose, CA 95134-2088 (US)**

(74) Representative:
**Schwabe - Sandmair - Marx**
**Stuntzstrasse 16**
**81677 München (DE)**

## (54) Object-oriented system for mapping structered information to different structured information

(57) An object-oriented system and computer program product for mapping structured information to different structured information, which allows a user to interactively define the mapping. The present invention operates as an object-oriented user tool by accepting interactive input from a user of a source input, by processing the input to display the source input in a format for accepting and processing user commands to create or edit a transformation map of source components to target components. Interactive user input is then accepted and processed for selection of an input file to be transformed and selection of a transformation map to be used for the requested transformation. Interactive user input is accepted and processed for selection of individual components of the first structured information format for mapping, and for selection of options for the target components. Exemplary options for the target components are a null value, the source component itself, a single selected target component, or plural selected target components. Interactive user input is accepted for processing to assign attribute values to components of the second structured information format. Exemplary options for the sources of attribute values are attribute values obtained from the source components, system attribute values, no value, attribute values input interactively by the user, and content of element. Interactive user input is then accepted and processed to initiate processing of a transformation of the source input file in the first structured information format to a target output file in the second structured information format.

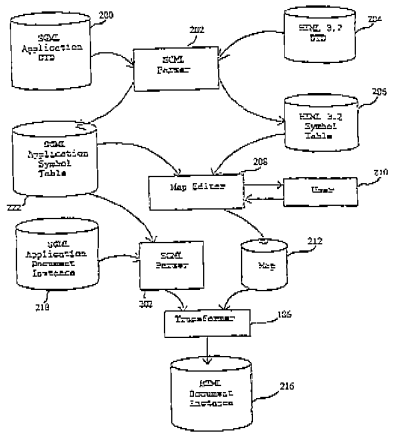


Fig. 5

EP 0 926 607 A2

## Description

### CROSS-REFERENCES TO RELATED APPLICATION

**[0001]** This application is related to and being concurrently filed with another patent application: U.S. Patent Application S/N 08/XXX,XXX, Attorney Docket No. 5244-0063-2X, entitled "Method and Apparatus For Mapping Structured Information to Different Structured Information" filed on _____, 1997, and incorporated herein by reference.

### BACKGROUND OF THE INVENTION

#### Field of the Invention

**[0002]** This invention relates generally to mapping structured information to different structured information in an object-oriented framework. The present invention relates more specifically to processing a document encoded in a markup language format, a database information format, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, or a DOS file name scheme, transforming it into another markup language format, another database information format, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, or a DOS file name scheme, in an object-oriented framework. The invention is more specifically related to a system and computer program product for mapping in which a user interactively defines the mapping for the transformation in an object-oriented framework.

**[0003]** This invention also relates generally to providing a user interface for mapping structured information to different structured information. The present invention relates more specifically to providing a user interface for processing a document encoded in a markup language format, a database information format, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, or a DOS file name scheme, transforming it into another markup language format, another database information format, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, or a DOS file name scheme. The invention is more specifically related to a method and apparatus for providing a user interface for mapping in which a user interactively defines the mapping for the transformation.

#### Discussion of the Background

**[0004]** Standard Generalized Markup Language ("SGML") is an information management standard adopted by the International Organization for Standardization ("ISO"), as ISO 8879:1986, as a means for providing platform-independent and application-independent documents that retain content, indexing, and linked information. SGML provides a grammarlike mechanism for users to define the structure of their documents and the tags they will use to denote the structure in individual documents. A complete description of SGML is provided in Goldfarb, C. F., *The SGML Handbook*, Oxford University Press, Oxford, 1990, and McGrath, S., *Parseme.1st: SGML for Software Developers*, Prentice Hall PTR, New Jersey, 1998, which are incorporated herein by reference.

**[0005]** HyperText Markup Language ("HTML") is an application of SGML that uses tags to mark elements, such as text or graphics, in a document to indicate how Web browsers should display these elements to the user and should respond to user actions such as activation of a link by means of a key press or mouse click. HTML is used for documents on the World Wide Web. HTML 2.0, defined by the Internet Engineering Task Force ("IETF"), includes features of HTML common to all Web browsers as of 1995, and was the first version of HTML widely used on the World Wide Web. Future HTML development will be carried out by the World Wide Web Consortium ("W3C"). HTML 3.2, the latest proposed standard, incorporates features widely implemented as of early 1996. A description of SGML and HTML features is given in Bradley, N., *The Concise (SGML) Companion*, Addison Wesley Longman, New York, 1997, which is incorporated herein by reference.

**[0006]** Object Oriented Programming ("OOP") is a programming methodology in which a program is viewed as a collection of discrete objects that are self-contained collections of data structures and routines that interact with other objects. Many high-level languages, including C++, support the declaration of a class. The class is typically a template, or detailed description, of the objects, or instances of objects, which will be created, or instantiated, by a constructor function during program execution and destroyed by a destructor function when the object is no longer needed. A conversational reference to a class includes all of the objects currently in existence as a result of constructor calls. A class is made up of data items, structures, and methods. Data items correspond to variables of prior programming art. Structures are named groupings of related data items and other structures. Methods correspond to functions and subroutines of prior programming art.

**[0007]** An object-oriented framework is a reusable basic design structure, consisting of abstract and concrete classes, that assists in building applications.

**[0008]** Pointers, used for accessing specific objects, data items, and methods, are data items which contain system equivalents of absolute addresses in computer memory. Null pointers, or zero pointers, are pointer variables or literals

which have been assigned a system value, for example, zero, denoting that a specific pointer is currently pointing to a null, or non-existent item. References and reference variables are generally data items which contain system equivalents of absolute addresses in computer memory.

**[0009]** A string variable or a string literal is a data structure composed of a sequence of characters of the character set of a particular application. A null string, a nil string, or an empty string is a string which contains no characters.

**[0010]** The three main features of object oriented programming are inheritance, encapsulation, and polymorphism. Inheritance allows a programmer to establish a general class with features which are desirable for a wide range of objects. For example, if a programmer designs a class polygon having certain features such as a closed convex shape made up of plural straight lines joined pairwise at vertices, it is then possible to construct polygon subclasses such as triangles, quadrilaterals, pentagons, and hexagons, all having the shared properties of the parent class polygon, with additional constraints on the number of sides to be allowed for the objects generated. It is also possible, for example, to have subclasses of class quadrilateral such as rectangle and rhombus. A class square inherits all features of the class rectangle and additionally has all of its sides equal in length. The class polygon is considered an abstract class, in that instantiations of actual objects is performed only in its subclasses. However, the class polygon establishes certain properties inherent to all of the non-abstract, or concrete subclasses for inheritance purposes.

**[0011]** Encapsulation and polymorphism have already been described, and are already well known, in patents relating to object oriented systems. A comprehensive discussion of OOP is provided in Coad, P. and Yourdon, E., *Object-Oriented Analysis, Second Edition*, Prentice-Hall, Inc., New Jersey, 1991, and in Booch, G., *Object-Oriented Analysis and Design with Applications, Second Edition*, Addison Wesley Longman, California, 1994, which are incorporated herein by reference.

**[0012]** A Graphical User Interface ("GUI") is an environment that represents programs, files, and options by means of icons, menus, and dialog boxes on a screen. An icon is an image, displayed on a screen or other output device, that can be manipulated by a user. By serving as a visual pictorial representation of a function that is available, an icon generates a user-friendly interface by freeing the user of the burden of having to remember commands or type them on a keyboard. A menu is a list of options from which the user can make a selection to perform a desired action. A dialog box is a special window, or area, displayed on a screen or other output device, to solicit a response from the user. In a GUI, the user can select and activate options by pointing and clicking with a mouse or by keystrokes on the keyboard. The preceding descriptions were derived from definitions given in the *Computer Dictionary, Third Edition*, Microsoft Press, Washington, 1997.

**[0013]** ISO and International Electrotechnical Commission ("IEC") form a specialized system for worldwide standardization. ISO/IEC 9070:1991(E) is an international standard which is applied to an assignment or unique owner prefixes to owners of public text conforming to ISO 8879. The standard describes the procedures for making an assignment and the method for constructing registered owner names from them. Procedures for self-assignment of owner prefixes by standards bodies and other organizations are also specified. ISO/IEC 9070:1991(E) is incorporated herein by reference.

**[0014]** UNIX and DOS are well-known operating systems for computers. Both UNIX and DOS support a file naming scheme which involve a path from a root directory, through descendant directories, to leaf nodes which are non-directory file names.

**[0015]** Processing systems are known in which a data processor converts a document encoded in a markup language automatically to another format. For example, Balise software from Computing Art, Inc. processes documents encoded in SGML to convert them to a formatted output for user viewing. However, this software does not allow the user to interactively define the mapping of SGML tags to another format.

## SUMMARY OF THE INVENTION

**[0016]** Accordingly, one object of this invention is to provide a novel object-oriented system and computer program product which can process information encoded in a structured information format to transform the information into another structured information format, and which allows a user to interactively define the mapping for the transformation. Exemplary structured information formats include markup language formats, database information formats, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, and a DOS file name scheme. Exemplary users include human users, software methods, and software objects.

**[0017]** It is a further object of this invention to provide a novel object-oriented system and computer program product for conversion of Standard Generalized Markup Language ("SGML") documents into HyperText Markup Language ("HTML") documents, allowing a user to interactively define the mapping for the transformation.

**[0018]** It is a further object of this invention to provide a novel object-oriented system and computer program product for conversion of information in a database format into information in a different database format, which allows a user to interactively define the mapping for the transformation.

**[0019]** It is a further object of this invention to provide a novel object-oriented system and computer program product

for conversion of information in an ISO/IEC 9070 naming scheme into a UNIX file name scheme, which allows a user to interactively define the mapping for the transformation.

**[0020]** It is a further object of this invention to provide a novel object-oriented system and computer program product for conversion of information from an ISO/IEC 9070 naming scheme into a DOS file name scheme, which allows a user to interactively define the mapping for the transformation.

**[0021]** These and other objects are accomplished by an object-oriented system and computer program product for processing information encoded in a structured information format, to transform the information into another structured information format, which allows a user to interactively define the mapping for the transformation.

**[0022]** An exemplary transformation for the present invention is conversion of SGML documents into HTML documents. For explanation of this example, the present invention has been developed as an object-oriented tool to allow a user to define the transformation of an SGML document into an HTML document or other structured format, for example, a database information format. The user tool for this example is currently implemented in the format of a Graphical User Interface ("GUI") using Object Oriented Programming ("OOP") technology. For this example, the current invention is designed to provide a user with an object-oriented graphic tool to transform documents written in a cryptic SGML format into another structured format for greater viewing ease and for greater portability of documents and information. A user interface object and a map creator object allow the user to select an option of performing a default or conditional mapping. The user is allowed to select an input SGML Document Type Definition ("DTD") object, or a currently existing map object. If the user selects an input SGML DTD object, the user interface object requests that a ParserService object process elements of the SGML DTD into component parts to produce an SGML symbol table object. The user interface object displays individual source component objects of the input for the user to input a selection. A user input object is utilized for accepting the selection. The user interface object and the map creator object provide the user with options for transformation of the individual source component objects such as a mapping of a source component object to a target null value, a mapping of a source component object to itself, a mapping of a source component object to a single target component object, or a mapping of a single source component object to plural target component objects. If the user selects a conditional mapping, then the map creator object checks for special cases, such as a history of an element being referenced previously, and processes special cases using further interactive input from the user by the user interface object.

**[0023]** The user interface object and map creator object also provide the user with options for assigning attribute values for the target components. Exemplary options are attribute values obtained from the source components, system attribute values, no value, and attribute values input interactively by the user using the user input object.

**[0024]** The user interface object and map creator object allow the user to interactively select options for transformation, and options for assigning attribute values for the target components, and the selected options are input to objects for properties and objects for attributes. These objects are processed by the map creator object to create a transformation rule object for the source component object.

**[0025]** The invention accepts and processes interactive user input, using the user input object, for making plural changes to any of the component mapping values the user desires until the user inputs a command to cease the interactive input and create a transformation map. The map creator object initiates processing of the transformation rules to create a transformation map object.

**[0026]** The user interface object accepts user input into a user input object for selecting an input source file for transformation to a target output file using an already existing map object specified interactively by the user. The user input is then processed, and the requested input file and map are then processed to transform the input file into the requested output file format. The created output file is then sent to the user specified destination.

**[0027]** Another object of this invention is to provide a novel method, apparatus, and computer program product which provides a graphical user interface for processing information encoded in a structured information format to transform the information into another structured information format, and which allows a user to interactively define the mapping for the transformation. Exemplary structured information formats include markup language formats, database information formats, an ISO/IEC 9070 naming scheme, a UNIX file name scheme, and a DOS file name scheme.

**[0028]** It is a further object of this invention to provide a novel method, apparatus, and computer program product which provides a graphical user interface for defining conversion of Standard Generalized Markup Language ("SGML") documents into HyperText Markup Language ("HTML") documents, which allows a user to interactively define the mapping for the transformation.

**[0029]** It is a further object of this invention to provide a novel method, apparatus, and computer program product which provides a graphical user interface for defining conversion of information in a database format into information in a different database format, which allows a user to interactively define the mapping for the transformation.

**[0030]** It is a further object of this invention to provide a novel method, apparatus, and computer program product which provides a graphical user interface for defining conversion of information from an ISO/IEC 9070 naming scheme into a UNIX file name scheme, which allows a user to interactively define the mapping for the transformation.

**[0031]** It is a further object of this invention to provide a novel method, apparatus, and computer program product

which provides a graphical user interface for defining conversion or information from an ISO/IEC 9070 naming scheme into a DOS file name scheme, which allows a user to interactively define the mapping for the transformation.

**[0032]** These and other objects are accomplished by a method, apparatus, and computer program product which provides a graphical user interface for processing information encoded in a structured information format, such as a markup language format, or such as a database information format, to transform the information into another structured information format, such as a markup language format, or such as another database information format, which allows a user to interactively define the mapping for the transformation.

**[0033]** An exemplary transformation for the present invention is conversion of SGML documents into HTML documents. For explanation of this example, the present invention has been developed as a tool to allow a user to define the transformation of an SGML document into an HTML document or other structured format, for example, a database information format. The user tool for this example is currently implemented in the format of a Graphical User Interface ("GUI") using Object Oriented Programming ("OOP") technology.

**[0034]** For this example, the current invention is designed to provide a user with a graphic tool to transform documents written in a cryptic SGML format into another structured format for greater viewing ease and far greater portability of documents and information. The user interface provides the user with selectable options of performing a default or conditional mapping. The user interface provides the user with selectable options of selecting an input SGML Document Type Definition ("DTD") or a currently existing map. The user interface displays the input for the user to select individual source components of the input. The user interface provides the user with selectable options for transformation of the individual source components such as a mapping of a source component to a target null value, a mapping of a source component to itself, a mapping of a source component to a single target component, or a mapping of a single source component to plural target components. If the user selects a conditional mapping, then special cases, such as a history of an element being referenced previously, are checked and processed using further interactive input from the user using the user interface.

**[0035]** The user interface also provides selectable options to the user for assigning attribute values for the target components. Exemplary options are attribute values obtained from the source components, system attribute values, no value, and attribute values input interactively by the user using the user interface.

**[0036]** The user interface allows the user to interactively select options for transformation, and options for assigning attribute values for the target components, and the selected options are processed to create a transformation rule for the source component.

**[0037]** The invention accepts interactive user input, to be processed by a map creator, for making plural changes to any of the component mapping values the user desires until the user inputs a command to cease the interactive input and create a transformation map. The transformation rules are processed by a map creator to create the transformation map.

**[0038]** The invention accepts user input for selecting an input source file for transformation to a target output file using an already existing map specified interactively by the user. The user input is then processed, and the requested input file and map are then processed to transform the input file into the requested output file format. The created output file is then sent to the user specified destination.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0039]** A more complete appreciation of the invention and many of the attendant advantages thereof will be readily obtained as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings, wherein:

Fig. 1A illustrates an exemplary Standard Generalized Markup Language ("SGML") Document Type Definition ("DTD");

Fig. 1B illustrates an exemplary mapping of SGML to HyperText Markup Language ("HTML");

Fig. 1C illustrates an exemplary SGML document;

Fig. 1D illustrates an exemplary HTML document output from a transformation of the SGML document;

Fig. 2 illustrates an exemplary browser output generated using the HTML document shown in Fig. 1D;

Fig. 3A illustrates, in tree format, the hierarchical nature of an SGML document and Fig. 3B illustrates the more "flat" structure of an HTML document;

Fig. 4 illustrates a design of the major components for the SGML to HTML mapping and transformation;

Fig. 5 illustrates, in a data flow diagram format, the flow of data through the SGML to HTML mapping and transformation;

Fig. 6A illustrates the flow of data and interaction of files through the mapping and transformation of information in one structured format to information in another structured format;

Fig. 6B illustrates the flow of data and interaction of files through the SGML to HTML mapping and transformation;

Fig. 7 illustrates a file organization of a map class object for the SGML to HTML mapping and transformation;
Fig. 8A illustrates a map class structure for the map module of the SGML to HTML mapping and transformation;
Fig. 8B illustrates the major classes within the map module of Fig. 8A;
Fig. 8C(1) illustrates a map class structure for a source SGML tag attribute class of the SGML to HTML mapping and transformation;
Fig. 8C(2) illustrates a map class structure for a source SGML content class of the SGML to HTML mapping and transformation;
Fig. 8C(3) illustrates a map class structure far a map service class of the SGML to HTML mapping and transformation;
Fig. 8C(4) illustrates a map class structure for a map create and edit service class of the SGML to HTML mapping and transformation;
Fig. 9 illustrates the hierarchical interaction among major modules of the SGML to HTML mapping and transformation;
Fig. 10 illustrates an exemplary main application window for the SGML to HTML mapping and transformation;
Fig. 11 illustrates exemplary dialog boxes for opening and saving a file;
Fig. 12A illustrates an exemplary window for the Map Processing Option of the SGML to HTML mapping and transformation;
Fig. 12B illustrates an exemplary window for the SGML to HTML Map Editor;
Fig. 12C illustrates an exemplary window for the SGML to HTML Map Editor with sample data displayed in exemplary dialog windows;
Fig. 13 illustrates a class diagram for the Menu Manager for the SGML to HTML mapping and transformation;
Fig. 14 illustrates an object message diagram for startup of the system of the SGML to HTML mapping and transformation;
Fig. 15 illustrates an object message diagram for opening an SGML document for the first time;
Fig. 16 illustrates an object message diagram for opening a new SGML document;
Fig. 17 illustrates the design of the GUI (Graphical User Interface) for the SGML to HTML mapping and transformation;
Figs. 18A(1)-18A(3) illustrate, in object message diagram format, the behavior among the objects of the classes for editing a map for the SGML to HTML mapping and transformation;
Figs. 18B(1)-18C(3) illustrate, i object message diagram format, the behavior of the objects of the classes for assigning values to HTML attributes;
Fig. 19 illustrates a hardware configuration for implementation of the SGML to HTML mapping and transformation;
Fig. 20A illustrates an exemplary public identifier in ISO/IEC 9070 format;
Fig. 20B illustrates an exemplary mapping of ISO/IEC 9070 to a UNIX file name format;
Fig. 20C illustrates an exemplary UNIX file name resulting from mapping the public identifier of Fig. 20A using the map of Fig. 20B;
Fig. 20D illustrates an exemplary user interface display for mapping a public identifier in ISO/IEC 9070 format to a UNIX file name format;
Fig. 20E illustrates an exemplary user interface display for mapping a registered owner field in ISO/IEC 9070 format to a UNIX file name format;
Fig. 20F illustrates an exemplary user interface for selections for a character mapping of a prefix, owner-name component separator in ISO/IEC 9070 format to the UNIX file name format;
Fig. 20G illustrates an exemplary user interface for mapping an owner name character in ISO/IEC 9070 format to valid characters of the UNIX file name format; and
Fig. 20H illustrates an exemplary user interface for a user to map a registered owner component in ISO/IEC 9070 format to a UNIX file name format.

## BRIEF DESCRIPTION OF THE APPENDICES

**[0040]**

Appendix A is an exemplary Standard Generalized Markup Language ("SGML") Document Type Definition ("DTD") corresponding to the tree structure or Fig. 3A;
Appendix B is an exemplary map of SGML elements from the SGML DTD of Appendix A to HTML elements to produce documents which correspond to the tree structure of Fig. 3B;
Appendix C is an exemplary SGML document which conforms to the SGML DTD of Appendix A;
Appendix D is an HTML document which is generated by using the map of Appendix B to transform the SGML document of Appendix C into HTML elements.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

**[0041]** Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views, and more particularly to Fig. 1A thereof, there is illustrated an exemplary Standard Generalized Markup Language ("SGML") Document Type Definition ("DTD"). Figs. 1A-1D are presented to illustrate sample inputs and outputs of the SGML to HTML mapping and transformation. The functions performed during the transformation which generate the outputs are described in detail below, and with respect to Figs. 4-6B and the object message diagrams of Figs. 18A(1)-18C(3).

**[0042]** Figs. 1A-1D and Appendices A-D show exemplary SGML DTDs, SGML documents, maps, and HTML documents produced from transforming the SGML documents using the maps. Figs. 3A-3B show exemplary tree structures for the SGML DTD of Appendix A and the HTML structure resulting from transforming the SGML DTD using the map of Appendix B. Figs. 1A-1D, Appendices A-D, and Figs. 3A-3B illustrate mapping SGML DTDs to HTML DTDs. The problem of DTD to DTD mapping is the default mapping from an SGML instance to an HTML instance based upon the DTDs. SGML tags are either mapped to zero or more HTML tags, and the sources of HTML attributes must be specified in the mapping. A more concise mathematical expression of the problem is given below.

**[0043]** Let SS be the space generated by the SGML DTD where SS $=\{S_i \mid i=0,...,m\}$, $S_i=\langle$Tag Name$_i$, Attribute Set$_i\rangle$, and Attribute Set$_i$=(Attribute$_j$ of Tag Name$_i$ $\mid$ j=0,...,$n_i$}+$\phi$. Similarly, let HH be the space generated by the HTML DTD specified by W3C (world Wide Web Consortium), that is, HH=$\{H_i \mid i=0,...,k\}$ where $H_i$ corresponds to $S_i$ above. Further, let HG be the space generated by HH consisting of the set of ordered members of HH. Then, HG={null, $\langle H_0\rangle$, $\langle H_1\rangle$,..., $\langle H_0, H_1\rangle$, $\langle H_0, H_2\rangle$,..., $\langle H_1, H_0\rangle$,...}. The sequence of legal HTML tags to be mapped are likely to be found in HG. Then the SGML tag to HTML tag mapping is equivalent to the function F : SS $\rightarrow$ HG + {not-assigned}.

**[0044]** For purposes of this discussion, {} denotes a set, $\langle ... \rangle$ denotes an ordered set, and + denotes union.

**[0045]** Let HGG be a set generated from SS, F(SS), and HH. HGG consists of the ordered set of triplets or null. A triplet consists of $S_i$, F($S_i$) and $\langle$HTMLTagName, an Attribute$\rangle$ where HTMLTagName belongs to one of $H_j$ in F($S_i$). Assume $H_0$ has $Attr_0$ and $Attr_1$, $H_6$ has $Attr_0$, and $S_0$ and $S_1$ are mapped to $\langle H_0\rangle$ and $\langle H_3, H_6\rangle$, respectively. Then HGG={null, $\langle S_0, \langle H_0\rangle, \langle H_0$Tag Name, $Attr_0\rangle\rangle$, $\langle S_0, \langle H_0\rangle, \langle H_0$Tag Name, $Attr_1\rangle\rangle$,...,$\langle S_1, \langle H_3, H_6\rangle, \langle H_6$ Tag Name, $Attr_0\rangle\rangle$,...}. Also, let SAtr be the source of the HTML Attribute value. Then Satr = AS+AC+{user inputs}+Null, where AS=the set of ordered pairs of tag name and one attribute name, and AC=the set of tag names with character data content. Then the identification of the attribute source is a function G mapping from HGG to Satr, denoted G: HGG$\rightarrow$SAtr.

**[0046]** A complete description of SGML is provided in Goldfarb, C. F., *The SGML Handbook*, Oxford University Press, Oxford, 1990, and McGrath, S., *Parseme.1st: SGML for Software Developers*, Prentice Hall PTR, New Jersey, 1998, which are incorporated herein by reference.

**[0047]** The exemplary SGML document of Fig. 1C, together with the exemplary SGML DTD of Fig. 1A, and the exemplary mapping of Fig. 1B are utilized in a transformation process to generate the HTML document of Fig. 1D. The SGML DTD of Fig. 1A and the SGML document of Fig. 1C are together parsed to produce the structural components of the SGML document of Fig. 1C. These components are then utilized in conjunction with the map of Fig. 1B to transform the SGML document of Fig. 1C into the HTML document of Fig. 1D. Further details of the parsing and transformation are explained below, and with respect to Figs. 4-6B and Figs. 18A(1)-18C(3).

**[0048]** In Fig. 1A, line 22 is a comment line containing the name of the SGML DTD file. Line 24 is a declaration of an element t containing a model group including elements t1, t2, t3, and t4. The '?' of line 24 indicates that an element is optional. The '*' of line 24 indicates that the model group may occur any number of times in a valid element t, and may also be absent. For this example, the SGML document of Fig. 1C illustrates a valid element t containing elements t1, t2, t3, and t4 in a group in lines 64-70.

**[0049]** Line 26 of Fig. 1A is a declaration of an element t1 having content type CDATA. The type CDATA means that the element may have a value that consists of general characters. For this example, the SGML document of Fig. 1C includes an element t1 on line 64 having as content the string of general characters '1. Hi Larry'. Usually, content of an element is delimited by a start tag for the element before the content, and an end tag for the element after the content. A start tag typically includes the character '<' followed by the name of the element, followed by optional element information such as attribute information, followed by '>'. An end tag then includes the characters '</' followed by the name of the element, followed by '>'. In SGML, the delimiters '<' and '>' can, by definition, be replaced by other characters.

**[0050]** Line 28 of Fig. 1A is a declaration for an attribute list for the element t1. An attribute is a property of an element that takes on different values for different instances of elements. For example, an element 'person' typically has an attribute list of attributes 'name', 'age', and 'haircolor'. A particular first person has name="Joe Smith", age ="27", and haircolor="brown", while a second person has name="Sally Jones", age="45", and haircolor="red". If the second person desires, her haircolor attribute is easily changed to haircolor="blond" by an assignment of a different value. For the example of Fig. 1A, on line 28, the attribute list includes an attribute 'name', of type CDATA. The character string '#REQUIRED' is an attribute value indicating that the attribute must be specified. For this example, in the SGML document of Fig. 1C the element t1 on line 64 has a general character content value of "hilarry" assigned to the name

attribute of this element t1.

**[0051]** Line 30 of Fig. 1A is a declaration for an element t2, of type CDATA. Line 32 is a declaration of an element t3, of type CDATA. Line 34 is a declaration of an element t4, of type CDATA.

**[0052]** In the SGML to HTML mapping of Fig. 1B, line 42 illustrates a mapping rule of the element t of line 24 to a string of HTML tags and text including '⟨html⟩⟨title⟩Title⟨/title⟩'. Line 44 illustrates a mapping rule of the element t1 of line 26 to a string of HTML tags '⟨H3⟩⟨A NAME="the source is t1's name"⟩'. The sentence between the double quotes of line 44 is a rule rather than an attribute value. Line 46 illustrates a mapping rule of the element t2 of line 30 to an HTML tag '⟨P⟩'. Line 48 illustrates a mapping rule of the element t3 of line 32 to an HTML tag '⟨P⟩'. Line 50 illustrates a mapping rule of the element t4 of line 34 to a string of HTML tags '⟨P⟩⟨A HREF="# the source is t1's name"⟩'. The sentence between the double quotes of line 50 is a rule rather than an attribute value.

**[0053]** Referring to the exemplary SGML document of Fig. 1C, line 60 shows the document type of the SGML document to be 't', with the DTD corresponding to the SGML document found in the system file "sample1.dtd". Line 62 contains the document start tag '⟨t⟩', which indicates that the lines following line 62 are assumed to follow the format defined in the DTD of Fig. 1A for the element t of line 24. Lines 64, 66, 68, and 70 are exemplary constituent parts of the element t shown on line 24 of Fig. 1A, defined for the exemplary SGML document of Fig. 1C. Line 72 contains the document end tag '⟨/t⟩', signifying the end of the document.

**[0054]** The HTML document of Fig. 1D is the output of the transformation process utilizing the SGML DTD file of Fig. 1A, the mapping of Fig. 1B, and the SGML document of Fig. 1C. Lines 82, 84, 86, 88, 90, 92, and 94 of Fig. 1D are generated from the information contained in each of Figs. 1A-1C.

**[0055]** The processing of the exemplary input files illustrated in Figs. 1A-1C to produce the output document illustrated in Fig. 1D will now be described. First, the SGML document line 60 of Fig. 1C is analyzed to determine the document type of the input SGML document and the name of the system file where the SGML documents DTD is stored. This causes the transformer to output the DOCTYPE HTML tag illustrated in line 80 of Fig. 1D, and to open the referenced system file for accessing the DTD for the current SGML document. A check is performed to determine that the DTD does correspond with the current SGML document. Next, the SGML tag of line 62 is parsed so that the current SGML tag becomes '⟨t⟩'. The map of Fig. 1B is referenced to determine that the current SGML tag is the start tag for the current SGML document, and maps to the HTML tag string '⟨html⟩⟨title⟩Title⟨/title⟩'. An HTML tag '⟨html⟩' is saved for the current SGML tag '⟨t⟩' and is output to line 82 of Fig. 1D. The HTML tag substring '⟨title⟩Title⟨/title⟩' is output to line 84 of Fig. 1D.

**[0056]** The first SGML tag in the string f line 64 of Fig. 1C is now obtained. The current SGML tag becomes '⟨t1⟩'. The transformer obtains the current attribute name and attribute value for the current SGML tag, obtaining an attribute called 'name' with a value "hilarry". The DTD of Fig. 1A is examined to determine that the SGML tag corresponds to the SGML element defined in line 26 of Fig. 1A, with its corresponding attribute list established in line 28 of Fig. 1A. The map of Fig. 1B is analyzed to determine that the current SGML tag's rule is line 44. The mapped HTML string '⟨H3⟩⟨A NAME=' followed by the current value of the name attribute for the SGML element t1, which is currently "hilarry", is then output to the HTML document on line 86 of Fig. 1D. An '⟩' is then output to terminate the tag. The HTML tags '⟨H3⟩' and '⟨A⟩' are saved for the current SGML tag. Next, the parser recognizes text that will be output to the HTML file on line 86 of Fig. 1D. The parser then recognizes the SGML end tag '⟨/t1⟩', at which point end tags for all the HTML tags currently saved for '⟨t1⟩' are output to the HTML document on line 86 of Fig. 1D in reverse order from which the tags were saved.

**[0057]** Next, the parser recognizes SGML tag '⟨t2⟩' from line 66 of Fig. 1C. The transformer utilizes the map rule line 46 of Fig. 1B to output HTML tag '⟨P⟩', shown on line 88 of Fig. 1D, and to save the HTML tag for the current SGML tag '⟨t2⟩'. The parser then recognizes text which is output to the HTML file, as shown on line 88 of Fig. 1D. The parser now recognizes SGML end tag '⟨/t2⟩' as terminating the text, at which point an end tag '⟨/P⟩' for the HTML tag currently saved for '⟨t2⟩' is output to the HTML document on line 88 of Fig. 1D.

**[0058]** The parser now recognizes SGML tag '⟨t3⟩' from line 68 of Fig. 1C. The transformer utilizes the map rule of line 48 of Fig. 1B to output HTML tag '⟨P⟩', shown on line 90 of Fig. 1D, and to save the HTML tag for the current SGML tag '⟨t3⟩'. The parser now recognizes text which is output to the HTML file, as shown on line 90 of Fig. 1D. Next, the parser recognizes SGML end tag '⟨/t3⟩' as terminating the text, at which point the end tag '⟨/P⟩' for the HTML tag currently saved for SGML tag '⟨t3⟩' is output to the HTML document on line 90 of Fig. 1D.

**[0059]** Next, the first SGML tag in the string of line 70 of Fig. 1C is obtained. The current SGML tag is now '⟨t4⟩'. The transformer obtains the current attribute name and attribute value for an SGML tag, obtaining an attribute called 'name' with a value "hilarry". The map of Fig. 1B is analyzed to determine that the current SGML tag's rule is line 50. The mapped HTML string '⟨P⟩⟨A HREF="#' followed by the current value of the name attribute, which is currently "hilarry", is then output to the HTML document on line 92 of Fig. 1D. An '"⟩' is then output to terminate the tag. The HTML tags '⟨P⟩' and '⟨A⟩' are saved for the current SGML tag '⟨t4⟩'. Next, the parser recognizes text that will be output to the HTML file on line 92 of Fig. 1D. The parser now recognizes the SGML end tag '⟨/t4⟩', terminating the text, at which point end tags for all the HTML tags currently saved for '⟨t4⟩' are output to the HTML document on line 92 of Fig. 1D in reverse order from which the tags were saved for '⟨t4⟩'.

**[0060]** Next, the parser recognizes the end of the SGML document by recognizing a '⟨/t⟩' tag of line 72 of Fig. 1C. This is interpreted to indicate an end tag for the SGML tag '⟨t⟩'. The HTML tags saved for the SGML tag '⟨t⟩' are then obtained and an end tag for the HTML tag '⟨html⟩', the only tag saved for '⟨t⟩', is output to the HTML document as shown on line 94 of Fig. 1D. This terminates the current processing of the documents.

**[0061]** Fig. 2 shows an output 100 resulting from opening the HTML output document of Fig. 1C as an exemplary What You See Is What You Get ("WYSIWYG") output of a Web browser on a user's computer screen. This output is in a format typically preferred by users of the World Wide Web on the Internet. Users employ Web browser programs to request HTML files, and other file types, from servers on the Internet. The browser downloads a requested HTML file and opens it to display formatted text and images on the user's computer screen. A browser does not have to download a file but is also capable of displaying an HTML file stored local to the computer running the browser or a local area network connected thereto.

**[0062]** Referring to Fig. 2, the 'Title' line is generated by a browser utilizing line 84 of Fig. 1D. Line 84 includes a start tag '⟨title⟩' and an end tag '⟨/title⟩' to delimit the text of the title to be displayed by the browser, usually in a title bar at the top of the user's computer screen. The '⟨H3⟩' of line 86 of Fig. 1D instructs a Web browser to output non-tag text in a larger, more bold format than normal text output. As the only text appearing after the HTML start tag '⟨H3⟩' is ' 1. Hi Larry', and this is the only text appearing before the HTML end tag '⟨/H3⟩', the text appears on a computer screen enlarged and bold in comparison with the surrounding text. The '⟨P⟩' start tag of line 88 instructs a Web browser to display non-tag text delimited by the start tag and its corresponding end tag in a new paragraph. New paragraphs start on a now line in the output. The end tag '⟨/P⟩' of line 88 instructs a Web browser that this is the end of the current paragraph, and that any non-tag text following this tag will start on a new line on screen.

**[0063]** On line 86 of Fig. 1D, the tag containing '⟨A name=' is an anchor tag. Anchor tags are used to set place markers in text, and to establish highlighted text that can be clicked on with a mouse to cause a jump to the text containing the place marker. For this tag, a place marker is established for the browser in the non-tag text following the next '⟩' until the '⟨/A⟩' end tag is encountered. Line 92 contains another type of anchor tag containing '⟨A HREF ='. The text appearing on line 92 between the anchor tag's '⟩' and its corresponding end tag '⟨/A⟩' appears on the screen of Fig. 2 as underlined text '(Back to the Hi Larry greeting.)'. This text is typically displayed in a different color from the surrounding text on the screen. When a user clicks a mouse on this underlined text, the text marked by the reference anchor tag of line 86 is pulled in for the user's viewing, surrounded by its neighboring text.

**[0064]** Fig. 3A and Fig. 3B illustrate the transformation of a hierarchical SGML document tree structure to the more "flat" tree structure of an HTML document. Fig. 3A illustrates a hierarchical SGML document tree structure, whereas Fig. 3B illustrates the corresponding "more flat" tree structure of the HTML document corresponding to the SGML document graphically displayed in Fig. 3A.

**[0065]** The trees of Fig 3A and Fig 3B are derived from documents illustrated in Appendices A-D. Appendix A shows an exemplary SGML DTD. The tree structure of Fig 3A is derived from the SGML DTD of Appendix A. The tree of Fig 3A has a root node test 110 which has children nodes front 112 and section 114. The node front 112 has children nodes title 116, author 118, and keywords 120. The node section 114 has children nodes number 122, title 124, para 126, and subsec 1 128. The node author 118 has children nodes fname 130, surname 132, and title 134. The node subsec 1 128 has children nodes number 136, title 138, para 140, and subsec 2 142. The node subsec 2 142 has children nodes number 144, title 146, and para 148. The tree structure of Fig. 3A which corresponds to the SGML DTD of Appendix A has five levels and twenty nodes.

**[0066]** The tree structure of Fig. 3B corresponds to a generalized HTML document that results from utilizing the SGML DTD of Appendix A and a mapping exemplified in Appendix B. Appendix C shows an exemplary SGML document to be processed through the mapping shown in Appendix B to give an HTML document exemplified in Appendix D. The tree structure of Fig. 3B has a root node html 150 having two children nodes, head 152 and body 154. The head node 152 has a child node title 156. The body node 154 has children h3 158 and p 160. The node p 160 has a child strong 162. In contrast to the tree structure of Fig 3A which has five levels and twenty nodes, the tree structure corresponding to the resulting HTML document has only four levels and seven nodes.

**[0067]** Fig. 4 illustrates an overview of major modules of the SGML to HTML mapping and transformation. A Map Module 184 interacts with a Parser 182 and a GUI 180 to create the actual mapping from an SGML document to an HTML document. A Transformer 186 interacts with the Map Module 184, the Parser 182, and the GUI 180 to transform the SGML document into the HTML document. A Service module 188 contains utility objects which can be utilized by all the modules for utility processing such as file handling. The GUI 180 handles interaction between a user and the system. The Parser 182 analyzes and breaks down input documents into recognizable component parts to be passed to other modules of the system. For example, in processing the exemplary SGML document of Fig. 1C, Parser 182 analyzes the input SGML document recognizing a DTD to generate a symbol table which can be passed to other modules of the system for processing documents. The Parser 182 recognizes line 60 of Fig. 1C as a 'DOCTYPE' tag and processes a DTD specified by a system file "sample.dtd" shown in Fig. 1A to generate the symbol table. The Parser 182 would then recognize line 62 of Fig. 1C contents as a start tag for the element t, and would transmit the tag and other

tag information to Transformer 186. Transformer 186 controls the processing of the SGML to HTML mapping and transformation, requesting information and data from the Map Module 184, the Parser 182, and the Service 188 modules when needed.

**[0068]** Fig. 5 illustrates a data flow diagram showing the flow of data through the SGML to HTML mapping and transformation. An SGML Application DTD 200 and HTML 3.2 DTD 204 are input to an SGML Parser 202. This SGML Parser 202 corresponds to the Parser 182 of Fig. 4. An SGML Application Symbol Table 222 and an HTML 3.2 Symbol Table 206 are output from the SGML Parser 202 to be utilized as input to a Map Editor 208, along with an interactive User 210 input. The Map Editor 208 is contained within the GUI 180 of Fig. 4. The Map Editor 208 outputs a Map 212. The SGML Application Symbol Table 222 and an SGML Application Document Instance 218 are together input to the SGML Parser 202 to give output to be used as input, along with the Map 212, to the Transformer 186. Transformer 186 corresponds to the Transformer 186 of Fig. 4. Transformer 186 then outputs an HTML Document Instance 216. The SGML Application DTD 200 and SGML Application Document Instance 218 are exemplified in Fig. 1A and Fig. 1C, respectively. The HTML Document Instance 216 is exemplified in Fig. 1D. The Map 212 is exemplified in Fig. 1B.

**[0069]** Fig. 6A is a more generalized data flow diagram showing exemplary paths taken by data flowing through the generalized mapping and transformation of information in one structured format to information in another structured format. A Structural Description of System A 230, together with a Structural Description of System B 232 and interactive User 210 input, are input to a Map Editor 208 to output the Map 212. The Map 212 and an Instance of System A 238 are then utilized by the Transformer 186 to output an Instance of System B 244.

**[0070]** Fig. 6B is a more generalized data flow diagram showing exemplary paths taken by data flowing through the SGML to HTML mapping and transformation. An SGML DTD 200, together with an SGML Document 218 and an HTML DTD 260, are input to the Mapping Editor 208 to output the Map 212. The Map 212 and the SGML Document 218 are then utilized by the Transformer 186 to output an HTML Document 216. The HTML Document 216 corresponds to the HTML Document Instance 216 of Fig. 5. The HTML Document 216 is then input to a Browser 262 for user viewing.

**[0071]** The SGML DTD 200, input to an SGML Editor 256, yields output to the SGML Document 218. A Database Design 250, input to the Mapping Editor 208, along with the SGML DTD 200 and the SGML Document 218, yield output to the Map 212 and a Data Base 254. The Map 212 and the SGML Document 218 are input to the Transformer 186 to yield output, which, together with the output from the Mapping Editor 208, are input to the Data Base 254. The Map 212 corresponds to the Map 212 of Fig. 5. The SGML Document 218 corresponds to the SGML Application Document Instance 218 of Fig. 5. The SGML DTD 200 corresponds to the SGML Application DTD 200 of Fig. 5. The Transformer 186 corresponds to the Transformer 186 of Fig. 5. The Mapping Editor 208 corresponds to the Map Editor 208 of Fig. 5. Arrows illustrate different paths the documents and data files take for different requests of a user.

**[0072]** Fig. 7 shows a hierarchical view of the DTD Map class object that can be stored in a file. The invention has been implemented using object oriented techniques, although any programming technique and/or hardware may be used to implement the invention. For purposes of this description, a class is a description of the structure and behavior of an object, while an object is an instance of the item described by a class. Objects typically communicate by passing objects and messages to each other. In structure, objects contain other objects or structures as components, as well as variables and methods.

**[0073]** Interpreting the horizontal lines of Fig. 7 from left to right, begin and end delimiters delimit each object in the file. List Begin and List End delimiters delimit lists from left to right. When a DTD Map Object exists, one or more SGML tag objects are placed between the SGML Tag List Begin 363 and SGML Tag List End 365.

**[0074]** The file begins with a Header 360 followed by a DTD Map 361. The DTD Map 361 includes, first, a DTD Map Begin 362 followed by an SGML Tag List Begin 363, followed by at least one SGML Tag 364-1 through an SGML Tag 364-n. The sequence of one or more SGML tags is followed by an SGML Tag List End 365, followed by a DTD Map End 366. Each SGML Tag 364-1 through 364-n includes an SGML Tag Begin 367, an SGML Tag Name 368, followed by an SGML Tag Empty State 369, followed by an SGML Tag Assignment Type 370, followed by an HTML Tag List 371, followed by an SGML Tag End 372.

**[0075]** Each HTML Tag List 371 is delimited by an HTML Tag List Begin 373 at the beginning and an HTML Tag List End 375 at the end with the list including at least one HTML Tag 374-1 through HTML Tag 374-m following the HTML Tag List Begin 373.

**[0076]** Each HTML Tag 374-1 through 374-m is delimited by an HTML Tag Begin 376 at the beginning and an HTML Tag End 380 at the end. Following the HTML Tag Begin 376 is an HTML Tag Name 377, followed by an HTML Tag Empty State 378, followed by an HTML Attribute List 379, followed by a delimiter HTML Tag End 380.

**[0077]** Each HTML Attribute List 379 is delimited by an HTML Attribute List Begin 381 at the beginning and an HTML Attribute List End 383 at the end. Following the delimiter HTML Attribute List Begin 381 is at least one HTML Attribute 382-1 through an HTML Attribute 382-P, followed by an ending delimiter HTML Attribute List End 383.

**[0078]** Each HTML Attribute 382-1 through 382-p is delimited by an HTML Attribute Begin 384 at the beginning and an HTML Attribute End 389 at the end. Following the HTML Attribute Begin 384 is an HTML Attribute Name 385, followed by an HTML Attribute Source Type 386, followed by an HTML Attribute Source 1 387, followed by an HTML

Attribute Source 2 388, if one exists. The delimiter HTML Attribute End 389 terminates the listing of contents.

**[0079]** Fig. 8A shows major class dependencies 424 for the DTD Map object. For purposes of explanation of the figures that follow, arrows show class dependencies, meaning that an object having an arrow pointing to it is contained within the object originating the arrow. A Map object 400 includes a pointer to an object DTDMap 402. A pointer is a value that represents an absolute address of an item in computer memory. A pointer to an object is used to access the information stored for the implementation of a particular object by, minimally, referencing the pointer name and the field or function name within the object. Viewing Fig. 7 and Fig. 8A together, the DTDMap 402 of Fig. 8A, corresponding to the DTD Map 361 of Fig. 7, includes, via pointers, an SGMLTagList 404 corresponding to the SGML Tag 364-1 through 364-n of Fig. 7. The SGMLTagList 404 of Fig. 8A includes, via pointers, a class SGMLTag 406 corresponding to each of the SGML Tags 364-1 through 364-n of Fig. 7. The SGMLTag class 406 of Fig. 8A includes, via pointers, HTMLTagList 408, which corresponds to the HTML Tag List 371 of Fig. 7. The HTMLTagList 408 of Fig. 8A includes, via pointers, an HTMLTag 410 which corresponds to each of the HTML Tag 374-1 through HTML Tag 374-m of Fig. 7. The HTMLTag 410 of Fig. 8A includes, via pointers, an HTMLAttrList 412 class which corresponds to the HTML Attribute List 379 of Fig. 7. The HTMLAttrList 412 of Fig. 8A includes, via pointers, an HTMLAttr 414 class which corresponds to the HTML Attributes 382-1 through 382-p of Fig. 7. The HTMLAttr 414 class of Fig. 8A includes, via pointers, a class derived from an abstract class, denoted by an 'A' inside a triangle, HTMLAttrSource 416 which corresponds to the HTML Attribute Source Type 386 of Fig. 7. An abstract class is typically defined as a model to be used for defining other closely related classes which may, for example, need to exhibit similar behavior in a system. By defining an abstract class, the other classes are defined as inheriting the structure and methods of the parent abstract class. The hollow arrows of Fig. 8A denote inheritance of classes. HTML attributes may be obtained from different sources. Therefore, a UserInput 418 class is shown to inherit from the abstract class HTMLAttrSource 416. Also, an SGMLContent 422 class inherits from HTMLAttrSource 416, as does an SGMLTagAttr 420 class.

**[0080]** Fig. 8B shows major classes within the Map Module 276, illustrating the major dependencies among the classes. The classes 424 illustrated inside the dashed line rectangle are the classes 424 of Fig. 8A. A class MapService 452 includes a class Transformer 450, a class DTDMapTransformerService 456, a class SrcSGMLTagAttr 462, a class SrcSGMLContent 464, a class Map 400, a class MapEdit 460, and a class MapCreateEditService 454. The class MapEdit 460 includes the class DTDMapEdit 466. The class DTDMapEdit 466 has dependencies with the class DTDMap 402, the class SGMLTagList 404, SGMLTag 405, HTMLTagList 408, HTMLAttrList 412, HTMLTag 410, HTMLAttr 414, SGMLTagAttr 420, SGMLContent 422, and UserInput 418. The class HTMLAttrSource 416 has dependencies with the class SrcSGMLTagAttr 462 and the class SrcSGMLContent 464. The class DTDMapTransformerService 456 has dependencies with the class SrcSGMLTagAttr 462 and the class SrcSGMLContent 464. The class SrcSGMLTagAttr 462 has a dependency with the class SGMLTagAttr 420 and the class SrcSGMLContent 464 has a dependency with SGMLContent 422. The class MapCreateEditService 454 has dependencies with the class DTDMapEdit 466 and the class MapEdit 460. The class Map 400 contains DTDMap 402, DTDMapTransformerService 456, SrcSGMLTagAttr 462, and the class SrcSGMLContent 464. The functionalities of these classes and their objects are explained with regard to Figs. 18A(1)-18C(3).

**[0081]** Data items and objects in software generally involve dynamic allocation of computer storage resources at some stage in a request for execution of program code. Pointer variables, containing addresses of data items, methods, or objects, are available to be passed among objects during execution of code. As objects and data items are constructed and destructed dynamically, an object using a pointer or reference to a data item, for example, may reference the item after it has been destructed, possibly causing a system failure. A facility for registering objects and data items as they are created and requested gives objects a means to verify the current existence and usage of objects and data items before reference. A destructor verifies the current usage of an object or data item before destruction so that other objects using the object or data item may successfully complete their usage before destruction. An exemplary use of registering objects and data items is assignment of attribute values to HTML attributes, as discussed below with regard to Figs. 8C(1)-8C(4) and Figs. 18A(1)-18C(3).

**[0082]** Fig. 8C(1) illustrates a class structure for a SrcSGMLTagAttr 462 class of the SGML to HTML mapping and transformation. Fig. 8C(2) illustrates a class structure for a SrcSGMLContent 464 class of the SGML to HTML mapping and transformation. Fig. 8C(3) illustrates a class structure for the MapService 452 class of the SGML to HTML mapping and transformation. As described previously with regard to Fig. 8B, the class HTMLAttrSource 416 includes references to the class SrcSGMLTagAttr 462 and the class SrcSGMLContent 464. The HTMLAttrSource 416 contains an AtrSrcSGMLTagAttr, which is a reference to a SrcSGMLTagAttr 462, and an AtrSrcSGMLContent, which is a reference to a SrcSGMLContent 464. SrcSGMLTagAttr 462 contains a method registerSGMLTagNameAndAttributeName, used for registering SGML tag attributes in the MapService 452, so that attributes that have already been registered are available to be unregistered from the HTMLAttrSource 416 by using a method unregisterTagAttrKeyAndMapEntry in SrcSGMLTagAttr 462 through a virtual function unregisterFromSourceMap. SrcSGMLTagAttr 462 also contains a method setValueForAttributeOfTag to be used at document instance processing time to transform an SGML document to an SGML document.

**[0083]** SrcSGMLContent 464 contains a method registerSGMLTagName, used for registering SGML tag content in the SrcSGMLContent 464, so that attributes that have already been registered are available to be unregistered from the HTMLAttrSource 416 by using a method unregisterSGMLTagName in SrcSGMLContent 464. SrcSGMLContent 464 also contains a method setValueForTag to be used at document instance processing time to transform an SGML document to an HTML document.

**[0084]** Fig. 8C(4) illustrates a class structure for the MapCreateEditService 454 class of the SGML to HTML mapping and transformation. MapCreateEditService 454 was discussed previously with regard to Fig. 8B, and functionalities of the class and object are explained with regard to Figs. 18A(1)-18C(3). A method setSelectedSGMLTagToBeNullAssigned in MapCreateEditService 454 sets a selected SGML tag to be mapped to a null value. A method setSelectedSGMLTagToBeNotAssigned in MapCreateEditService 454 sets a selected SGML tag to be kept in the mapping. A method getAttributeAssignmentInformationForHTMLAttribute in MapCreateEditService 454 gets assignment information for assigned values to HTML attributes. Methods assignHTMLAttributeWithSGMLAttribute, assignHTMLAttributeWithSGMLContent, assignHTMLAttributeWithSystem, assignHTMLAttributeWithNoValue, assignHTMLAttributeWithUserInput in MapCreateEditService 454 assign values to the HTML attributes.

**[0085]** Fig. 9 illustrates a hierarchical view of major modules for implementation of the GUI for the SGML to HTML mapping and transformation. An Application Window 470 initiates execution of a Menu Manager 472. The Menu Manager 472 initiates execution of a File Service 474, Editor for Map 476, View 478, Map 480 and Message Dialog Service 482. The Map 480 module corresponds to the Map class 400 of Fig. 8A. The View 478 module is included in the GUI 270 of Fig. 4. The functionalities of these modules are explained with regard to Figs. 18A(1)-18C(3).

**[0086]** Fig. 10 shows an exemplary computer screen output of a Main Application Window 510. The window includes a title bar 512, a menu bar 514, a tool bar 516, and a viewing window work space 518. In order for a user to perform any menu operation, the user must select one of the items in the menu bar 514. When the user makes a selection, a sub-menu (or pull-down menu) displays all operations available for selection from the main menu. For example, if the user selected the File option, a sub-menu appears to display the options Open SGML, Open DTD, Open Map, Save Map, Map Save As, Save HTML, HTML Save As, Close SGML, and Exit. The Open SGML option allows the user to select an SGML document to open. The Open DTD option allows the user to select a DTD to open. The Open Map option allows the user to select a map to open. The Save Map Option allows the user to save a map onto disk. The Map Save As option allows the user to save the map onto the disk with the option of selecting a new file name for the saved map. The Save HTML option allows the user to save an HTML document onto disk. The HTML Save As option allows the user to save an HTML document onto disk with the option of selecting a new file name for the HTML file. The Close SGML option allows the user to close an SGML document and Exit option allows the user to exit the Application Window processing of the conversion.

**[0087]** If the user selects an Edit option from the menu bar 514, a sub-menu appears to display options Create Map and Edit Map. The Create Map option allows the user to create a map that can be used, to transform an SGML document to an HTML document. The Edit Map option allows the user to modify an existing map. A sub-menu for a View option displays the options to View SGML and to View HTML. These options are designed to display an SGML document or an HTML document when selected by a user. A View SGML option allows the user to display an SGML document in the work space of the main application window. A View HTML option allows the user to display an HTML document in the work space of the main application window. If the user selects the Map option, a sub-menu appears to display an option Run Map. Selecting Run Map initiates the transformation to transform an input SGML document to an HTML document. If the user selects the Option button, a sub-menu appears to display options of Incremental and Reference. An Incremental option allows the user to perform an incremental mapping of an SGML document to an HTML document. After an SGML tag is mapped into HTML tags the SGML document is then transformed into its corresponding HTML document. This occurs after each SGML tag is mapped. A Reference option allows the user to display the reference in transforming an SGML document to an HTML document.

**[0088]** Referring to Fig. 11, dialog boxes for opening an SGML file and saving an HTML file are shown. Fig. 11 shows an exemplary file open dialog box 600 which would be displayed responding to an Open SGML option from the sub-menu displayed after the user selected the File option in Fig. 10. A Filter text edit box 602 is displayed allowing the user to choose a type of file to be opened. Candidate directories for files are displayed in a Directories list box 604. Candidate files for opening will be displayed in a Files list box 606. A Selection text edit box 608 displays the file name that the user selects for opening. An OK button 610 allows the user to approve the selection shown in the Selection text edit box 608. A Filter button 612 allows the user to request a display of all files of a given type, in the Files list box 606, as described in the Filter text edit box 602. A Cancel button 614 allows the user to choose termination and cancellation of the current request to open a file.

**[0089]** Fig. 11 also shows an exemplary Save HTML file dialog box 616 that is displayed as a result of the user selecting the Save HTML option from the sub-menu displayed after the user selected the File button in Fig. 10. A Filter text edit box 618 allows the user to select a type of file for saving the current HTML file. Candidate directories for files are displayed in a Directories list box 620. Candidate files of a given type are displayed in a Files list box 622. A Selection

text edit box 624 allows the user to select a file name for saving the file. An OK button 626 allows the user to approve an operation and complete the operation of saving a file. A Filter button 628 allows the user to request a display of all files of a given type, in the Files list box 622, as described in the Filter text edit box 618. A Cancel button 630 allows the user to terminate and cancel the current request to save the current HTML file.

**[0090]** Fig. 12A and Fig. 12B display exemplary Map Edit Option and Map Edit Dialog boxes utilized for creation and editing of a map. The user is allowed to create a new map or edit an already existing map. If the user selects the Edit button from the Main Application Window 510 of Fig. 10, and either the Edit Map or Create Map option is selected, the Map Edit Option 690 dialog box of Fig. 12A is displayed to allow the user to select whether the Default mapping of the SGML document or a Conditional mapping of the SGML document should be used to create or edit the map. The Default mapping, selected by clicking on a Default button 692, is the user defined tag mapping set up by the user interaction with the SGML to HTML Map Edit dialog box 700 of Fig. 12B. The Conditional mapping, selected by clicking on a Conditional button 694, involves defining the conditional or special mappings. After the user selects one of the options from the Map Edit Option 690, then the Map Edit dialog box 700 of Fig. 12B is displayed to allow the user to interact with the system in defining a map. If the Create Map option is selected, the user is allowed to create a new map. Both the Map Edit Option dialog box 690 and the Map Edit dialog box 700 are used for creating a map and for editing an existing map.

**[0091]** Referring to Fig. 12B, a display of the Map Edit dialog box 700 shows a display of a list of SGML Tags 702, the current HTML Tag list 704 that an SGML tag selected from the SGML Tag list 702 maps to, and a list of Legal HTML Tags 706 that can be added into the current HTML Tag list 704. For a given SGML Tag 702, the user selects the Legal HTML Tag 706 by double clicking a mouse on an HTML tag from the Legal HTML Tag list 706. This will add the HTML Tag to the Current HTML Tag list 704. If the Current HTML Tag list 704 contains a list of HTML Tags that the SGML Tag 702 maps into, a new HTML Tag will be added below the HTML Tag that is selected in the Current HTML Tag 704 list. If an HTML Tag is inserted into the current HTML Tag 704 list, the HTML Tag(s) following the inserted Tag must be deleted, as they may no longer be legal. A Clear HTML Tag 708 button will clear the current HTML Tag 704 list. A Delete HTML Tag 710 button will delete the HTML Tag selected in the Current HTML Tag 704 list. The HTML Tags following the deleted HTML Tag in the Current HTML Tag 704 list must be deleted since they may no longer be legal. An Undo 712 button will undo the last clear, delete or insert operation. These buttons are easily modified to a menu operation format by one skilled in the art of computing. A Map SGML Tag 714 button allows the user to map the SGML Tag 702 to the HTML Tag list in the Current HTML Tag 704 list and then allows the user to select the next SGML tag to map. If a Done 716 button is selected, the remaining SGML Tags 702 will not be mapped. If a Cancel 718 button is selected, all previous SGML to HTML map information will be disregarded. Two possible selections in the Legal HTML Tag list 706 are Null Assigned and Not Assigned. Null Assigned deletes the SGML Tag 702 so that the SGML tag 702 will not be mapped and will not be displayed after transformation. Not Assigned leaves the SGML Tag 702 as is, so that the SGML Tag 702 will not be mapped to HTML but will be displayed as is after transformation.

**[0092]** An explanation of tag attribute assignment is provided with regard to Figs. 18A(1)-18C(3).

**[0093]** Fig. 12C shows exemplary data in the tag list boxes of the Map Edit dialog box 700 previously discussed for Fig. 12B. An explanation of the processing of the data is provided with regard to Figs. 18A(1)-18C(3).

**[0094]** Fig. 13 illustrates an exemplary class diagram displaying relationships among the classes of the SGML to HTML mapping and transformation for the GUI. An Application Window 772 manages the handling of the display of the application window of the GUI. A Menu Manager 778 handles all the tasks and objects associated with menu operations. A File Service 782 handles open and save operations associated with files. An ntEntity 790 is a general system representation of an SGML document, an SGML DTD, an HTML document, or an HTML DTD. A Symbol Table 770 is the system representation of an input document after it has been processed by a Parser Service 774. A MessageDialogService 776 handles the output of messages to the system used. A View Document 786 class handles the display of SGML or HTML documents upon user request. A Map Service 780 handles the creation and editing, through a MapCreateEditService 792, of a Map 788, which is the system representation of the rules to be utilized in the transformation of an SGML document to an HTML document. A GUIEditorForMap 784 handles the GUI interface for the user to dynamically create or edit the Map 788.

**[0095]** Fig. 14 shows an object message diagram displaying the behavior of the system among objects of the classes for the startup of the system. The object diagram illustrates major software objects enclosed in cloud shapes. Object method calls are illustrated with an Arabic numeral preceding a colon, followed by the name of the object method. The numeric ordering illustrated by the Arabic numerals followed by colons illustrate a stepwise logical flow of execution, and a flow of object data and messages, through the diagram. For a more detailed description of object diagrams and design, see Booch, G., *Object-Oriented Analysis and Design with Applications, Second Edition*, Addison Wesley Longman, California, 1994, which is incorporated herein by reference.

**[0096]** An Application Window 772 is the object which generates the main application window of Fig. 10. It is the first object created when the system starts execution. It contains all the necessary information and functions to display the main application window of Fig. 10. An HTMLSymbolTable 800 is an object, which is the system representation of the

HTML DTD, created by the Application Window 772, in a call Create (new) 802 through ParserService 774. The HTML Symbol Table 800 exists throughout the lifetime of the system. A MenuManager 778 is an object created by the Application Window 772, in a call Create (new) 804, to handle all menu operations associated with the menu of the main Application Window 772. The Application Window 772 passes the HTMLSymbolTable 800 object it created to the MenuManager 778 so that the MenuManager 778 can pass it to any other objects which require it. The MenuManager 778 creates and manages all objects necessary carry to out menu operations.

**[0097]** A MessageDialogService 776 is an object created by the MenuManager 778, in a call Create (new) 806, to allow message dialog boxes to display any kind of message to the user from anywhere in the system. Exemplary messages are error messages, warnings or instructions to the user, or any other type of message required. The MenuManager 778 passes the MessageDialogService 776 to other objects which may need to display messages to the user.

**[0098]** A MapService 780 is an object created by the MenuManager 778, in a call Create (new) 808, to handle map related objects. The MenuManager 778 initiates a call MapServiceInit 810 to initialize the state of the newly created MapService 780. For this example, a map is an object which describes how the SGML document will be transformed into an HTML document. The MenuManager 778 passes the HTMLSymbolTable 800 and the MessageDialogService 776 to the MapService 780 so that it has information about the HTML DTD and can send messages to the user.

**[0099]** A MapCreateEditService 792 is an object created by the MapService 780, in a call Create (new) 812, to handle the creation of a map or the modification of an existing map. The MapService 780 passes the HTMLSymbolTable 800 to the MapCreateEditService 792 so that it has information about an HTML DTD. The MenuManager 778 receives the MapCreateEditService object 792 from Map Service 780, in a call getMapCreateEditServiceObject 814, so that it may create or edit a map at any time.

**[0100]** A FileService 782 is an object created by the MenuManager 778, in a call Create (new) 816, to handle the tasks of opening and saving a file. The file corresponds to an SGML document, an SGML DTD, a map, or an HTML document. The user requests actions for files by selecting the File button exemplified in the menu bar 514 in Fig. 10. The File Service 782 creates an Open or Save dialog box to allow the user to choose the file the user wants to open or save, as exemplified in Fig. 11. MenuManager 778 passes MessageDialogService 776 to File Service 782 so that it may display messages to the user. The MenuManager 778 also passes the MapCreateEditService 792 to the File Service 782.

**[0101]** A GUIEditorForMap 784 is an object created by the MenuManager 778, in a call Create (new) 818, to handle the task of allowing the user to create a map or modify an existing map through a dialog box, as exemplified in Figs. 12B-12C. The MenuManager 778 passes the MessageDialogService 776 to the GUIEditorForMap 784 for displaying messages to a user. The MenuManager 778 passes the MapCreateEditService 792 to GUIEditorForMap 784 to create or modify a map.

**[0102]** A View Document 786 is an object created by the MenuManager 778, in a call Create (new) 820, to handle the task of displaying an SGML or HTML document through a display window. The user requests document viewing by selecting the View button from the menu bar 514 exemplified in Fig. 10. The Menu Manager 778 passes the MessageDialogService 776 to the ViewDocument 786 so that it may display messages to the user.

**[0103]** Fig. 15 shows an object message diagram to display the dynamic relationships that exist among the objects of the invention when an SGML document is being opened for the first time. A User 830 requests, from an Application Window 772, opening an SGML document file using a call OpenSGML 840. This is accomplished by the user's selection of the File button in the menu bar 514 of Fig. 10, followed by selection of the Open SGML option in the resulting sub-menu. The Application Window 772 sends a call OpenSGML 842 to a MenuManager 778 to process the user request. The MenuManager 778 then sends a call OpenSGML 844 to a FileService 782. The FileService 782 initiates a call getFileName 846 to request a file name for the SGML document from the User 830. A User 830 response, a FileName 846, is returned to the FileService 782. The FileService 782 sends a request isFound 848, accompanied by a FileName 848, to an IOService 832 to determine, by a response YES 848, that the FileName 846 returned by the User 830 exists. The FileService 782 then sends a request isReadable 850, accompanied by a FileName 850, to the IOService 832 to determine, by a response YES 850, that the file is also readable by the system. The FileService 782 then sends a request getEntityObject 852 to IOService 832, accompanied by the FileName 852, so that IOService 832 may obtain and return an ntEntity 852, which is associated with an external entity such as an SGML document, and its corresponding DTD, requested by the User 830.

**[0104]** The MenuManager 778 sends a request getSGMLNtEntity 854 to the FileService 782 to receive the ntEntity 854 from the FileService 782. The MenuManager 778 then sends the ntEntity 856 to a Parser Service 774 with a call getSymbolTable 856. The Parser Service 774 then generates a SymbolTable 856 for the SGML document, checks that it is a valid symbol table for the SGML document, and returns the SymbolTable 856 to the MenuManager 778. The MenuManager 778 then sends the ntEntity 858 to a MapCreateEditService 792 using a call useThisSGMLDoc 858, and sends the SymbolTable 860 to the MapCreateEditService 792 using a call UseThisSymbolTable 860. The MapCreateEditService 792 then handles further processing of the requested document.

**[0105]** Fig. 16 shows an object message diagram to display the dynamic relationships that exist among the objects

of the invention as an SGML document as being opened, with an existing SGML document already opened. A User 830 requests, from an Application Window 772, opening an SGML document file by a call OpenSGML 890. This is accomplished by the user's selection of the File button in the menu bar 514 of Fig. 10, followed by selection of the Open SGML option in the resulting sub-menu. The Application Window 772 sends a call OpenSGML 892 to a MenuManager 778 to process the user request. The MenuManager 778 then sends a call closeHTMLWindow 894 to a ViewDocument 786 so that if there is an open window displaying an HTML document, it will be closed. If there is an HTML document that has not been saved, the MenuManager 778 sends a request SaveHTML 896 to a MessageDialogService 776. The MessageDialogService 776 then sends a request SaveHTML 898 to the User 830 as a message asking if the user wishes to save the currently displayed HTML file. A User 830 response of NO 898 is returned to the MessageDialogService 776, which then returns a NO 896 to the MenuManager 778. The MenuManager 778 then sends a call Destroy(delete) 900 to an HTMLntEntity 880, representing the HTML file. The MenuManager 778 then sends a call closeSGMLWindow 902 to the ViewDocument 786 for the ViewDocument 786 to process closing of the display window of an opened SGML document. The MenuManager 778 then sends a call Destroy(delete) 904 to an SGML ntEntity 882, representing an opened SGML file. The MenuManager then sends a call OpenSGML 906 to a FileService 782. The FileService 782 initiates a call getFileName 908 to request a file name for the SGML document from the User 830. The User 830 response, a FileName 908, is returned to the FileService 782. The FileService 782 sends a request isFound 910, accompanied by a FileName 910, to an IOService 832 to determine, via a response YES 910, that the FileName 908 returned by the User 830 exists. The FileService 782 then sends a request isReadable 912, accompanied by a FileName 912, to the IOService 832 to determine, by a response YES 912, that the file is also readable by the system. The FileService 782 then sends a request getEntityObject 914, accompanied by a FileName 914, to the IOService 832, so that the IOService 832 may obtain and return an ntEntity 914. The ntEntity 906 is then returned to the MenuManager 778 in response to the request OpenSGML 906.

**[0106]** The MenuManager 778 then sends an ntEntity 916 to a Parser Service 774 with a call getSymbolTable 916. The Parser Service 774 then generates a SymbolTable 916 for the SGML document and DTD and returns the SymbolTable 916 to the MenuManager 778. The MenuManager 778 then sends an ntEntity 918 and a Map 918 to a MapService 780 using a call areThisMapAndTheSGMLDocConsistent 918, to determine if the existing Map 918 can be used with the new SGML document, ntEntity 918. A response NO 918 is returned to the MenuManager 778 if the Map 918 cannot be used. The MenuManager 778 then sends a request isMapSaved 920 to the MapService 780, to receive a response of YES 920, if the Map 918 is saved. The MenuManager 778 then sends a call destroy(erase & delete) 922 to an SGMLSymbolTable 884, to destroy the system's current SGML file represented in the format of a symbol table. The MenuManager 778 then sends a call Destroy (delete) 924 to a Map 788 to destroy the system's current map for a previous SGML document. The MenuManager 778 then sends a call resetMap 926 to the MapService 780 to initialize a new map for the new SGML document to be processed. The MenuManager 778 then sends a call useThisSGMLDoc 928 to a MapCreateEditService 792, sending an ntEntity 928 obtained from IOService 832. The MenuManager 778 then sends a call UseThisSymbolTable 930, along with a SymbolTable 930 obtained from ParserService 774, to the MapCreateEditService 792. The MapCreateEditService 792 then handles further processing of the requested SGML document.

**[0107]** Fig. 17 illustrates, in a class diagram format, the design of the Map Editor GUI. The Application Window 772, the MenuManager 778, the GUIEditorForMap 784, the SymbolTable 770, and the MapCreateEditService 792 have been described previously with regard to Figs. 13-15. In Fig. 17, a MapEditDialog 950 is contained by the GUIEditorForMap 784. The MapEditDialog 950 manages the handling of the display of the Map Edit dialog box. This includes determining what type of operation or requests the user can perform at a given point in the mapping of an SGML element. A MapTag 952 interacts with MapCreateEditService 792 and SymbolTable 770 to handle tasks and objects associated with mapping an SGML element to an HTML element. An AssignAttribute 954 interacts with MapCreateEditService 792 and SymbolTable 770 to handle tasks and objects associated with assigning values to the attributes of HTML elements.

**[0108]** Figs. 18A(1)-18C(3) are object message diagrams showing the behavior of the SGML to HTML mapping for assigning values to HTML attributes and for mapping SGML tags to HTML tags. As the object diagram of Figs. 18A(1)-18C(3) was large, it was divided over a plurality of drawing sheets. However, these drawing sheets, when taken together, constitute one object diagram. The MapEditDialog 950, the MapTag 952, the AssignAttribute 954, and the MapCreateEditService 792 have been described previously with regard to Fig. 17. The GUIEditorForMap 784 has been described previously with regard to Fig. 13. The HTMLSymbolTable 800 has been described previously with regard to Fig. 14. The SGMLSymbolTable 884 has been described previously with regard to Fig. 16.

**[0109]** As an exemplary manner of operating, the GUIEditorForMap 784 of Fig. 18A(1) creates the Map Edit dialog box of Fig. 12B by a function call EditMap 960 to call a constructor of the MapEditDialog 950.

**[0110]** To fill the SGML Tag list box 702 of Fig. 12B, MapEditDialog 950 gets a first SGMLTag 962, with a YES message 962, from MapTag 952 through a function getNextSGMLTag 962, and displays the SGML tag in the SGML Tag list box 702 of Fig. 12B. MapTag 952 gets a first SGMLTag 964 of Fig. 18A(2), with a YES message 964, from the SGML-

SymbolTable 884 through a function call getFirstElement 964. An alternative GUI design is to keep a list of all SGML tags already processed in the display.

**[0111]** The user selects an SGML tag to map by double clicking the mouse on the SGML tag in the SGML Tag list box 702 of Fig. 12B and the SGMLTag 966 of Fig. 18A(1) is sent to MapTag 952 through a function call selectedSGMLTag 966 of Fig. 18A(1). MapTag 952 of Fig. 18A(2) sends the SGMLTag 967 to MapCreateEditService 792 of Fig. 18A(2) through a function call selectedSGMLTag 967 of Fig. 18A(2).

**[0112]** To fill the Current HTML Tag list box 704 of Fig. 12B, the MapEditDialog 950 box of Fig. 18A(1) then gets an HTMLTag 968, with a YES message 968, to which the selected SGML tag maps, from MapTag 952 through a function call getNextCurrentHTMLTag 968. The function will get one HTMLTag 968 at a time and display it in the Current HTML Tag list box 704 of Fig. 12B. MapTag 952 of Fig. 18A(2) gets a next HTMLTag 970, along with a YES message 970, from the MapCreateEditService 792 using a function call getExistingHTMLTagsinMap 970. If an SGML tag is being assigned for the first time, NOT ASSIGNED will be displayed in the Current HTML Tag list box 704 of Fig. 12B.

**[0113]** To fill the Legal HTML Tag list box 706 of Fig. 12B, the MapEditDialog 950 box of Fig. 18A(1) sends a last HTML tag 972 in the Current HTML Tag list box 704 of Fig. 12B to MapTag 952 through a function call setLastCurrentHTMLTagInList 972. Then the MapEditDialog 950 box gets legal HTMLTags 974, along with a YES message 974, through a function call getLegalHTMLTag 974 from MapTag 952. The function will get one HTMLTag 974 at a time and display it in the Legal HTML Tag list box 706 of Fig. 12B. The legal HTML tags are those which can follow the last HTML tag in the current HTML Tag list box 704 of Fig. 12B. In Fig. 18A(2), the MapTag 952 gets legal HTMLTags 976, along with a YES message 976, one at a time, from HTMLSymbolTable 800, through a function call getFirstLegalElement 976, or MapTag 952 will get legal HTMLTags 978, along with a YES message 978, from HTMLSymbolTable 800 through a function call GetNextLegalElement 978.

**[0114]** To add an HTML tag from the Legal HTML Tag list box 706 of Fig. 12B to the Current HTML Tag list box 704 of Fig. 12B, the user selects an HTML tag from the Legal HTML Tag list box 706 of Fig. 12B by double clicking the mouse on the HTML tag. The HTML tag is added to the Current HTML Tag list box 704. The selected HTMLTag 980 of Fig. 18A(1) is sent to MapTag 952 through a function call addSelectedHTMLTag 980. In Fig. 18A(2), MapTag 952 sends an HTMLTag 982 to MapCreateEditService 792 through a function call selectedHTMLTag 982. MapTag 952 informs MapCreateEditService 792 that the HTMLTag 982 must be added to the current map through a function call addSelectedHTMLTagToCurrentMappingList 984. MapEditDialog 950 of Fig. 18A(1) sets an HTML tag 986 as the last tag in the list by a function call setLastCurrentHTMLTagInList 986 to MapTag 952.

**[0115]** In order to update the list of Legal HTML Tags 706 of Fig 12B, MapEditDialog 950 of Fig. 18A(1) obtains an HTMLTag 988, with a YES message 988 if there exists a legal tag, from MapTag 952 using a function call getLegalHTMLTag 988. MapTag 952 then obtains an HTMLTag 990, with a YES message 990, from HTMLSymbolTable 800 using a function call getFirstLegalElement 990, if it is the first legal element requested, or MapTag 952 obtains an HTMLTag 992, with a YES message 992, from HTMLSymbolTable 800 using a function call getNextLegalElement 992, if it is not the first legal element requested.

**[0116]** To fill the HTML Tag Attribute list box 720 of Fig. 12B, MapEditDialog 950 of Fig. 18A(1) sends an HTMLTag 994 in the Current HTML Tag list box 704 of Fig. 12B to AssignAttribute 954 through a function call selectedHTMLTag 994. Then the MapEditDialog 950 gets attributes of the HTML tag, HTMLAttribute 996, along with a YES message 996, from AssignAttribute 954 through a function call getNextHTMLAttribute 996. The function getNextHTMLAttribute 996 gets one HTMLAttribute 996 at a time and displays them in the HTML Tag Attribute list box 720 of Fig. 12B. AssignAttribute 954 of Fig. 18A(3) gets, one at a time, an HTMLAttribute 998, along with a YES message 998, of the HTMLTag 998 from HTMLSymbolTable 800 through a function call getFirstAttributeOfElement 998. If it is not the first attribute, AssignAttribute 954 gets an HTMLAttribute 1002 of the HTMLTag 998, along with a YES message 1002, from HTMLSymbolTable 800 through a function call getNextAttribute 1002. For each attribute AssignAttribute 954 gets from HTMLSymbolTable 800, and returns to MapEditDialog 950 of Fig. 18A(1), MapEditDialog 950 checks with AssignAttribute 954 to see if the attribute is of the required type through a function call IsCurrentAttributeOfRequiredType 1000, with a NO message 1000 indicating it is not, to be obtained from HTMLSymbolTable 800. AssignAttribute 954 of Fig. 18A(3) then checks with HTMLSymbolTable 800 to see if the attribute is of the required type through a function call IsCurrentAttributeOfRequiredType 1001, with a NO message 1001 indicating it is not.

**[0117]** To select an attribute to assign it a value, the user selects an attribute in the HTML Tag Attribute list box 720 of Fig. 12B by double clicking the mouse on the attribute. The MapEditDialog box 950 of Fig. 18A(1) sends a selected HTMLAttribute 1004 to AssignAttribute 954 through a function call selectedHTMLAttribute 1004. Depending upon the attribute type to which the HTMLAttribute 1006 is assigned, the MapEditDialog box 950 of Fig. 18A(1) will take different actions. For example, AssignAttribute 954 of Fig. 18A(3) gets an SGMLAttribute type 1006, an SGMLTag 1006, and an SGMLAttribute 1006 by sending an HTMLAttribute 1006 to MapCreateEditService 792 using a function call getAttributeAssignmentInformationForHTMLAttribute 1006. The MapEditDialog 950 box of Fig. 18A(1) gets an SGMLAttribute type 1008 for the selected HTML Attribute 1004 from AssignAttribute 954 through a function call getAttributeType 1008. The SGML Attribute radio button 726 of Fig. 12B is displayed depressed. The MapEditDialog 950 of Fig. 18A(1)gets a

source SGMLTag 1010 and a source SGMLAttribute 1010 assigned to the HTML Attribute 1004 through a function call getSourceSGMLTagAndAttribute 1010 and displays them in the Source SGML Tag list box 722 of Fig. 12B, and the SGML Tag Attribute list box 724 of Fig. 12B. Next, the MapEditDialog 950 gets a source SGMLTag 1012, one SGMLTag 1012 at a time, which can be assigned to the HTML attribute 1004, along with a YES message 1012, from AssignAttribute 954 through a function call getNextSourceSGMLTag 1012, and displays them in the Source SGML Tag list box 722 of Fig. 12B. AssignAttribute 954 of Fig. 18A(3) gets source SGML Tags 1014, with a YES message 1014 if getting the first element, from SGMLSymbolTable 884 through a function call GetFirstElement 1014, or, one by one, gets a source SGML Tag 1018, with a YES message 1018, through a call GetNextElement 1018 if it is not the first element. AssignAttribute 954 verifies that the source SGML Tag 1014 has an attribute by checking with the SGMLSymbolTable 884 through a function call elementHasAttribute 1016 to obtain a YES 1016 response, for the first element if it has attributes. If it is not the first element, AssignAttribute 954 verifies that the source SGML Tags 1018 have attributes by checking with the SGMLSymbolTable 884 through a function call elementHasAttribute 1020 to obtain a YES 1020 response. The MapEditDialog 950 of Fig. 18A(1) gets all attributes SGMLAttribute 1022, one SGMLAttribute 1022 at a time, with a YES message 1022, of the previously assigned source SGML tag from AssignAttribute 954 through a function call getNextSourceSGMLAttribute 1022 and displays the attributes in the source SGML Tag Attribute list box 724 in Fig. 12B. AssignAttribute 954 of Fig. 18A(3) gets SGML attributes 1024, and a YES message 1024, from SGMLSymbolTable 884 by sending an SGML tag 1024 through a function call getFirstAttributeOfElemont 1024 if it is the first attribute of the element, or it gets SGMLAttributes 1026, one SGMLAttribute 1026 at a time, along with a YES message 1026, from SGMLSymbolTable 884 through a function call getNextAttribute 1026. The user selects an SGML tag from the Source SGML Tag list box 722 of Fig. 12B by double clicking the mouse on the SGML tag. The current implementation supports selection by double clicking the mouse on the selection. However, any alternative selection technique can be used, such as highlighting the selection and pressing the Enter or Return key. MapEditDialog 950 of Fig. 18C(1) sends a selected source SGMLTag 1110 to AssignAttribute 954 through a function call selectedSourceSGMLTag 1110. Then MapEditDialog 950 of Fig. 18A(1) gets all SGMLAttributes 1022 of the selected source SGMLTag 1110, one SGMLAttribute 1022 at a time, along with a YES message 1022, from AssignAttribute 954 through a function call getNextSourceSGMLAttribute 1022 and displays them in the SGML Tag Attribute list box 724 of Fig. 12B. AssignAttribute 954 of Fig. 18A(3) gets the SGMLAttributes 1024, along with a YES message 1024, from SGMLSymbolTable 884 through a function call getFirstAttributeOfElement 1024 if it is requesting the first attribute, or AssignAttribute 954 gets the SGMLAttributes 1026, one SGMLAttribute 1026 at a time, along with a YES message 1026, from SGMLSymbolTable 884 through a function call getNextAttribute 1026 if it is not the first attribute.

**[0118]** The user selects an SGML attribute from the source SGML Attribute list box 724 of Fig. 12B by double clicking the mouse on the SGML attribute. MapEditDialog 950 of Fig. 18C(1) sends a selected source SGMLAttribute 1112 to AssignAttribute 954 through a function call selectedSourceSGMLAttribute 1112. The user selects the Assign button 736 of Fig. 12B. MapEditDialog 950 lets AssignAttribute 954 know that an SGML attribute was assigned to the HTML attribute through a function call assignedSGMLAttribute 1114 of Fig. 18C(1). AssignAttribute 954 of Fig. 18C(3) sends, for the HTML attribute that is being assigned, an HTMLAttribute 1116, a source SGMLTag 1116, and the source SGMLAttribute 1116 to MapCreateEditService 792 through a function call assignHTMLAttributeWithSGMLAttribute 1116.

**[0119]** If SGML content was assigned to the HTML attribute, then the SGML Content radio button 728 of Fig. 12B is depressed. The MapEditDialog box 950 of Fig. 18B(1) gets a content SGMLTag 1048 assigned to the HTML attribute from AssignAttribute 954 through a function call getContentSGMLTag 1048, along with a YES message 1048, and displays it in the Source SGML Tag list box 722 of Fig. 12B. The MapEditDialog box 950 gets all the content SGMLTags 1052, one SGMLTag 1052 at a time, which can be assigned to the HTML attribute, along with a YES message 1052, from AssignAttribute 954 through a function call getNextContentSGMLTag 1052 and displays them in the Source SGML Tag list box 722 of Fig. 12B. AssignAttribute 954 of Fig. 18B(3) gets content SGMLTags 1054, along with a YES message 1054, from SGMLSymbolTable 884 through a function call getFirstPrimitiveElement 1054 if it is the first SGML tag requested, or AssignAttribute 954 gets content SGMLTags 1056, one SGMLTag 1056 at a time, along with a YES message 1056, from SGMLSymbolTable 884 through a function call getNextPrimitiveElement 1056.

**[0120]** The user selects the content SGML tag from the Source SGML Tag list box 722 of Fig. 12B by double clicking the mouse on the SGML tag. MapEditDialog 950 of Fig. 18B(1) sends the selected content SGMLTag 1058 to AssignAttribute 954 through a function call selectedContentSGMLTag 1058.

**[0121]** The user selects the Assign button 736 of Fig. 12B. MapEditDialog 950 of Fig. 18B(1) lets AssignAttribute 954 know that an SGML content was assigned to the HTML attribute through a function call assignSGMLContent 1060. AssignAttribute 954 of Fig. 18B(3) sends an HTMLAttribute 1062 that is being assigned the SGML content and a content SGMLTag 1062 to MapCreateEditService 792 through a function call assignHTMLAttributeWithSGMLContent 1062.

**[0122]** If System was assigned to the HTML attribute, the System radio button 730 of Fig. 12B is depressed. MapEditDialog 950 takes no further action. The user selects the Assign button 736 of Fig. 12B. MapEditDialog 950 of Fig. 18B(1) lets AssignAttribute 954 know that a system value was assigned to the HTMLAttribute through a function call

assignSystem 1049. AssignAttribute 954 of Fig. 18B(3) sends the HTMLAttribute 1051 that is being assigned a system value to MapCreateEditService 792 through a function call assignHTMLAttributeWithSystem 1051.

**[0123]** If No Value was assigned to the HTML attribute, the No Value radio button 732 is depressed. MapEditDialog 950 will take no further action. The user selects the Assign button 736 of Fig. 12B. MapEditDialog 950 of Fig. 18C(1) lets AssignAttribute 954 know that no value is assigned to the HTML attribute through a function call assignNoValue 1102. AssignAttribute 954 of Fig. 18C(3) sends the HTMLAttributes 1104 to be assigned no value to MapCreateEditService 792 through a function call assignHTMLAttributeWithNoValue 1104.

**[0124]** If User Input was assigned to the HTML attribute, the User Input radio button 734 is depressed. MapEditDialog 950 of Fig. 18C(1) gets a UserInput 1100 from AssignAttribute 954 through a function call getUserInputData 1100 and then displays the information in the User Input text edit box 731 of Fig. 12B. The user enters data into the text edit box 731 of Fig. 12B. MapEditDialog 950 of Fig. 18C(1) sends a UserInputData 1106 to AssignAttribute 954 through a function call selectedUserInputData 1106. The user then selects the Assign button 736 of Fig. 12B. MapEditDialog 950 of Fig. 18C(1) lets AssignAttribute 954 know that a user input is to be assigned to the HTML attribute through the function assignUserInput 1108. AssignAttribute 954 of Fig. 18C(3) sends an HTMLAttribute 1109 that is being assigned a user input value, and the UserInputData 1109, to MapCreateEditService 792 through a function call assignHTMLAttributeWithUserInput 1109.

**[0125]** For any of the HTML attribute source types assigned to the HTML attribute, the user has options to change the source type of the HTML attribute. For example, if the No Value radio button 732 of Fig. 12B is depressed for a selected HTML attribute, the user has an option to change the source type of the HTML attribute by pressing any of the other radio buttons such as the User Input radio button 734 of Fig. 12B. Depending upon which radio button the user selects, the user will need to enter more information before the HTML, attribute is assigned a value.

**[0126]** For the radio buttons SGML Attribute 726, SGML Content 728, and User Input 734, the user can change the input selection more than one time. For example, the user can change the user input in the text edit box 731 more than one time. As another example, the user can select one content SGML tag in the Source SGML Tag list box 722, and later decide to select another content SGML tag. The most recent value given by the user is the value assigned to the HTML attribute when the Assign button 736 is selected.

**[0127]** The user repeats selection of an attribute to assign it a value, selecting a radio button for an HTML attribute source type, assigning a value to the HTML attribute, and selecting the Assign button 736 of Fig. 12B until the user has assigned all the HTML attributes desired. The user then selects the Assign Done button 740 of Fig. 12B. MapEditDialog 950 of Fig. 18B(1) lets MapCreateEditService 792 know that it is done assigning values to the attributes through a function call assignDoneSelected 1064 to AssignAttribute 954, which sends a call assignDoneSelected 1066 to MapCreateEditService 792 of Fig. 18B(3).

**[0128]** The user repeats adding more HTML tags to the Current HTML Tag list box 704 of Fig.12B at will. At any time the user can clear the Current HTML Tag list box 704 or delete HTML tags from the Current HTML Tag list box 704 by selecting the Clear HTML button 708 or the Delete HTML button 710 of Fig. 12B. Once the user has completed mapping one SGML tag, the user selects the Map SGML Tag button 714. MapEditDialog 950 of Fig. 18B(1) informs MapTag 952 that the user has completed mapping the selected SGML tag through a function call doneMappingSGMLTag 1068. MapTag 952 of Fig. 18B(2) informs MapCreateEditService 792 that the user has completed mapping the selected SGML tag through the function finishOneMapping 1070. MapEditDialog 950 of Fig. 18B(1) requests a next SGMLTag 1072 for mapping from MapTag 952, along with a YES message 1072, using a function call getNextSGMLTag 1072 and displays the SGMLTag 1072 in the SGML Tag list box 702 of Fig. 12B. MapTag 952 of Fig. 18B(2) then obtains a next SGMLTag 1074, along with a YES message 1074, from SGMLSymbolTable 884 using a function call getNextElement 1074. MapTag 952 then returns the SGMLTag 1072 to MapEditDialog 950 of Fig. 18B(1) in response to the function call getNextSGMLTag 1072.

**[0129]** The user repeats processing the map for all the SGML tags the user wants to map. When the user is finished with the map editor, the user selects the Done button 716 of Fig. 12B. MapEditDialog 950 of Fig. 18B(1) informs MapTag 952 that it is done mapping through a function call doneMapping 1076. MapTag 952 of Fig. 18B(2) informs MapCreateEditService 792 that the mapping of the SGML tags is completed through a function call finishCreatingMap 1078.

**[0130]** Fig. 19 illustrates an exemplary hardware configuration upon which the invention may be implemented. A Workstation 1200 has component parts a Display Controller 1202, a Central Processing Unit ("CPU") 1204, a Random Access Memory ("RAM") 1206, a Read Only Memory ("ROM") 1208, an Input Controller 1210, connected to a Keyboard 1212 and a Mouse 1214, a System Bus 1220, a Hard Disk 1222 and a Floppy Drive 1224 connected to a Disk Controller 1226, a Comm Controller 1228 connected to a Network 1230, and an Input/Output ("I/O") Controller 1232 connected to a Hard Disk 1236 and a Printer 1234, and a Cathode Ray Tube ("CRT") 1238 connected to the Display Controller 1202. The System Bus 1220 connects the CPU 1204, the RAM 1206, the ROM 1208, the Input Controller 1210, the Disk Controller 1226, the Comm Controller 1228, the I/O Controller 1232, and the Display Controller 1202 for transmitting data over the connection line.

**[0131]** For example, the computer code generated for execution is loaded into the RAM 1206 for execution by the CPU 1204, using the System Bus 1220, with input files stored on the Hard Disk 1236, with other input coming from the Keyboard 1212 and the Mouse 1214 through the Input Controller 1210, and from the Hard Disk 1222 and the Floppy Drive 1224, through the Disk Controller 1226, onto the System Bus 1220. The System Bus 1220 interacts with the ROM 1208, the Network 1230, and the Comm Controller 1228. The GUI of the system can be displayed on the CRT 1238 through the Display Controller 1202, and on output to the Printer 1234 or to the Hard Disk 1236 through the I/O Controller 1232.

**[0132]** Other implementations of the map creator and editor for transforming a first structured information format to a second structured information format are possible using the procedures described previously for Figs. 1A-19. For example, variable names in a first database format may be mapped to variable names in a second database format. Another exemplary implementation is a mapping of public identifiers in an ISO/IEC 9070 format to file names in a UNIX file system format.

**[0133]** For other implementations, the same techniques described with regard to the SGML to HTML mapping and transformation are utilized, with structure of information defined differently from an SGML DTD. In general terms, a parser breaks down an input source file into source components and their structure, based upon a structure format specified for the input source file, for map creating and editing. The source components and their structure are presented to the user for interactive selection of components of the first structure, with candidate target components of the second structure presented to the user for selection of target components for the mapping of the source components for creation of rules for a transformation map. An exemplary implementation of a mapping of public identifiers in an ISO/IEC 9070 format to file names in a UNIX file system format is discussed below with regard to Figs. 20A-20H.

**[0134]** Fig. 20A illustrates a public identifier 1400 in ISO/IEC 9070 standard format. An owner name is made up of a registered owner name and an unregistered owner name. For this example, the owner name is 'XYZ'. The public identifier further has an object name, separated from the owner name by '//'. For this example, the object name is 'font::metric::x-offset::622'.

**[0135]** Mapping one system structure to another system structure involves transformation of strings in one allowable character set to strings of another allowable character set. For example, computer programming languages are defined as having an alphabet of acceptable characters for forming valid variable names. A first programming language may be defined as allowing the '-' (hyphen) character to be embedded in a valid variable name, but not the '_' (underscore) character. A second programming language may be defined as allowing the '_' character, but not the '-' character. A mapping of valid variable names of the first programming language to valid variable names of the second programming language would involve mapping occurrences of '-' to a different character, such as '_', which is acceptable in the second programming language environment.

**[0136]** In the example of mapping the ISO/IEC 9070 naming scheme to the UNIX file name scheme, the separator '//' is allowed in ISO/IEC 9070, but is not a valid character sequence in the UNIX file system naming conventions. A user wishing to map valid ISO/IEC 9070 names to valid UNIX file names needs to transform every occurrence of the separator '//' into a valid UNIX file name character string.

**[0137]** Fig. 20B illustrates an exemplary mapping of an ISO/IEC 9070 public identifier to a UNIX file name format. The exemplary mapping maps the structured public identifier to a flat UNIX file name. Lines 1420, 1422, 1424, and 1426 illustrate rules to map component name strings to identical strings in the UNIX format. Line 1428 illustrates a rule to map an ISO/IEC 9070 owner name component separator '::', which is not widely used, to the character '_', which is a valid UNIX character. Line 1430 illustrates a rule to map an ISO/IEC 9070 owner name, object name component separator '//', which is not a valid string in the UNIX file format, to '__' (two underscore characters).

**[0138]** Fig. 20C illustrates an exemplary UNIX file name 1440 resulting from mapping the exemplary ISO/IEC 9070 name of Fig. 20A through the mapping of Fig. 20B. Ownername 'XYZ' of line 1400 of Fig. 20A is mapped to 'XYZ' using the rule of line 1420 of Fig. 200. The '//' is mapped to '__' using the rule of line 1430 of Fig. 20B. The three substrings '::' in the object name of public identifier 1400 of Fig. 20A are each mapped to a character '_' using the rule 1428 of Fig. 20B.

**[0139]** Fig. 20D illustrates an exemplary user interface for mapping a public identifier 1502 of ISO/IEC 9070 to a UNIX file system format 1504. A map editor and creator 1500 maps names in the ISO/IEC 9070 convention to names in the UNIX file system convention. An exemplary mapping starts with system display 1502 of public identifier components and a display of valid UNIX file name components 1504. The public identifier components 1502 are registered owner name, unregistered owner name, and object name. A user selects one of these options. If owner name is selected, then the user is asked to select from prefix and owner-name components 1522 in Fig. 20E. User options presented are a create directory 1506, a map 1508, a merge all 1510, a next 1512, and a previous 1514. The create directory 1506 option allows the user to create a new directory name in the UNIX file system window 1504. The map option 1508 allows the user to request that a map be created at the time of request. The merge all 1510 option allows the user to create a UNIX file name 1504 by merging all the components of the public identifier name 1502 into a flat file name 1504. The next 1512 option allows the user to step to a next screen. The previous 1514 option allows the user to back up to the previous screen.

**[0140]** Fig. 20E illustrates an exemplary user interface 1520 for a registered owner 1522 component of the ISO/IEC 9070 public identifier 1502 of Fig. 20D. User interrace 1520 options presented are a window 1522 showing a prefix and an owner-name component. User options are a map individually 1524, a merge both 1526, a next 1527, and a previous 1528. The map individually 1524 option allows the user to map individual components of the ISO/IEC 9070 name 1522 to individual components of the UNIX file system scheme 1504 of Fig. 20D. The merge both 1526 option allows the user to merge components of the registered owner name 1522 into one flat UNIX file name or directory name. The next 1527 option allows the user to step to a next screen. The previous 1528 option allows the user to back up to the previous screen.

**[0141]** Fig. 20F illustrates an exemplary user interface 1530 for mapping the prefix, owner name component separator to the UNIX legal character set format 1534. The registered owner component has a prefix and an owner-name component separator "::" 1532 which is not a widely used character string in the UNIX environment. The user is allowed to map the '::' separator 1532 to any of the valid characters in the UNIX file system character set 1534, with a mapping to '_' as a default mapping. User options are a map 1536, a next 1537, and a previous 1538. The map 1536 option allows the user to select creating a map, with the assumption that the user has finished selecting options for creation of the map. The next 1537 option allows the user to step to a next screen. The previous 1538 option allows the user to back up to the previous screen.

**[0142]** Fig. 20G illustrates an exemplary user interface 1540 for mapping an owner name character 1542 to valid characters 1544 of the UNIX file system format. The user is given the options of mapping special characters 1542 which are valid in the ISO/IEC 9070 scheme to characters which are valid in the UNIX file system scheme 1544. A mapping of a character in the ISO/IEC 9070 scheme 1542 is set to '_' as a default mapping. The user is given options of a map 1546, a next 1548, and a previous 1550. The map 1546 option allows the user to select creating a map, with the assumption that the user has finished selecting options for creation of the map. The next 1548 option allows the user to step to a next screen. The previous 1550 option allows the user to back up to the previous screen.

**[0143]** Fig. 20H illustrates an exemplary user interface 1560 for the user to map a registered owner component 1562 to a UNIX file scheme format 1564. The user is allowed to select a prefix component of the registered owner name or an owner name component other than prefix 1562. The user is allowed to select a directory option in the UNIX scheme 1564. The user is also allowed to select a file with object name option in the UNIX file scheme 1564. The user is given an option of a map 1566 for creating the map with the options currently selected. The user is also given an option of a previous 1568 to back up to the previous screen.

**[0144]** The present invention has been described using an exemplary implementation of a mapping creator and editor for an SGML to HTML transformer with user interaction for creation and editing of the map, and an exemplary mapping creator and editor for an ISO/IEC 9070 to a UNIX file format transformer. The example shown in this disclosure uses OOP and Windows GUI techniques to implement the user interface, map processing, and transformation. However, the user interface can be implemented using text line queries or menus. Programming methodologies other than OOP can be used for implementing the processing. References to storage areas can be made by techniques other than using pointers.

**[0145]** This invention may be conveniently implemented using a conventional general purpose digital computer or microprocessor programmed according to the teachings of the present specification, as will be apparent to those skilled in the computer art. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those skilled in the software art. The invention may also be implemented by the preparation of application specific integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

**[0146]** The present invention includes a computer program product which is a storage medium including instructions which can be used to program a computer to perform a process of the invention. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, CD-ROMs, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, or any type of media suitable for storing electronic instructions.

**[0147]** Obviously, numerous modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope of the appended claims, the invention may be practiced otherwise than as specifically described herein.

# Appendix A

```
<!doctype test [
<!element test - o (front, section+)
<!element front - o (title, author, keywords?)>
<!element title - o (#PCDATA)>
<!element author - o (fname, surname, title?)>
<!element (fname, surname) - o (#PCDATA)>
<!element keywords - o (#PCDATA)>
<!element section - o (number?, title?, para*, subsec1*)>
<!element subsec1 - o (number?, title?, para*, subsec2*)>
<!element subsec2 - o (number?, title?, para+)>
<!element number - o (#PCDATA)>
<!element para - o (#PCDATA)>
]>
```

# Appendix B

| | | |
|---|---|---|
| 30 | Mapping | |
| 32 | Front → | Null |
| 34 | Title → | H3 |
| 36 | Author → | Null |
| 38 | Frame → | P |
| 40 | Surname → | P |
| 42 | Keywords → | P |
| 44 | Section → | Null |
| 46 | Number → | P strong |
| 48 | Para → | P |
| 50 | Subsec 1 → | Null |
| 52 | Subsec 2 → | Null |

# Appendix C

```
<test>
<front>
<title>
Test mapping
</title>
<author>
<fname>
Tester
</fname>
<surname>
Giver
</surname>
</author>
<keywords>
Mapping
</keywords>
</front>
<section>
<number>
1
</number>
<title>
First Major Section
</title>
<para>
The first major section para.
</para>
<subsec1>
<number>
1.1
</number>
<title>
Subsection 1.1
</title>
<para>
This is a para in the subsec1.1
</para>
</subsec1>
<subsec1>
  <number>
1.2
</number>
<title>
Subsection 1.2
</title>
<para>
This is a subsection 1.2
</para>
</subsec1>
</section>
<section>
<number>
2
</number>
```

```
<title>
Second Major Section
</title>
<para>
The second major section para.
</para>
<subsec1>
<number>
2.1
</number>
<title>
Subsection 2.1
</title>
<para>
This is a para in the subsec 2.1
</para>
</subsec1>
<subsec1>
<number>
2.2
</number>
<title>
Subsection 2.2
</title>
<para>
This is a subsection 2.2
</para>
</subsec1>
</section>
</test>
```

# Appendix D

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<html>

<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<meta name="GENERATOR" content="Microsoft FrontPage 2.0">
<title>test</title>
</head>

<body bgcolor="#FFFFFF">

<h3>Test mapping</h3>

<p>Tester</p>

<p>Giver</p>

<p>Mapping</p>

<p><strong>1</strong></p>

<h3>First Major Section</h3>

<p>The first major section para.</p>

<p><strong>1.1</strong></p>

<h3>Subsection 1.1</h3>

<p>This is a para in the subsec1.1</p>

<p><strong>1.2</strong></p>

<h3>Subsection 1.2</h3>

<p>This is a subsection 1.2</p>

<p><strong>2</strong></p>

<h3>Second Major Section</h3>

<p>The second major section para.</p>

<p><strong>2.1</strong></p>

<h3>Subsection 2.1</h3>

<p>This is a para in the subsec 2.1</p>

<p><strong>2.2</strong></p>

<h3>Subsection 2.2</h3>

<p>This is a subsection 2.2</p>
</body>
</html>
```

## Claims

1. An object-oriented system for processing structured information for implementation by a computer in an object-oriented framework, comprising:

   a storage means;
   a first obtaining means for obtaining an interactive input from a user;
   a second obtaining means for obtaining a first structural description of a first structured information format;
   a third obtaining means for obtaining a second structural description of a second structured information format;
   means for creating a rule to transform an element of the first structured information format into an element of the second structured information format utilizing the interactive input from the user, the first structural description, and the second structural description; and
   means for outputting the rule,
   wherein at least one of the first obtaining means, the second obtaining means, the third obtaining means, the means for creating, and the means for outputting includes a software object.

2. A system according to Claim 1, wherein the first structured information format includes an ISO/IEC 9070 public identifier naming format, the second structured information format includes an operating system file name format, and the means for creating comprises:

   means for creating a rule to transform an ISO/IEC 9070 public identifier naming format element into an operating system file name format element utilizing the interactive input from the user, a structural description of the ISO/IEC 9070 public identifier naming format, and a structural description of the operating system file name format.

3. A system according to Claim 1, wherein the first structured information format includes a first database variable name format, the second structured information format includes a second database variable name format, and the means for creating comprises:

   means for creating a rule to transform a first database variable name format element into a second database variable name format element utilizing the interactive input from the user, a structural description of the first database variable name format, and a structural description of the second database variable name format.

4. A system according to Claim 1, wherein the structured information includes a markup language format, the first structured information format includes a first markup language format, the second structured information format includes a second markup language format, and the means for creating comprises:

   means for creating a rule to transform a first markup language format element into a second markup language format element utilizing the interactive input from the user, a structural description of the first markup language format, and a structural description of the second markup language format.

5. A system according to Claim 4, wherein the first markup language format includes a Standard Generalized Markup Language ("SGML"), the second markup language format includes a HyperText Markup Language ("HTML"), and the means for creating comprises:

   means for creating a rule to transform an SGML element of the first markup language format into an HTML element of the second markup language format utilizing the interactive input from the user, the first structural description which includes an SGML Document Type Definition ("DTD"), and the second structural description which includes an HTML DTD.

6. A system according to Claim 1, wherein the means for creating comprises:

   a map creator object.

7. A system according to Claim 6, wherein the means for outputting the rule comprises:

   an object method for outputting the rule to a map object.

8. A system according to Claim 6, wherein the map creator object comprises:

a reference to a software object for an element for transformation of the first structured information format;
a reference to a software object for an element of the second structured information format, for transformation of the element of the first structured information format;
a reference to a software object for a property of the element of the second structured information format, for transformation of the element of the first structured information format;
a reference to a software object for an attribute value of the element of the second structured information format, for transformation of the element of the first structured information format;
an object method for obtaining the element for transformation of the first structured information format, which has been interactively selected by the user, using the software object for the element for transformation of the first structured information format;
an object method for obtaining the element of the second structured information format which corresponds to the element of the first structured information format, which has been interactively selected by the user, using the software object for the element of the second structured information format;
an object method for determining a property of the element of the second structured information format which has been selected by the user, using the software object for a property of the element of the second structured information format;
an object method for obtaining a second structured information format attribute value which has been interactively input by a user, using the software object for the attribute value of the element of the second structured information format; and
an object method for assigning the attribute value which has been interactively input by a user to the second structured information format attribute value.

9. A system according to Claim 8, wherein the map creator object further comprises:

a reference to a software object for registering an instance of an element for transformation of the first structured information format; and
a reference to a software object for unregistering the instance of an element for transformation of the first structured information format when the element is no longer needed by the map creator

10. A system according to Claim 1, further comprising:

means for processing an element of the first structured information format into a plurality of first structured information format components.

11. A system according to Claim 10, wherein the means for processing an element of the first structured information format into a plurality of first structured information format components comprises:

a parser object.

12. A system according to Claim 11, wherein the parser object comprises:

a reference to the element of the first structured information format;
a reference to a storage area for the plurality of first structured information format components; and
an object method for processing the element of the first structured information format into the plurality of first structured information format components for storage in the storage area for the plurality of first structured information format components.

13. A system according to Claim 12, further comprising:

means for utilizing the rule to transform the element of the first structured information format into the element of the second structured information format.

14. A system according to Claim 13, wherein the means for utilizing the rule to transform the element of the first structured information format into the element of the second structured information format comprises:

a transformer object.

**15.** A system according to Claim 1, wherein the means for obtaining an interactive input from a user comprises:

a user interface object.

**16.** A system according to Claim 15, wherein the user interface object comprises:

a reference to a software object for user input;
an object method for obtaining interactive input from the user of a selection of an element for transformation of a first structured information format using the software object for user input; and
an object method for obtaining interactive input from the user of a selection of an element of a second structured information format which corresponds to the element of the first structured information format using the software object for user input.

**17.** A system according to Claim 15, wherein the user interface object further comprises:

a reference to a software object for user input of a selection of a transformation to be performed on the element of the first structured information format; and
an object method for obtaining interactive input from the user of the selection of a transformation to be performed on the element of the first structured information format using the software object for user input of the selection of the transformation.

**18.** A system according to Claim 17, wherein the object method for creating a rule to transform an element of a first structured information format into an element of a second structured information format utilizing the interactive input from the user further comprises:

a reference to a software object for a rule to be created; and
an object method for creating a rule to map the second element of the first structured information format to a null string using the software object for the rule to be created, when the selection of a transformation which has been input by the user indicates a null transformation is to be performed.

**19.** A system according to Claim 17, wherein the object method for creating a rule to transform an element of a first structured information format into an element of a second structured information format utilizing the interactive input from the user further comprises:

a reference to a software object for a rule to be created; and
an object method for creating a rule to map the second element of the first structured information format to a copy of the second element of the first structured information format using the software object for the rule to be created, when the selection of a transformation which has been input by the user indicates a transformation of the second element of the first structured information format to a copy of the second element of the first structured information format is to be performed.

**20.** A system according to Claim 15, wherein the user interface object further comprises:

a reference to a software object for an interactive user input of a source for inputting the second structured information format attribute value;
a reference to a software object for an interactive user input of the second structured information format attribute value;
an object method for obtaining interactive input from the user of the source for inputting the second structured information format attribute value using the software object for the interactive user input of the source for inputting the second structured information format attribute value; and
an object method for obtaining interactive input from the user of the second structured information format attribute value using the software object for the interactive user input of the second structured information format attribute value.

**21.** A system according to Claim 20, further comprising:

a reference to a software object for a rule to be created;
an object method for examining the source which has been input by the user; and

an object method for assigning a null value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates no source is to be used.

**22.** A system according to Claim 20, further comprising:

a reference to a software object for a rule to be created;
an object method for examining the source which has been input by the user; and
an object method for assigning a system value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a system source is to be used.

**23.** A system according to Claim 20, further comprising:

a reference to a software object for a rule to be created;
an object method for examining the source which has been input by the user; and
an object method for assigning a first structured information format attribute value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a first structured information format attribute source is to be used.

**24.** A system according to Claim 20, further comprising:

a reference to a software object for a rule to be created;
an object method for examining the source which has been input by the user; and
an object method for assigning a first structured information format content value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a first structured information format content source is to be used.

**25.** A system according to Claim 20, further comprising:

a reference to a storage buffer for the source which has been input by the user;
an object method for examining the source which has been input by the user using the storage buffer for the source which has been input by the user;
an object method for interactively inputting a user input value, when the source which has been input by the user indicates a user input source is to be used; and
an object method for assigning the user input value to the second structured information format attribute value, when the source which has been input by the user indicates a user input source is to be used.

**26.** A system according to Claim 1, wherein the user comprises:

a software object.

**27.** An object-oriented computer program product for processing structured information for implementation by a computer in an object-oriented framework, comprising:

a storage means;
a first obtaining means for obtaining an interactive input from a user;
a second obtaining means for obtaining a first structural description of a first structured information format;
a third obtaining means for obtaining a second structural description of a second structured information format;
means for creating a rule to transform an element of the first structured information format into an element of the second structured information format utilizing the interactive input from the user, the first structural description, and the second structural description; and means for outputting the rule,
wherein at least one of the first obtaining means, the second obtaining means, the third obtaining means, the means for creating, and the means for outputting includes a software object.

**28.** A computer program product according to Claim 27, wherein the first structured information format includes ISO/IEC 9070 public identifier naming format, the second structured information format includes an operating system file name format and the means for creating comprises:

means for creating a rule to transform an element of a first structured information format which includes an ISO/IEC 9070 public identifier element into an element of a second structured information format which includes an operating system file name format element utilizing the interactive input from the user, the first structural description which includes a structural description of the ISO/IEC 9070 public identifier format, and the second structural description which includes a structural description of the operating system file name format.

**29.** A computer program product according to Claim 27, wherein the structured information includes database variable names, the first structured information format includes a first database variable name format, the second structured information format includes a second database variable name format, and the means for creating comprises:

means for creating a rule to transform an element of a first structured information format which includes a first database variable name format element into an element of a second structured information format which includes a second database variable name format element utilizing the interactive input from the user, the first structural description which includes a structural description of the first database variable name format, and the second structural description which includes a structural description of the second database variable name format.

**30.** A computer program product according to Claim 27, wherein the structured information includes markup language, the first structured information format includes a first markup language, the second structured information format includes a second markup language, and the means for creating comprises:

means for creating a rule to transform an element of a first structured information format which includes a first markup language element into an element of a second structured information format which includes a second markup language element utilizing the interactive input from the user, the first structural description which includes a structural description of the first markup language, and the second structural description which includes a structural description of the second markup language.

**31.** A computer program product according to Claim 30, wherein the first markup language includes SGML, the second markup language includes HTML, and the means for creating further comprises:

means for creating a rule to transform an element of a first markup language which includes an SGML element into an element of a second markup language which includes an HTML element utilizing the interactive input from the user, the first structural description which includes an SGML DTD, and the second structural description which includes an HTML DTD.

**32.** A computer implemented method to provide a graphical user interface for creating a mapping of a first structural description to a second structural description, comprising the steps of:

displaying an element for transformation of a first structural description;
displaying a list of candidate elements of a second structural description;
inputting, from a user, a first selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
storing the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description as a rule.

**33.** A method according to Claim 32, wherein the first structural description includes an ISO/IEC 9070 public identifier naming format, the second structural description includes an operating system file name format, and the step of displaying the element for transformation comprises:

displaying the element for transformation which includes an element of the ISO/IEC 9070 public identifier naming format; and
the step of displaying the list of candidate elements comprises:
displaying the list of candidate elements which includes a list of operating system file name candidate elements.

**34.** A method according to Claim 32, wherein the first structural description includes a first database variable name format, the second structural description includes a second database variable name format, and the step of displaying the element for transformation comprises:

displaying the element for transformation which includes an element of the first database variable name format; and
the step of displaying the list of candidate elements comprises:
displaying the list of candidate elements which includes a list of second database variable name format candidate elements.

**35.** A method according to Claim 32, further comprising the steps of:

obtaining the stored rule;
displaying the element for transformation of the first structural description;
displaying the first selected element of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;
displaying the list of candidate elements of the second structural description;
inputting, from the user, a second selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
storing the correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description as a rule.

**36.** A method according to Claim 32, further comprising the steps of:

displaying an icon for the user to input a request to clear the first selection which is being displayed;
inputting the request to clear the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed; and
clearing the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed.

**37.** A method according to Claim 32, wherein the first structural description includes a first markup language, the second structural description includes a second markup language, and the step of displaying the element for transformation comprises:

displaying the element for transformation which includes an element of the first markup language; and
the step of displaying the list of candidate elements comprises:
displaying the list of candidate elements which includes a list of second markup language candidate elements.

**38.** A method according to Claim 37, wherein the first markup language includes a Standard Generalized Markup Language ("SGML"), the second markup language includes a HyperText Markup Language ("HTML"), and the step of displaying the element for transformation comprises:

displaying the element for transformation which includes an element of SGML; and
the step of displaying the list of candidate elements comprises:
displaying the list of candidate elements which includes a list of HTML candidate elements.

**39.** A method according to Claim 32, wherein the storing step comprises:

storing the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description as a rule in a list of rules for transformation.

**40.** A method according to Claim 39, further comprising the steps of:

displaying a second element for transformation of the first structural description;
displaying a list of candidate elements of the second structural description;
inputting, from the user, a second selection of one of the candidate elements of the second structural description which defines a correspondence between the second element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description for a transformation of the second element of the first structural description to the second structural description; and
storing the correspondence between the second element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description as a rule in a list of rules for transformation.

**41.** A method according to Claim 39, wherein the inputting step further comprises:

displaying an icon for the user to input a request to store the first selection correspondence as a rule in the list of rules for transformation; and
the storing step further comprises:
inputting the request to store the correspondence as a rule in the list of rules for transformation, when the user inputs the request to store the first selection as a rule; and
displaying a second element for transformation of the first structural description, when the user inputs the request to store the first selection as a rule.

**42.** A method according to Claim 39, wherein the inputting step further comprises:

displaying an icon for the user to input a request to store the correspondence as a rule in the list of rules for transformation; and
the storing step further comprises:
inputting the request to store the correspondence as a rule in the list of rules for transformation, when the user inputs the request to store the first selection as a rule in the list of rules for transformation; and
storing the list of rules for transformation as a map.

**43.** A method according to Claim 39, further comprising the steps of:

displaying an icon for the user to input a request to delete the list of rules for transformation;
inputting the request to delete the list of rules for transformation, when the user inputs the request to delete the list or rules for transformation; and
deleting the list of rules for transformation, when the user inputs a request to delete the list of rules for transformation.

**44.** A method according to Claim 32, further comprising the steps of:

displaying the first selection of one of the candidate elements of the second structural description which defines the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description.

**45.** A method according to Claim 44, wherein the inputting step comprises:

inputting, from the user, a first ordered list of a plurality of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;
the step of displaying the first selection comprises:
displaying the first ordered list of the plurality of the candidate elements of the second markup language which defines a correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
the storing step comprises:
storing the correspondence between the element for transformation of the first structural description and the

first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description as a rule.

**46.** A method according to Claim 45, further comprising the steps of:

obtaining the stored rule;
displaying the element for transformation of the first structural description;
displaying the first ordered list of elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;
displaying the list of candidate elements of the second structural description;
inputting, from the user, a second ordered list of a plurality of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the second ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
storing the correspondence between the element for transformation of the first structural description and the second ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description as a rule.

**47.** A method according to Claim 45, further comprising the steps of:

displaying an icon for the user to input a request to clear the first ordered list which is being displayed;
inputting the request to clear the first ordered list which is being displayed, when the user inputs the request to clear the first ordered list which is being displayed; and
clearing the first ordered list which is being displayed, when the user inputs the request to clear the first ordered list which is being displayed.

**48.** A method according to Claim 32, further comprising the steps of:

displaying an attribute of the first selection of one of the candidate elements of the second structural description which corresponds to a transformation of the element of the first structural description to the second structural description, for assignment of an attribute value of the second structural description;
displaying a plurality of icons representing sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed;
displaying the element of the first structural description;
displaying an attribute list of the element of the first structural description;
inputting a user input of a selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed; and
processing the user input of the selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed, when the user inputs the selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed.

**49.** A method according to Claim 48, wherein the processing step further comprises:

assigning a null value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates no source is to be used.

**50.** A method according to Claim 48, wherein the processing step further comprises:

assigning a system value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a system value is to be used.

**51.** A method according to Claim 48, wherein the processing step further comprises:

assigning a first structural description attribute value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a first structural description

attribute value source is to be used.

**52.** A method according to Claim 48, wherein the processing step further comprises:

assigning a first structural description content value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a first structural description content value source is to be used.

**53.** A method according to Claim 48, wherein the processing step further comprises:

assigning a user input value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a user input value source is to be used.

**54.** A method according to Claim 53, wherein the assigning step comprises:

displaying a text input area for the user to input a value to be assigned;
inputting the value entered by the user in the text input area; and
assigning the value input by the user to the attribute of the first selection which is being displayed.

**55.** A method according to Claim 32, wherein the step of displaying a list of candidate elements of a second structural description further comprises:

displaying a candidate for requesting removal of the first structural description element in the transformation; and
displaying a candidate for requesting ignoring of the first structural description element in the transformation.

**56.** An apparatus for providing a graphical user interface for creating a mapping of a first structural description to a second structural description, comprising:

an element displaying means for displaying an element for transformation of a first structural description;
a list displaying means for displaying a list of candidate elements of a second structural description;
a user inputting means for inputting, from a user, a first selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
a storing means for storing the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description as a rule.

**57.** An apparatus according to Claim 56, wherein the first structural description includes an ISO/IEC 9070 public identifier naming format, the second structural description includes an operating system file name format, and the element displaying means further comprises:

means for displaying the element for transformation which includes an element of the ISO/IEC 9070 public identifier naming format; and
the list displaying means further comprises:
means for displaying the list of candidate elements which includes a list of operating system file name candidate elements.

**58.** An apparatus according to Claim 56, wherein the first structural description includes a first database variable name format, the second structural description includes a second database variable name format, and the element displaying means further comprises:

means for displaying the element for transformation which includes an element of the first database variable name format; and
the list displaying means further comprises:
means for displaying the list of candidate elements which includes a list of second database variable name for-

mat candidate elements.

**59.** An apparatus according to Claim 56, further comprising:

means for obtaining the stored rule;
means for displaying the element for transformation of the first structural description;
means for displaying the first selected element of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;
means for displaying the list of candidate elements of the second structural description;
means for inputting, from the user, a second selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
means for storing the correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description as a rule.

**60.** An apparatus according to Claim 56, further comprising:

means for displaying an icon for the user to input a request to clear the first selection which is being displayed;
means for inputting the request to clear the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed; and
means for clearing the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed.

**61.** An apparatus according to Claim 56, wherein the first structural description includes a first markup language, the second structural description includes a second markup language, and the element displaying means further comprises:

means for displaying the element for transformation which includes an element of the first markup language; and
the list displaying means further comprises:
means for displaying the list of candidate elements which includes a list of second markup language candidate elements.

**62.** An apparatus according to Claim 61, wherein the first markup language includes an SGML, the second markup language includes an HTML, and the element displaying means further comprises:

means for displaying the element for transformation which includes an element of SGML; and
the list displaying means comprises:
means for displaying the list of candidate elements which includes a list of HTML candidate elements.

**63.** An apparatus according to Claim 56, wherein the storing means comprises:

means for storing the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description as a rule in a list of rules for transformation.

**64.** An apparatus according to Claim 63, further comprising:

means for displaying a second element for transformation of the first structural description;
means for displaying a list of candidate elements of the second structural description;
means for inputting, from the user, a second selection of one of the candidate elements of the second structural description which defines a correspondence between the second element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description for a transformation of the second element of the first structural description to the second structural description; and

means for storing the correspondence between the second element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description as a rule in a list of rules for transformation.

**65.** An apparatus according to Claim 63, further comprising:

means for displaying an icon for the user to input a request to store the first selection correspondence as a rule in the list of rules for transformation;
means for inputting the request to store the correspondence as a rule in the list of rules for transformation, when the user inputs the request to store the first selection as a rule; and
means for displaying a second element for transformation of the first structural description, when the user inputs the request to store the first selection as a rule.

**66.** An apparatus according to Claim 63, further comprising:

means for displaying an icon for the user to input a request to store the correspondence as a rule in the list of rules for transformation;
means for inputting the request to store the correspondence as a rule in the list of rules for transformation, when the user inputs the request to store the first selection as a rule in the list of rules for transformation; and
means for storing the list of rules for transformation as a map.

**67.** An apparatus according to Claim 63, further comprising:

means for displaying an icon for the user to input a request to delete the list of rules for transformation;
means for inputting the request to delete the list of rules for transformation, when the user inputs the request to delete the list of rules for transformation; and
means for deleting the list of rules for transformation, when the user inputs a request to delete the list of rules for transformation.

**68.** An apparatus according to Claim 56, further comprising:

means for displaying the first selection of one of the candidate elements of the second structural description which defines the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description.

**69.** An apparatus according to Claim 68, wherein the user inputting means comprises:

means for inputting, from the user, a first ordered list of a plurality of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;
the means for displaying the first selection comprises:
means for displaying the first ordered list of the plurality of the candidate elements of the second markup language which defines a correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
the storing means comprises:
means for storing the correspondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description as a rule.

**70.** An apparatus according to Claim 69, further comprising:

means for obtaining the stored rule;
means for displaying the element for transformation of the first structural description;
means for displaying the first ordered list of elements of the second structural description which defines a cor-

respondence between the element for transformation of the first structural description and the first ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;

means for displaying the list of candidate elements of the second structural description;

means for inputting, from the user, a second ordered list of a plurality of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the second ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and

means for storing the correspondence between the element for transformation of the first structural description and the second ordered list of the plurality of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description as a rule.

**71.** An apparatus according to Claim 69, further comprising:

means for displaying an icon for the user to input a request to clear the first ordered list which is being displayed;

means for inputting the request to clear the first ordered list which is being displayed, when the user inputs the request to clear the first ordered list which is being displayed; and

means for clearing the first ordered list which is being displayed, when the user inputs the request to clear the first ordered list which is being displayed.

**72.** An apparatus according to Claim 56, further comprising:

means for displaying an attribute of the first selection of one of the candidate elements of the second structural description which corresponds to a transformation of the element of the first structural description to the second structural description, for assignment of an attribute value of the second structural description;

means for displaying a plurality of icons representing sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed;

means for displaying the element of the first structural description;

means for displaying an attribute list of the element of the first structural description;

means for inputting a user input of a selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed; and

an input processing means for processing the user input of the selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed, when the user inputs the selection of sources for obtaining the attribute value to be assigned to the attribute of the first selection which is being displayed.

**73.** An apparatus according to Claim 72, wherein the input processing means further comprises:

means for assigning a null value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates no source is to be used.

**74.** An apparatus according to Claim 72, wherein the input processing means further comprises:

means for assigning a system value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a system value is to be used.

**75.** An apparatus according to Claim 72, wherein the input processing means further comprises:

means for assigning a first structural description attribute value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a first structural description attribute value source is to be used.

**76.** An apparatus according to Claim 72, wherein the input processing means further comprises:

an assigning means for assigning a first structural description content value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a first struc-

tural description content value source is to be used.

77. An apparatus according to Claim 72, wherein the input processing means further comprises:

means for assigning a user input value to the attribute of the first selection which is being displayed, when the selection of sources which has been input by the user indicates a user input value source is to be used.

78. An apparatus according to Claim 77, wherein the assigning means comprises:

means for displaying a text input area for the user to input a value to be assigned;
means for inputting the value entered by the user in the text input area; and
means for assigning the value input by the user to the attribute of the first selection which is being displayed.

79. An apparatus according to Claim 56, wherein the list displaying means further comprises:

means for displaying a candidate for requesting removal of the element for transformation of the first structural description in the transformation; and
means for displaying a candidate for requesting ignoring of the element for transformation of the first structural description in the transformation.

80. A computer program product including a computer readable medium for providing a graphical user interface for creating a mapping of a first structural description to a second structural description, comprising:

element displaying means for displaying an element for transformation of a first structural description;
list displaying means for displaying a list of candidate elements of a second structural description;
user inputting means for inputting, from a user, a first selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and
storing means for storing the correspondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description as a rule.

81. A computer program product according to Claim 80, wherein the first structural description includes ISO/IEC 9070 public identifier format, the second structural description includes an operating system file name format, and the element displaying means comprises:

means for displaying the element for transformation which includes an element of the ISO/IEC 9070 public identifier format; and
the list displaying means comprises:
means for displaying the list of candidate elements which includes a list of operating system file name candidate elements.

82. A computer program product according to Claim 80, wherein the first structural description includes a first database variable name format, the second structural description includes a second database variable name format, and the element displaying means comprises:

means for displaying the element for transformation which includes an element of the first database variable name format; and
the list displaying means comprises:
means for displaying the list of candidate elements which includes a list of second database variable name format candidate elements.

83. A computer program product according to Claim 80, further comprising:

means for obtaining the stored rule;
means for displaying the element for transformation of the first structural description;
means for displaying the first selected element of the second structural description which defines a corre-

spondence between the element for transformation of the first structural description and the first selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description;

means for displaying the list of candidate elements of the second structural description;

means for inputting, from the user, a second selection of one of the candidate elements of the second structural description which defines a correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description for a transformation of the element of the first structural description to the second structural description; and

means for storing the correspondence between the element for transformation of the first structural description and the second selection of one of the candidate elements of the second structural description as a rule.

**84.** A computer program product according to Claim 80, further comprising:

means for displaying an icon for the user to input a request to clear the first selection which is being displayed;

means for inputting the request to clear the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed; and

means for clearing the first selection which is being displayed, when the user inputs the request to clear the first selection which is being displayed.

**85.** A computer program product according to Claim 80, wherein the first structural description includes a first markup language, the second structural description includes a second markup language, and the element displaying means comprises:

means for displaying the element for transformation which includes an element of the first markup language; and

the list displaying means comprises:

means for displaying the list of candidate elements which includes a list of second markup language candidate elements.

**86.** A computer program product according to Claim 85, wherein the first markup language includes SGML, the second markup language includes HTML, and the element displaying means further comprises:

means for displaying the element for transformation which includes an element of SGML; and

the list displaying means further comprises:

means for displaying the list of candidate elements which includes a list of HTML candidate elements.

```
22 <!--    sample1.dtd   >      -->
24 <!ELEMENT t - - (t1?, t2?, t3?, t4?) ->
26 <!ELEMENT t1 - - CDATA>
28 <!ATTLIST t1 name CDATA #REQUIRED>
30 <!ELEMENT t2 - - CDATA>
32 <!ELEMENT t3 - - CDATA>
34 <!ELEMENT t4 - - CDATA>
```

Fig. 1A

```
42 t--> <html><title>Title</title>
44 t1--><H3><A NAME="the source is t1's name">
46 t2--> <P>
48 t3--> <P>
50 t4--> <P><A HREF="# the source is t1's name">
```

Fig. 1B

```
60 <!DOCTYPE t system "sample1.dtd">
62 <t>
64 <t1 name="hilarry">1.  Hi Larry</t1>
66 <t2> This is the most fun I have ever had.</t2>
68 <t3> It must be great for you as well.</t3>
70 <t4> (Back to the Hi Larry greeting.)</t4>
72 </t>
```

Fig. 1C

```
80 <DOCTYPE HTML Public "-//W3C//DTD HTML 3.2 Final//EN">
82 <html>
84 <title>Title</title>
86 <H3><A name="hilarry">1.  Hi Larry</A></H3>
88 <P> This is the most fun I have ever had.</P>
90 <P> It must be great for you as well.</P>
92 <P><A HREF="#hilarry"> (Back to the Hi Larry greeting.)</A></P>
94 </html>
```

Fig. 1D

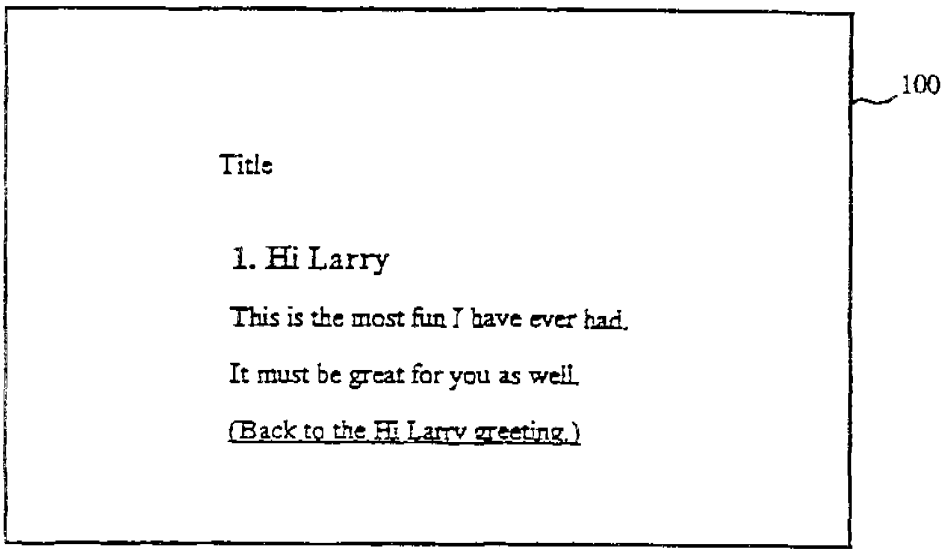

100
Title
1. Hi Larry
This is the most fun I have ever had.
It must be great for you as well.
(Back to the Hi Larry greeting.)

Fig. 2

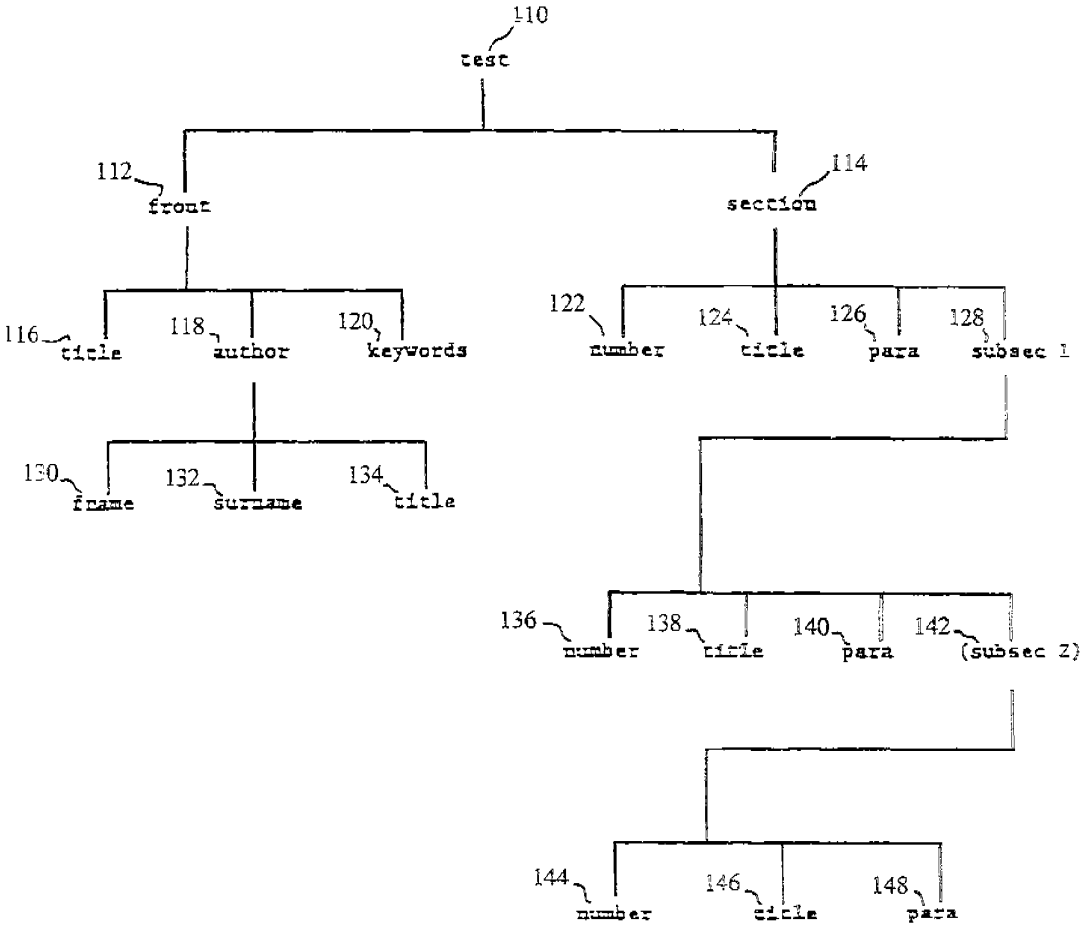

110
test
112
front
114
section
116
title
118
author
120
keywords
122
number
124
title
126
para
128
subsec 1
130
fname
132
surname
134
title
136
number
138
title
140
para
142
(subsec 2)
144
number
146
title
148
para

Fig. 3A

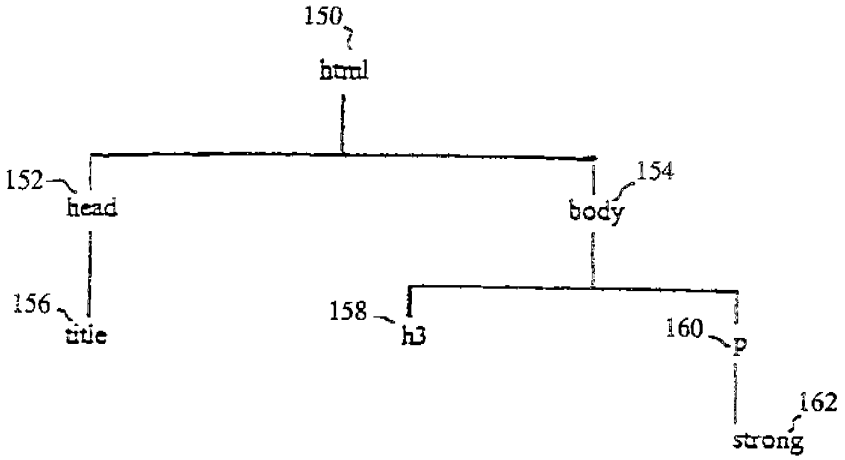



Fig. 3B

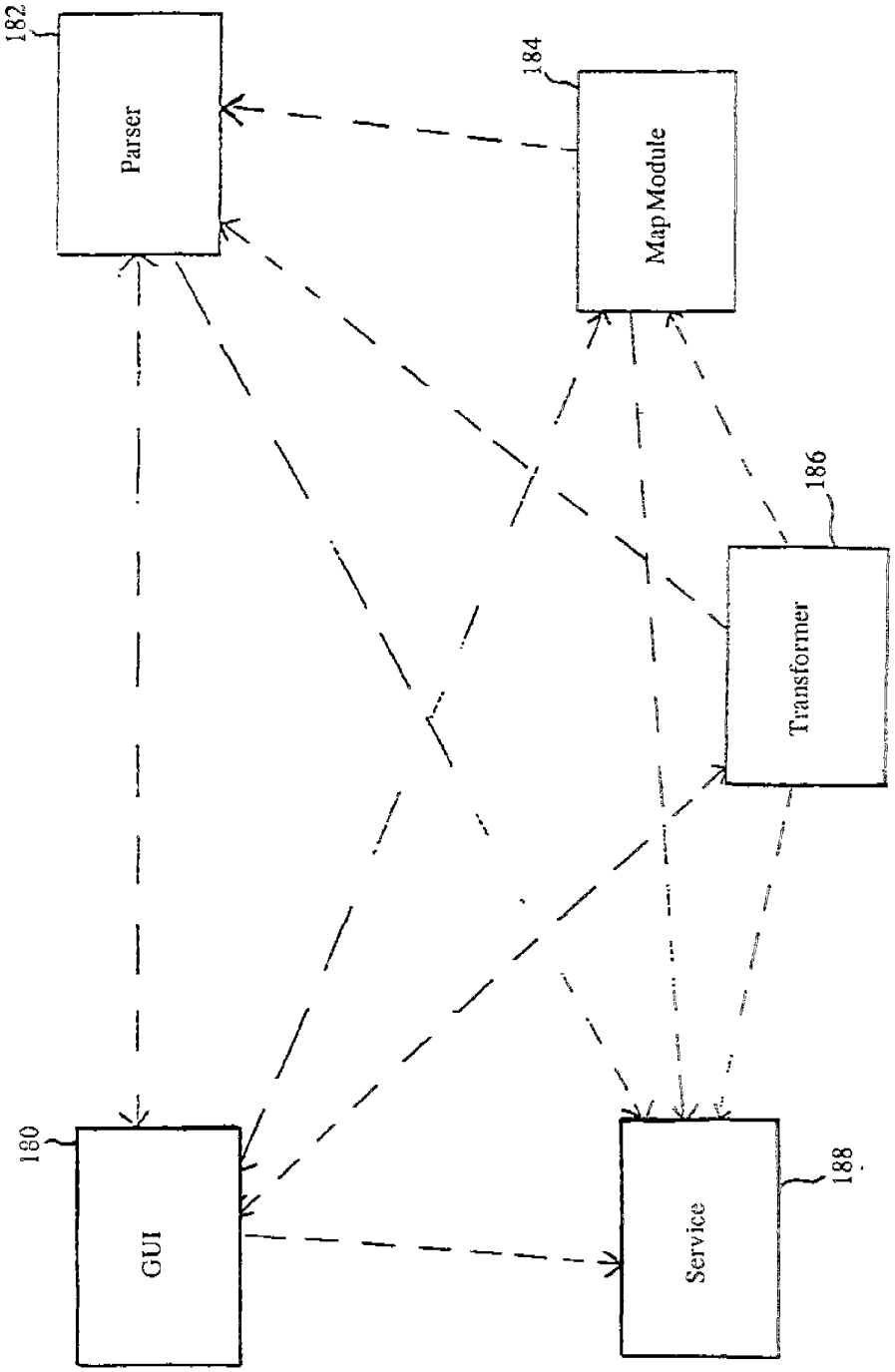

182
Parser
184
Map Module
186
Transformer
180
GUI
188
Service

Fig. 4

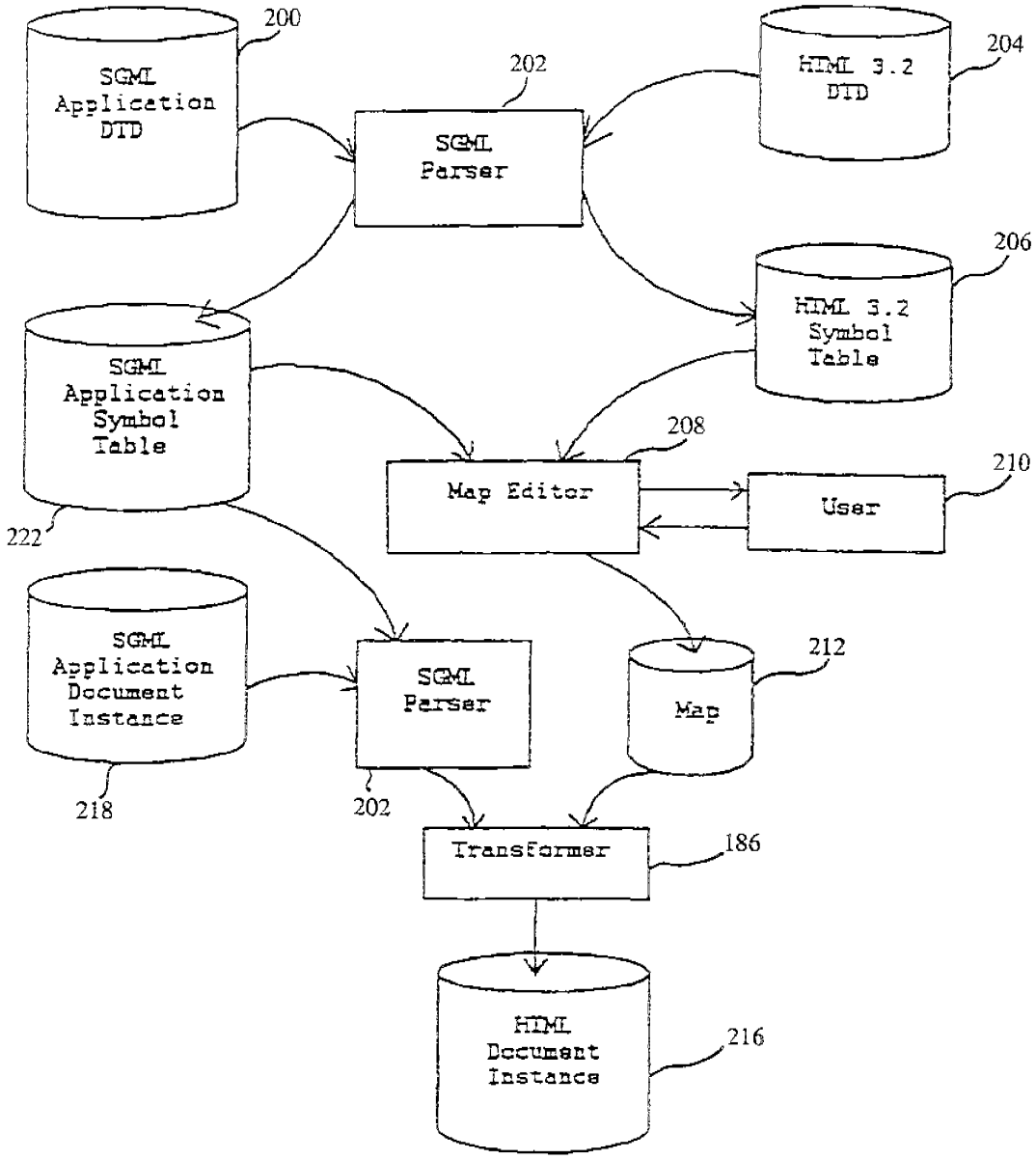

SGML Application DTD
200
SGML Parser
202
HTML 3.2 DTD
204
HTML 3.2 Symbol Table
206
SGML Application Symbol Table
222
Map Editor
208
User
210
SGML Application Document Instance
218
SGML Parser
202
Map
212
Transformer
186
HTML Document Instance
216

Fig. 5

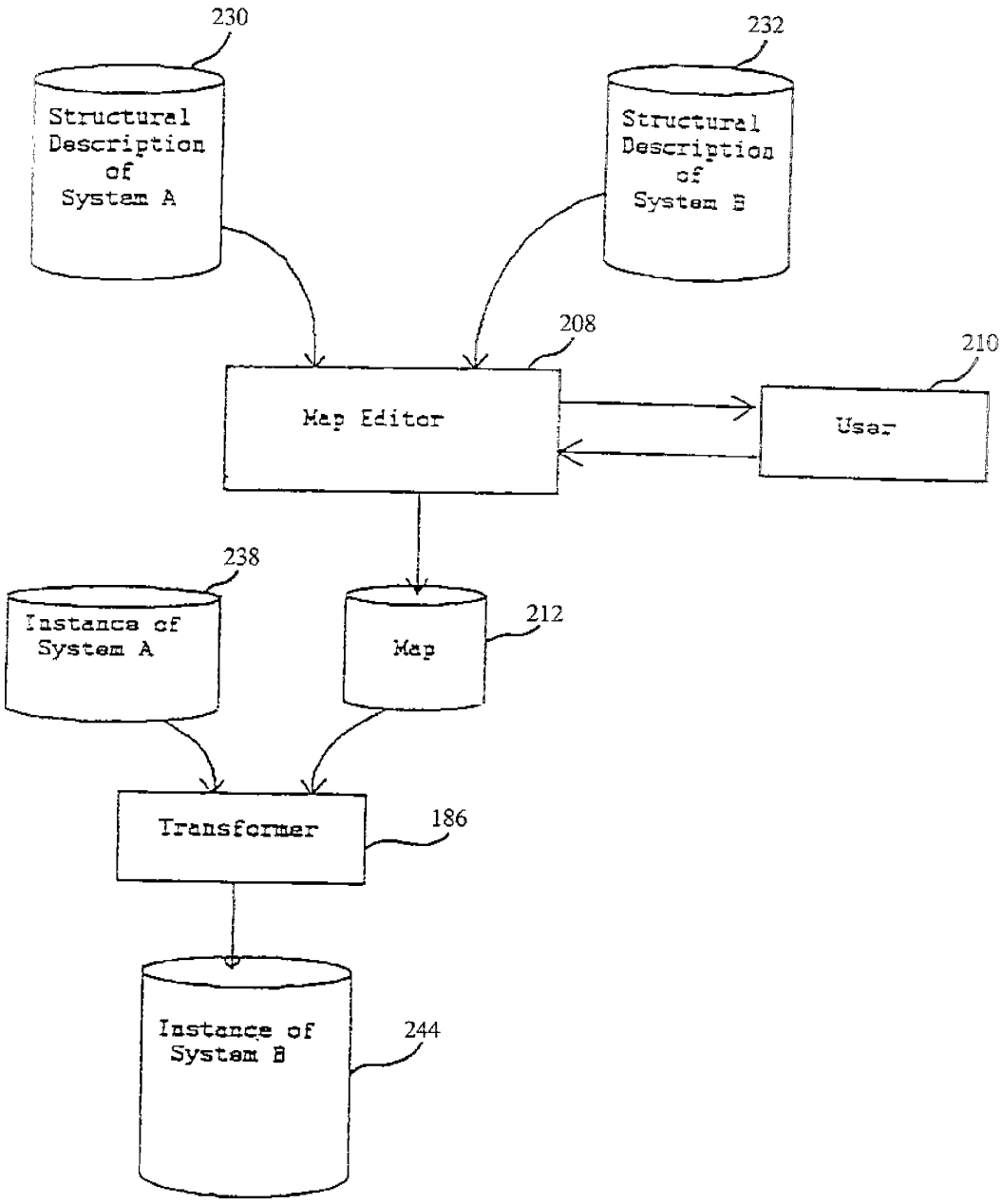

230
Structural Description of System A
232
Structural Description of System B
208
Map Editor
210
User
238
Instance of System A
212
Map
Transformer
186
Instance of System B
244

Fig. 6A

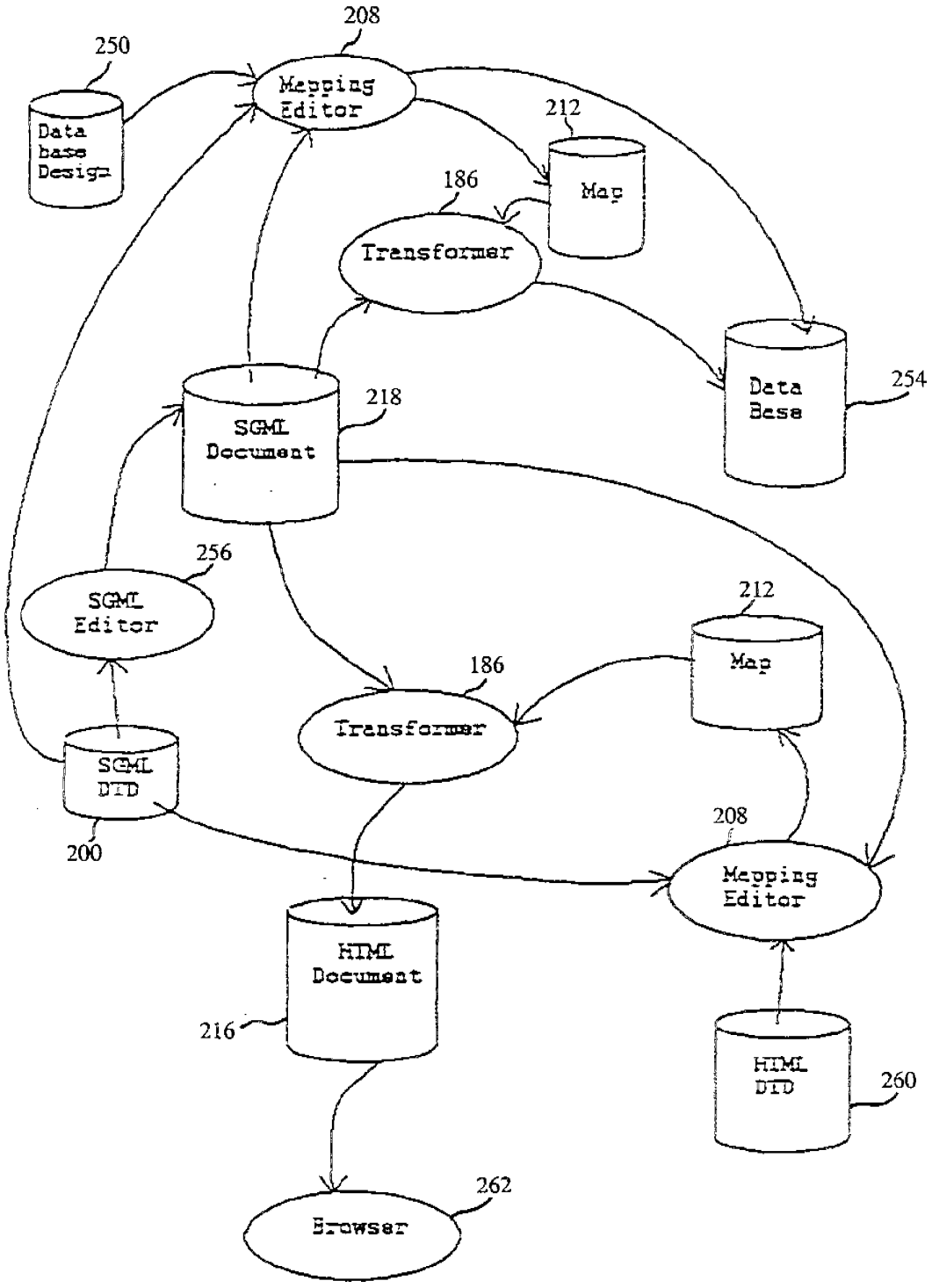

208
250
Mapping Editor
Data base Design
212
Map
186
Transformer
Data Base
254
218
SGML Document
256
SGML Editor
212
Map
186
Transformer
SGML DTD
200
208
Mapping Editor
HTML Document
216
HTML DTD
260
262
Browser

Fig. 6B

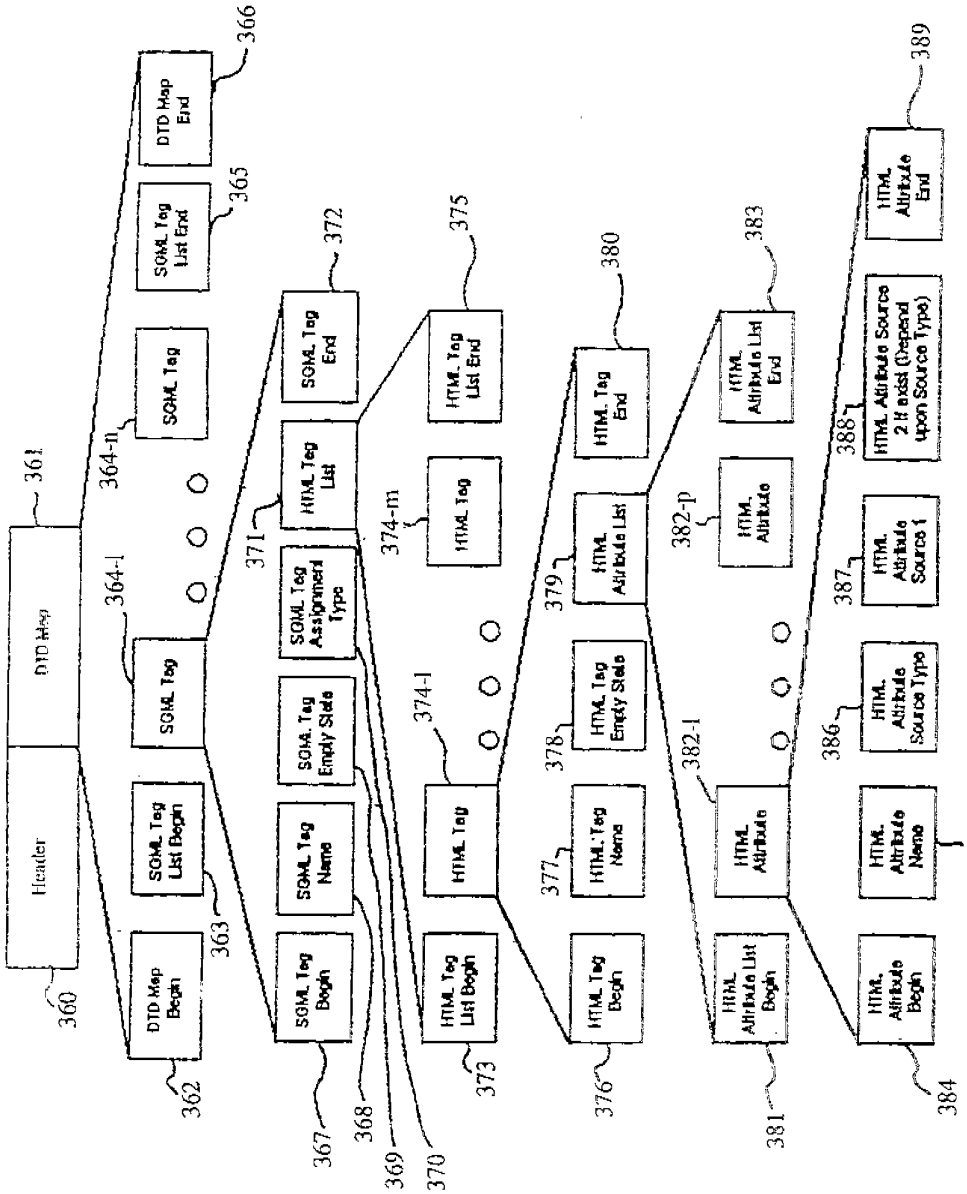

Header
DTD Map
DTD Map Begin
SGML Tag List Begin
SGML Tag
SGML Tag List End
DTD Map End
SGML Tag Begin
SGML Tag Name
SGML Tag Empty Slash
SGML Tag Assignment Type
HTML Tag List
SGML Tag End
HTML Tag List Begin
HTML Tag
HTML Tag End
HTML Tag Begin
HTML Tag Name
HTML Tag Empty Slash
HTML Attribute List
HTML Attribute List End
HTML Attribute List Begin
HTML Attribute
HTML Attribute Name
HTML Attribute Source Type
HTML Attribute Source 1
HTML Attribute Source 2 If exist (Depend upon Source Type)
HTML Attribute End
360
361
362
363
364-1
364-n
365
366
367
368
369
370
371
372
373
374-1
374-m
375
376
377
378
379
380
381
382-1
382-p
383
384
385
386
387
388
389

Fig. 7

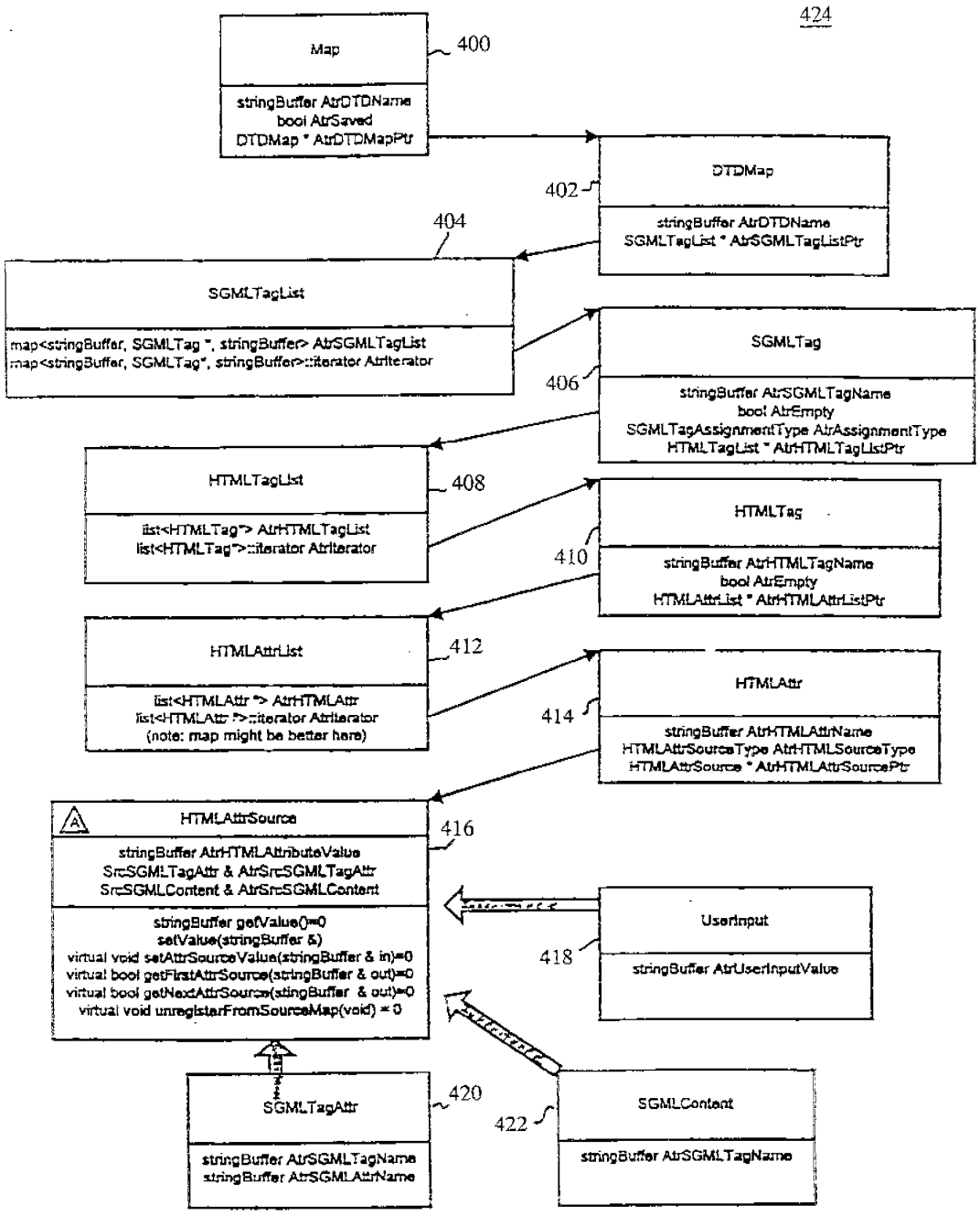

424
Map
400
stringBuffer AtrDTDName
bool AtrSaved
DTDMap * AtrDTDMapPtr
DTDMap
402
stringBuffer AtrDTDName
SGMLTagList * AtrSGMLTagListPtr
404
SGMLTagList
map<stringBuffer, SGMLTag *, stringBuffer> AtrSGMLTagList
map<stringBuffer, SGMLTag*, stringBuffer>::iterator AtrIterator
SGMLTag
406
stringBuffer AtrSGMLTagName
bool AtrEmpty
SGMLTagAssignmentType AtrAssignmentType
HTMLTagList * AtrHTMLTagListPtr
HTMLTagList
408
list<HTMLTag*> AtrHTMLTagList
list<HTMLTag*>::iterator AtrIterator
HTMLTag
410
stringBuffer AtrHTMLTagName
bool AtrEmpty
HTMLAttrList * AtrHTMLAttrListPtr
HTMLAttrList
412
list<HTMLAttr *> AtrHTMLAttr
list<HTMLAttr *>::iterator AtrIterator
(note: map might be better here)
HTMLAttr
414
stringBuffer AtrHTMLAttrName
HTMLAttrSourceType AtrHTMLSourceType
HTMLAttrSource * AtrHTMLAttrSourcePtr
HTMLAttrSource
416
stringBuffer AtrHTMLAttributeValue
SrcSGMLTagAttr & AtrSrcSGMLTagAttr
SrcSGMLContent & AtrSrcSGMLContent
stringBuffer getValue()=0
setValue(stringBuffer &)
virtual void setAttrSourceValue(stringBuffer & in)=0
virtual bool getFirstAttrSource(stringBuffer & out)=0
virtual bool getNextAttrSource(stringBuffer & out)=0
virtual void unregisterFromSourceMap(void) = 0
UserInput
418
stringBuffer AtrUserInputValue
SGMLTagAttr
420
stringBuffer AtrSGMLTagName
stringBuffer AtrSGMLAttrName
SGMLContent
422
stringBuffer AtrSGMLTagName

Fig. 8A

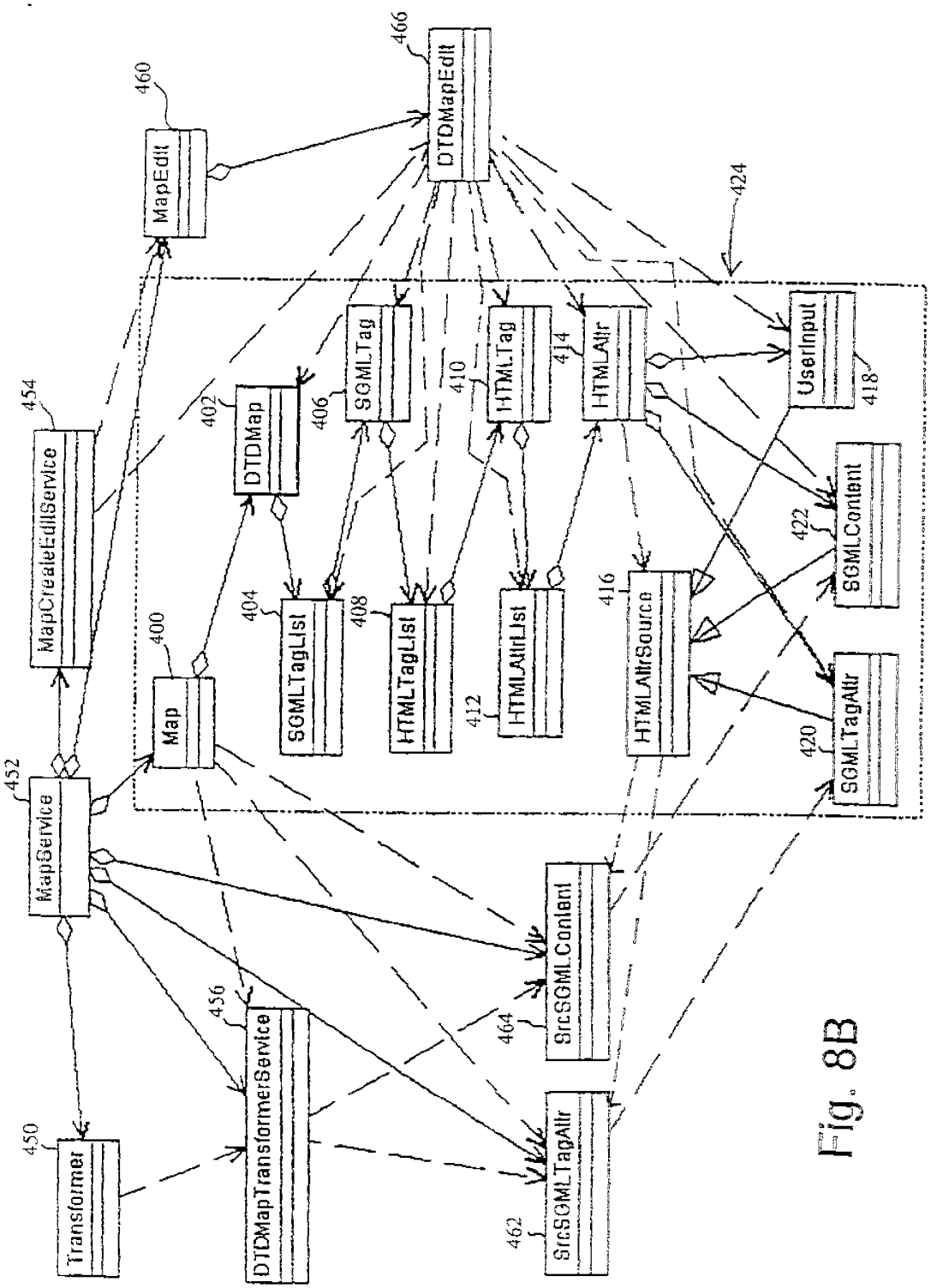

MapService
MapCreateEditService
MapEdit
DTDMapEdit
Map
DTDMap
SGMLTagList
HTMLTagList
HTMLAttrList
HTMLAttrSource
SGMLTag
HTMLTag
HTMLAttr
UserInput
SGMLContent
SGMLTagAttr
Transformer
DTDMapTransformerService
SrcSGMLContent
SrcSGMLTagAttr
400
402
404
406
408
410
412
414
416
418
420
422
424
450
452
454
456
460
462
464
466

Fig. 8B

462

| SrcSGMLTagAttr |
| --- |
| stringBuffer AtrCurrentSGMLTagName<br>stringBuffer AtrCurrentAttribute<br>stringBuffer AtrCurrentHTMLAttributeSourceValue<br>stringBuffer AtrTagAttrKey<br>map<stringBuffer, list<SGMLTagAttr *>, stringBuffer> AtrKeyOfSGMLTagAndAttribute |
| void registerSGMLTagNameAndAttributeName(SGMLTagAttr *)<br>void unregisterTagAttrKeyAndMapEntry(SGMLTagAttr *)<br>void setValueForAttributeOfTag<br>(stringBuffer & Tag, stringBuffer & Attribute, stringBuffer & value)<br>void reset(void) |

Fig. 8C(1)

464

| SrcSGMLContent |
| --- |
| stringBuffer AtrCurrentSGMLTagName<br>stringBuffer AtrCurrentHTMLAttributeSourceVal<br>map<stringBuffer, list<SGMLContent *>, stringBuffer> AtrTableWithSGMLTagKey |
| void registerSGMLTagName(SGMLContent *)<br>void unregisterSGMLTagName(SGML Content *)<br>void setValueForTag(stringBuffer & Tag, stringBuffer & value)<br>void reset(void) |

Fig. 8C(2)

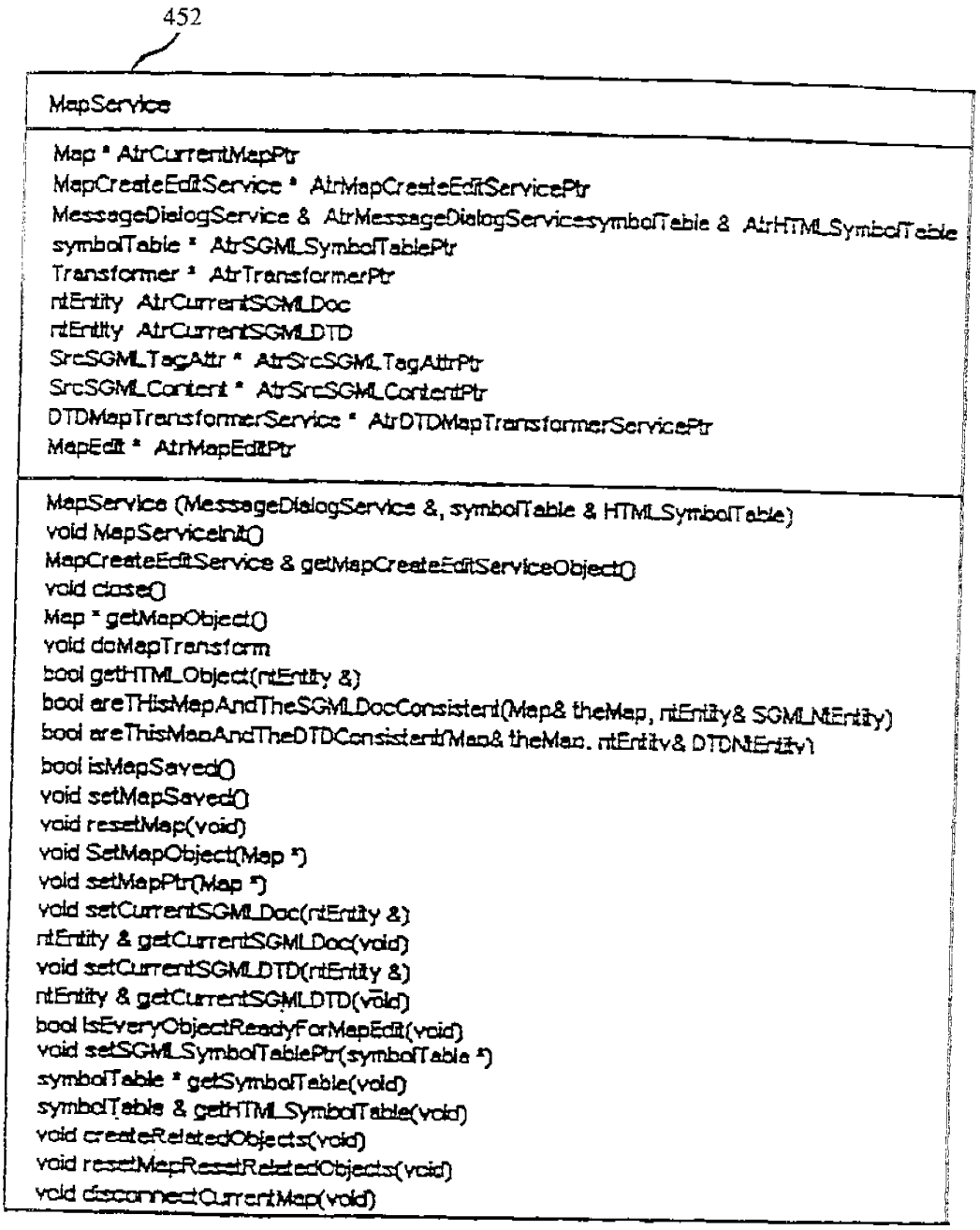

452
MapService
Map * AtrCurrentMapPtr
MapCreateEditService * AtrMapCreateEditServicePtr
MessageDialogService & AtrMessageDialogServicesymbolTable & AtrHTMLSymbolTable
symbolTable * AtrSGMLSymbolTablePtr
Transformer * AtrTransformerPtr
ntEntity AtrCurrentSGMLDoc
ntEntity AtrCurrentSGMLDTD
SrcSGMLTagAttr * AtrSrcSGMLTagAttrPtr
SrcSGMLContent * AtrSrcSGMLContentPtr
DTDMapTransformerService * AtrDTDMapTransformerServicePtr
MapEdit * AtrMapEditPtr
MapService (MessageDialogService &, symbolTable & HTMLSymbolTable)
void MapServiceInit()
MapCreateEditService & getMapCreateEditServiceObject()
void close()
Map * getMapObject()
void doMapTransform
bool getHTML_Object(ntEntity &)
bool areThisMapAndTheSGMLDocConsistent(Map& theMap, ntEntity& SGMLNtEntity)
bool areThisMapAndTheDTDConsistent(Map& theMap, ntEntity& DTDNtEntity)
bool isMapSaved()
void setMapSaved()
void resetMap(void)
void SetMapObject(Map *)
void setMapPtr(Map *)
void setCurrentSGMLDoc(ntEntity &)
ntEntity & getCurrentSGMLDoc(void)
void setCurrentSGMLDTD(ntEntity &)
ntEntity & getCurrentSGMLDTD(void)
bool IsEveryObjectReadyForMapEdit(void)
void setSGMLSymbolTablePtr(symbolTable *)
symbolTable * getSymbolTable(void)
symbolTable & getHTMLSymbolTable(void)
void createRelatedObjects(void)
void resetMapResetRelatedObjects(void)
void disconnectCurrentMap(void)

Fig. 8C(3)

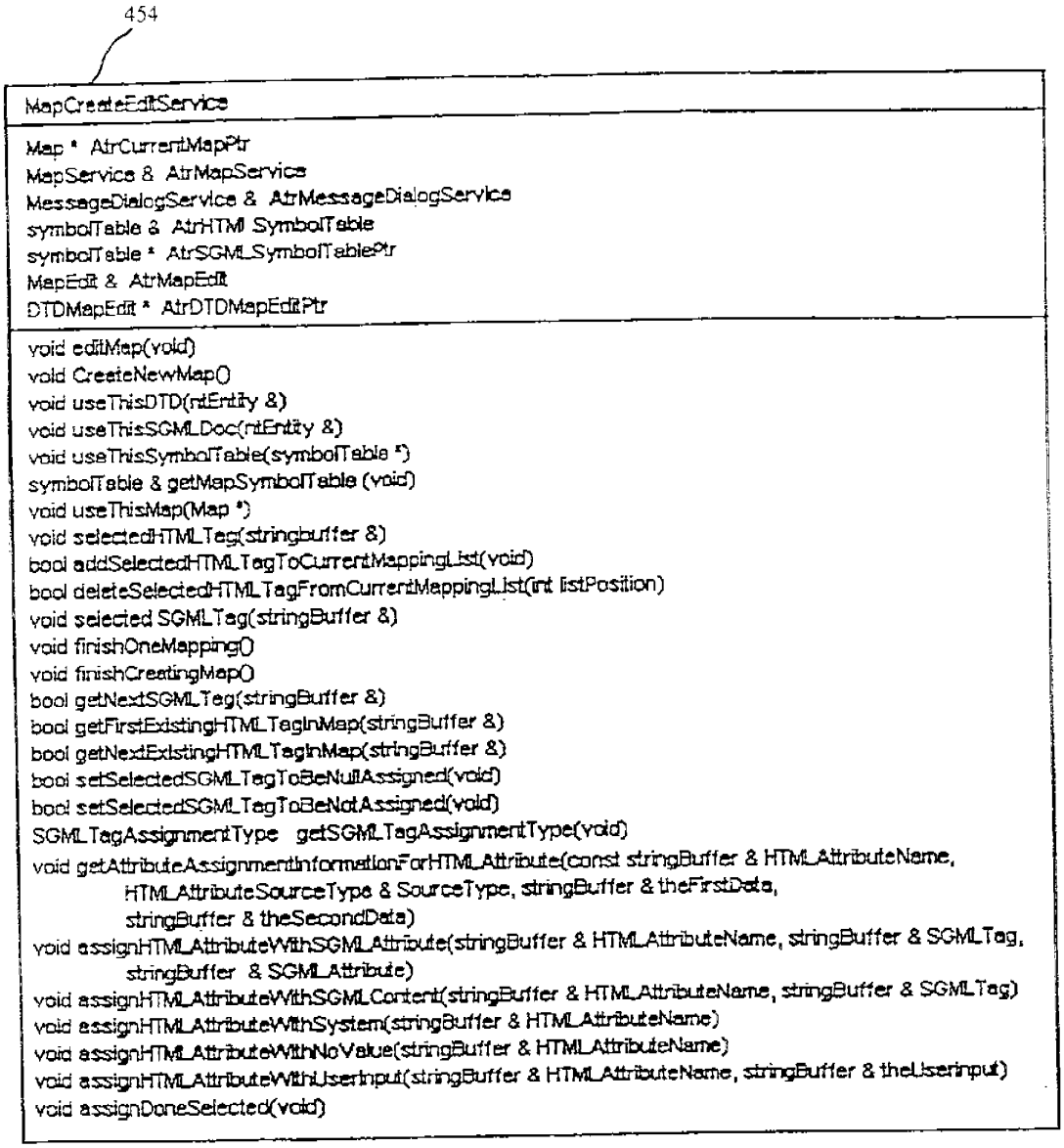

454
MapCreateEditService
Map * AtrCurrentMapPtr
MapService & AtrMapService
MessageDialogService & AtrMessageDialogService
symbolTable & AtrHTMl SymbolTable
symbolTable * AtrSGMLSymbolTablePtr
MapEdit & AtrMapEdit
DTDMapEdit * AtrDTDMapEditPtr
void editMap(void)
void CreateNewMap()
void useThisDTD(ntEntity &)
void useThisSGMLDoc(ntEntity &)
void useThisSymbolTable(symbolTable *)
symbolTable & getMapSymbolTable (void)
void useThisMap(Map *)
void selectedHTMLTag(stringbuffer &)
bool addSelectedHTMLTagToCurrentMappingList(void)
bool deleteSelectedHTMLTagFromCurrentMappingList(int listPosition)
void selected SGMLTag(stringBuffer &)
void finishOneMapping()
void finishCreatingMap()
bool getNextSGMLTag(stringBuffer &)
bool getFirstExistingHTMLTagInMap(stringBuffer &)
bool getNextExistingHTMLTagInMap(stringBuffer &)
bool setSelectedSGMLTagToBeNullAssigned(void)
bool setSelectedSGMLTagToBeNotAssigned(void)
SGMLTagAssignmentType getSGMLTagAssignmentType(void)
void getAttributeAssignmentInformationForHTMLAttribute(const stringBuffer & HTMLAttributeName,
HTMLAttributeSourceType & SourceType, stringBuffer & theFirstData,
stringBuffer & theSecondData)
void assignHTMLAttributeWithSGMLAttribute(stringBuffer & HTMLAttributeName, stringBuffer & SGMLTag,
stringBuffer & SGMLAttribute)
void assignHTMLAttributeWithSGMLContent(stringBuffer & HTMLAttributeName, stringBuffer & SGMLTag)
void assignHTMLAttributeWithSystem(stringBuffer & HTMLAttributeName)
void assignHTMLAttributeWithNoValue(stringBuffer & HTMLAttributeName)
void assignHTMLAttributeWithUserInput(stringBuffer & HTMLAttributeName, stringBuffer & theUserInput)
void assignDoneSelected(void)

Fig. 8C(4)

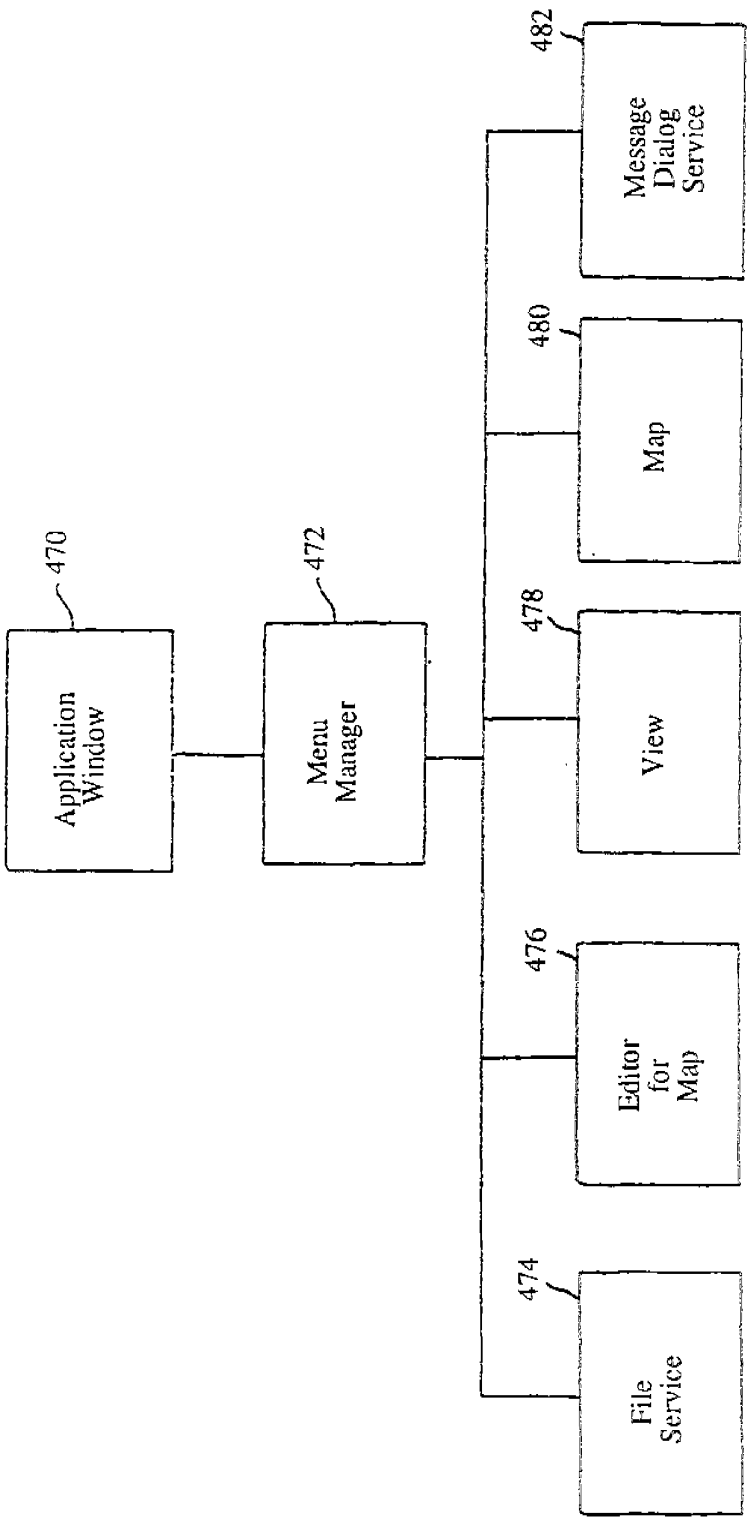

Application Window
470
Menu Manager
472
Message Dialog Service
482
Map
480
View
478
Editor for Map
476
File Service
474

Fig. 9

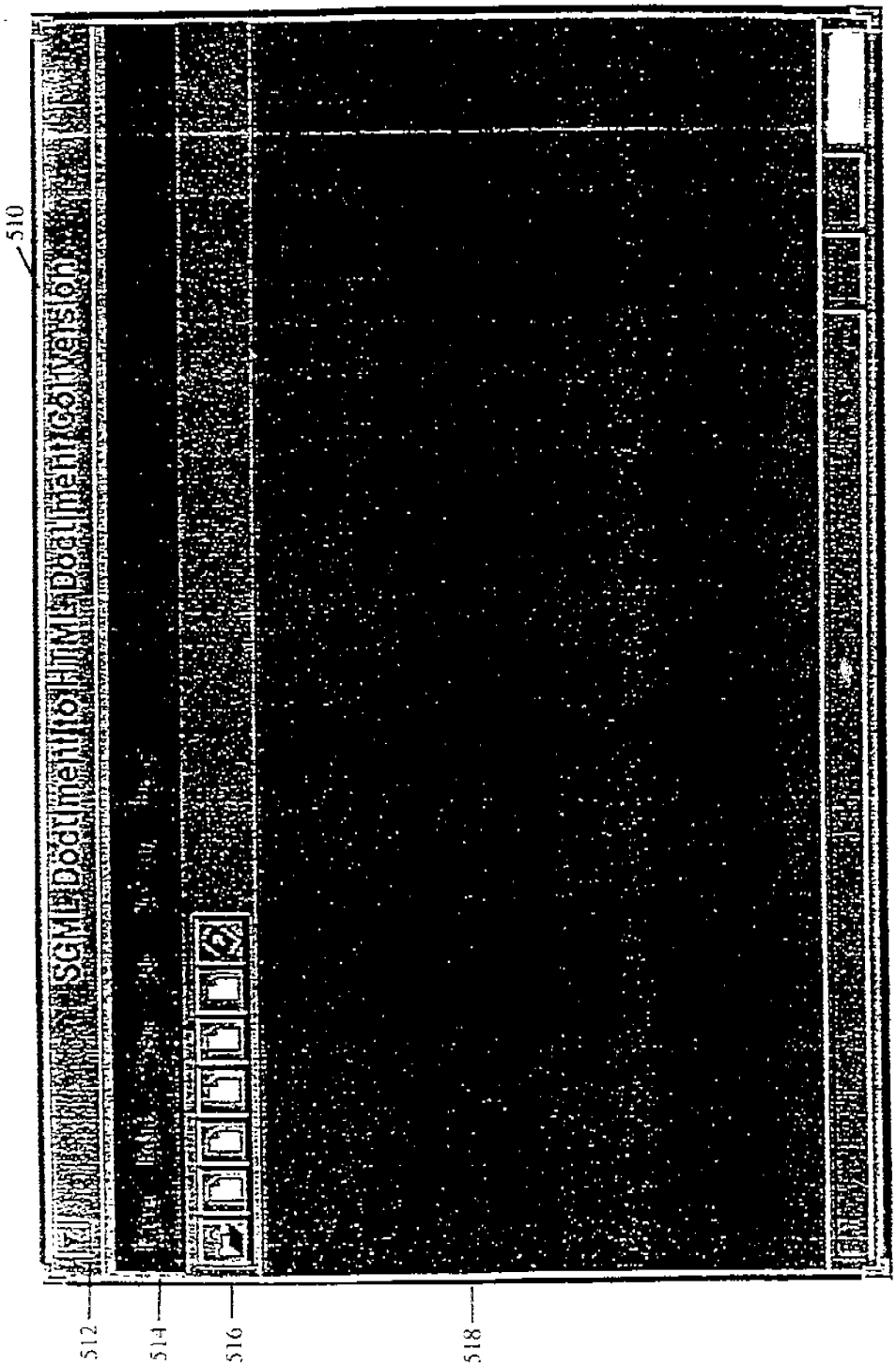

510
SGML Document to HTML Document Conversion
512
514
516
518

Fig. 10

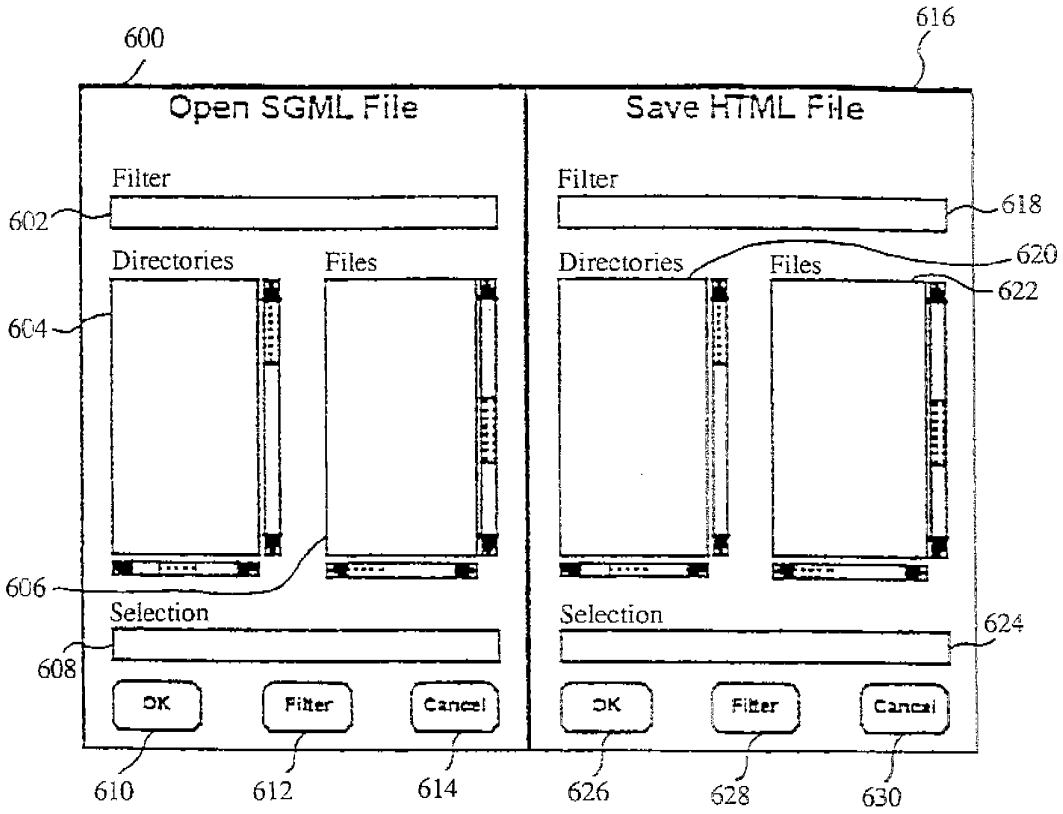

600
616
Open SGML File
Save HTML File
Filter
Filter
602
618
Directories
Files
Directories
Files
620
622
604
606
Selection
Selection
624
608
OK
Filter
Cancel
OK
Filter
Cancel
610
612
614
626
628
630

Fig. 11

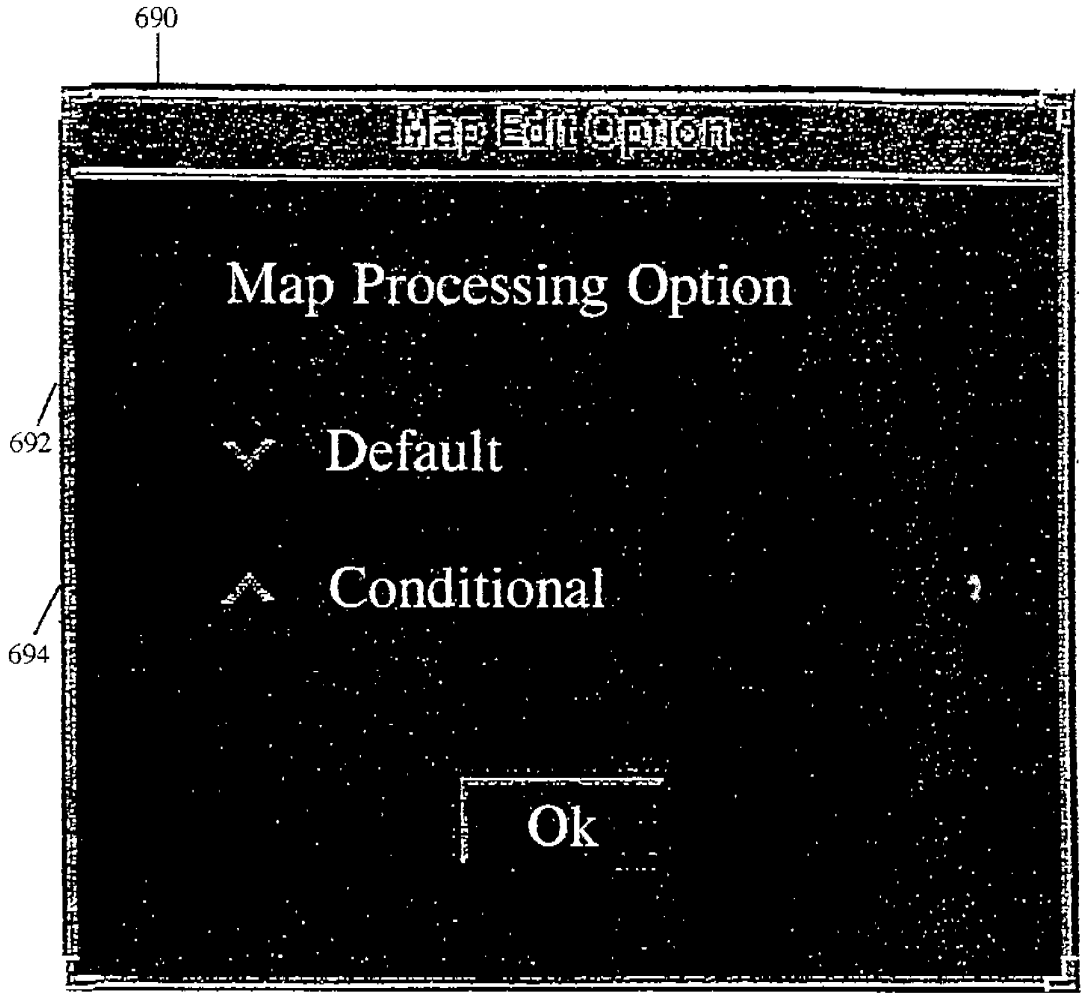

690
Map Edit Option
Map Processing Option
692
Default
Conditional
694
Ok

Fig. 12A

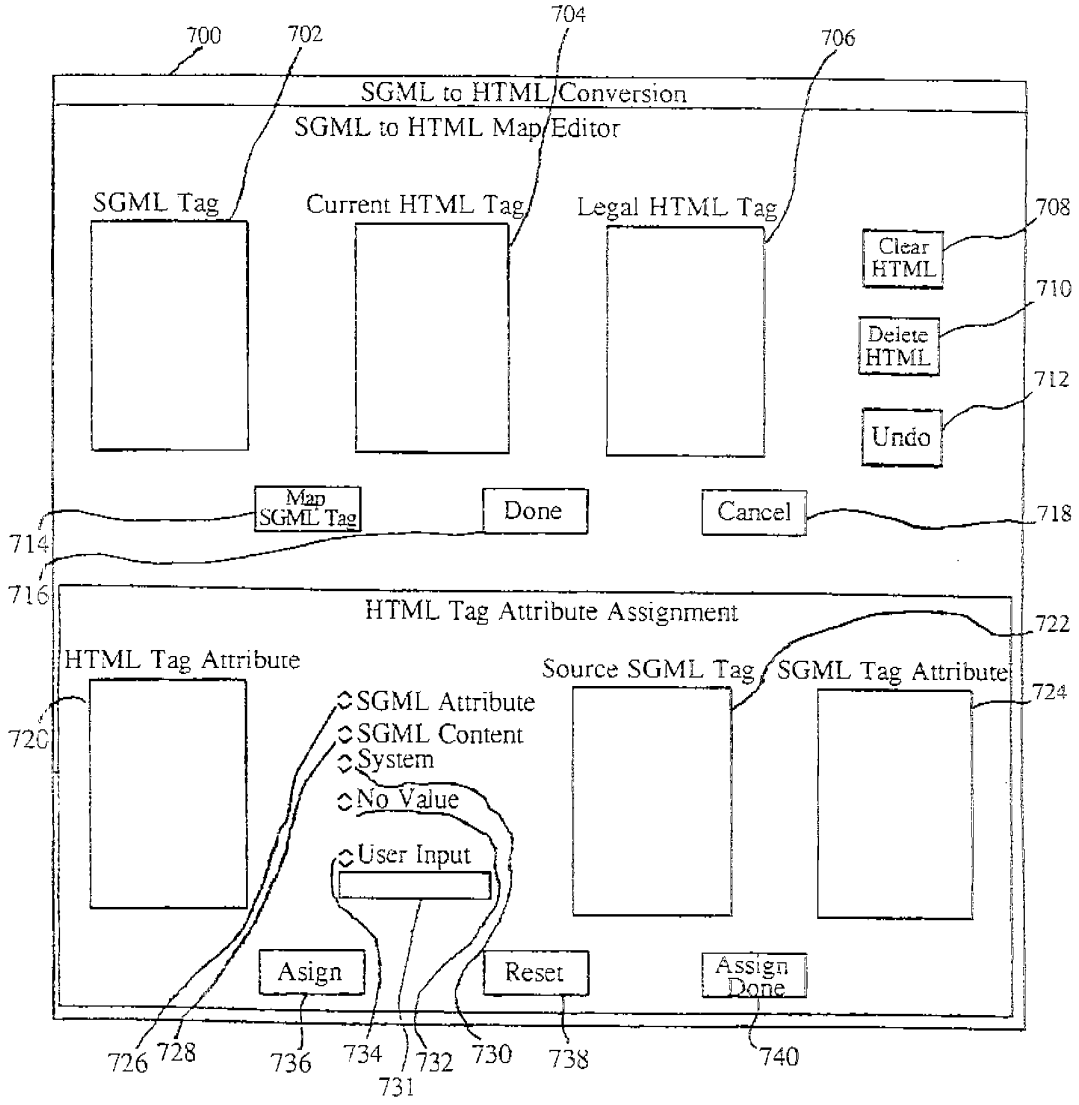

700
702
704
706
SGML to HTML Conversion
SGML to HTML Map Editor
SGML Tag
Current HTML Tag
Legal HTML Tag
Clear HTML
708
Delete HTML
710
Undo
712
Map SGML Tag
Done
Cancel
718
714
716
HTML Tag Attribute Assignment
722
HTML Tag Attribute
Source SGML Tag
SGML Tag Attribute
724
720
SGML Attribute
SGML Content
System
No Value
User Input
Asign
Reset
Assign Done
726
728
736
734
732
731
730
738
740

Fig. 12B

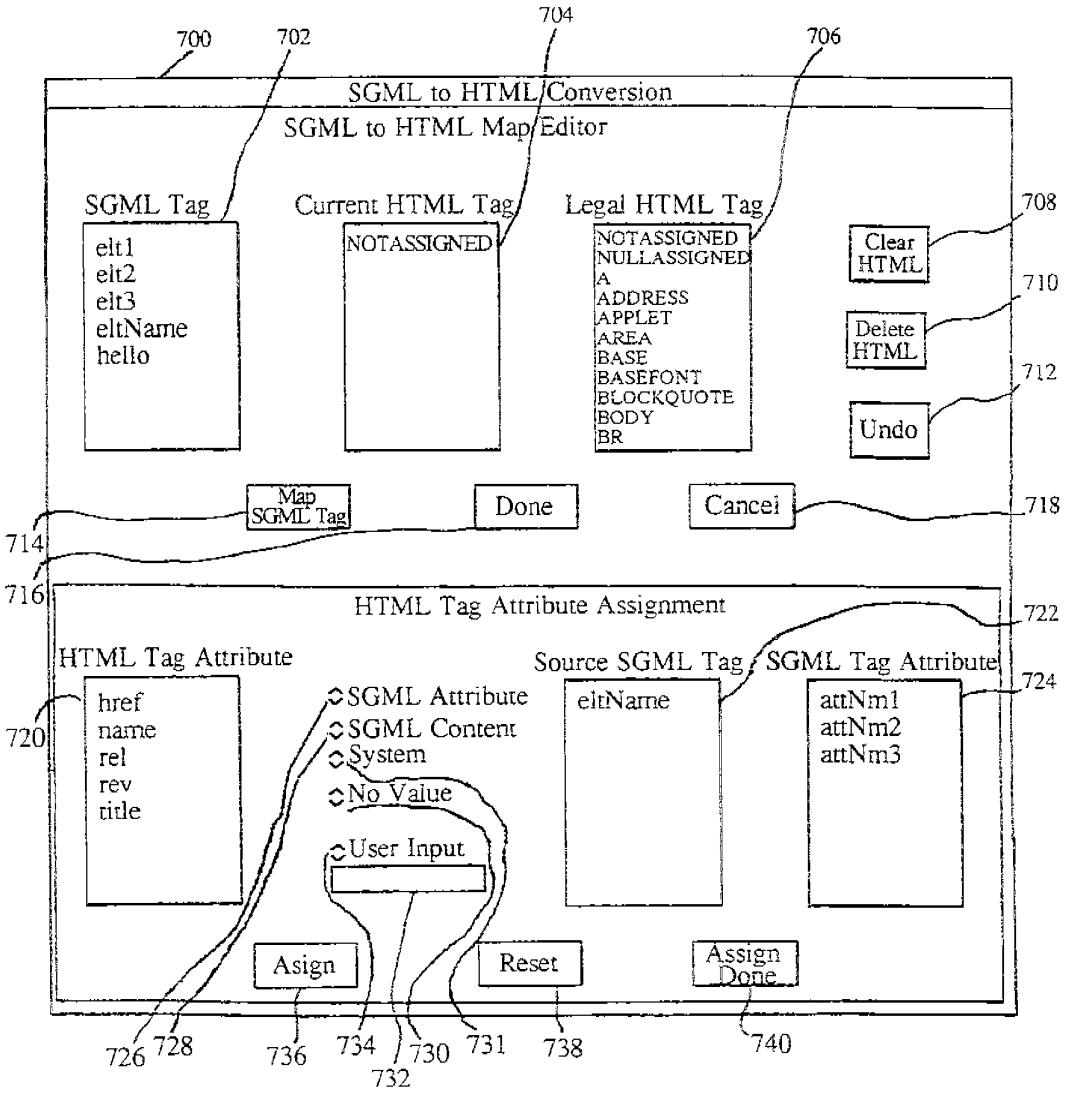

700
702
704
706
SGML to HTML Conversion
SGML to HTML Map Editor
SGML Tag
elt1
elt2
elt3
eltName
hello
Current HTML Tag
NOTASSIGNED
Legal HTML Tag
NOTASSIGNED
NULLASSIGNED
A
ADDRESS
APPLET
AREA
BASE
BASEFONT
BLOCKQUOTE
BODY
BR
Clear HTML
708
Delete HTML
710
Undo
712
Map SGML Tag
Done
Cancel
718
714
716
HTML Tag Attribute Assignment
722
HTML Tag Attribute
720
href
name
rel
rev
title
SGML Attribute
SGML Content
System
No Value
User Input
Source SGML Tag
eltName
SGML Tag Attribute
attNm1
attNm2
attNm3
724
Asign
Reset
Assign Done
726
728
736
734
730
732
731
738
740

Fig. 12C

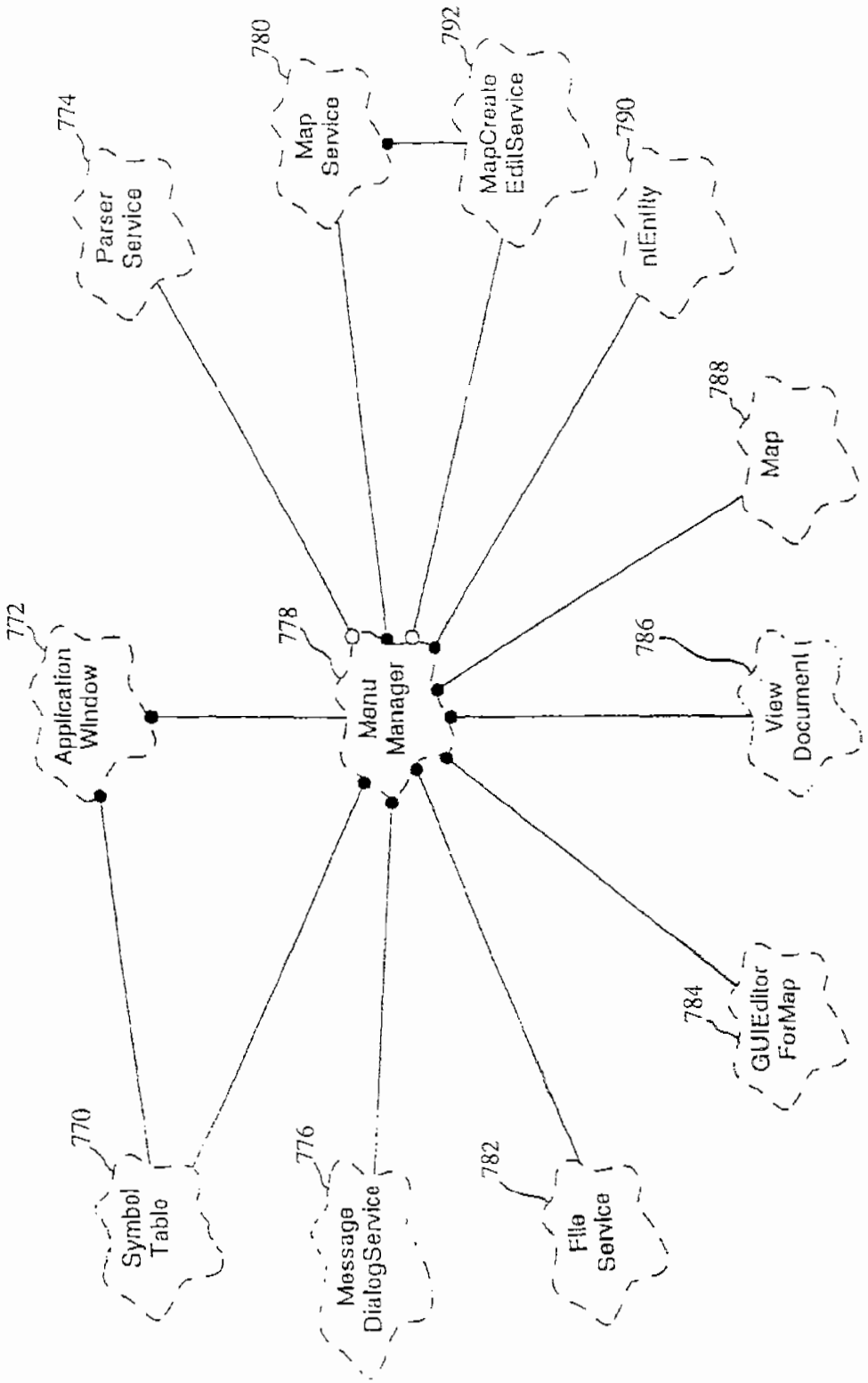

772
Application Window
778
Menu Manager
770
Symbol Table
776
Message DialogService
782
File Service
784
GUIEditor ForMap
786
View Document
788
Map
790
nEntity
792
MapCreate EditService
780
Map Service
774
Parser Service

Fig. 13

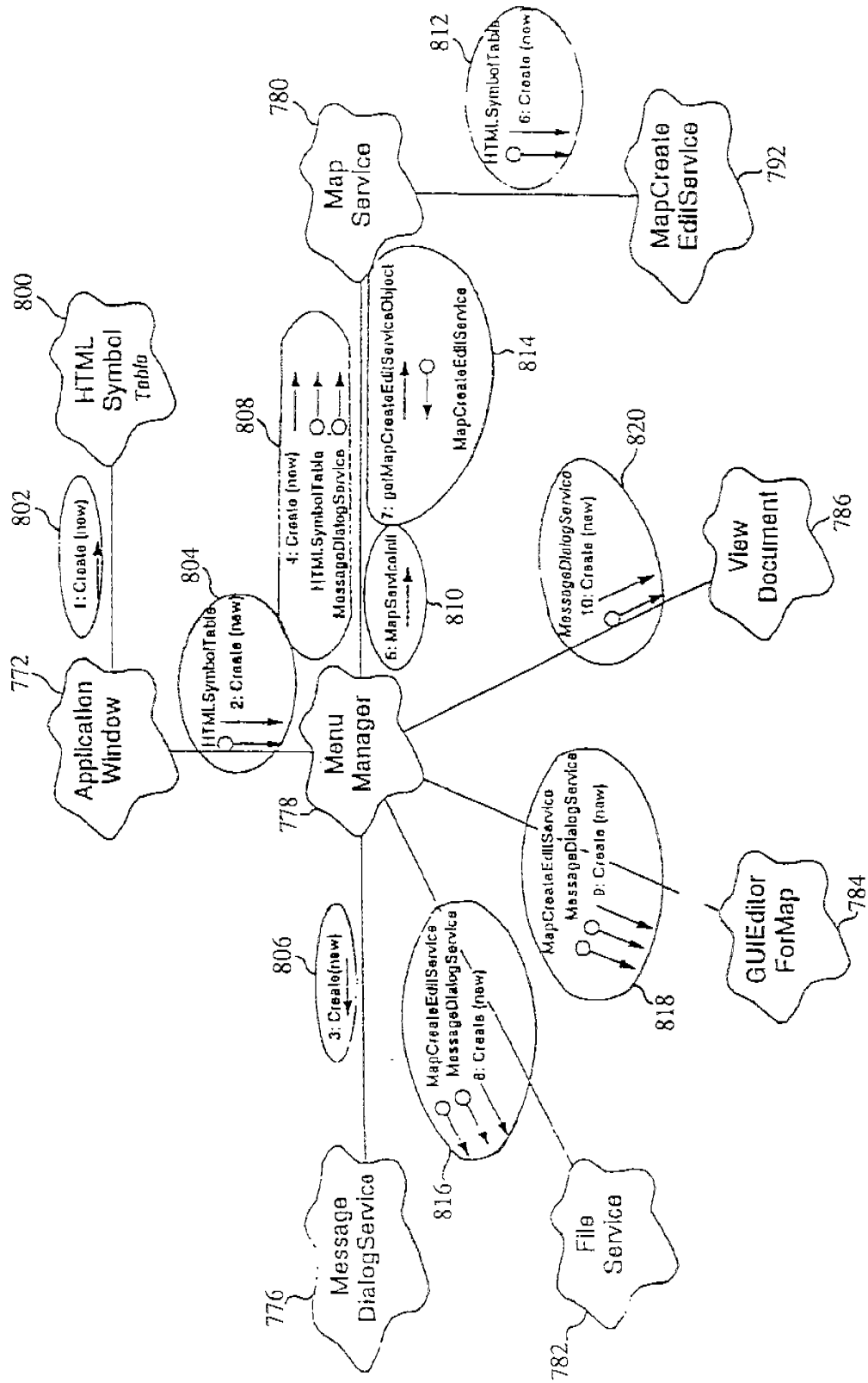

Application Window
772
HTML Symbol Table
800
1: Create (new)
802
HTMLSymbolTable
2: Create (new)
804
Menu Manager
778
4: Create (new)
HTMLSymbolTable
MessageDialogService
808
6: MapServiceInit
810
7: getMapCreateEditServiceObject
MapCreateEditService
814
Map Service
780
HTMLSymbolTable
6: Create (new)
812
MapCreate EditService
792
MessageDialogService
10: Create (new)
820
View Document
786
3: Create(new)
806
MapCreateEditService
MessageDialogService
8: Create (new)
816
MapCreateEditService
MessageDialogService
8: Create (new)
818
GUIEditor ForMap
784
Message DialogService
776
File Service
782

Fig. 14

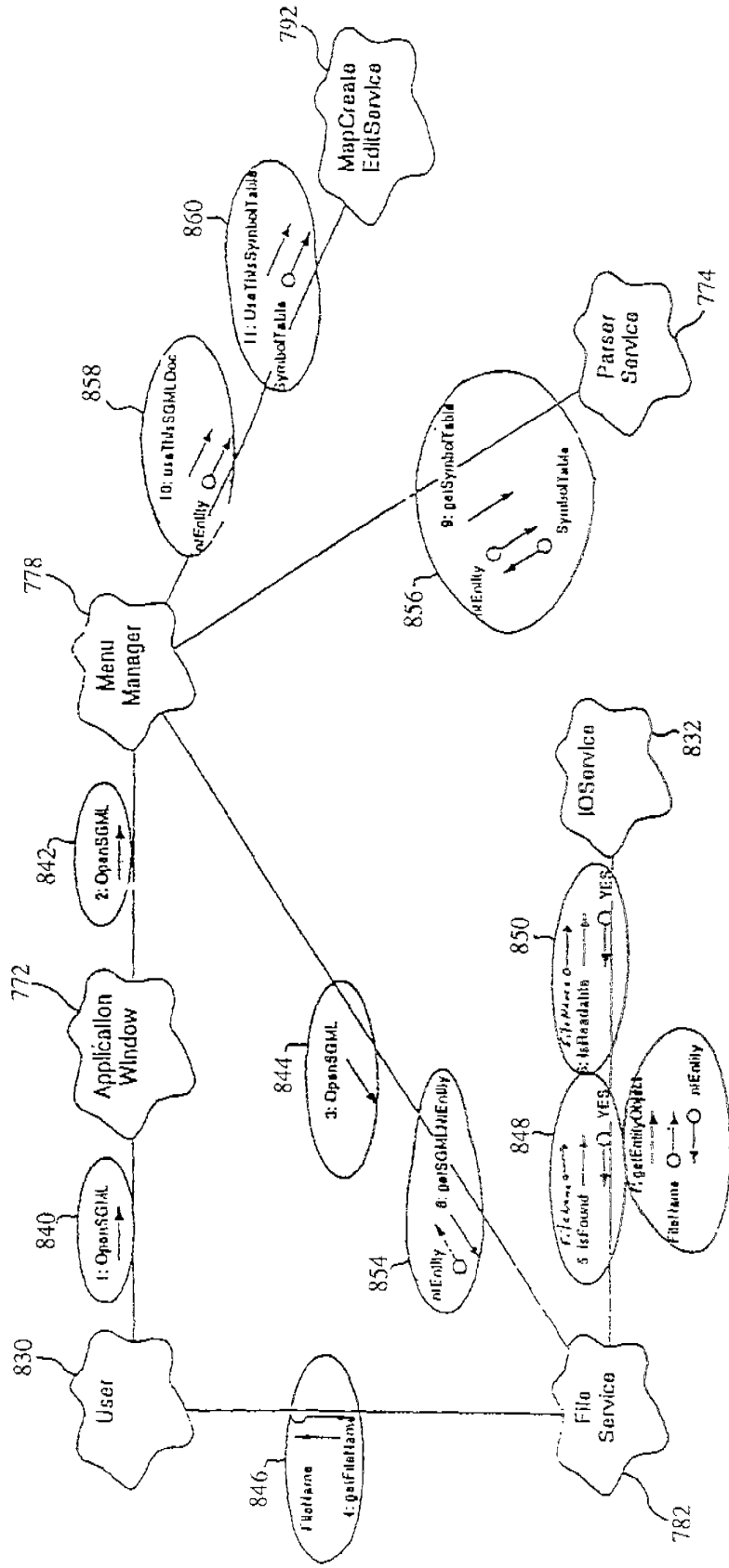

User
Application Window
Menu Manager
File Service
MapCreate EditService
Parser Service
IOService
1: OpenSGML
2: OpenSGML
3: OpenSGML
4: getFileName
5: isFound
6: isReadable
7: getEntity
8: getSGMLEntity
9: getSymbolTable
10: useThisSGMLDoc
11: UseThisSymbolTable
SymbolTable
pfEntity
FileName
YES
830
840
842
844
846
848
850
852
854
856
858
860
772
778
782
792
774
832

Fig. 15

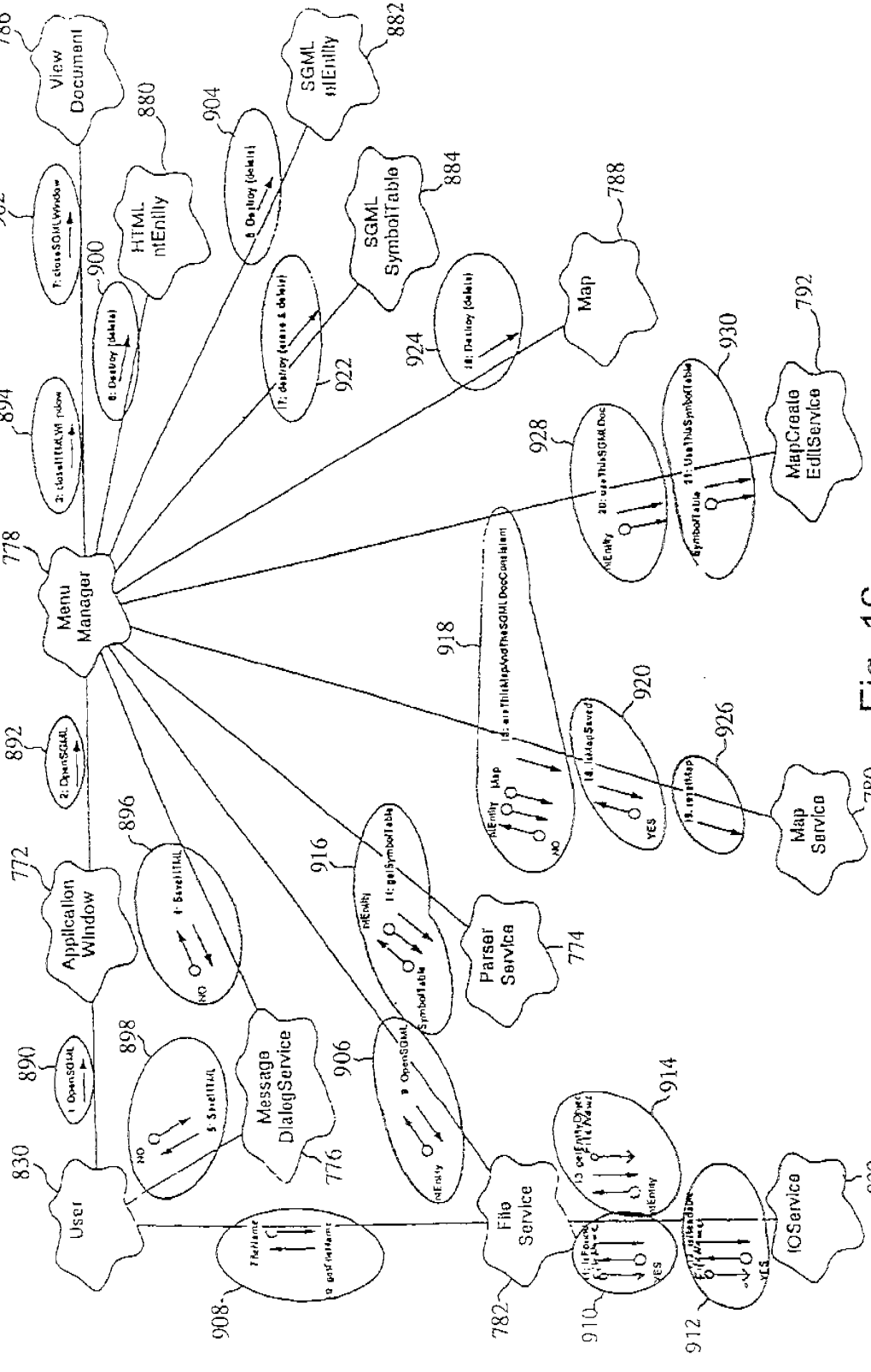

786
View Document
902
880
HTML nEntity
904
SGML nEntity
882
900
894
884
SGML SymbolTable
778
Menu Manager
922
924
788
Map
892
772
Application Window
896
916
918
928
920
926
930
792
MapCreate EditService
Map Service
780
Parser Service
774
898
906
914
890
Message DialogService
776
830
User
908
782
File Service
910
912
IOService
832

Fig. 16

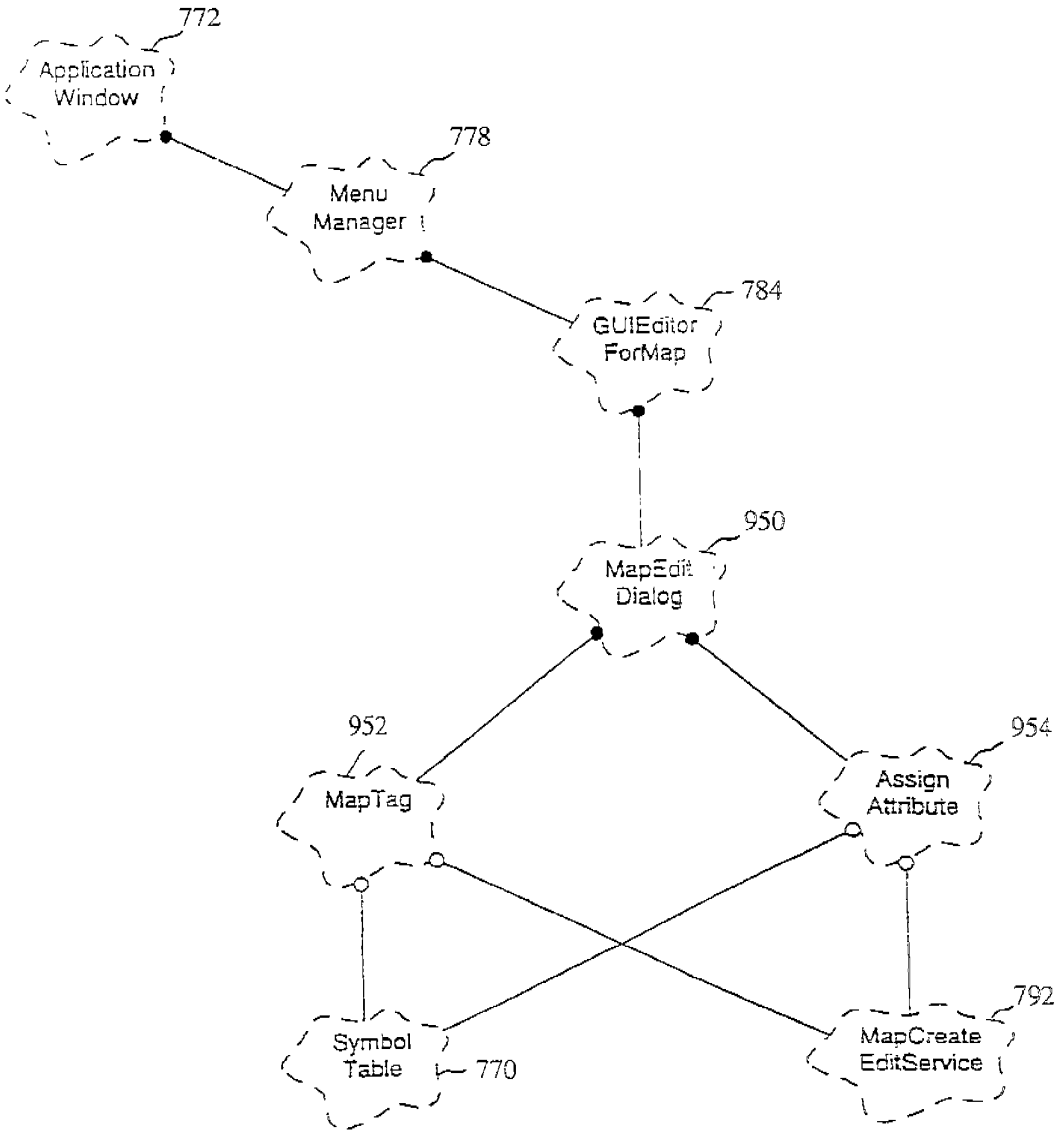

772
Application Window
778
Menu Manager
784
GUIEditor ForMap
950
MapEdit Dialog
952
MapTag
954
Assign Attribute
Symbol Table
770
792
MapCreate EditService

Fig. 17

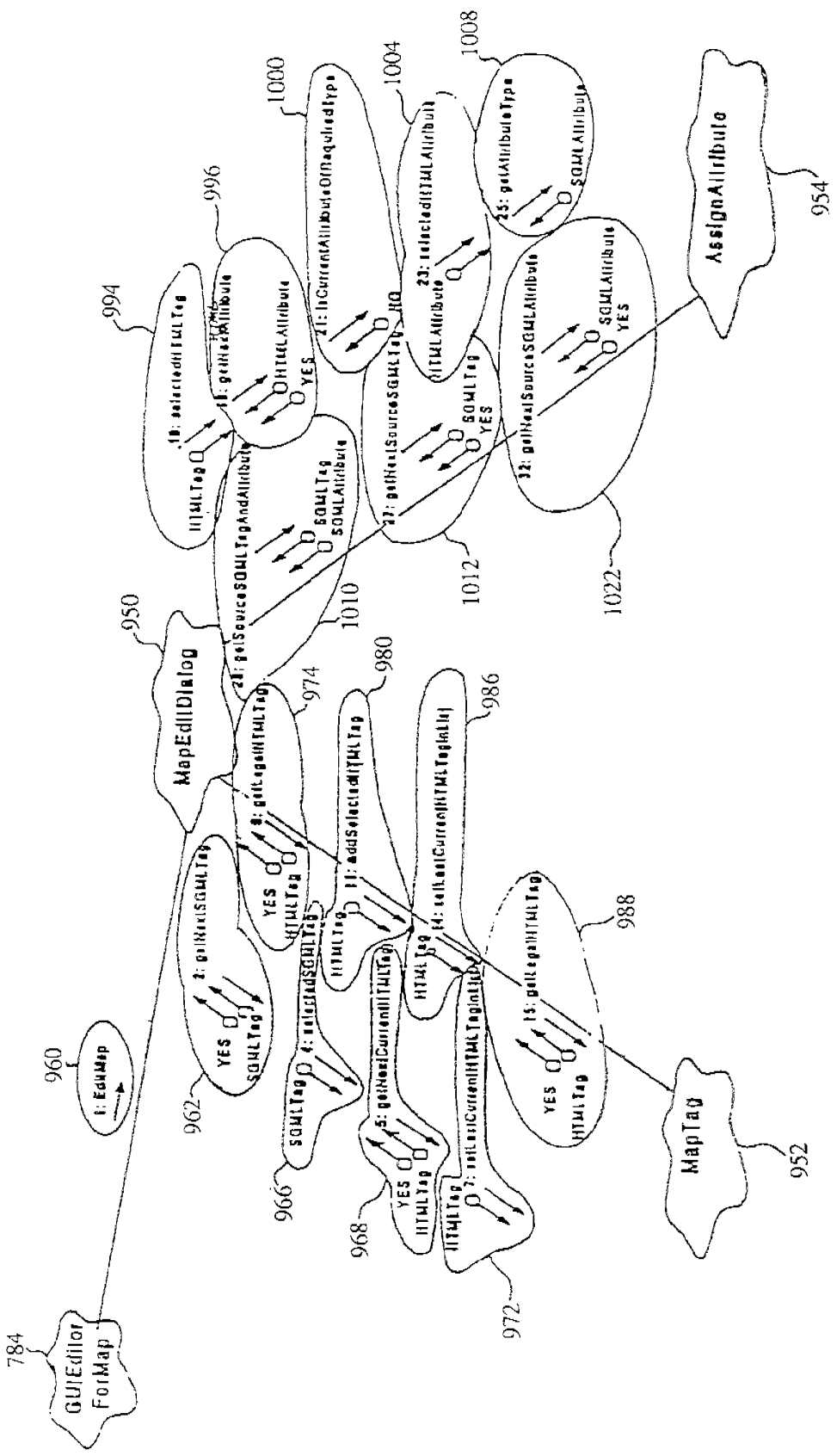

GUIEditor ForMap
784
1: EditMap
960
MapEditDialog
950
MapTag
952
AssignAttribute
954
962
966
968
972
974
980
986
988
994
996
1000
1004
1008
1010
1012
1022
YES
HTMLTag
SGMLTag
SGMLAttribute

Fig. 18A(1)

Fig. 18A(2)

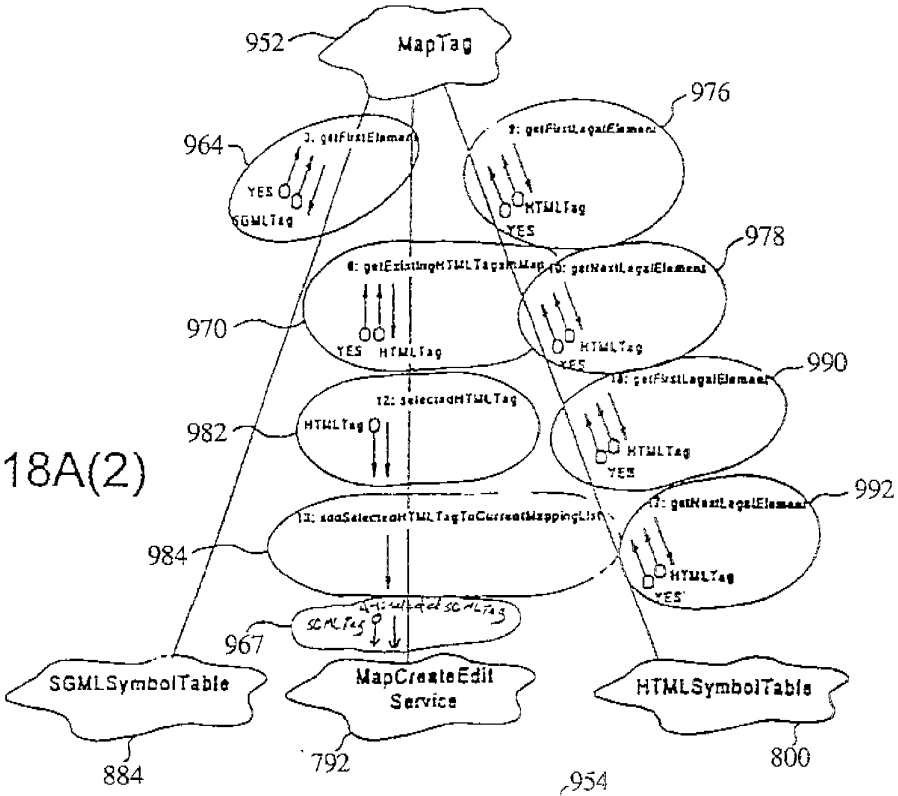
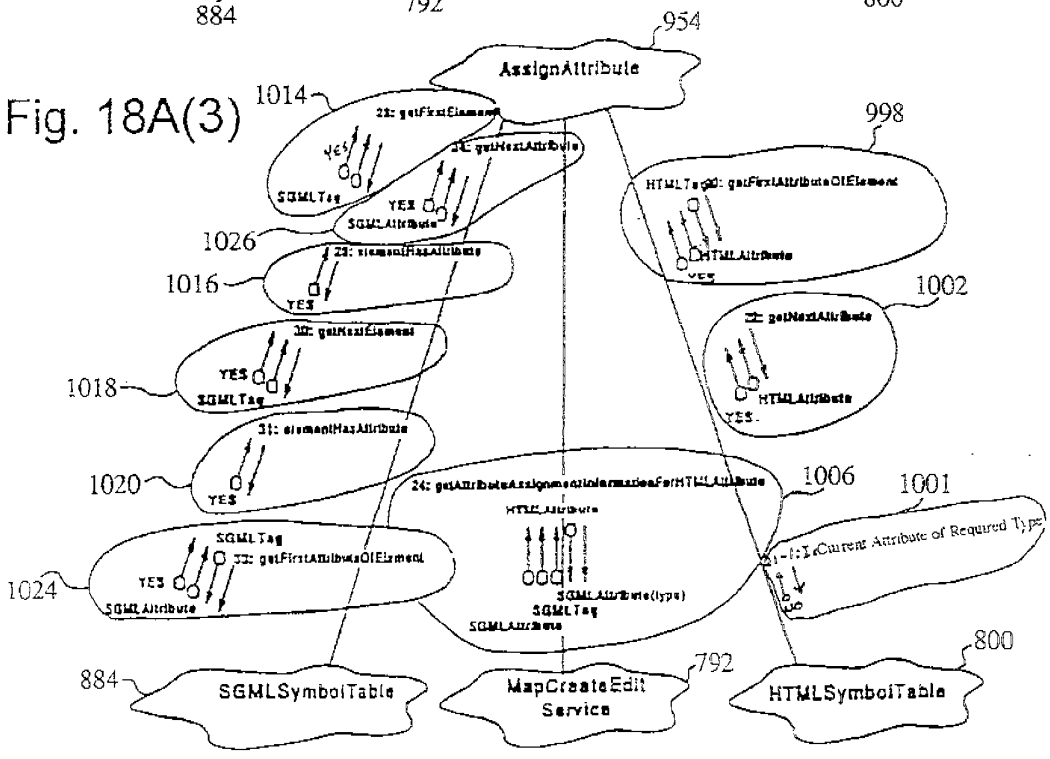



Fig. 18A(3)



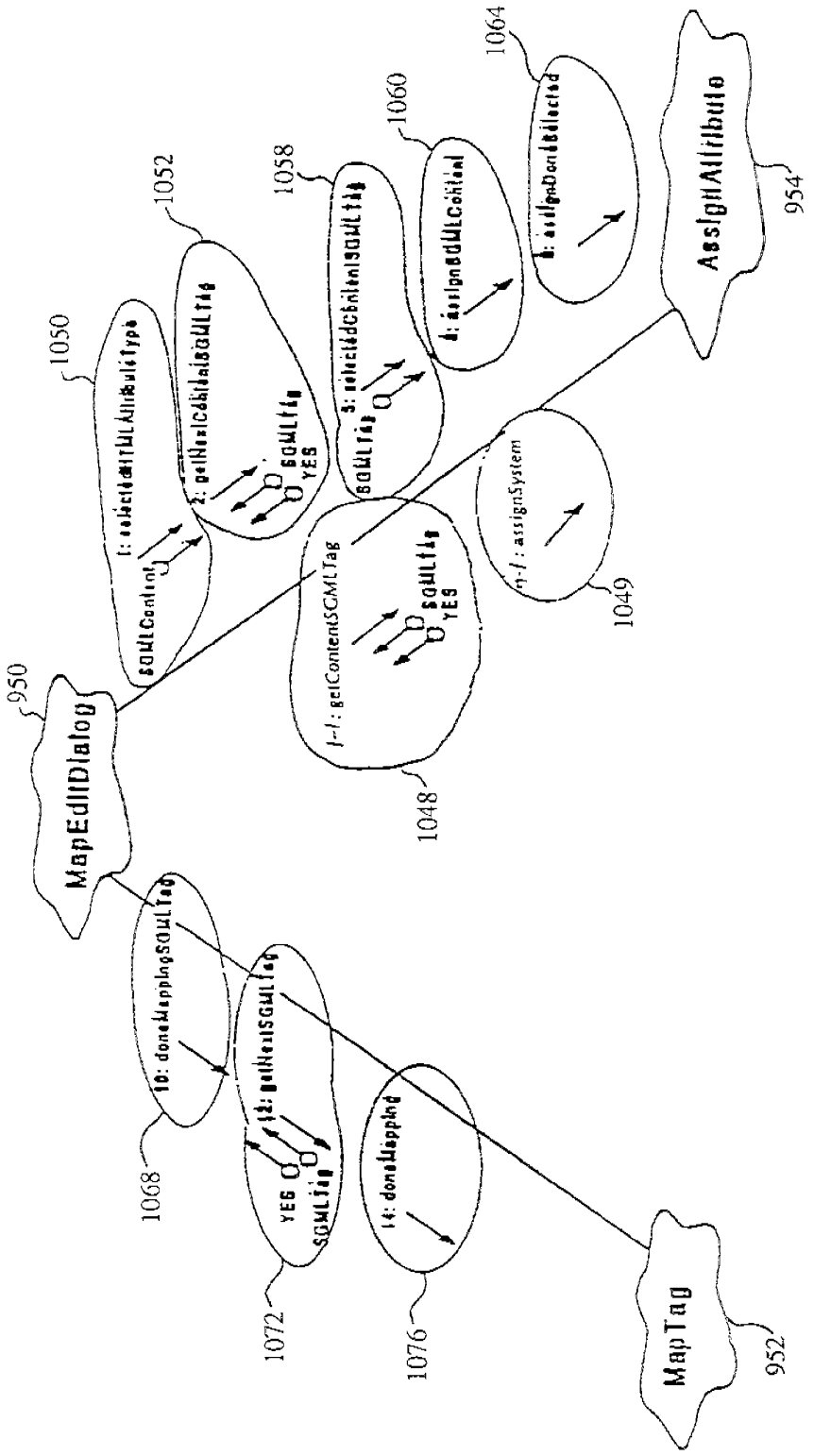

MapEdIIDialog
950
1050
1052
1058
1060
1064
954
1068
1072
1076
1048
1049
MapTag
952
YES
YES

Fig. 18B(1)

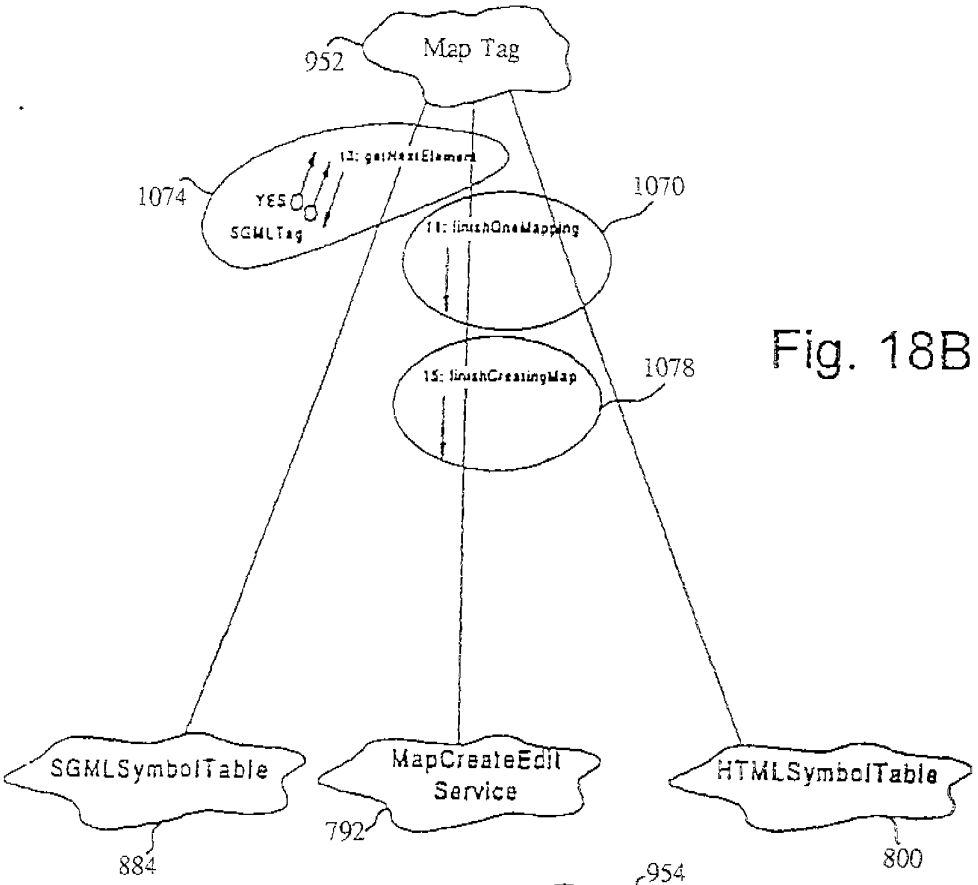



Fig. 18B(2)

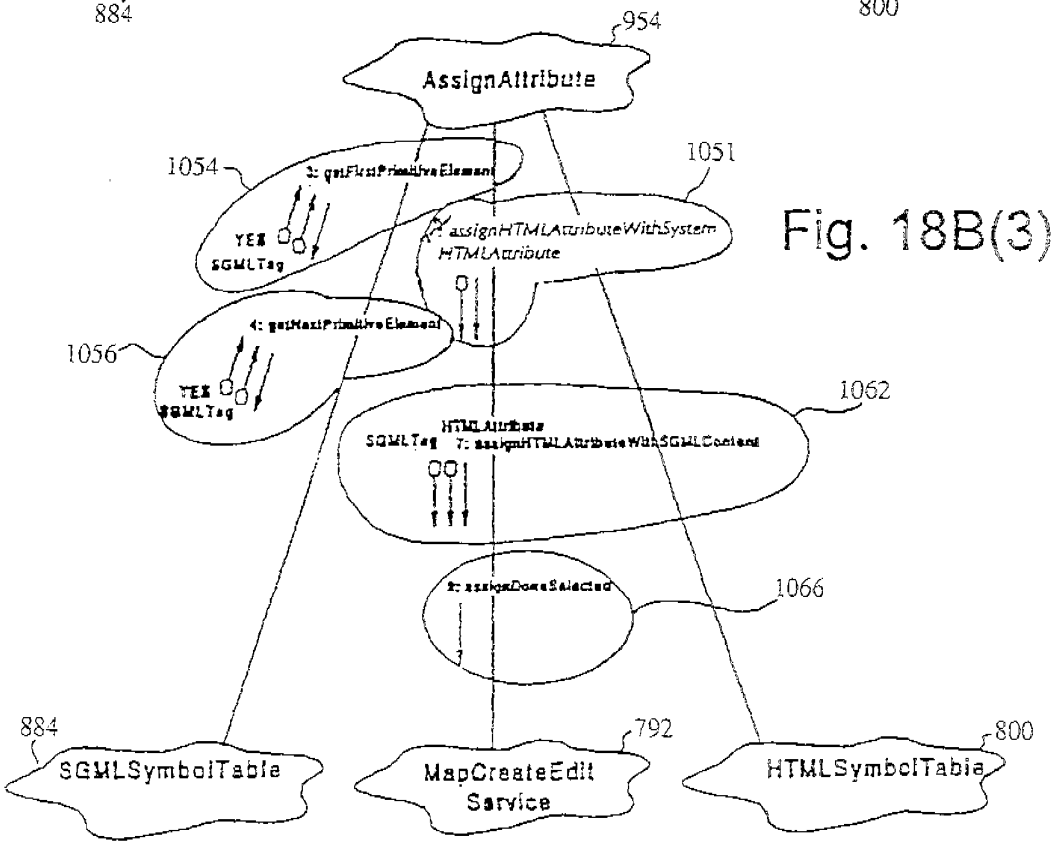



Fig. 18B(3)

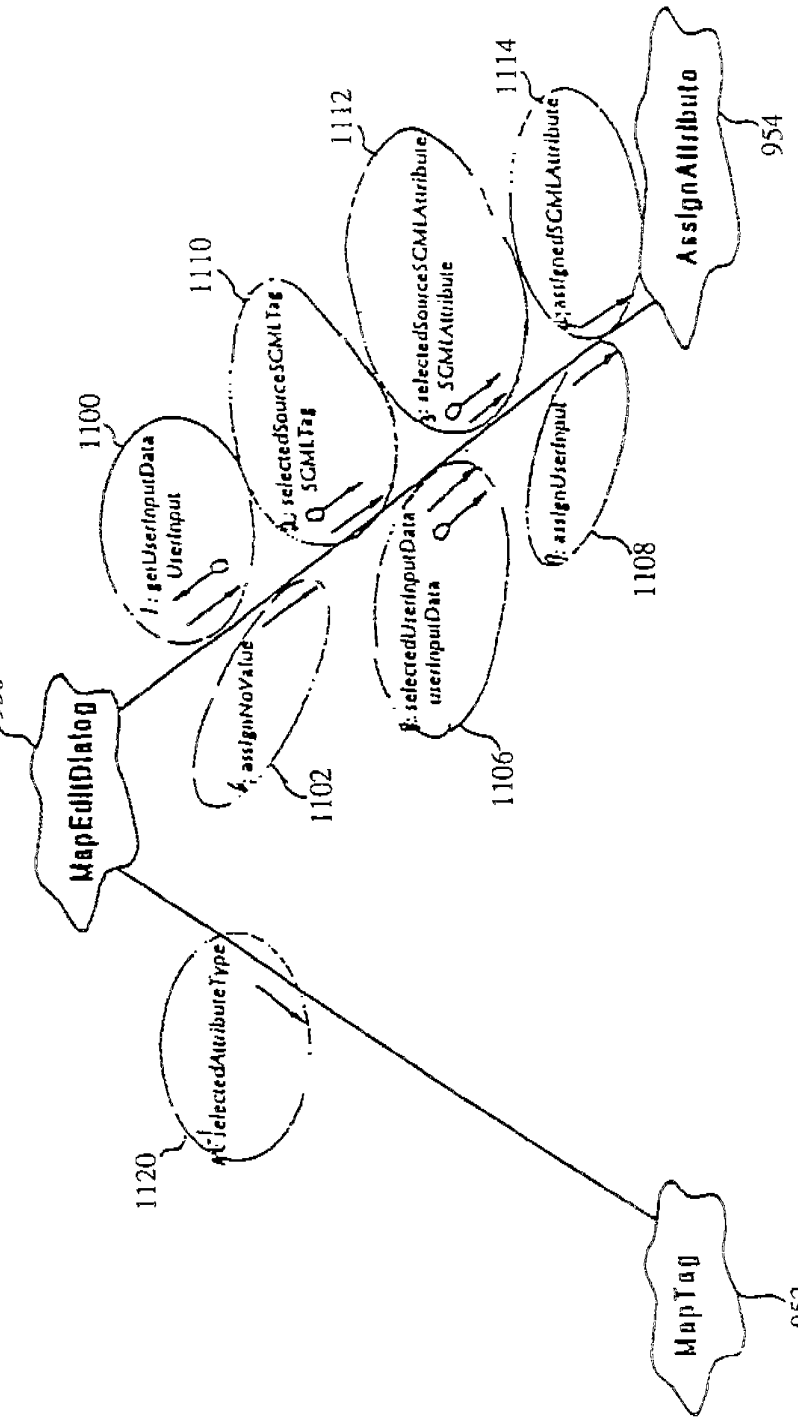

950
MapEditDialog
1100
1: getUserInputData
UserInput
1110
2: selectedSourceSGMLTag
SGMLTag
1112
3: selectedSourceSGMLAttribute
SGMLAttribute
1114
4: assignedSGMLAttribute
AssignAttribute
954
4: assignNoValue
1102
5: selectedUserInputData
userInputData
1106
6: assignUserInput
1108
1120
7: selectedAttributeType
MapTag
952

Fig. 18C(1)

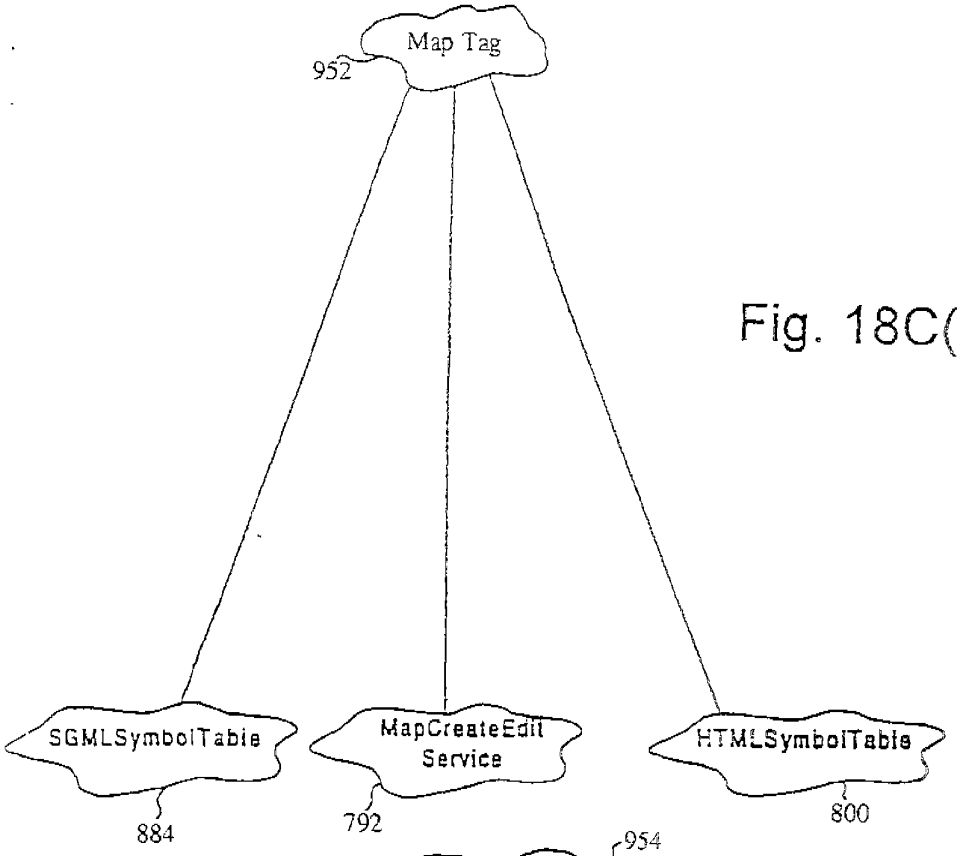



Fig. 18C(2)

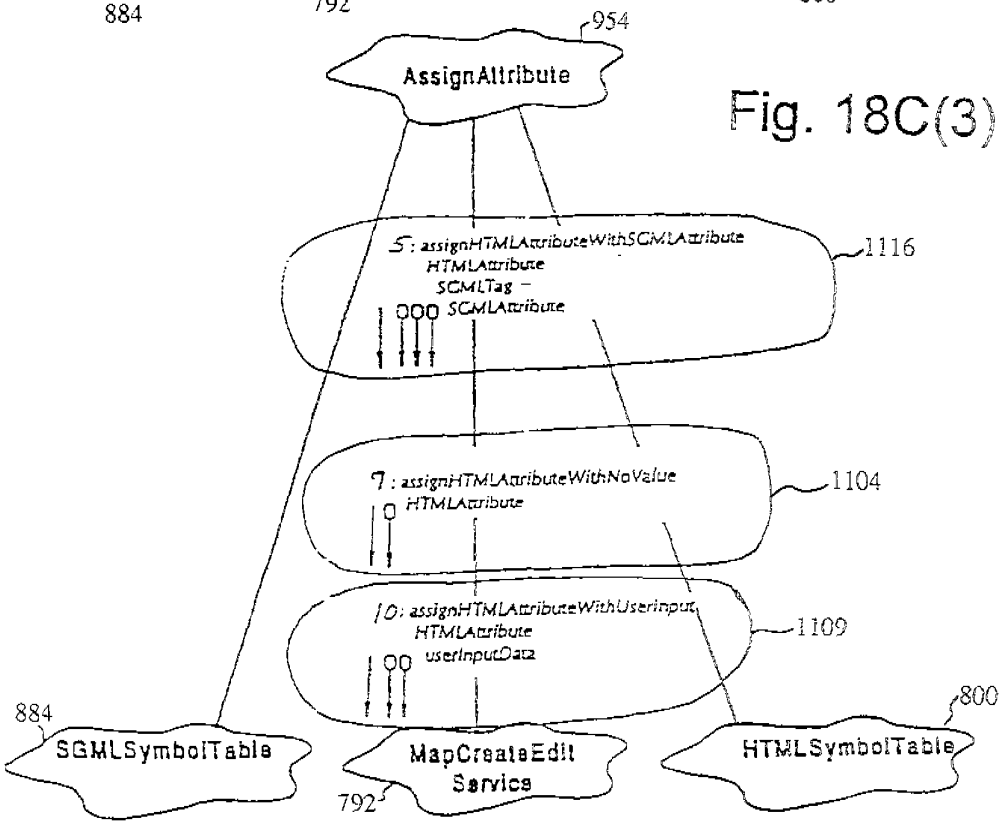



Fig. 18C(3)

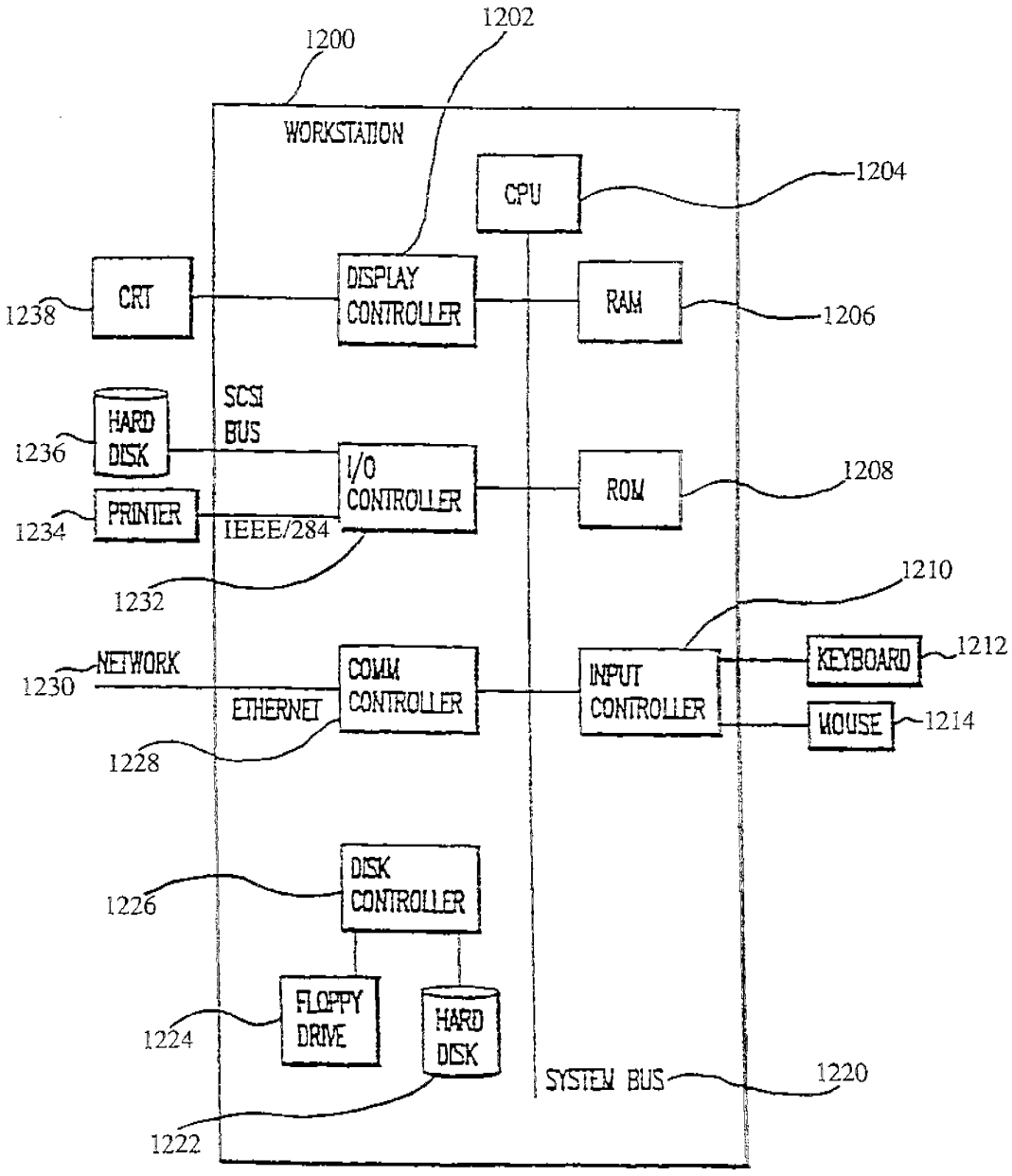

1200
1202
WORKSTATION
CPU
1204
CRT
1238
DISPLAY CONTROLLER
RAM
1206
SCSI BUS
HARD DISK
1236
I/O CONTROLLER
ROM
1208
PRINTER
1234
IEEE/284
1232
1210
NETWORK
1230
COMM CONTROLLER
INPUT CONTROLLER
KEYBOARD
1212
ETHERNET
MOUSE
1214
1228
DISK CONTROLLER
1226
FLOPPY DRIVE
1224
HARD DISK
SYSTEM BUS
1220
1222

Fig. 19

1400 XYZ//font::metric::x-offset::622

Fig. 20A

| | | | |
|---|---|---|---|
| 1420 | ownemame | -> | ownemame |
| 1422 | ownerdescription | -> | ownerdescription |
| 1424 | objectname | -> | objectname |
| 1426 | objectdescription | -> | objectdescription |
| 1428 | '::' | -> | '_' |
| 1430 | '//' | -> | '__' |

Fig. 20B

1440 XYZ__font_metric_x-offset_622

Fig. 20C

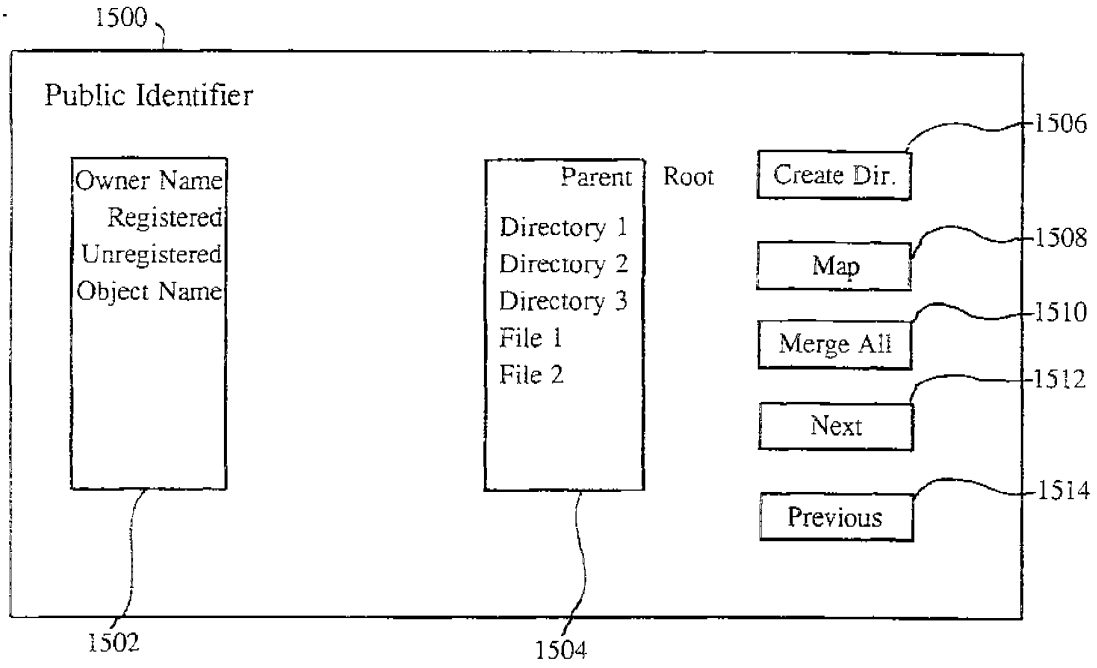



Fig. 20D

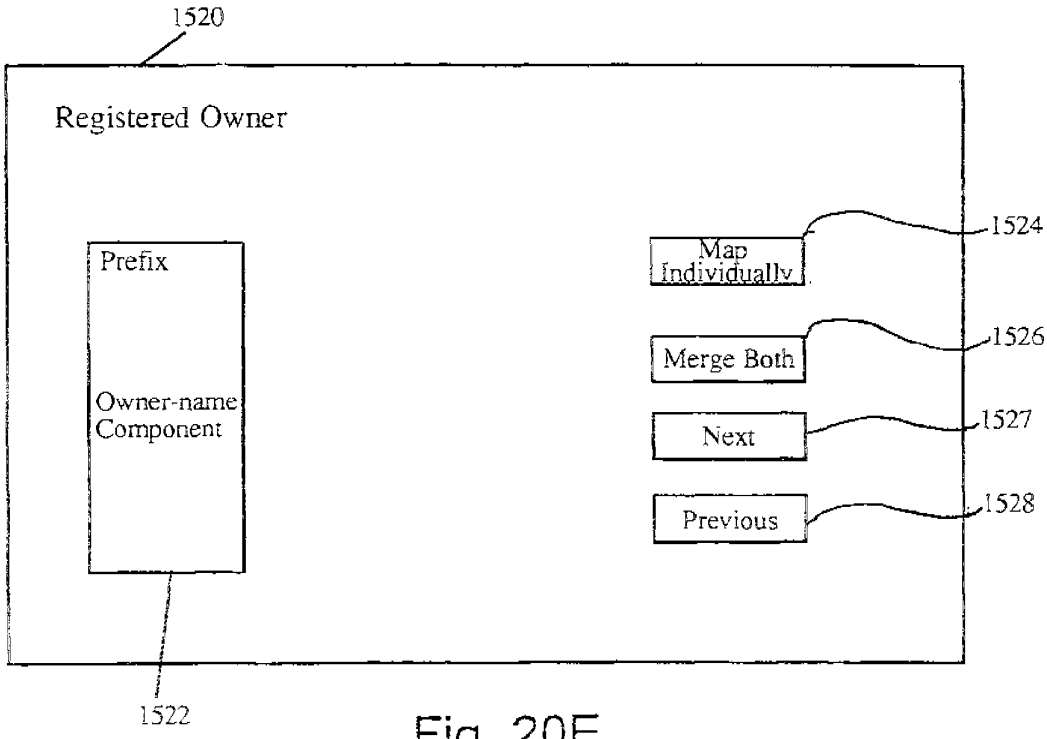



Fig. 20E

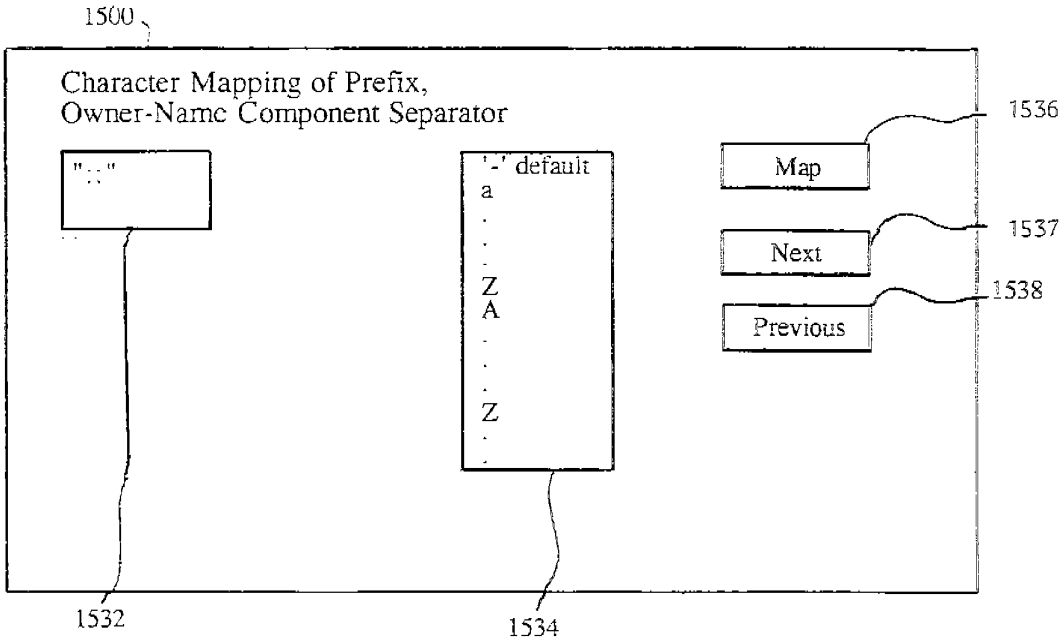

1500
Character Mapping of Prefix,
Owner-Name Component Separator
"::"
'-' default
a
Z
A
Z
Map
Next
Previous
1536
1537
1538
1532
1534

Fig. 20F

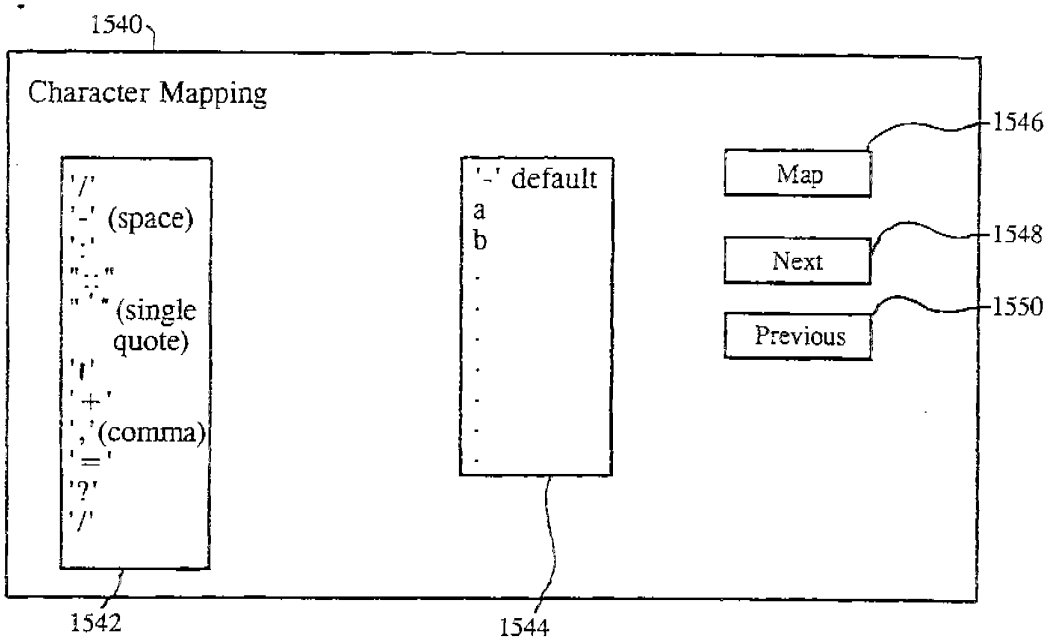



Fig. 20G

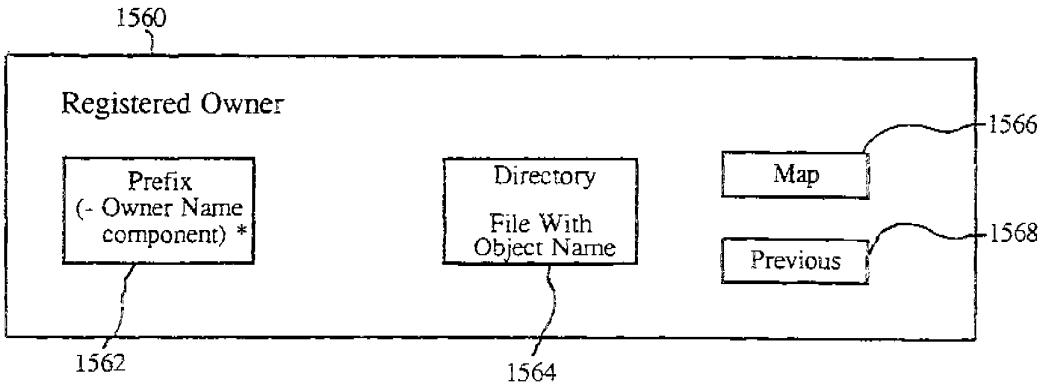



Fig. 20H

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 949 571 A2

# (12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
13.10.1999 Bulletin 1999/41

(51) Int Cl.[6]: G06F 17/22

(21) Application number: 99302718.4

(22) Date of filing: 07.04.1999

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 07.04.1998 US 80909 P
29.01.1999 US 239295

(71) Applicant: XEROX CORPORATION
Rochester, New York 14644 (US)

(72) Inventors:
- Bickmore, Timothy W.
Somerville, Massachusetts 02144 (US)
- Girgensohn, Andreas
Menlo Park, California (US)
- Schilit, William N.
Menlo Park, California 94025 (US)
- Sullivan, Joseph W.
San Francisco, California 94107 (US)

(74) Representative: Skone James, Robert Edmund
GILL JENNINGS & EVERY
Broadgate House
7 Eldon Street
London EC2M 7LH (GB)

## (54) Document re-authoring systems and methods for providing device-independent access to the world wide web

(57) An automatic re-authoring system and method re-author a document originally designed for display on a desktop computer screen for display on a smaller display screen, such as those used with a PDA or a cellular telephone. The automatic re-authoring system and method input a document to be re-authored and re-authoring parameters, such as display screen size, default font and the like. The automatic re-authoring system and method convert the document into a number of pages, where each page is fully displayable with only at most a minimal amount of scrolling on the display screen of the PDA or cellular phone. At each stage of the re-authoring, a number of different transformations are applied to the original document or a selected re-authored page. The selected re-authored page is the best page resulting from the previous re-authoring stage. The best page at each stage is determined based on the re-authoring parameters and the content of the document being re-authored.

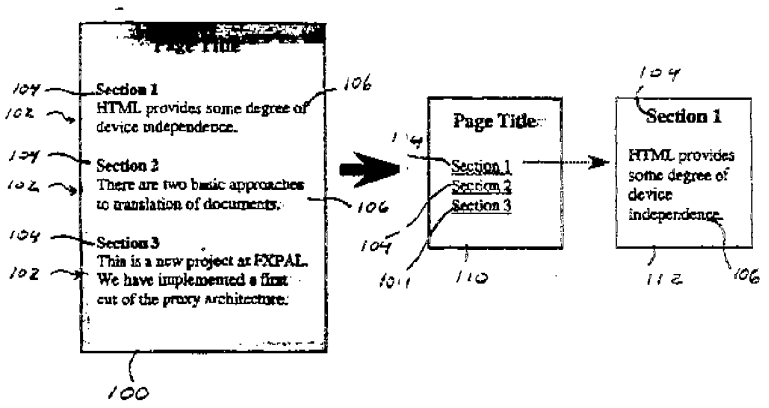



FIG. 1

EP 0 949 571 A2

## Description

**[0001]** This invention is directed to document re-authoring systems and methods that automatically re-author arbitrary documents from the world-wide web to display the documents appropriately on small screen devices, such as personal digital assistants (PDAs) and cellular phones, providing device-independent access to the web.

**[0002]** Access to world-wide web documents from personal electronic devices has been demonstrated in research projects such as those described in J. Bartlett, "Experience with a Wireless World Wide Web Client", IEEE COMPCON 95, San Francisco, CA, March 1995; S. Gessler et al., "PDAs as Mobile WWW Browsers", Second International World Wide Web Conference. Chicago, IL, October 1994; G. Voelker et al., "Mobisaic: An Information System for a Mobile Wireless Computing Environment", Workshop on Mobile Computing Systems and Applications. Santa Cruz, CA, December 1994; and T. Watson, "Application Design for Wireless Computing", 1994 Mobile Computing Systems and Applications Workshop Position Paper, August 1994. Such access is now a commercial reality. General Magic's Presto! Links for Sony's MagicLink, and AllPen's NetHopper for the Newton and Sharp's MI-10 all provide WWW browsers for PDA class devices, while the Nokia 9000 Communicator and Samsung's Duett provide web access capabilities from cellular phones.

**[0003]** Unfortunately, most documents on the world-wide web and other distributed networks are designed for display on desktop computers with color monitors having at least 640x480 resolution. Many pages are designed with even larger resolution monitors in mind. In contrast, most PDA class devices and cellular phone displays are much smaller. This difference in display area can lead to a ratio of designed vs. available display area from 4-to-1 to 100-to-1, or greater, making direct presentation of most worldwide web documents on these small devices aesthetically unpleasant, un-navigable, and in the worst case, completely undecipherable. This presents a central problem in accessing world-wide web pages using these small devices: how to display arbitrary web documents, such as HTML documents, that have been designed for desktop systems on personal electronic devices that have much more limited display capabilities.

**[0004]** Technologies already provide computational mobility and wireless connectivity, but the standard solutions to viewing documents and web pages on tiny screens are to either increase the screen resolution, which is great if the user happens to carry a magnifying glass, or to provide the ability to FAX or print to a local hardcopy device, which is both inconvenient and contradicts the rationale for having electronic documents in the first place. There are five general approaches to displaying web documents on small screen devices: device-specific authoring; multiple-device authoring; client-side navigation; automatic re-authoring; and web page filtering. Device-specific authoring involves authoring a set of web documents for a particular display device, such as, for example, a cellular phone outfitted with a display and communications software, such as the Nokia 9000. The basic philosophy in this approach is that users of such specialty devices will only have access to a select set of services. Thus, the document for these services must be designed up-front for the accessing device's particular display system. Information may be provided from the distributed network at large, but the desired pages must be pre-defined, and custom information extraction and page formatting software must be written to deliver the information to the small device. This is the approach taken in Unwired Planet's UP.Link service, which uses a proprietary mark-up language (HDML).

**[0005]** In multiple-device authoring, a range of target devices is identified. Then, mappings from a single source document to a set of rendered documents are defined to cover the devices within the identified range. One example of this is the StretchText approach discussed in I. Cooper et al., "PDA Web Browsers: Implementation Issues" University of Kent at Canterbury Computing Laboratory WWW Page, November 1995. In StretchText, portions of the document, potentially down to the word level, can be tagged with a 'level of abstraction' measure. Upon receiving the document, users can specify the level of abstraction they wish to view and are presented with the corresponding detail or lack of detail.

**[0006]** Another example of multiple-device authoring is HTML cascading style sheets (CSS), as described in H. Lie et al. "Cascading Style Sheets", WWW Consortium, September 1996. In cascading style sheets, a single style sheet defines a set of display attributes for different structural portions of a document. For example, all top-level section headings can be defined to be displayed in red 18-point Times font. A series of style sheets may be attached to a document, each with a weight describing that style sheet's desirability to the document's author. The user can also specify a default style sheet. The browser used by the user to access the distributed network can also define a "default" style sheet. Although the author's style sheets normally override the user's style sheets, the user can selectively enable or disable the author's style sheets, providing the user with the ability to tailor the rendering of the document to the user's particular display.

**[0007]** In client-side navigation, the user is given the ability to interactively navigate within a single web page by altering the portion of the single web page that is displayed at any given time. A very trivial example of this is the use of scroll bars in the document display area. A much more sophisticated approach is that taken in the PAD++ system, as described in B. Bederson et al., "Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics", Proceedings of ACM UIST '94, ACM Press, 1994, in which the user is free to zoom and pan the device display over

the document with infinite resolution. Active Outlining, as described in J. Hsu et al., "Active Outlining for HTML Documents: An X-Mosaic Implementation", Second International World Wide Web Conference, Chicago, IL, October 1994, has also been implemented as a client-side navigation technique, in which the user can dynamically expand and collapse sections of the document under the respective section headings. Other techniques that fall into this category include semi-transparent widgets, as described in T. Kamba et al., "Using small screen space more efficiently", Proceedings, Computer-Human Interactions: CHI 96, Vancouver, BC, Canada, April 1996, and the Magic Lens system, as described in E. Bier et al., "Toolglass and Magic Lenses: The See-through Interface", SIGGRAPH '93 Conference Proceedings 1993.

**[0008]** Automatic document re-authoring involves developing software that can take an arbitrary document, such as an HTML document, designed to be displayed on a desktop-sized monitor, along with characteristics of the target display device, and re-author the arbitrary document through a series of transformations, so that the arbitrary document can be appropriately displayed on the target display device. This process can be performed either by the client, by the server, or by an intermediary proxy server, such as an HTTP proxy server , that exists solely to provide these transformation services. An example of this latter approach is the UC Berkeley Pythia proxy server, as described in A. Fox et al., "Reducing WWW Latency and Bandwidth Requirements by Real-Time Distillation", Fifth International World Wide Web Conference, Paris, France, May 1996, which performs transformations on web page images. However, the focus of the Pythia proxy server is solely on minimizing page retrieval time. Spyglass Prism is a commercial product that performs automatic re-authoring of HTML documents using fixed transformations associated with page tags or embedded object types. For example, Prism will reduce all JPEG images by 50%.

**[0009]** Finally, web page filtering lets a user see only those portions of a page that user is interested in. Filtering may be performed on an intermediate server, such as an HTTP proxy server, to conserve wireless bandwidth and device memory. However, filtering could also be performed by the client device as a display-management technique. Filter specifications can be based on keyword or regular expression matching, or on page structure navigation and extraction commands. Filtering can be either specified using visual tools or using a scripting language.

**[0010]** Each of the five approaches, device-specific authoring, multiple-device authoring, client-side navigation, automatic re-authoring and web page filtering, has specific benefits and drawbacks. Device-specific authoring will typically yield the best-looking results due to the direct involvement of human designer. However, device-specific authoring limits the user's access to a small, select set of documents that have been authored for that specific device. Multiple-device authoring, while requiring less total effort per document than device-specific authoring, still requires significantly more manual design work than simply authoring a single version of a document for a single desktop platform. Client-side navigation will work well if a good set of viewing techniques can be developed. However, client-side navigation requires that the entire document be delivered to the client device at once, which may waste valuable wireless bandwidth and memory. Furthermore, the 'peephole' approach taken in PAD++ seems very awkward to use for large documents, and the active outlining technique has limited applicability, as most web pages do not use a strict section/subsection organization, or use that organization incorrectly.

**[0011]** Automatic re-authoring is thus the ideal approach to providing broad access to web documents or other web content from a wide range of devices, if automatic re-authoring can be made to produce legible, navigable and aesthetically pleasing re-authored documents without loss of information.

**[0012]** This invention provides systems and methods that automatically re-author documents designed for a larger display area for display on a smaller display area.

**[0013]** This invention separately provides systems and methods that automatically transform a document into a plurality of linked subdocuments, where each subdocument requires less display area.

**[0014]** This invention separately provides systems and methods that automatically apply a plurality of different transforms to an original document to generate a plurality of sets of linked subdocuments.

**[0015]** This invention further provides systems and methods that automatically apply the plurality of different transforms to at least one of the plurality of sets of linked subdocuments to generate additional linked subdocuments.

**[0016]** This invention further provides systems and methods that analyze a main subdocument of each set of linked subdocuments to determine a best one of the main subdocuments.

**[0017]** This invention additionally provides systems and methods that determine if the best main subdocument can be displayed in the smaller display area, and if not, that apply further transforms to that main subdocument to further reduce the required display area.

**[0018]** This invention separately provides systems and methods that filter a document to extract a desired portion of the document that is displayable in a smaller display area.

**[0019]** This invention separately provides systems and methods that filter a document to extract a described portion based on a predefined script.

**[0020]** This invention separately provides systems and methods that generate scripts usable to filter a document to extract a desired portion.

**[0021]** This invention separately provides a scripting language usable to write scripts for filtering a document to

extract a desired portion.

**[0022]** In one exemplary embodiment, the document re-authoring systems and methods of this invention are implemented on an HTTP proxy that dynamically re-authors requested web pages using a heuristic planning technique and a set of structural page transformations to achieve the best-looking document for a given display size. The automatic document re-authoring according to the systems and methods of this invention can be performed either by the client, by the server, or, in one exemplary embodiment, by an intermediary HTTP proxy server that exists solely to provide these transformation services. Additionally, the automatic document re-authoring systems and methods according to this invention can be performed on a combination of these devices.

**[0023]** The automatic document re-authoring systems and methods of this invention work well with displays found in PDAs. However, when the document re-authoring systems and methods of this invention are applied to the very limited displays found on current cellular phones, the document re-authoring systems and methods of this invention sometimes produces pages that are difficult to navigate. When accessing a distributed network, such as the Internet or an intranet, from a cellular phone, most users are mainly interested in accessing very specific information. The document filtering systems and methods of this invention provide those users with manual control in defining the information they would like to be displayed. The document filtering systems and methods of this invention return only a small portion of a page that is easily navigable. The document filtering systems and methods of this invention are ideal in those situations in which the user is monitoring a particular page whose layout is fixed but whose content is changing, since those users can tune the filters to the format of the page.

**[0024]** The automatic document re-authoring and document filtering systems and methods of this invention provide an automatic document re-authoring capability coupled with document filtering to provide access to arbitrary documents on a distributed network, such as the Internet or an intranet, to devices with limited communications bandwidth and small displays.

**[0025]** The automatic document re-authoring and document filtering systems and methods ofthis invention intercept requests for documents from a distributed network and return re-authored versions of the requested documents rather than the original requested documents.

**[0026]** In the larger context of mobile and ubiquitous computing, the automatic document re-authoring and document filtering systems and methods of this invention provide a key technology for giving users view-mobility over platforms.

**[0027]** These and other features and advantages of this invention are described in or are apparent from the following detailed description of the preferred embodiments.

**[0028]** The preferred embodiments of this invention will be described in detail, with reference to the following figures, wherein:

Fig. 1 illustrates re-authoring of a document into a section list page and a number of section pages according to one exemplary embodiment of the document re-authoring systems and methods of this invention;

Fig. 2 illustrates a layout table that can be re-authored into a plurality of linked cells according to one exemplary embodiment of the document re-authoring systems and methods of this invention;

Fig. 3 illustrates how a document can be re-authored into different re-authored states based on applying different transformations according to one exemplary embodiment of the re-authoring systems and methods of this invention;

Fig. 4 illustrates one exemplary embodiment of a control form for supplying display information to the HTTP proxy server according to the document re-authoring system and method of this invention;

Fig. 5 illustrates one exemplary embodiment of re-authoring an exemplary document according to the document re-authoring systems and methods of this invention;

Fig. 6 is a block diagram outlining one exemplary embodiment of the invention in which the document re-authoring systems and methods of this invention are used;

Fig. 7 is a block diagram outlining one exemplary embodiment of the document flow in the document re-authoring systems and methods of this invention;

Fig. 8 is a functional block diagram outlining one exemplary embodiment of a document re-authoring system according to this invention;

Fig. 9 is one exemplary embodiment of the document version search space of the document re-authoring systems and methods of this invention;

Fig. 10 is one exemplary embodiment of an image and the abstract syntax tree generated from that image according to this invention;

Figs. 11A and 11B outline one exemplary embodiment of a method for document re-authoring according to this invention;

Fig. 12 is one exemplary embodiment of a method for performing elision transformation according to this invention;

Fig. 13 is one exemplary embodiment of a method for performing table transformation according to this invention;

Fig. 14 is one exemplary embodiment of a method for performing image reduction transformation according to this

invention;
Fig. 15 is a functional block diagram outlining one exemplary embodiment of a document re-authoring system 600 of this invention including the document filtering according to this invention;
Fig. 16 is one exemplary embodiment of the document flow during document filtering and re-authoring according to this invention;
Fig. 17 shows an exemplary embodiment of using the document filtering systems and methods of this invention to navigate within the abstract syntax tree generated from the image shown in Fig. 10; and
Fig. 18 illustrates further navigation within the abstract syntax tree of Fig. 10 according to the document filtering systems and methods of this invention.

**[0029]** In the following discussion of the document re-authoring and document filtering systems and methods of this invention, the terms "web page", "web document" and "document" are intended to encompass any set of information retrieved as a single entity from a distributed network, such as an intranet, the Internet, the World Wide Web portion of the Internet or any other known or later developed distributed network. This information can include text strings, images, tables of text strings and images, links to other web pages and formatting information that defines the layout of the text strings, images, tables and links within the web page.

**[0030]** There are many possible automatic document re-authoring techniques, which can be categorized along two dimensions: syntactic vs. semantic techniques and transformation vs. elision techniques. Syntactic techniques operate on the structure of the document, while semantic techniques rely on some understanding of the content. Elision techniques basically remove some information, leaving everything else untouched, while transformation techniques involve modifying some aspect of the document's presentation or content. Table 1 illustrates these dimensions, along with examples of each category:

TABLE 1

| Examples of different types of automatic document re-authoring techniques | | |
|---|---|---|
| | Elide | Transform |
| Syntactic | Section Outlining | Image Reduction |
| Semantic | Removing Irrelevant Content | Text Summarizing |

**[0031]** In order to gain an understanding of the processes required by an automated document re-authoring system, a study was conducted to assess the characteristics of typical web pages, and to identify candidate re-authoring techniques through the process of re-authoring several web pages by hand.

**[0032]** A collection of'typical' web pages, the Xerox Corporate web site, was initially selected to focus the study. This collection of 3,188 web pages is representative of a state-of-the-art, professionally-designed web site. A variety of statistics were collected on these pages using a web crawler, to help gain an understanding of the structure and content of a typical page. These statistics generally agree with other, larger-scale studies that have been performed across the entire web.

**[0033]** Next, a subset of the pages in the Xerox web site was selected for manual re-authoring. A set of pages from the Xerox 1995 Annual Report was selected and converted by hand for display on a Sharp Zaurus PDA with a 320x240 pixel screen. Detailed notes were kept of the design strategies and techniques used.

Some of the design heuristics learned during this process were:

**[0034]** Keeping at least some of the original images is important to maintain the look and feel of the original document. Common techniques include keeping only the first image, or keeping only the first and last images, i.e., the bookend images, and eliding the rest.

**[0035]** Section headers, i.e., the H1 - H6 tags in HTML, are not often used correctly. The section headers are more frequently used to achieve a particular font size and style, such as, for example, bold, if the section headers are used at all. Thus, the section headers cannot be relied upon to provide a structural outline for most documents. Instead, documents with many text blocks can be reduced by replacing each text block with the first sentence or phrase of each block, i.e., first sentence elision.

**[0036]** An initial rule of thumb for images is to reduce all images in size by a standard percentage, dictated by the ratio of the display area that the document was authored for to the display area of the target device. However, images which contain text or numbers can only be reduced by a small amount before their contents become illegible.

**[0037]** Semantic elision can be performed on sidebars that present information which is tangential to the main concepts presented in a page. Many of the Xerox pages had such sidebars, which were simply eliminated in the reduced

versions.

**[0038]** Semantic elision can also be performed on images that do not contribute any information to the page, but serve only to improve its aesthetics.

**[0039]** Pages can be categorized, and then re-authored based on their category. Two examples of these are banners and link tables. Banners primarily contain a set of images and a small number of navigation links, often only one, that serve to establish an aesthetic look, but contain little or no content. When space is at a premium, these can usually be omitted entirely. Link table pages are primarily sets of hypertext links to other pages, and thus contain very little additional content These link table pages can usually be re-formatted into a more compact form that just lists the links in a text block.

**[0040]** Whitespace, which is taken for granted on a large display, is at a premium on small devices. Several techniques were discovered for reducing the amount of whitespace in a page. Sequences of paragraphs , i.e., HTML "P" tags, or breaks, i.e., HTML "BR" tags, can be collapsed into one such paragraph or break. Lists, i.e., HTML "UL", "OL", and/or "DL" tags, take up valuable horizontal space with their indenting and bullets. These lists can be re-formatted into simple text blocks with breaks between successive items, as described in Cooper et al.

**[0041]** In conclusion, to perform document re-authoring two things are required: a set of re-authoring techniques, i. e., a set of page transformations, and a strategy for applying the page transformations. Of the techniques used in the manual re-authoring study, those most amenable to codification were the syntactic elision techniques, including section outlining, first sentence elision, and image elision, and the syntactic transformation techniques, including image size reduction and font size reduction. The design strategy learned during the study included a ranking of the transformation techniques, i.e., try this before that, and a set of conditions under which each transformation or combination of transformations should be applied.

**[0042]** Following the results of the study discussed above, there are two major elements to the document re-authoring systems and methods of this invention: a collection of individual re-authoring techniques that transform documents in various ways; and automated document re-authoring systems and methods that implement a design strategy by selecting the best combination of techniques for a given document/display size pair.

**[0043]** The Section Header Outlining transform provide a very good method for reducing the required display size for structured documents, such as technical papers and reports. The outlining process is shown in Fig. 1.

**[0044]** As shown in Fig. 1, the document 100 is converted into a list of sections page 110 and each section is elided into a page 111. That is, the contents 106 of each section 102 of the document 100 is elided from the document 100 and each section header 104 is converted in o a hypertext link. When the hypertext link for any section is selected, the corresponding page 111 of elided content is loaded into the browser. When confronted with multiple section levels (sections, sub-sections, sub-subsections, etc.), there are two approaches to performing the elision. The first approach is full outlining, which works by keeping only the section headers and eliding all content, with the results looking like a table of contents for a book. The second approach is to-level outlining. In the to-level outlining, a cutoff level in the section hierarchy is determined and all content below that level, including lower-level section headers, is elided, but all content above that level is kept.

**[0045]** Since most pages have text blocks, even when no section headers are present, the First Sentence Elision transform can be a good way of reducing required screen area. In this technique, each text block is replaced with its first sentence, or, alternatively, its first phrase up to some natural break point. This first sentence or phrase is also made into a hypertext link to the original text block.

**[0046]** The Indexed Segment transform first attempts to find page elements that can logically be partitioned, such as ordered or unordered lists, sequences of paragraphs on tables. This transform takes an input page, segments the content into sub-pages by allocating some number of items to each, and builds and prepends an index page to the collection of sub-pages. The Indexed Segment transform then starts filling output pages with these elements in order until each page is "full" relative to the client's display size. If a single logical element cannot fit on a single output page, then the Indexed Segment transform performs a secondary partitioning that partitions text blocks on paragraph or sentence boundaries.

**[0047]** In the Indexed Segment transform, as much style information as possible is retained for the output elements, by outputting each element embedded within all of its ancestor partitions' HTML tags. The Indexed Segment transform then constructs an index page by copying a section header or first sentence from each element to be output, concatenating the copied portion onto an index page, and creating a hypertext link from each copied portion to the appropriate sub-page. It should be appreciated that the index page itself may also need to be segmented. In the Indexed Segment transform, "Next" and "Previous" navigation links between sequential sub-pages are also added for navigational convenience.

**[0048]** The Table transform recognizes when a table, i.e., the presentation of information arranged in a rectangular grid, on a page cannot be directly sent to the client. In these cases, the Table transform generates one sub-page per table cell, using a top-down, left-to-right order. Tables nested within tables are processed in the same manner. The Table transform uses heuristics to determine when table columns are being used as "navigational sidebars," which is

a common practice in commercial HTML web pages. In this case, the Table transform moves these cells to the end of the list of sub-pages as these cells tend to carry very little content.

**[0049]** Fig. 2 shows a nested table, marking tables with thicker borders than table cells. In the table 120 show in Fig. 2, the cell 122 is identified a as sidebar and will be placed after the cell 128. All of the other cells are placed in their natural order. The six portions of the cell 124, such as the subcells 125 and 126, are each placed in their own subpage between the subpages containing the subcells 123 and 127, unless they contain only whitespace.

**[0050]** As one can see from the example, nested tables and sidebars complicate the processing of tables. This is especially true if the sidebar is part of an inner table. In that situation, the sidebar should be moved to the end of the inner table, rather than to the end of any surrounding tables. In one exemplary embodiment of the document re-authoring systems and methods of this inventory, the sidebars are moved one table at a time and then all table cells are processed at once, rather than grouping the cells by table.

**[0051]** Images present one of the most difficult problems for automatic document re-authoring, because the decision of whether to keep, reduce, or eliminate a given image should be based on an understanding of the content and role of the image on the page. However, Image Reduction transforms and Image Elision transforms can be applied without content understanding, as long as users are provided a mechanism by which the users can retrieve the original images. In one exemplary embodiment of the systems and methods of this invention, the Image Reduction transform reduces all images in a page by one of a set of pre-defined scaling factors, such as 25%, 50%, and 75%, and making the reduced images into hypertext links that link the reduced images back to the original images.

**[0052]** In addition to the Image Reduction transform, three Syntactic Elision transforms have also been developed for image, the Elide All transform, the First Image Only transform, and the Bookends transform. In the Elide All transform, all images are elided from the document. In the First Image Only transform, all but the first image are elided from the document. In the Bookends transform, all but the first and last images are elided from the document. The elided images are each replaced with their HTML "ALT" text when it is available. Alternatively, the elided images are each replaced with a standard icon when no ALT text is available. The ALT text or standard icon for each elided image is also made into a hypertext link to that original image.

**[0053]** In one exemplary embodiment of the document re-authoring systems and methods of this inventory, if screen space is too limited or the client device cannot display images, the images are removed from the document. However, the removed images may be used as anchors for hypertext links via a client-side image map. It should also be appreciated that if such images are removed, the web site represented by the HTML document can be rendered non-navigable. To accommodate this, in one exemplary embodiment of the document re-authoring systems and methods of this inventory, a transform that extracts the hypertext links from such images and formats them into a text list of link anchors is used. The labels for the text list are extracted from the HTML "ALT" tags of the image map, if present, or from part of the Uniform Resource Locator of the link. This transformation preserves links attached to images for navigation when removing the images.

**[0054]** The overall process of deciding which combination of transforms to apply to a given page for a given client display seems at first to require some form of human artistic ability. However, the automatic document re-authoring systems and methods of this invention capture many of the heuristics used in the manual re-authoring exercise, and do a fairly good job of producing good-looking pages for a given display.

**[0055]** Individual page transformations are ordered by their desirability. In order to determine which combination of transformations should be applied to a given document, the document re-authoring systems and methods of this invention performs a depth-first search of the document transformation space, using many heuristics that describe preconditions for transformations and combinations of transformations. The depth-first search ensures that a "good enough" version of the document is found by using a combination of the most desirable transformations. Only if the more desirable transformations are not applicable or do not reduce the document enough, are the less favored transformations used.

**[0056]** The document re-authoring systems and methods of this invention search a document transformation space in a best-first manner. Each state in this search space represents a version of the document, with the initial state representing the original 'as-authored' document. Each state is tagged with a number representing a measure of merit that represents the quality of the document version at that state. The measure of merit, i.e., the evaluation function or value, for each state is a rough estimate of the screen area required to display the entire document as that document exists in that state. A state can be expanded into a successor state by applying a single transformation technique to the re-authored document as it exists in that state.

**[0057]** At every step in the search process, the most-promising state of the document, i.e., the state with the smallest current display area requirements, is selected and a transformation is applied to transform the document from its current state to a more promising state of the document, if possible. As soon as a state is created that contains a document version that is 'good enough', the search can be halted and that version of the document is returned to the client device for rendering. Alternatively, the search is continued until all content of the original page is contained or represented in a set of good-enough subpages. If the search is exhausted and no document version can be found that is good enough,

then the best document found during the search is returned to the client device for rendering. If there are hard size constraints that are not met by the best document, a more destructive transformation is applied that breaks documents up in the middle of paragraphs.

**[0058]** Fig. 3 shows how different transformations applied to a document 200 result in different resulting re-authored sub-pages 210, 220 and 230. Depending on the information supplied by the user to the systems and methods of this invention, one of the sub-pages 210, 220 and 230 would be selected as the "best" re-authored page. Then, if further re-authoring is required, for example, to generate good-enough subpages for the content removed from the first sub-page, or if the best sub-page is not yet "good enough", additional transformations could be applied to the subpages resulting from the selected best re-authored sub-page 210, 220 or 230 or to further re-author the selected best re-authored subpage 210, 220 or 230.

**[0059]** Heuristic information is used in several places by the document re-authoring systems and methods according to this invention, including: the order in which various transformation techniques are applied to a given state; the pre-conditions for each transformation technique; and the determination of when a document version or subpage is 'good enough'. In general, transformations which make minor changes to the document are preferred over those which make more extensive changes. For example, reducing images by 25% is preferable to reducing the images by 75%.

**[0060]** The pre-conditions for each transformation technique specify the other transformations with which that transformation can be combined. For example, it makes no sense to apply both full outlining and first sentence elision to the same document. The pre-conditions also specify the requirements on the content and structure of the document that the technique is being applied to. For example, the Full Outlining transform should be applied only when there are at least three section headers in the document or sub-page being re-authored. The current condition for 'good enough' is fairly simplistic. That is, the search is stopped when the area required by a document or sub-page is a predetermined multiple of the screen area of the client display. In general, this predetermined multiple is greater than I, and, in one exemplary embodiment, is 2.5. This higher multiple merely assumes that the user doesn't mind scrolling the display a little in one direction.

**[0061]** When a transformation is applied to a document it can result in the document's contents being split into multiple, smaller "sub-pages", as shown in Fig. 2. However, each of these sub-pages may still be too large to download and display on the client. To address this problem, the document re-authoring systems and methods of this invention keep a list of the sub-pages generated by each sequence of transformations attached to the state representing the resulting document version. Once the good-enough version of the document is selected, which is really only a good-enough version of the *first* sub-page delivered to the client, the list of generated sub-pages for that version is added to a global list of pages to be re-authored. The document re-authoring systems and methods of this invention then re-author each of these to-be-re-authored pages until all of the resulting sub-pages can be delivered to the client. This procedure is shown in pseudocode below, where "reauthor" refers to the best-first re-authoring process described above for a single input page.

```
Digestor(initial_page)
    to_be_reauthored = { initial_page }
    to_deliver = {}
    while(to_be_reauthored != {})
        next_page = pop(to_be_reauthored)
        best_version_state = reauthor(next_page)
        to_deliver.append(best_version_state.page)
        to_be_reauthored.append(best_version_state.sub_pages)
return to_deliver
```

**[0062]** All re-authored sub-pages are cached as transformed parse trees. As the user navigates a transformed document and requests subpages, the corresponding parse trees are rendered and sent to the client.

**[0063]** The document re-authoring systems and methods of this invention re-author document by first parsing the document and constructing a parse tree or abstract syntax tree (AST) representation of the document. The document re-authoring systems and methods of this invention then apply a series of transformations to the parse tree. Then, the document re-authoring systems and methods of this invention map each resulting transformed parse tree back into a document representation, which may be in a document format that is different from the input format of the original document.

**[0064]** Document transforms are implemented using a standard procedure that includes a condition function that takes a state node in the document version space and returns true if the transform should be applied to the state, and

an action function that is called when the transform is actually applied to a state to produce a new state containing a new document version, a new measure of quality, and the resulting sub-pages. Three types of transforms can be defined—1) those which are always run on a page before the planning process starts; 2) those used in the best-first planning process; and 3) those which are always run on a page before it translated from the final abstract syntax tree back into a surface form such as HTML.

**[0065]** Transformations manipulate the parse tree, in the state they are applied to, in order to produce a new version of the document. The manipulations are similar to those described in S. Bonhomme et al., "Interactively Restructuring HTML Documents", Fifth International World Wide Web Conference, Paris, France, May 1996. Whenever portions ofthe parse tree are elided or transformed, an HTML hypertext link is added into the parse tree to reference the node identifiers of all affected parse tree subtrees, enabling users to request the original portions of the document that have been modified during re-authoring.

**[0066]** The document re-authoring systems and methods of this invention also keep track of which combinations of transforms have already been tried, via a global list of transform sets, assuming that all transformations are commutative, to ensure that no duplicate states are ever constructed.

**[0067]** One exemplary document re-authoring system and method according to this invention, as described above, has been implemented as an HTTP proxy server. The HTTP proxy server accepts a request for an HTML document, retrieves the document from the specified HTTP server, parses the HTML document, constructs the parse tree, or abstract syntax tree, from the retrieved HTML document, labels each of the parse tree nodes with a unique identifier, and then retrieves any embedded images so that the size of the retrieved images can be determined, as necessary. Once this has been accomplished, the document re-authoring systems and methods of this invention are initialized with a state containing the parse tree for the original retrieved document. During each re-authoring cycle, the document re-authoring systems and methods of this invention select the state with the best document version so far, then select the best applicable transformation technique and apply the selected transformation, resulting in a new state and a new document version being generated. It is assumed that the convolution of transformations is always commutative, and several checks are used by the re-authoring software systems and methods of this invention to ensure that redundant states are not constructed.

**[0068]** In one exemplary embodiment of the document re-authoring systems and methods of this invention, fifteen transformation techniques were implemented: FullOutline, Outline ToH1, Outline ToH2, Outline ToH3, Outline ToH4, Outline ToH5, Outline ToH6, FirstsentenceElision, ReduceImages25%, ReduceImages50%, ReduceImages75%, ElideAllInmges, FirstImageOnly, BookendImages, and ReduceFontSize.

**[0069]** This exemplary embodiment of the document re-authoring systems and methods of this invention has been implemented in the Java programming language. In addition to functioning as a true proxy server, this HTTP proxy server system can also respond to requests for certain uniform resource locators with documents generated by the HTTP proxy server itself. This is used to provide the user with forms-based control over the HTTP proxy server and the document re-authoring systems and methods. This exemplary embodiment of the document re-authoring system can process even very compex pages in less than 2 seconds on a 200Mhz Pentium, using Symantec's Java JIT compiler.

**[0070]** The first thing that a user of the document re-authoring software systems and methods of this invention must do is specify the size of display for the device being used and indicate the font size of the default browser font being used. This information is needed in order to estimate the screen area requirements of text blocks. To do this, the user requests a specific control uniform resource locator from the HTTP proxy server, resulting in delivery of the form 300 shown in Fig. 4.

**[0071]** Once a user has configured the document re-authoring system, the user can start retrieving documents from a distributed network, such as the World Wide Web. The original page 400 and the re-authored page 410 shown in Fig. 5 illustrate the re-authoring capability of the document re-authoring systems and methods of this invention. In this example, this exemplary embodiment of the document re-authoring systems and methods of this invention chose to use 25% image reduction in combination with first sentence elision to render the displayed page 410 from the original page 400. The re-authored page 410 is then displayed on a browser window 420. In this exemplary embodiment of the re-authoring systems and methods of this invention, immediately following retrieval of a page, the user can request a trace of the re-authoring session to determine which transformations had been applied, by requesting another control uniform resource locator from the HTTP proxy server.

**[0072]** Fig. 6 shows one exemplary embodiment of an environment 500 in which the automatic document re-authoring systems and methods and/or the automatic document filtering systems and methods of this invention will be implemented. As shown in Fig. 6, the environment 500 includes a limited display area device 510 that has a display having a display area that is significantly limited relative to the display area of a monitor for a desktop or a laptop computer. As shown in Fig. 6, the environment 500 further includes a transmitter/receiver communication system 550, a host node 570 of a distributed network and the remaining portions 590 of the distributed network.

**[0073]** In the environment 500, the limited display area device 510 will normally be a personal digital assistance

(PDA), a cellular phone or the like that is connected by a wireless communication channel 530 to the transmitter/receiver communication system 550. Thus, as shown in Fig. 6, the limited display area device 510 will normally include an antenna 520, while the transmitter/receiver communication system 550 will normally include a corresponding antenna 540. The limited display area device 510 will normally communicate with the transmitter/recciver communication system 550 over the wireless communications channel 530 using radio frequency signals transmitted between the antennas 520 and 540.

**[0074]** The transmitter/receiver communication system 550 converts the analog or digital signals received from the limited display area device 510 over the communications channel 530 in to a form usable by the host node 570 of the distributed network. The transmitter/receiver communication system 550 then outputs the signals received over the communications channel 530 over a communication link 560 to the host node 570 of the distributed network. It should be appreciated that the communication link 560 can be any known or later-developed communication structure capable of transmitting the appropriate signals between the transmitter/receiver communication system 550 and the host node of the distributed network 570. Because the exact structure of the transmitter/receiver communication system 550 and the communication link 560 will be a matter of design choice depending upon how these elements are implemented, but such design choices will be readily apparent and predictable to those of ordinary skill in the art, these elements will not be further described.

**[0075]** It should also be appreciated that the limited display area device 510 can also be connected to the host node 570 of the distributed network by other than the wireless communication channel 530, such as a communication link 522. That is, the communication link 522 could be any other known communications structure, such as a local area network, a wide area network, a modem connection over the public switched telephone network or a cable television system, or the like. For example, the user of the limited display area device 510, rather than communicating over the wireless communication channel 530, could connect the limited display area device 510 to the public switch telephone network using a modem. The user would then dial directly into the host node 570 of the distributed network.

**[0076]** Regardless of how the host node 570 of the distributed network is ultimately connected to the limited display area device 510, once the host node 570 of the distributed network receives a request for a document to be transmitted to the limited display area device 510, the host node 570 of the distributed network first determines if the requested document is located locally on the host node 570 of the distributed network. If the requested document is not located locally, the host node 570 of the distributed network communicates over a communication structure 580 to the remaining portions 590 of the distributed network to request the document. The particular node of the remaining portions 590 of the distributed network storing that document ultimately will receive the request from the host node 570 over the communication structure 580 and will return the requested document to the host node 570 over the communication structure 580. It should be appreciated that the communication structure 580 can be any known or later-developed communication structure and protocol system for linking together widely located nodes of a distributed network.

**[0077]** Once the host node 570 of the distributed network receives the requested document, an HTTP proxy server executing on the host node 570 of the distributed network re-authors the requested document based on the previously-provided information about the limited display area device 510. A first re-authored page is ther transmitted by the host node 570 over either the wireless communication link 530 or the communication link 522 to the limited display area device 510. As the user reviews the delivered page, the user may determine that viewing additional information removed from the re-authored page is required. In this case, the user will send a request over one of the wireless communication link 530 or the communication link 522 to the host node 570 of the distributed network to obtain the desired re-authored sub-page. The host node 570, in response to this request, transmits a further re-authored sub-page of the original document to the limited display area device 510 over one of the wireless communication channel 530 or the communication link 522.

**[0078]** Fig. 7 shows this information flow in greater detail. As shown in Fig. 7, when the user of the limited display area device 510 wishes to review a particular document presiding on a distributed network, the user sends a request for the particular document from the limited display area device 510 to an HTTP proxy server 571 residing on the host node 570 of the distributed network. The HTTP proxy server 571 then transmits the request for the particular document to the particular remote node 591 on the distributed network that stores the requested page. The particular remote node 591 returns the requested original document to a document re-authoring system 600 residing on the HTTP proxy server 571. The document re-authoring system 600 re-authors the original document into a plurality of subdocuments that are each capable, as closely as possible, of being displayed on the limited display area device 510. The document re-authoring system 600 then delivers the first re-authored to page to the limited display area device 510, while the other re-authored sub-pages are stored in a re-authored sub-page cache 636 of the document re-authoring system 600. Thus, when the user of the limited display area device 510 wishes to view information residing on one of the re-authored sub-pages stored in the re-authored sub-page cache 636, the user causes the limited display area device 510 to transmit a request for that sub-page. The requested cached sub-pages are delivered from the re-authored sub-page cache 636 to the limited display area device 510.

**[0079]** It should be appreciated that, while the HTTP server 571, the document re-authoring system 600 and the re-

authored subpage cache 636 are shown in Fig. 7 as independent elements, in general, these elements will be implemented as different portions of a single entity, such as different modules of a single software application.

**[0080]** Fig. 8 is a functional block diagram outlining in greater detail one exemplary embodiment of the document re-authoring system 600. As shown in Fig. 8, the document re-authoring system 600 includes a controller 610, an input/output interface 620, a memory 630, an abstract syntax tree generating circuit 640, a document size evaluation circuit 650, a transform circuit 660 and a tree-to-document remap circuit 670. each interconnected by a data/control bus 680. The communication links 522, 560 and 580 discussed above with respect to Fig. 6 are each connected to the input/output interface 620.

**[0081]** The memory 630 includes a number of functionally distinct portions, including an original page memory portion 631, a display device size memory portion 632, an abstract syntax tree memory portion 633, a search space portion 634, a transform memory 635, the re-authored page cache 636 described above with respect to Fig. 7, and a sub-pages to be re-authored list 637. The original page memory portion 631 stores the returned original document returned from the remote node 591 of the distributed network that stores the page requested by the limited display area device 510.

**[0082]** The display device size memory 632 stores a number of form documents used by the document re-authoring system 600 to obtain various parameters about the limited display area device 510 used by the document re-authoring system 600 to re-author a page for a particular limited display area device 510. The display device size memory 632 also stores the particular size parameters for at least one limited display area device 510. It should be appreciated there are a number of different possible ways of implementing the document re-authoring system 600 relative to the various parameters about the limited display area device 510. In one exemplary embodiment, the document re-authoring system 600 can store the various parameters for a particular limited display area device 510 only for as long as that limited display area device 510 remains continuously connected to the document re-authoring system 600. In this case, each time a particular limited area device 510 is reconnected to the document re-authoring 600, the document re-authoring system 600 would send the various forms used to obtain the various parameters about the limited display area device 510 and the user would be required to re-supply these various parameters each time the document re-authoring system 610 was initially accessed.

**[0083]** While this reduces the required size for the display device size memory 632 and does not require any system for identifying a particular limited display area device 510, this system places a larger burden on the user of the limited display area device 510 or requires a process for automating the supply of information from the limited display area device 510 to the document re-authoring system 600. This automation could be provided, for example, by the document re-authoring system 600 requesting the information from the limited display area device 510. If the information has already been entered by the user during a previous session with the document re-authonng system 600, and that information was stored at that time on the limited display area device 510, the user would not need to be actively involved in re supplying the information to document re-authoring system 600.

**[0084]** Alternatively, the information could be stored in the display device size memory 632, along with an identification code that the user can cause to be supplied from the limited display area device 510 when beginning a session with the documen re-authoring system 600. By supplying the identification code to the document re-authoring system 600, the user again would not be required to re-supply all of the various parameters about the limited display area device 510 each time the document re-authoring system 600 is accessed.

**[0085]** In any case, the document re-authoring system 600 uses the various parameter about the limited display area device 510, as described above, when re-authoring the original page stored in the original page memory 631 so that each re-authored page will fit, as closely as possible, on to the small display area of the limited display area device 510.

**[0086]** The abstract syntax tree memory portion 633 stores the abstract syntax tree generated form the original document stored in the original page memory 631 by the abstract syntax tree generating circuit 640. The transform memory portion 635 stores the various transforms described above, as well as the conditions under which each transform can be applied and the conditions regarding which transforms are not usable with various other ones of the transforms. The transform memory 635 also stores an indication of the desirability of applying any particular transform to a particular original or re-authored page. That is, as described above, the various transforms have general order that emphasis applying a more limited transform, such as reducing an image by a small about, over a more radical transform, such as reducing an image by a large amount or removing the image completely.

**[0087]** The re-authored page cache 636 stores the abstract syntax tree corresponding to each re-authored page or sub-page as the document size evaluation circuit indicates that the abstract syntax tree for a particular re-authored page or sub-page is good enough, based on the various parameters about the limited display area device 510 stored in the display device size memory 632. The sub-pages to be re-authored list 637 stores the abstract syntax trees for those sub-pages generated by transforming the original document or an earlier sub-page. These sub-pages will generally contain the images of any reduced-size images or any elided images, as well as the full text of any text segments that have had content elided from them.

**[0088]** Finally, the search space memory 634 stores a number of states generated by the transform circuit 660 as it

applies the various transforms stored in the transform 635 to either the original document stored in the original page memory 631 or to various sub-pages stored in the sub-pages to be re-authored list 637, based on the particular state of the search space currently being manipulated.

**[0089]** In particular, each state i in the search space 634 includes an evaluation value portion, a transformed abstract syntax tree portion and a sub-page list portion. The evaluation valued portion stores the evaluation value generated for the re-authored page or sub-page corresponding to the state i generated by the document size evaluation circuit 650. The transformed abstract syntax tree portion stores the transformed abstract syntax tree for the state i generated by the transform circuit 660 by applying one of the transforms in the transform memory 635 to the parent state to the state i. The sub-page list portion stores the list of sub-pages generated to store any original content removed from the page corresponding to the state i when the transform circuit 660 applies the particular transform used to generate that state i.

**[0090]** It should be appreciated that state 0 corresponds to the original document stored in the original page memory 631. In particular, the evaluation value portion of state 0 corresponds to the evaluation value generated for the original document before any re-authoring. In this state 0, the transformed abstract syntax tree portion stores the original un-transformed abstract syntax tree generated by the abstract syntax tree generating circuit for the original document. Finally, before state 0, the sub-page list will be empty, as the original document contains all ofthe original information and therefore, no sub-pages are required.

**[0091]** Fig. 9 graphically illustrates various states stored in the search space memory portion 634. In particular, Fig. 9 shows a document comprising a section header, a text paragraph, and an image. As shown in Fig. 9, in the initial state, i.e., state 0, the original document has not been transformed. This initial state also shows the original rating, i. e., the evaluation value, generated for the original document. Fig. 9 also shows the state 1 generated from the state 0 by applying the "elide all images" transformed to the document of state 0. As shown in state 1, the re-authored sub-page of state 1 contains the section header and the text but does not contain the image. Rather, in place of the image, the re-authored sub-page of state 1 contains a link labeled "IMG" that links the re-authored page of state 1 to the sub-page storing the image elided from the re-authored subpage of state 1. State 1 also indicates the evaluation value for this re-authored document. As shown in Fig. 9, the size requirements for the re-authored page are now one-quarter of the size requirements of the original, un-re-authored page.

**[0092]** Fig. 9 also indicates that two additional states, state 2 and state 3, were generated by applying other transforms to the document of state 0. Finally, Fig. 9 shows three additional states, state 4, state 5 and state 6, generated by applying additional transforms to the re-authored document of state 1 or to the sub-page of state 1. For example, if the sub-page containing the image is still too large to be displayed on the limited display area device 510, an intermediate sub-page generated by applying the "reduced image by 25%", the "reduce image by 50%", or the "reduce image by 75%" transforms to the image to obtain a re-authored document good enough to be displayed on the limited display area device 510.

**[0093]** Currently, in operation, the document re-authoring system 600 of Fig. 8 receives the returned original document over the communication link 580. The received or general document is input through the input/output interface 620 and is stored in the original page memory 631 under the control of the controller 610. Then, the abstract syntax tree generating circuit 640, under control of the controller 610, inputs the original document from the original page memory portion 631 and generates an abstract syntax tree from that original document. The abstract syntax tree generated by the abstract syntax tree generating circuit 640 is then stored in the abstract syntax tree memory portion 633 of the memory 630 under control of the controller 610.

**[0094]** The document size evaluation circuit 650 then inputs, under control of the controller 610, the abstract syntax tree corresponding to the original document stored in the original page memory 631 and the various parameters from the display device size memory 632 about the particular limited display area device 510 to which the re-authored documents are to be returned. The document size evaluation circuit 650 then generates an evaluation value and stores that evaluation value in state 0 of the search space memory portion 634. The document size evaluation circuit 650 also outputs an indication to the controller 610 whether the document of state 0 is good enough for outputting it to the limited display area device 510 over one of the communication links 522 or 560. If the original document is already good enough, the original document is immediately returned without further transformation.

**[0095]** Then, the transform circuit 660, under control of the controller 610, inputs the document of state 0, as represented by the abstract syntax tree for that state, and applies one of the transforms stored in the transform memory 635 to the abstract syntax tree of the input state. In particular, the transform circuit 660 first determines, for the current state i, whether the selected transform should be applied to the current state i of the document. For example, as described above, if the current state i of the document does not contain any images, there is no point in applying any of the image reduction or elision transforms to this state of the document. Furthermore, if the "elide all but first image" transform has already been applied to obtain the current state i of the image, there is no point of applying the "elide all but first and last images" transform to this current state i.

**[0096]** Assuming the current transform selected by the transform circuit 660 is properly applicable to the current state

i of the document, as indicated by the transformed abstract syntax tree for the current state i, the transform circuit 660 applies that transform to the abstract syntax tree for that state to generate a child state j. The child state j includes the further transformed abstract syntax tree and a sub-page list indicating the sub-pages that remain to be transformed based on the content elided from the original document necessary to reach this child state j. Finally, the document size evaluation circuit 650, under control of the controller 610, evaluates the document obtained in the child state j to determine if that resulting document is good enough for outputting to the limited display area device 510. That evaluation value is then stored in the newly-created child state j.

**[0097]** After the transform circuit 660 has generated the new child state j, the transform abstract syntax tree for that state j is output to the document size evaluation circuit 650 for evaluating the size requirements of the document corresponding to the state j.

**[0096]** Once the abstract syntax tree for the first page of the transformed document is determined to be good enough, that abstract syntax tree is output to the tree-to-document remap circuit 670, which renders the first re-authored sub-page from that abstract syntax tree. That first re-authored sub-page is output from the tree-to-document remap circuit 670 to the input/output interface 620 and ultimately is transmitted to the limited area display device 510. At the same time, the transform circuit 660 continues to apply additional transforms to any subpages resulting from transforming the original document into the first good-enough re-authored subpage. As each such subpage is transformed into a good-enough subpage, the abstract syntax tree for each such good-enough subpage is stored in the re-authored page cache 636 until a request for that subpage is received by the document re-authoring system 600 from the limited area display device 510.

**[0099]** Once a request for that subpage is received by the document re-authoring system 600, the abstract syntax tree for that requested subpage is output to the tree-to-document remap circuit 670, which renders the requested re-authored sub-page from that abstract syntax tree. That requested re-authored sub-page is output from the treeto-document remap circuit 670 to the input/output interface 620 and ultimately is transmitted to the limited area display device 510.

**[0100]** It should be understood that each ofthe circuits and other elements shown in Figs. 6-8 can be implemented as portions of suitably programmed general purpose computers. Alternatively, each of the circuits shown in Figs. 6-8 can be implemented as physically distinct hardware circuits within one or more ASICs, or using FPGAs, PDLs, PLAs, or PALs, or using discreet logic elements or discreet circuit elements. The particular form each of the circuits shown in Figs. 6-8 will take is a design choice and will be obvious and predictable to those of ordinary skill in the art.

**[0101]** It should also be appreciated that the links 522, 560 and 580 can by any known or later-developed device or system for connecting the limited display area device 510 to the host node 570 or the host node 570 to the transmitter/receiver communication system 550 or the remaining portions 590 of the distributed network. Thus, the links 522, 560 and 580 each be implemented as a direct cable connection, a connection over a wide-area network or a local-area network, a connection over an intranet, or a connection over the Internet. In general, the links 522, 560 and 580 can be any known or later-developed connection system or structure usable to connect the corresponding apparatus to the host node 570 over the distributed network.

**[0102]** It should further be appreciated that the document re-authoring system 600 is preferably implemented on a programmed general purpose computer. However, the document re-authoring system 600 can also be implemented on special purpose computer, a programmed microprocessor or microcontroller as a peripheral integrated circuit elements, and ASIC or other integrated circuit, a digital signal processor, a hardwired electronic or logic circuit such as a discreet element circuit, a programmable logic device such as PLD, PLA, FPGA or PAL, or the like. In general, any device, capable of implementing a finite state machine that is in turn capable of implementing the flowcharts shown in Figs. 11A-14, can be used to implement the document re-authoring system 600.

**[0103]** The memory 630 shown in Fig. 8 is preferably implemented using static or dynamic RAM. However, the memory 630 can also be implemented using a floppy disk and disk drive, a writeable optical disk and disk drive, a hard drive, flash memory or any other know or later-developed volatile or non-volatile alterable memory. In addition, the memory 630 can further include one or more portions storing control programs for the controller 610. In general, such control programs are preferably stored using non-volatile memory, such as flash memory, a ROM, a PROM, and EPROM or EEPROM, a CD-ROM and disk drive, or any other known or later-developed alterable or non-alterable non-volatile memory.

**[0104]** Fig. 10 shows another exemplary original document and the abstract syntax tree that is generated from that document. As shown in Fig. 10, the document includes an image, a table having two rows of three columns each, and a text paragraph. The resulting abstract syntax tree generated from this page includes a root node labeled "Page". Three intermediate nodes, "Image", "Table" and "Paragraph" corresponding to each of the image, the table and the text paragraph, respectively, extend from the root "Page" node. Furthermore, as shown in Fig. 10, two intermediate nodes, "Row 1" and "Row 2", corresponding to each of the two rows, respectivley, extend from the intermediate "table" node. Finally, three nodes, corresponding to each of the three cells in each row, respectively, extend from each of the "Row 1" and "Row 2" nodes.

**[0105]** To re-author the page shown in Fig. 10, for example, the first transform to be applied would generally replace the full size image with a node representing an image reduced by 25%. Then, a new abstract syntax tree having a root node corresponding to the full-sized image would be formed and linked by a hypertext link to the reduced image node of the transformed abstract syntax tree. If the re-authored page having the image reduced by 25% is not yet good enough, the image reduction transformation reducing the image by 50%, 75% and then completely removing the image would be applied.in turn to the original document until a good-enough image was obtained. In each case, the abstract syntax tree would contain a link from the transformed node corresponding to the image to the separate abstract syntax tree containing the full-sized image. If removing the image completely is still insufficient to result in a good-enough re-authored document, the table transform can be applied, as described above, to transform the table into a set of linked individual cells, or the First Sentence Elision transform can be applied to move the text paragraph into a separate subpage.

**[0106]** Figs. 11A and 11B are a flowchart outlining one exemplary method for re-authoring a page according to this invention. As shown in Fig. 11, control begins in step S100 and continues to step S110, where a user connects a device having a limited display area to a re-authoring system according to this invention. Then, in step S120, the re-authoring system transmits one or more parameter forms to the user to obtain the necessary information about the limited display area necessary to be able to re-author a requested page for display on the limited display area device. Then, in step S130, the re-authoring system inputs the parameter information from the user and stores the input parameter information in a memory. Control then continues to step S140.

**[0107]** As indicated above with respect to Figs. 6 and 7, the parameter information gathering process outlined in steps S120 and S130 can be automated so that the user does not have to be actively involved in performing steps S120 and S130. Alternatively, as shown in optional step S135, steps S120 and 130 can be replaced by step S135. In step S135, the user either actively inputs, or the limited display area device automatically outputs, an identification code to the re-authoring system identifying previously-stored parameter information for this particular limited display area device. Control then again continues to step S140.

**[0108]** In step S140, a request for a document on the distributed network is output to the re-authoring system from the user using the limited display area device. Then, in step S150, the re-authoring system obtains the requested document from the distributed network. Next, in step S160, the obtained document is parsed to build an abstract syntax tree of that document. Then, in step S170, an evaluation value for the obtained original document is generated from the abstract syntax tree. Control then continues to step S180.

**[0109]** In step S180, the evaluation value is analyzed to determine if the obtained document is good enough to b displayed on the limited display area device without any re-authoring. If so, control jumps to step S340. Otherwise, control continues to step S190.

**[0110]** In step S190, one or more pre-re-authoring transforms are applied to the abstract syntax tree of the obtained, original document. These pre-re-authoring transforms are used, for example, to remove portions of the original document that do not contain any content but that consume display area For example, such portions of the obtained document include banners and other graphical elements that are merely identifying links to other pages or portions of the page. These contentless images are replaced by text links. However, because such transforms do not actually remove any content from the image, re-authoring the page in this way does not require the removed portions to be retained. Other portions that can be removed without effecting the content of the original document include formatting commands that add whitespace and other contentless esthetic formatting to the original document. Finally, other transforms can be applied that convert the various fonts of a document to a single standard font to eliminate unnecessary display area requirements of large and complicated fonts.

**[0111]** Once the pre-re-authoring transforms are applied in step S190, control continues to step S200, where an evaluation value for the pre-re-authored original document is generated. Then, in step S210, the pre-re-authored documents evaluation value is checked to determine if the pre-re-authored document is good enough to be displayed on the limited display area device. If so, control again jumps to step S340. Otherwise, control continues to step S220.

**[0112]** In step S220, state 0 of the search space, corresponding to the pre-re-authored document, is selected as the current state of the search space. Then, in step S230, a first transform is selected as the current transform. Then, in step S240, a determination is made whether the current transform can be applied to the abstract syntax tree of the current state. As outlined above, various ones of the transforms have conditions that indicate whether that transform can be efficiently applied to the current re-authored document or whether the current transform is properly combinable with previously applied transforms. If the current re-authored document corresponding to the current state is such that the current transform can be efficiently applied and does not conflict with any previously applied transforms, control continues to step S250. Otherwise, control jumps to step S290.

**[0113]** In step S250, the current state is transformed to a child state using the current transform and the resulting child state, including the transformed abstract syntax tree and any resulting sub-pages, are added to the search space. Then, in step S260, an evaluation value is generated for the document corresponding to the transformed abstract syntax tree corresponding to the child state generated in step S250. Next, in step S270, the evaluation value is analyzed

to determine if the document corresponding to the child state generated in step S250 is good enough to be displayed on the limited display area device. If the evaluation value indicates the re-authored document or sub-page is good enough, control jumps to step S310. Otherwise, control continues to step S280.

**[0114]** In step S280, a determination is made whether all transforms have been applied to the current state. If all of the transforms have not been applied, control continues to step S290. Otherwise, control jumps to step S300.

**[0115]** In step S290, the next transform is selected as the current transform and control jumps back to step S240. In contrast, in step S300, the state of the search space having the best evaluation value is selected as the current state. Control then jumps back to step S230.

**[0116]** In step S310, the document or sub-page defined by the current state is added to the re-authored page cache as a first re-authored page or a next re-authored sub-page suitable for delivery to the requesting limited display area device. Then, in step S320, a determination is made whether there are any sub-pages resulting from the good-enough sub-page that has been added to the re-authored page cache. If there are any such sub-pages that still need to be re-authored, control continues to step S330. Otherwise, control jumps to step S340.

**[0117]** In step S330, a state of the search space corresponding to one of the sub-pages to be re-authored is selected as the current state. Control then jumps back to step S230. In contrast, since there are no further sub-pages that need to be re-authored, in step S340, the first re-authored page is output to the requesting limited display area device. Then, in step S350, the control routine ends.

**[0118]** Fig. 12 outlines one exemplary embodiment of an elision transform according to this invention. As shown in Fig. 12, the elision transform routine begins in step S400, and continues to step S410, where a portion of a current page or sub-page to be removed is selected. Then, in step S420, the selected portion is copied into a new sub-page. Next, in step S430, an identifier is generated for the selected portion. In general, the identifier will be generated using some content of the selected portion. For example, if the selected portion is a paragraph or other text string, the identifier will be the first sentence or the first portion of the first sentence of the selected text portion. If the selected portion is an image, the identifier could be a portion of text used to identify the image in the web document. Control then continues to step S440.

**[0119]** In step S440, a link is generated to link the current page or sub-page with generated sub-page. Then, in step S450, the selected portion is removed from the current page or sub-page and the identifier and the link are added to the current page. Next, in step S640, the control routine stops.

**[0120]** Fig. 13 outlines one exemplary embodiment of a table transform according to this invention. As shown in Fig. 13, the table transform begins in step S500 and continues to step S505, where a top level table is selected as the current table. Then, in step S510, the current table is checked to determine if there are any nested tables in the current table. If so, control continues to step S515. Otherwise, control jumps to step S520. In step S515, one nested table of the current table is selected as the new current table. Control then jumps back to step S510, to determine if there are nested tables in this nested table selected as the current table.

**[0121]** Once there are no nested tables in the current table, in step S520, the current table is checked to determine if there are any sidebars in the current table. If so, control continues to step S525. Otherwise, control jumps to step S535. In step S525, a link list is generated from all of the links in all of the sidebars of the current table. Then, in step S530, the link list is placed at the end of the current table. Control then continues to step S535.

**[0122]** In step S535, the current table is divided into two or more portions. In particular, as indicated above, one method for dividing the current table into portions is to divide each cell of the table into a separate portion. Then, in step S540, each portion of the current table is copied into a separate new sub-page, and "Next" and "Previous" links are added to each such sub-page. Next, in step S545, the current table is replaced with the set of linked sub-pages generated in step S540. Control then continues to step S550.

**[0123]** In step S550, the current table is checked to determine if it is the top level table. If not, there is at least one higher level table that still needs to be divided into portions. Accordingly, control continues to step S555. Otherwise, control jumps to step S560.

**[0124]** In step S555, the table that contains the current table is selected as the new current table. Control then jumps back to step S510, to determine if there any more nested tables in the current table. In contrast, in step S560, the control routine ends.

**[0125]** Fig. 14 is a flowchart outlining one exemplary embodiment of an image reduction transformation according to this invention. Beginning in step S600, the image reduction transformation continues to step S610, where the image to be reduced in the current sub-page is selected. Then, the reduced image is generated based on the reduction factor associated with the particular image reduction transformation being applied. Then, in step S630, the current sub-page is analyzed to determine if the selected image has been previously reduced. If so, control jumps to step S670. Otherwise, control continues to step S640.

**[0126]** In step S640, the selected image is copied to a new sub-page. Next, in step S650, a link to the new sub-page is generated. Then, in step S660, the full-size image is removed from the current page or sub-page, and the reduced image and the generated link are added to the current page to form the re-authored page. Control then jumps to step

S680.

**[0127]** In contrast, in step S670, rather than moving the full-sized image from the current sub-page, the old previously reduced image is removed from the current sub-page and the new reduced image is added to the current sub-page. However, because the current sub-page should already have a link to the previously-created sub-page containing the full-size image, it is not necessary to again add the link to the current sub-page or to create a new sub-page storing that full-sized image. Control then continues to step S680, where the control routine ends.

**[0128]** Even with perfect automatic re-authoring of documents, there is often simply too much information in a typical web document to make serendipitous cellular phone web browsing a pleasurable or profitable past-time, due to the very small, text-only-type display used in cellular phones. Typically, these devices and services will be used to find and present information that the user is specifically looking for. That is, these devices and services will be used for targeted information search and extraction. The document filtering systems and methods of this invention allow users to extract only portions of documents that they are interested in, via a simple, end-user scripting language that combines structural page navigation commands with regular expression pattern matching and report generation functions.

**[0129]** The SPHINX system, as described in R. Miller et al., "SPHINX: a framework for creating personal, site-specific Web crawlers", Seventh International World-Wide Web Conference, Brisbane, Australia, April 1998, provides a visual tool that lets users create custom "personal" web crawlers that are similar in functionality to the filtering mechanism of the systems and methods of this invention. The Internet Scrapbook, as described in A. Sugiura et al., "Internet Scrapbook: automating Web browsing tasks by programming-by-demonstration", Seventh International World-Wide Web Conference, Brisbane, Australia, April 1998, allow users to visually select elements from web pages and then updates these elements in a "scrapbook" when the web pages change, providing a function that is similar to the page element retrieval for a particular page of the systems and methods of this invention. Several commercial products also provide similar functionality for other applications, such as, for example, corporate reporting or database population. Lanacom's Headliner Pro, as described in Lanacom, Inc., http://www.headliner.com, and OnDisplay's CenterStage, as described in OnDisplay, Inc., http://www.ondisplay.com, both provide visual editors that let users specify which structural parts of web pages to extract. However, neither of those systems provide users with any ability to extract content based on regular expressions or keywords.

**[0130]** The document filtering systems and methods of this invention have the capability to extract partial information from a document based on commands written by a user in a high-level scripting language. The document filtering systems and methods of this invention combine page structure navigation, regular expression matching, site traversal, i.e., web crawling, and iterative matching, in addition to re-authoring of the extracted information using the document re-authoring systems and methods of this invention described above.

**[0131]** A filter script is simply entered into a text file and saved on a web server. The filter script is executed whenever a user requests its Uniform Resource Locator. A filter script will typically load a target web page, traverse to particular locations within that web page, which are described structurally and/or by regular expressions, extract the content found at those locations, and then send the extracted content through the document re-authoring system to be properly formatted before being returned to the user.

**[0132]** The document filtering systems and methods of this invention take advantage of the parse tree creation and navigation of the document re-authoring systems and methods of this invention, by providing a simple set of HTML document navigation options that use the concept of a "current context" in the HTML document. The current context is analogous to a "cursor" in database programming, in that it refers to a location within HTML the document.

**[0133]** In actuality, the current context refers to a node in the HTML parse tree. The navigation commands serve to move this reference around within the tree until a desired part of the HTML document is found, at which time the desired part can be extracted. For example, Fig. 10 shows an HTML document and its corresponding parse tree. When the document is first loaded, by executing a "GO URL" command, the current context is pointing at the root node of the parse tree, which essentially refers to the entire document.

**[0134]** Fig. 15 shows one exemplary embodiment of the document re-authoring system 600 further including a filter circuit 690 that implements the document filtering systems and methods outlined herein. In particular, the filter circuit 690, under control of the controller 610, inputs a requested filter, requested by the user over one of the communication links 522 or 560, that is supplied from a node of the distributed network storing such a filter over the communication link 580. The filter circuit 690 then inputs the requested document from the node of the distributed network storing the requested document and filters the requested document to extract the requested page elements. The filter circuit 690 stores these extracted page elements in the original page memory 631 in place of the original document initially stored there. The document re-authoring system 600 then operates on these extracted page elements as if they were the original document to be re-authored.

**[0135]** In extracting the page elements from the original document, the filter circuit 690 uses the abstract syntax tree generated by the abstract syntax tree generating circuit from the original document and stored in the abstract syntax tree memory 633.

**[0136]** Fig. 16 outlines one exemplary embodiment of the information flow when the requested document is also to

be filtered. As shown in Fig. 16, after a request for filter is output by the limited display area device 510 to the HTTP proxy server 571, the request for filter is forwarded by the HTTP proxy server 571 to a remote node 592 of the new distributed network that stores the requested filter. The remote node 592 storing the requested filter returns the requested filter to the document filter 690. The document filter 690 then requests, under control of the controller 610, the document from the remote node 591 of the distributed network that stores the request page. The remote node 591 storing the requested page returns the document to the document filter 690. The document filter 690 then filters the returned document using the filter returned from the remote node 592 and the abstract syntax tree generated by the abstract syntax tree generating circuit 640. The document filter 690 returns the extracted page elements to the document re-authoring system 600 where the extracted page elements are treated as an original document for re-authoring as described above.

**[0137]** There are three types of page navigation commands, those which go *into* the current context to select more specific content, those which go *out* from the current context to enclosing structures, and those which traverse the page sequentially from the start of the current context, for example, to navigate to the *next* structure of some kind, which may or may not be properly contained within the current context.

**[0138]** The simplest type of navigation command goes *into* the current context. For example, given the document and current context shown in Fig. 10, executing the command "GO ROW 2" results in the current context being moved to the second table row object within the current context, as shown in Fig. 17.

**[0139]** The current context can also be enlarged, i.e., moved up the parse tree towards the root node, by using a "GO ENCLOSING" command. For example, given the document and context shown in Fig. 17, a "GO ENCLOSING TABLE" command results in the current context shown in Fig. 18.

**[0140]** Finally, the current context can be moved forwards or backwards among the objects in a page in a sequential manner, as they appear to a user. This is accomplished by moving the current context forwards or backwards from its current location within a prefix traversal of the parse tree. This results in a search that first is performed within the current context, then continues with the objects that follow the current context on the page. For example, a "GO PREVIOUS IMAGE" command moves to the previous image found sequentially from the current context.

**[0141]** In addition to named page elements, navigation commands can also be specified using regular expressions. For example, a "GO NEXT" "DOW\sJONES\s*(\d+)\s*POINTS"" command moves the current context to the next match of the specified regular expression, using a prefix traversal of text blocks on the page. The filtering systems and methods of this invention are able to demarcate sub-expressions and recall them into output strings.

**[0142]** The simple navigation commands described above can also be used to navigate among a set of linked web pages through the use of the "LINKEDPAGE" page object type. For example, a "GO FIRST LINKEDPAGE" command moves to the first hypertext link within the current context, loads the referenced page and moves the current context to the root of that document's parse tree, while a "GO ENCLOSING LINKEDPAGE" command returns the current context to the hypertext link that led to the document currently being processed.

**[0143]** Traversal between pages is handled by a stack of script activations, each of which pairs script state information (including current context) with a particular Uniform Resource Locator and a parse tree. This facilitates rapid navigation back and forth among linked pages and is required to support the "GO ENCLOSING LINKEDPAGE" command.

**[0144]** Once the current context has been moved to a page object that is of interest, a "REPORT" command is used to extract it. The "REPORT" command can be issued several times within a filter script, in which case the extracted page elements are concatenated. The "REPORT" command can also be used to insert arbitrary strings into the output, which can contain sub-strings from regular expression pattern matching. For example, the "REPORT "Dow:\1"" command adds the string "Dow:" plus a substring identified by the identifier "1" extracted during a regular expression match to the filter's output.

**[0145]** Often the user does not know in advance how many page elements of a particular kind will exist on a web page. For example, the number of news article paragraphs in a daily e-zine will generally not be known in advance. The "FOREACH" command addresses this lack of information by executing a sequence of commands for every page element found within the current context that meets a specified criteria. When used with a "LINKEDPAGE" target, this provides the functionality of a web spider that can visit all of the linked pages within a web site. In the following examples the ellipses represent sequences of valid filter commands:

**[0146]** A "FOREACH PARAGRAPH" command moves to each paragraph within the current context in turn DO... END and executes the specified commands.

**[0147]** A "FOREACH LINKEDPAGE" command loads each page that is reachable through hypertext links from the DO... END current page in turn and executes the specified commands.

**[0148]** Whenever a filter encounters any kind of error, including navigation failures, regular expression matching failures, or web page retrieval error, it simply begins the next iteration of the innermost "FOREACH" loop in which the offending command is embedded. If the error occurred at the top level of a filter, the filter halts execution and produces any pending output.

**[0149]** The document re-authoring systems and methods of this invention do a good job of automatically re-authoring

documents for display on devices with small screens. One exemplary embodiment of the document re-authoring systems and methods of this invention have been informally tested on a wide range of pages for a number of screen sizes. This exemplary embodiment of the document re-authoring systems and methods of this invention produced output that is legible and navigable.

**[0150]** In one exemplary embodiment, the document re-authoring systems and methods of this invention simply add up the space requirements of all images and text to arrive at an estimate of the screen area requirements for a document. This is adequate for fairly dense documents with minimal structure, such as those in a Xerox Annual Report, but works poorly for documents with a lot of whitespace or which use advanced layout techniques, such as, for example, tables. In a second exemplary embodiment, the document re-authoring systems and methods of this invention includes a size estimator that performs much of the work performed by a browser in formatting each document version onto a display area. Factors other than required screen area may also need to be included, such as actual width requirements of the re-authored document, because users don't like to scroll horizontally, bandwidth requirements, and aesthetic measures.

**[0151]** Users should be able to adjust the various heuristics used in the document re-authoring systems and methods of this invention to suit their taste. For example, the user could specify the relative preference of the transformation techniques, or specify that some transforms not be used at all. At a higher level of abstraction, the user could express their preferences for a set of trade-offs, such as 'more content' vs. 'larger representation'. In addition, the re-authoring systems and methods of this invention could be moved to the client and coupled with the browser so that the user could dynamically apply and undo different transformations until the user achieves a result the user likes.

**[0152]** The automatic document re-authoring systems and methods of this invention, and in particular, the exemplary embodiment of the HTTP proxy server described above, are preferably implemented on a programmed general purpose computer. However, the automatic document re-authoring systems and methods of this invention, and in particular, the HTTP proxy server described above, can also be implemented on a special purpose computer, a programmed microprocessor or microcontroller and peripheral integrated circuit elements, an ASIC or other integrated circuit, a digital signal processor, a hardwired electronic or logic circuit such as a discrete element circuit, a programmable logic device such as a PLD, PLA, FPGA or PAL, or the like. In general, any device, capable of implementing a finite state machine, can be used to implement the automatic document re-authoring system and method of this invention, and in particular, the HTTP proxy server described above.

**[0153]** The automatic document re-authoring systems and methods according to this invention can be performed by invoking a stand-alone re-authoring program running on the HTTP proxy server described above, or can be performed through a plug-in to a conventional web browser, such as Netscape Navigator or the like.

**[0154]** Furthermore, while the automatic document re-authoring systems and methods of this invention have been described in relation to re-authoring documents obtained from the world-wide web, the automatic re-authoring systems and methods of this invention can be used to re-author documents obtained from any distributed network, such as a local area network, a wide area network, an intranet, the Internet, or any other distributed processing and storage network.

**[0155]** While this invention has been described in conjunction with the specific embodiments outlined above, it is evident that many alternatives, modifications and variations will be apparent to those skilled in the art. Accordingly, the preferred embodiments of the invention set forth above are intended to be illustrative, not limiting. Various changes may be made without departing from the spirit and scope of the invention.

## Claims

1. A method for automatically re-authoring a document, comprising:

   parsing the document;
   transforming the parsed document into a transformed document;
   generating an evaluation value from the transformed document;
   determining if the evaluation value meets at least one evaluation criterion,
   if the evalution value for the transformed document does not meet the at least one criterion, repeating the transforming, generating and determining steps using a different transform; and
   if the evaluation value for the transformed document meets the at least one criterion, outputting the transformed document.

2. The method of claim 1, wherein:

   parsing the document comprises generating an abstract syntax tree from the document; and
   transforming the parsed document comprises transforming the abstract syntax tree into at least one trans-

formed abstract syntax tree.

3. The method of claim 1 or claim 2, wherein transforming the parsed document comprises:

selecting a transform;
determining if the transform can properly be applied to the parsed document;
if the transform can properly be applied, transforming the parsed document into the transformed document using the selected transform; and
if the transform cannot properly be appliet, repeating the selecting and determining steps for a different transform.

4. The method of any of claims 1-3, wherein transforming the parsed document into the transformed document comprises at least one of outlining sections of the document, removing contentless portions from the document, removing content from the document, reducing a size of ad least one image within the document, removing at least one image from the document, removing at least one table cell from the document, and summarizing text within the document.

5. The method of claim 4. wherein:

outlining sections ofthe document preferably comprises:

identifying sections within the document,
identifying a section header and a document portion for each section,
placing each identified document portion into a separate subpage,
removing the identified document portions from the parsed document to form a transformed document containing only the identified sections headers,
converting each of the identified section headers into a link to the corresponding subpage, and
linking the separate subpages together and to the transformed document;

educing a size of at least one image within the document preferably comprises:

identifying at lead one image within the document,
placing each identified image into a separate subpage,
generating a reduced version of each identified image,
removing each identified image from the document and inserting the reduced version of each removed image to form the transformed document, and
adding, for each removed image, a link into the reduced version of that image to the subpage containing that removed image;

removing at least one image from the document preferably comprises one of removing all images from the document, removing all but the first image from the document, and removing all but the first and last images from the document;
removing at least one table cell from the document preferably comprises:

determining if the table contains any sidebars of links,
if the table contains any sidebars, converting the sidebars into a list of links as a last cell of the table,
identifying all but the first cell of the table,
adding each identified cell to a separate subpage,
replacing the table with the first cell to form the transformed document, and
linking the separate subpages together and to the transformed document, and
removing at least one table cell from the document preferably further comprises:

determining if that cell is a nested table,
if that cell is not a nested table, adding that cell to the separate subpage, and
if that cell is a nested table, repeating the determining, converting, identifying, adding, replacing and linking steps; and

removing contentless portions from the document preferably comprises at least one of replacing sequenc-

es of page breaks or paragraph breaks with a single page break or paragraph break, removing indenting from the document; converting text strings of the document to at least one of a single font and font size, removing bullets from the document, removing background space from the document and removing banner images from the document.

6. The method of any of claims 1-5, wherein, if no transform results in a transformed document that has an evaluation value that meets the a least one evaluation criterion, the method further comprises:

selecting the transformed document having the evaluation value that most closely meets the evaluation value; and
repeating the transforming, generating and determining steps on the selected transformed document using art additiond transform.

7. The method of any of claims 1-6, wherein:

transforming the parsed document into a transformed document comprises generating at least one subpage; and
when a transformed document meets the at least one evaluation criterion, the method further comprises:
generating an evaluation value for each generated subpage for that transformed document;
determining, for each subpage, if the evaluation value for that subpage meets the at least one evaluation criterion;
for each subpage, if the evaluation value for that subpage does not meet the at least one criterion, performing the transforming, generating and determining steps on that subpage using an additional one of the transforms to generate a transformed subpage; and
for each subpage, if that subpage meets the at least one criterion, identifying that subpage as ready to be output.

8. The method of claim 1, further comprising, after parsing the document:

optionally removing contentless portions from the document to form a pre-transformed document
generating an evaluation value from the document or the pre-transformed document;
determining if the evaluation value meets at least one evaluation criterion;
if the evaluation value for the document or the pre-transformed document does not meet the at least one criterion, performing the transforming, generating and determining steps using a first one of the transforms; and
if the evaluation valve for the document or the pre-transformed document meets the at least one criterion, outputting the document or the pre-transformed document without removing any content from the document.

9. The method of any of claims 1-8, wherein transforming the document comprises:

filtering the document to extract desired portions of the document; and
replacing the document with the extracted portions.

10. A document re-authoring system that automatically re-authors a document, comprising

a parse tree generating circuit that parses the document to generate a parse tree;
a transform circuit that transforms the parse tree using a first transform to generate a transformed parse tree representing a transformed document, and that preferably transforms the parse tree or the transformed parse tree using another transform to generate another transformed parse tree representing another transformed document; and
a document size evaluation circuit that evaluates the parse tree or the another transformed parse tree to determine if the document, the transformed document or the another transformed document meets at least one evaluation criterion;
wherein, when the document, the transformed document or the another transformed document meets the at least one evaluation criterion; the document, the transformed document or the another transformed document is output to a display device.

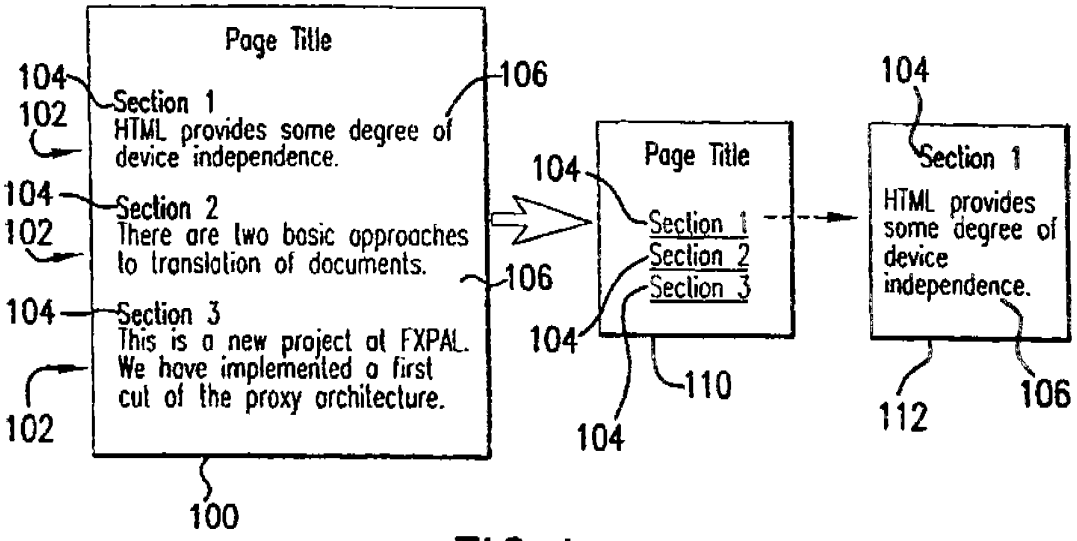

Page Title
104
Section 1
102
HTML provides some degree of device independence.
106
104
Section 2
102
There are two basic approaches to translation of documents.
106
104
Section 3
This is a new project at FXPAL. We have implemented a first cut of the proxy architecture.
102
100
104
Page Title
Section 1
Section 2
Section 3
104
110
104
104
Section 1
HTML provides some degree of device independence.
112
106

FIG.1

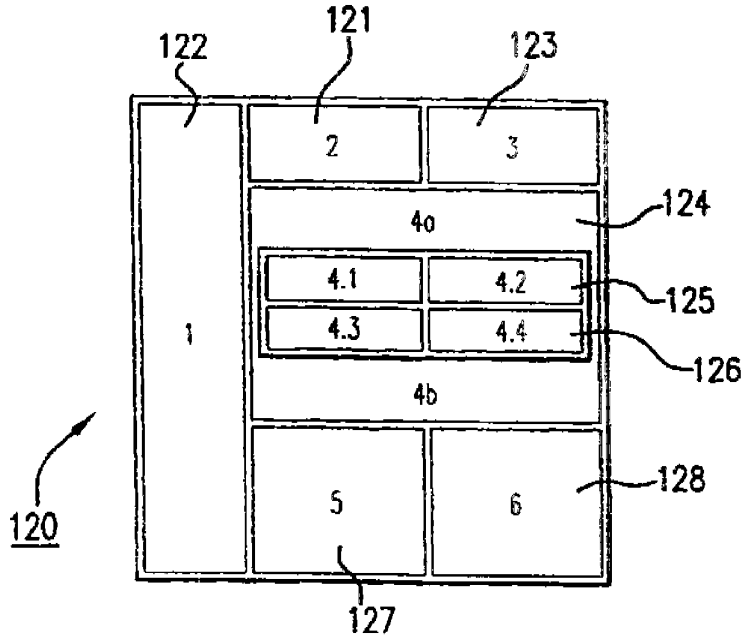

121
122
123
2
3
4a
124
4.1
4.2
125
1
4.3
4.4
126
4b
5
6
128
120
127

FIG.2

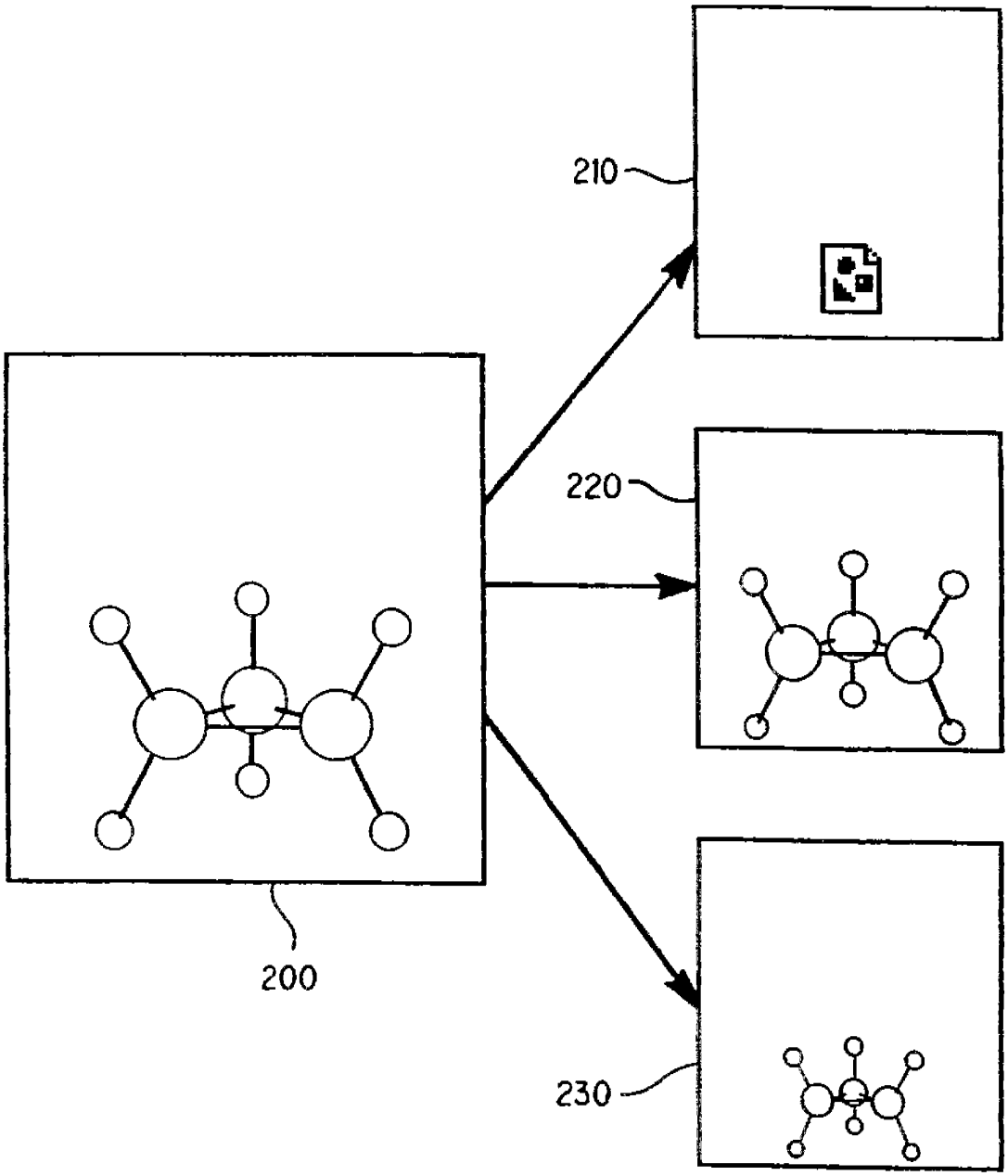



FIG. 3

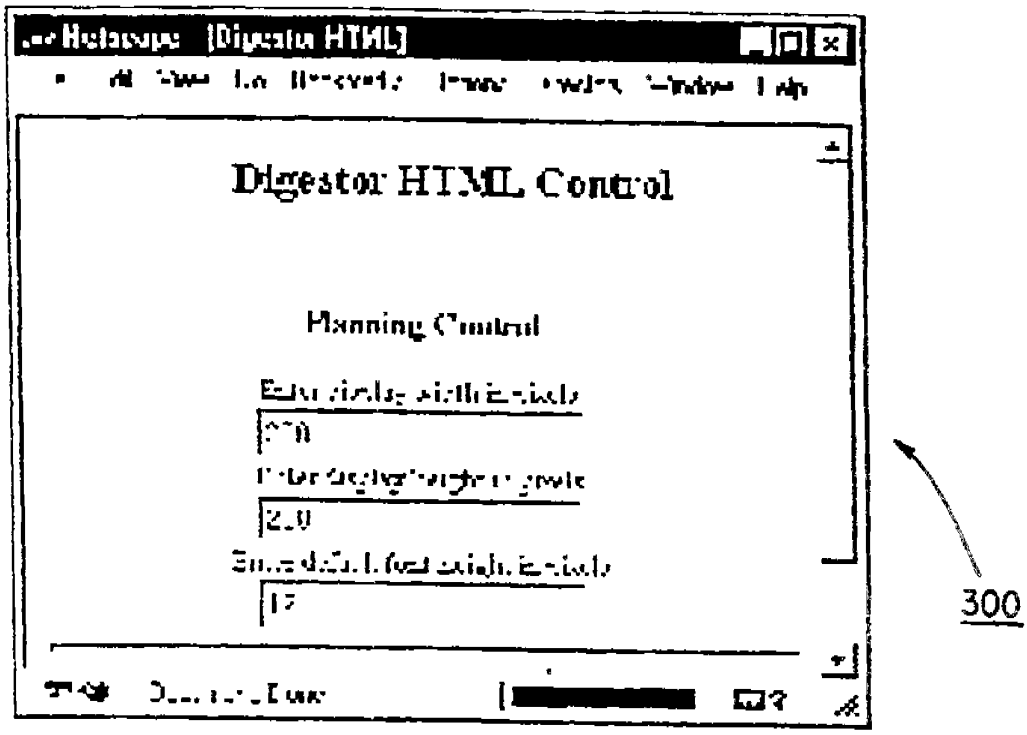



FIG. 4

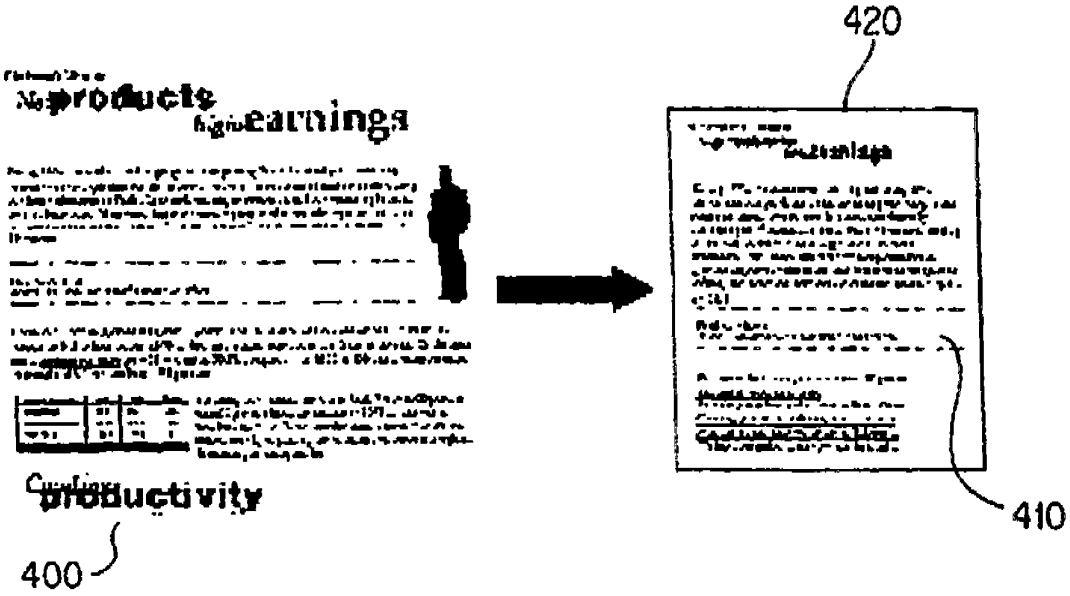



FIG. 5

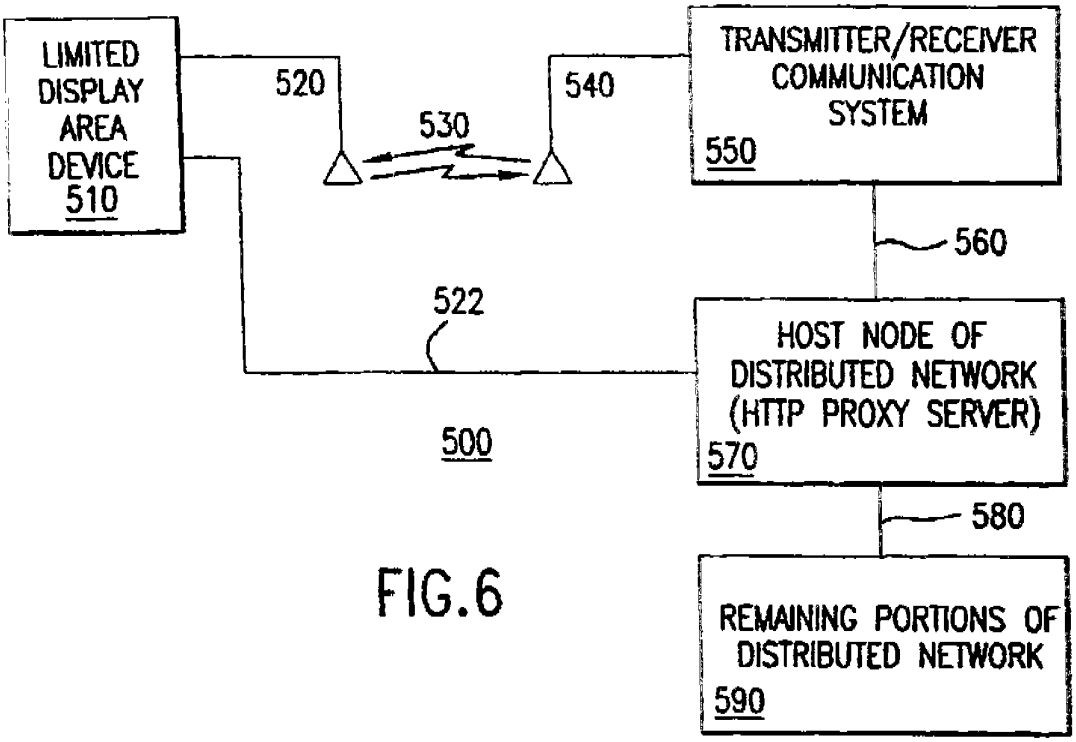

LIMITED DISPLAY AREA DEVICE 510
520
530
540
TRANSMITTER/RECEIVER COMMUNICATION SYSTEM 550
560
522
HOST NODE OF DISTRIBUTED NETWORK (HTTP PROXY SERVER) 570
500
580
REMAINING PORTIONS OF DISTRIBUTED NETWORK 590

FIG.6

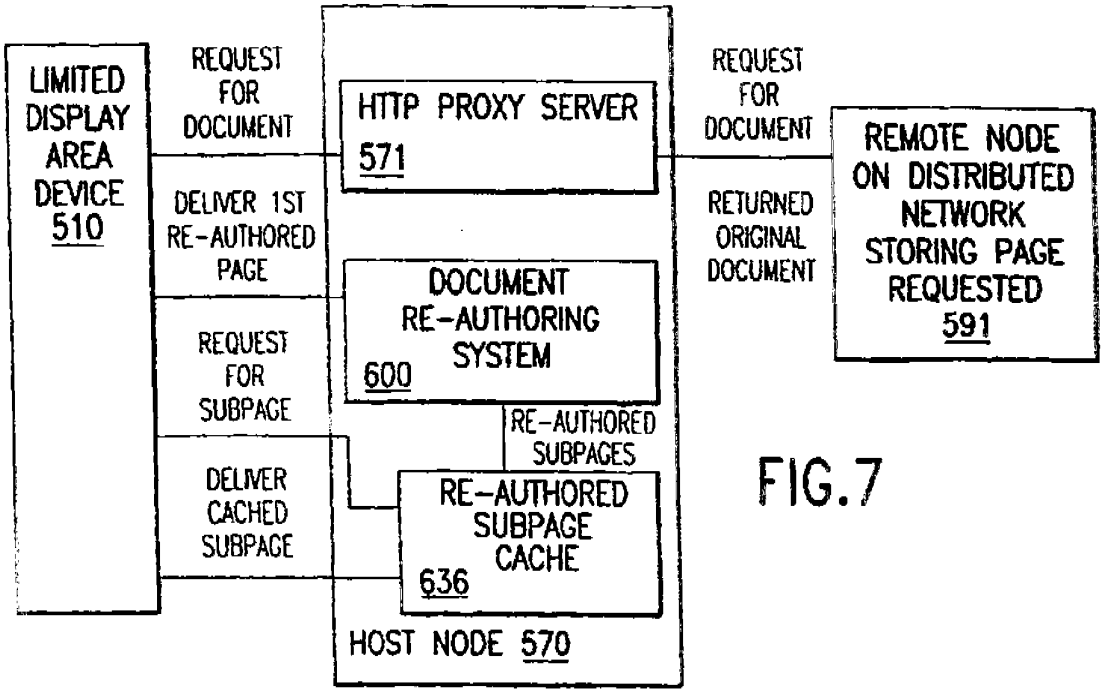

LIMITED DISPLAY AREA DEVICE 510
REQUEST FOR DOCUMENT
HTTP PROXY SERVER 571
REQUEST FOR DOCUMENT
REMOTE NODE ON DISTRIBUTED NETWORK STORING PAGE REQUESTED 591
DELIVER 1ST RE-AUTHORED PAGE
RETURNED ORIGINAL DOCUMENT
DOCUMENT RE-AUTHORING SYSTEM 600
REQUEST FOR SUBPAGE
RE-AUTHORED SUBPAGES
DELIVER CACHED SUBPAGE
RE-AUTHORED SUBPAGE CACHE 636
HOST NODE 570

FIG.7

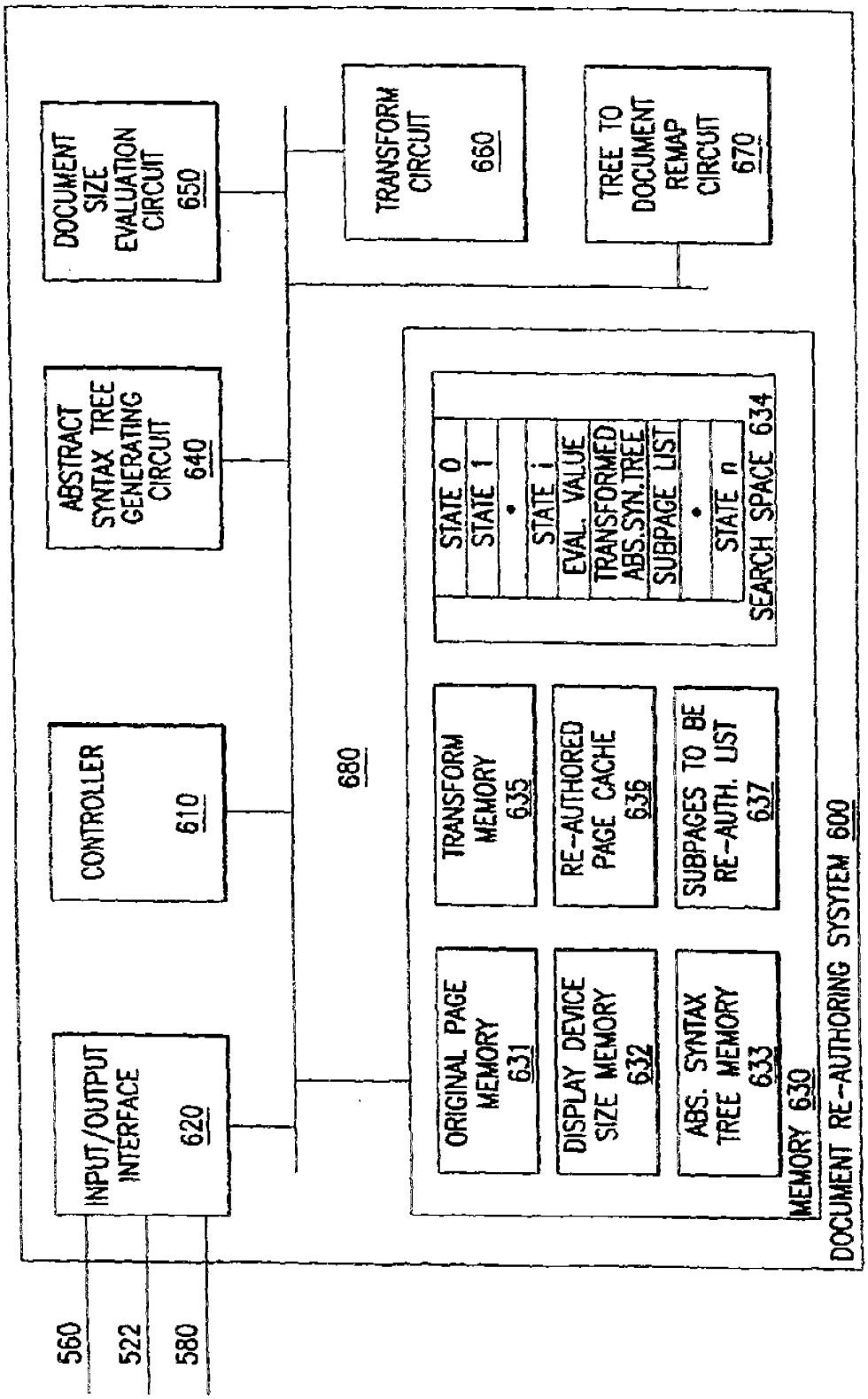

DOCUMENT RE-AUTHORING SYSTEM 600
MEMORY 630
ORIGINAL PAGE MEMORY 631
DISPLAY DEVICE SIZE MEMORY 632
ABS. SYNTAX TREE MEMORY 633
TRANSFORM MEMORY 635
RE-AUTHORED PAGE CACHE 636
SUBPAGES TO BE RE-AUTH. LIST 637
SEARCH SPACE 634
STATE 0
STATE 1
STATE i
EVAL. VALUE
TRANSFORMED ABS.SYN.TREE
SUBPAGE LIST
STATE n
INPUT/OUTPUT INTERFACE 620
CONTROLLER 610
ABSTRACT SYNTAX TREE GENERATING CIRCUIT 640
DOCUMENT SIZE EVALUATION CIRCUIT 650
TRANSFORM CIRCUIT 660
TREE TO DOCUMENT REMAP CIRCUIT 670
680
560
522
580

FIG.8

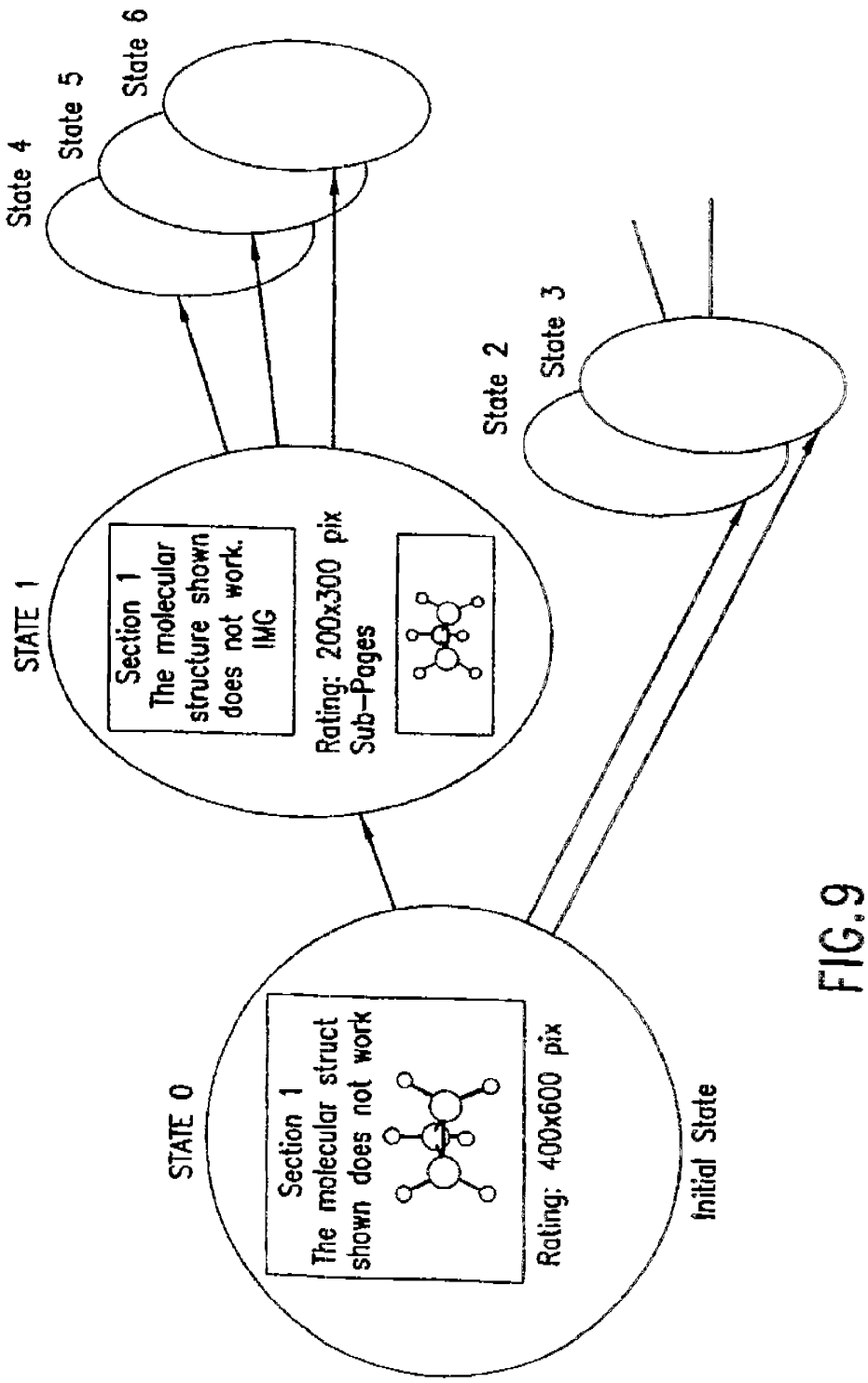

State 4
State 5
State 6
State 2
State 3
STATE 1
Section 1
The molecular structure shown does not work.
IMG
Rating: 200x300 pix
Sub-Pages
STATE 0
Section 1
The molecular struct shown does not work
Rating: 400x600 pix
Initial State

FIG.9

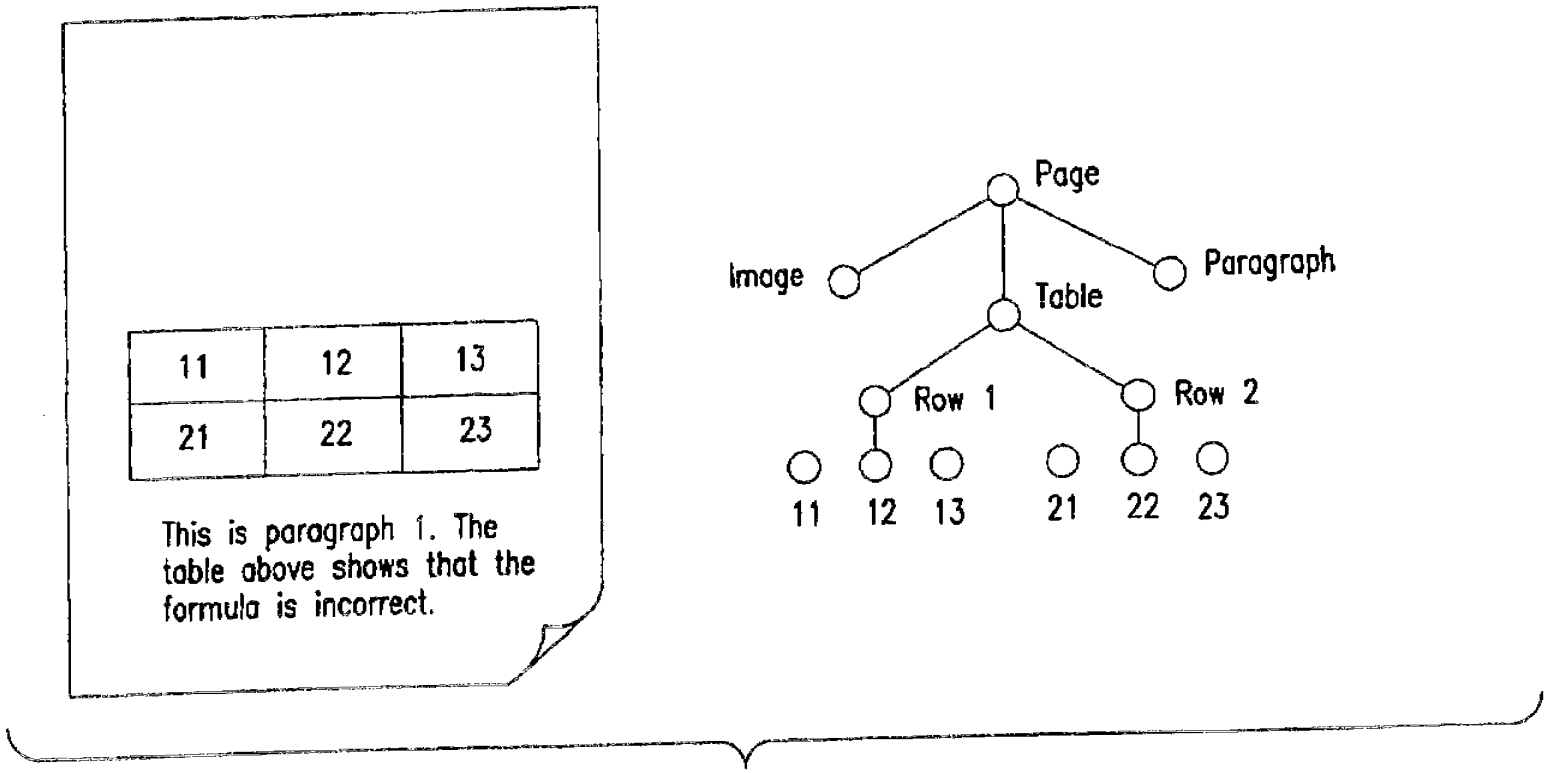



FIG.10

EP 0 949 571 A2

IPR2023-00330 Page 00515

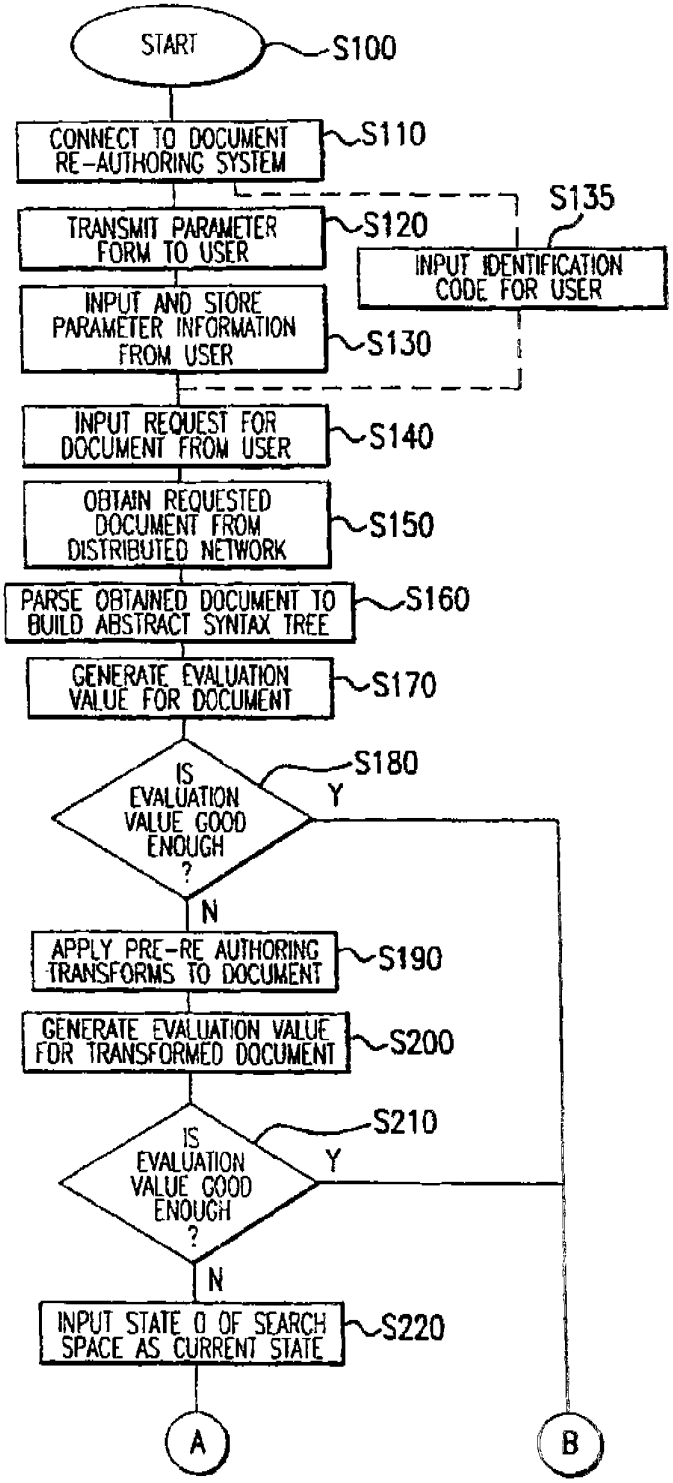

START
S100
CONNECT TO DOCUMENT RE-AUTHORING SYSTEM
S110
TRANSMIT PARAMETER FORM TO USER
S120
S135
INPUT IDENTIFICATION CODE FOR USER
INPUT AND STORE PARAMETER INFORMATION FROM USER
S130
INPUT REQUEST FOR DOCUMENT FROM USER
S140
OBTAIN REQUESTED DOCUMENT FROM DISTRIBUTED NETWORK
S150
PARSE OBTAINED DOCUMENT TO BUILD ABSTRACT SYNTAX TREE
S160
GENERATE EVALUATION VALUE FOR DOCUMENT
S170
IS EVALUATION VALUE GOOD ENOUGH ?
S180
Y
N
APPLY PRE-RE AUTHORING TRANSFORMS TO DOCUMENT
S190
GENERATE EVALUATION VALUE FOR TRANSFORMED DOCUMENT
S200
IS EVALUATION VALUE GOOD ENOUGH ?
S210
Y
N
INPUT STATE 0 OF SEARCH SPACE AS CURRENT STATE
S220
A
B

FIG.11A

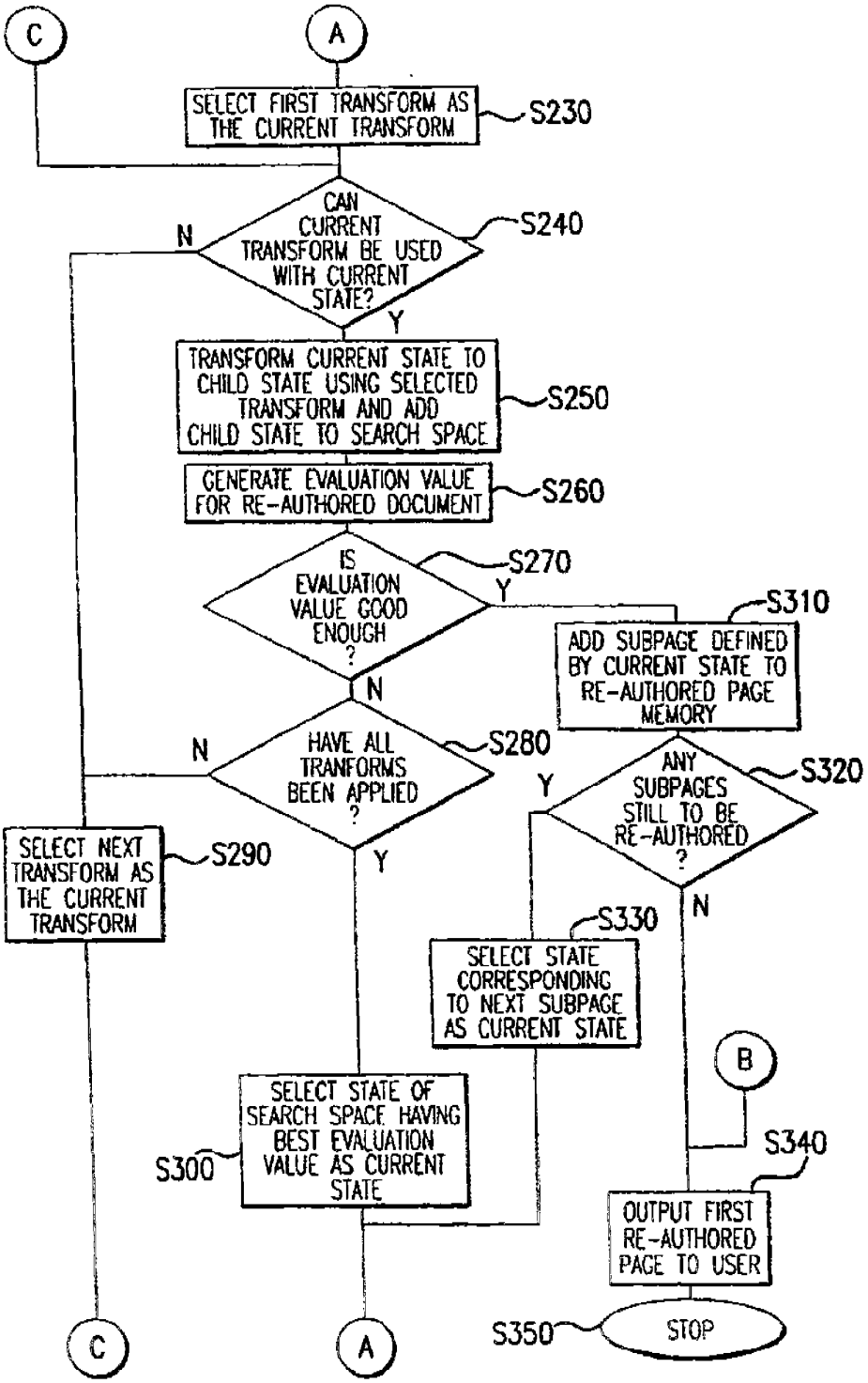

C
A
SELECT FIRST TRANSFORM AS THE CURRENT TRANSFORM
S230
CAN CURRENT TRANSFORM BE USED WITH CURRENT STATE?
S240
N
Y
TRANSFORM CURRENT STATE TO CHILD STATE USING SELECTED TRANSFORM AND ADD CHILD STATE TO SEARCH SPACE
S250
GENERATE EVALUATION VALUE FOR RE-AUTHORED DOCUMENT
S260
IS EVALUATION VALUE GOOD ENOUGH ?
S270
Y
N
S310
ADD SUBPAGE DEFINED BY CURRENT STATE TO RE-AUTHORED PAGE MEMORY
HAVE ALL TRANFORMS BEEN APPLIED ?
S280
N
Y
ANY SUBPAGES STILL TO BE RE-AUTHORED ?
S320
Y
N
SELECT NEXT TRANSFORM AS THE CURRENT TRANSFORM
S290
S330
SELECT STATE CORRESPONDING TO NEXT SUBPAGE AS CURRENT STATE
B
SELECT STATE OF SEARCH SPACE HAVING BEST EVALUATION VALUE AS CURRENT STATE
S300
S340
OUTPUT FIRST RE-AUTHORED PAGE TO USER
S350
STOP
C
A

FIG.11B

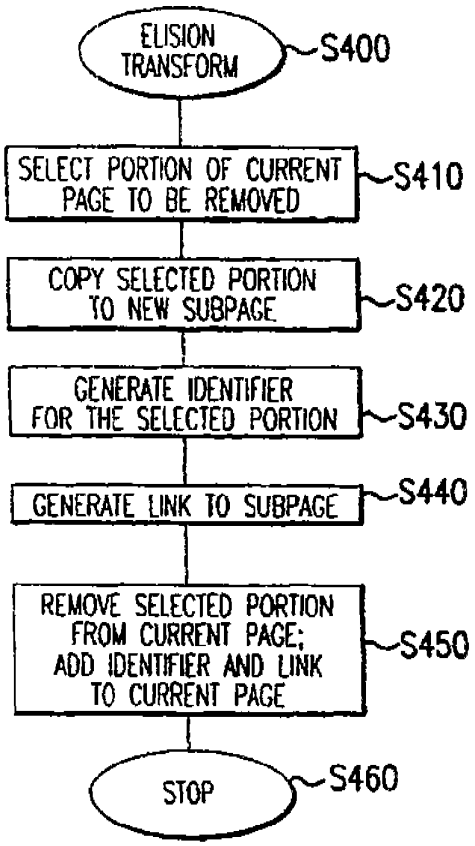

ELISION TRANSFORM
S400
SELECT PORTION OF CURRENT PAGE TO BE REMOVED
S410
COPY SELECTED PORTION TO NEW SUBPAGE
S420
GENERATE IDENTIFIER FOR THE SELECTED PORTION
S430
GENERATE LINK TO SUBPAGE
S440
REMOVE SELECTED PORTION FROM CURRENT PAGE; ADD IDENTIFIER AND LINK TO CURRENT PAGE
S450
STOP
S460

FIG.12

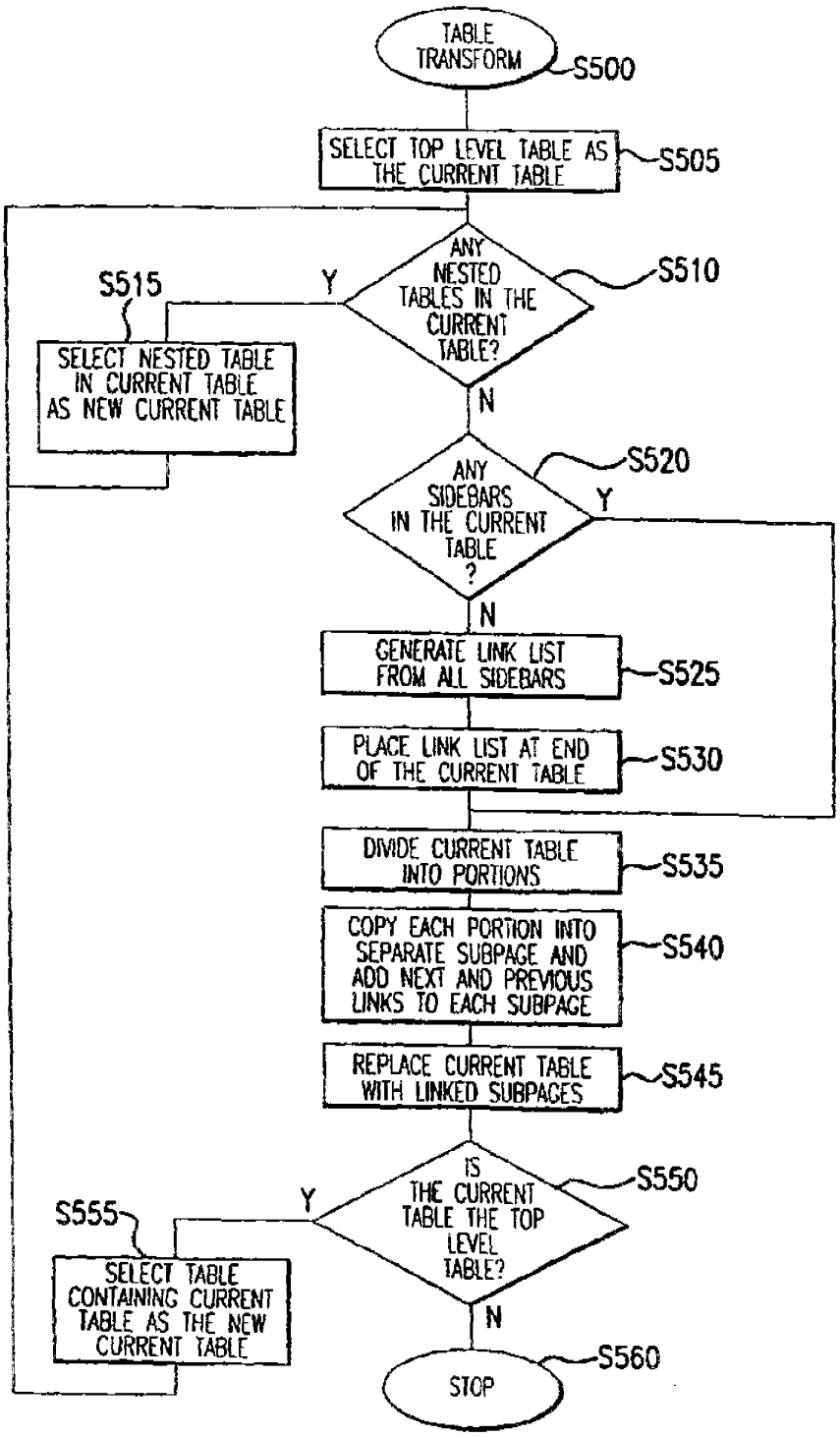

TABLE TRANSFORM
S500
SELECT TOP LEVEL TABLE AS THE CURRENT TABLE
S505
ANY NESTED TABLES IN THE CURRENT TABLE?
S510
Y
S515
SELECT NESTED TABLE IN CURRENT TABLE AS NEW CURRENT TABLE
N
ANY SIDEBARS IN THE CURRENT TABLE ?
S520
Y
N
GENERATE LINK LIST FROM ALL SIDEBARS
S525
PLACE LINK LIST AT END OF THE CURRENT TABLE
S530
DIVIDE CURRENT TABLE INTO PORTIONS
S535
COPY EACH PORTION INTO SEPARATE SUBPAGE AND ADD NEXT AND PREVIOUS LINKS TO EACH SUBPAGE
S540
REPLACE CURRENT TABLE WITH LINKED SUBPAGES
S545
IS THE CURRENT TABLE THE TOP LEVEL TABLE?
S550
Y
S555
SELECT TABLE CONTAINING CURRENT TABLE AS THE NEW CURRENT TABLE
N
STOP
S560

FIG.13

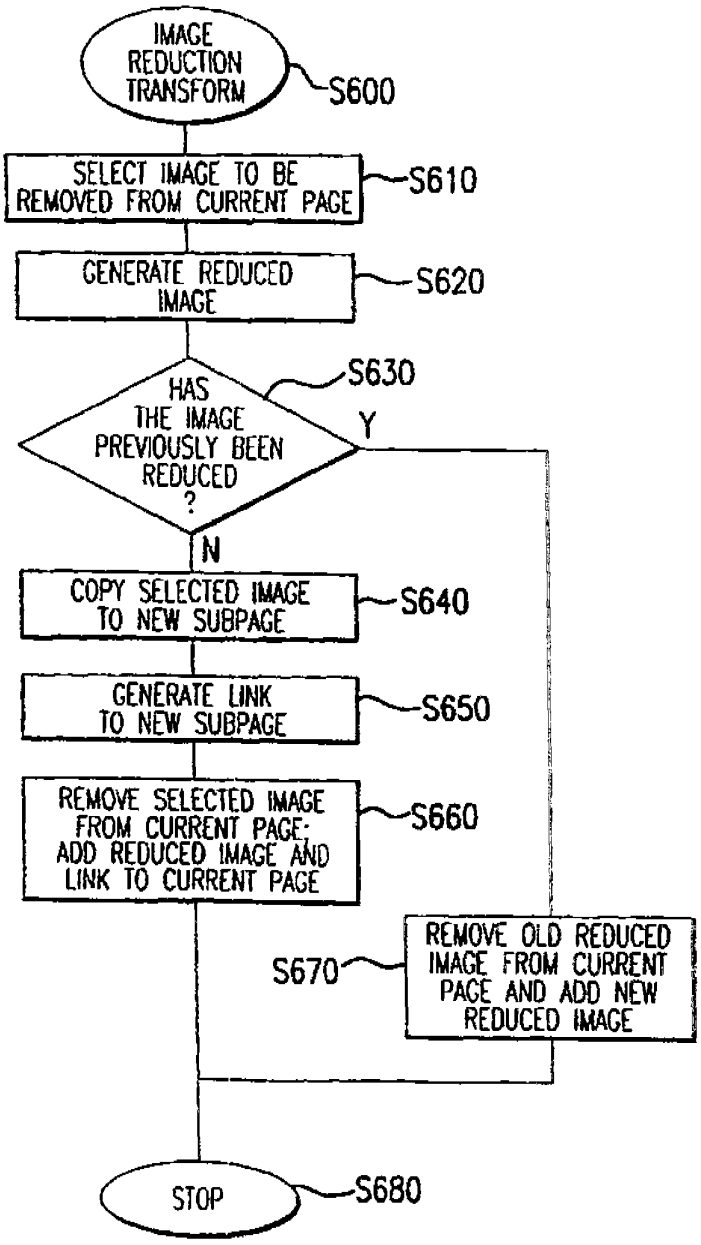

IMAGE REDUCTION TRANSFORM
S600
SELECT IMAGE TO BE REMOVED FROM CURRENT PAGE
S610
GENERATE REDUCED IMAGE
S620
HAS THE IMAGE PREVIOUSLY BEEN REDUCED ?
S630
Y
N
COPY SELECTED IMAGE TO NEW SUBPAGE
S640
GENERATE LINK TO NEW SUBPAGE
S650
REMOVE SELECTED IMAGE FROM CURRENT PAGE· ADD REDUCED IMAGE AND LINK TO CURRENT PAGE
S660
S670
REMOVE OLD REDUCED IMAGE FROM CURRENT PAGE AND ADD NEW REDUCED IMAGE
STOP
S680

FIG.14

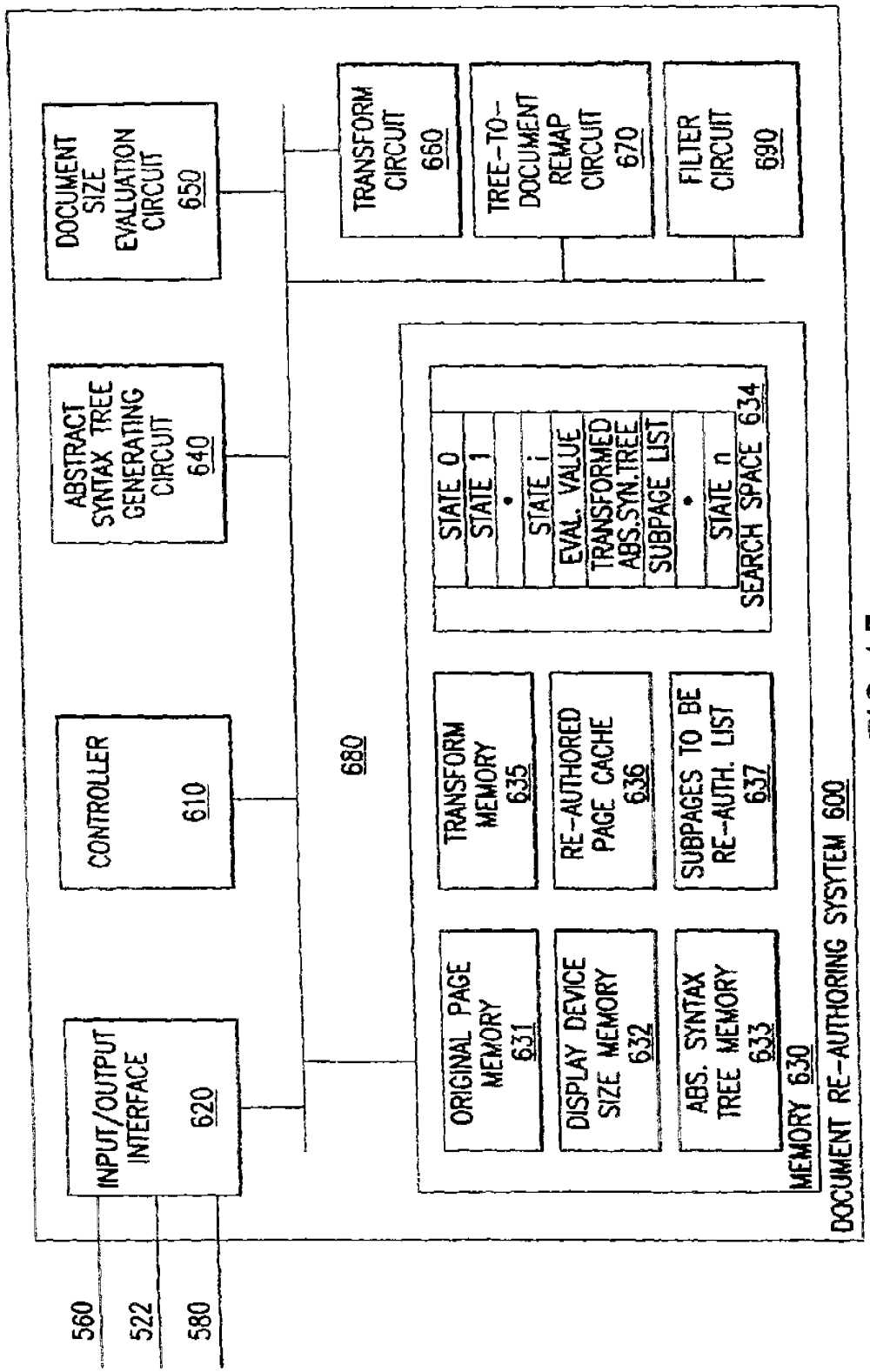

DOCUMENT RE-AUTHORING SYSTEM 600
INPUT/OUTPUT INTERFACE 620
CONTROLLER 610
680
DOCUMENT SIZE EVALUATION CIRCUIT 650
ABSTRACT SYNTAX TREE GENERATING CIRCUIT 640
TRANSFORM CIRCUIT 660
TREE-TO-DOCUMENT REMAP CIRCUIT 670
FILTER CIRCUIT 690
MEMORY 630
ORIGINAL PAGE MEMORY 631
DISPLAY DEVICE SIZE MEMORY 632
ABS. SYNTAX TREE MEMORY 633
TRANSFORM MEMORY 635
RE-AUTHORED PAGE CACHE 636
SUBPAGES TO BE RE-AUTH. LIST 637
SEARCH SPACE 634
STATE 0
STATE 1
STATE i
EVAL. VALUE
TRANSFORMED ABS.SYN.TREE
SUBPAGE LIST
STATE n
560
522
580

FIG.15

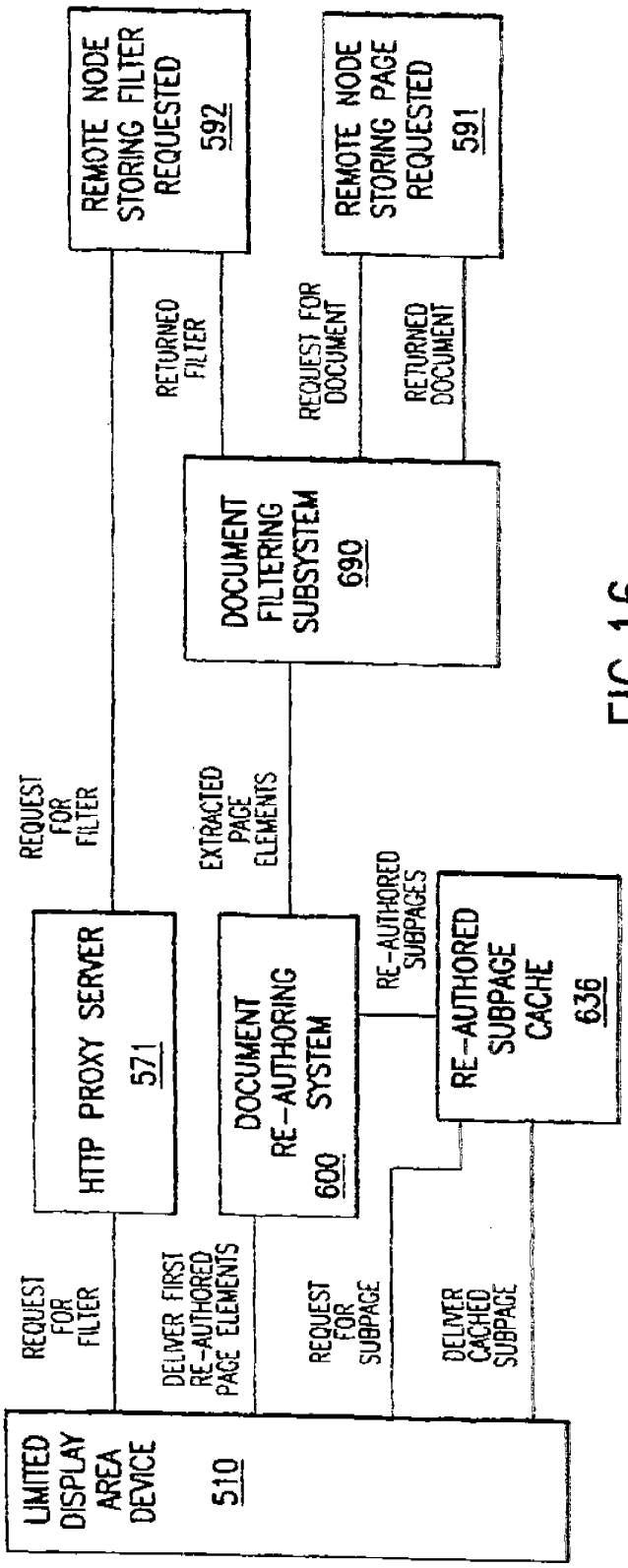

LIMITED DISPLAY AREA DEVICE 510
REQUEST FOR FILTER
DELIVER FIRST RE-AUTHORED PAGE ELEMENTS
REQUEST FOR SUBPAGE
DELIVER CACHED SUBPAGE
HTTP PROXY SERVER 571
DOCUMENT RE-AUTHORING SYSTEM 600
RE-AUTHORED SUBPAGES
RE-AUTHORED SUBPAGE CACHE 636
REQUEST FOR FILTER
EXTRACTED PAGE ELEMENTS
DOCUMENT FILTERING SUBSYSTEM 690
RETURNED FILTER
REQUEST FOR DOCUMENT
RETURNED DOCUMENT
REMOTE NODE STORING FILTER REQUESTED 592
REMOTE NODE STORING PAGE REQUESTED 591

FIG. 16

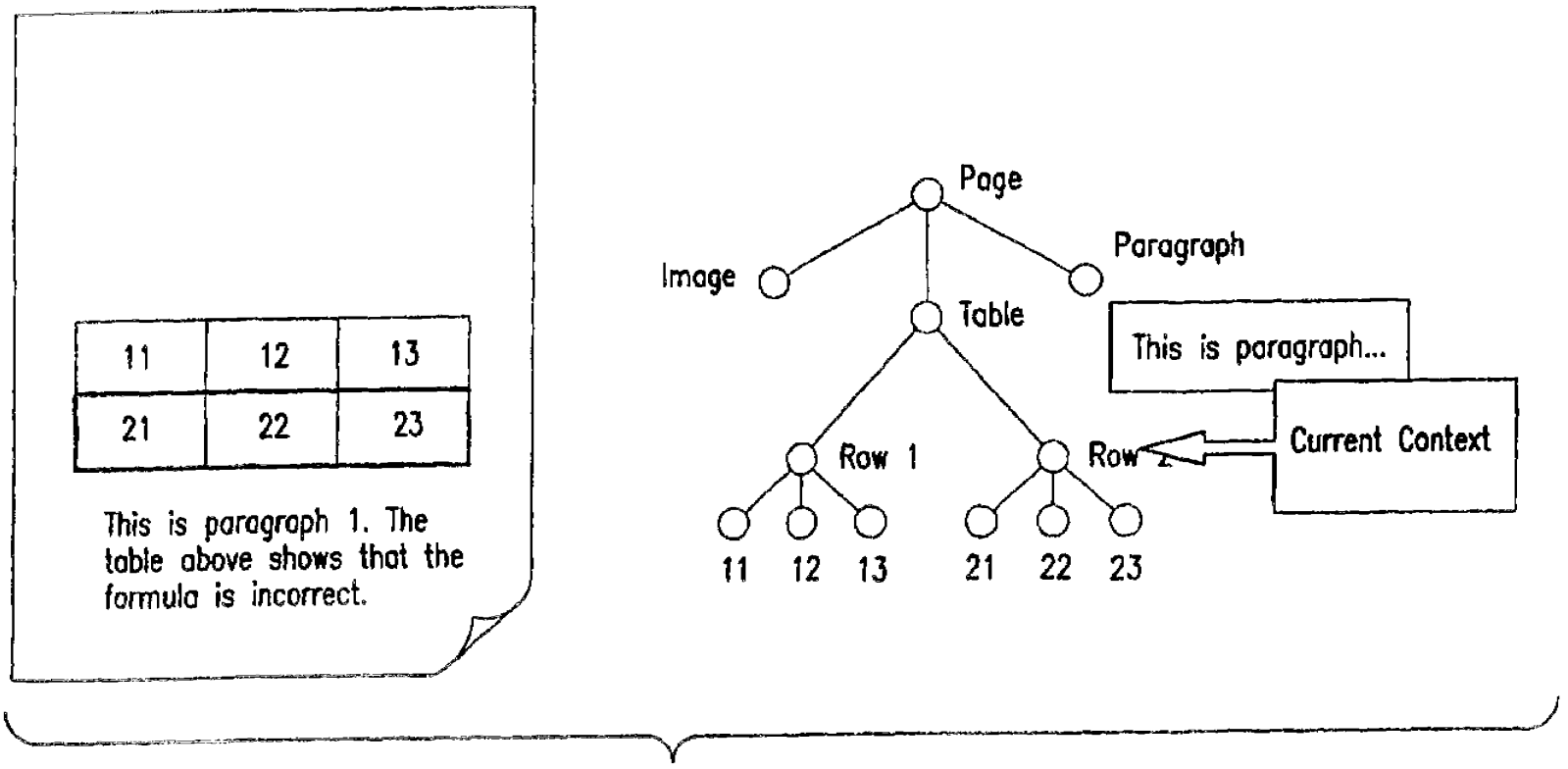



FIG.17

EP 0 949 571 A2

IPR2023-00330 Page 00523

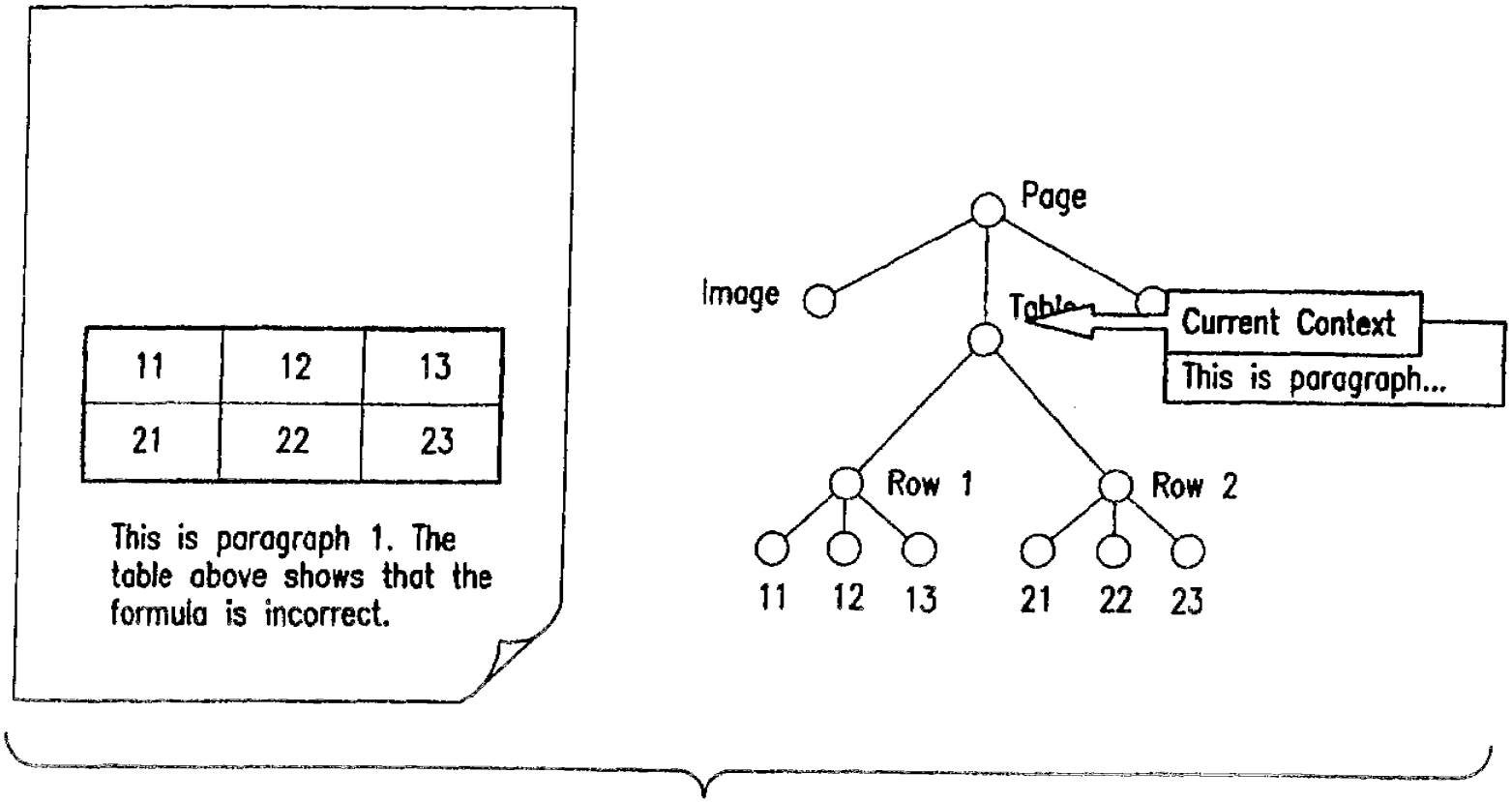



FIG.18

EP 0 949 571 A2

IPR2023-00330 Page 00524

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 949 571 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
05.01.2000 Bulletin 2000/01

(51) Int Cl.7: G06F 17/22, G06F 17/30

(43) Date of publication A2:
13.10.1999 Bulletin 1999/41

(21) Application number: 99302718.4

(22) Date of filing: 07.04.1999

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 07.04.1998 US 80909 P
29.01.1999 US 239295

(71) Applicant: XEROX CORPORATION
Rochester, New York 14644 (US)

(72) Inventors:
- Bickmore, Timothy W.
Somerville, Massachusetts 02144 (US)
- Girgensohn, Andreas
Menlo Park, California (US)
- Schilit, William N.
Menlo Park, California 94025 (US)
- Sullivan, Joseph W.
San Francisco, California 94107 (US)

(74) Representative: Skone James, Robert Edmund
GILL JENNINGS & EVERY
Broadgate House
7 Eldon Street
London EC2M 7LH (GB)

# (54) Document re-authoring systems and methods for providing device-independent access to the world wide web

(57) An automatic re-authoring system and method re-author a document originally designed for display on a desktop computer screen for display on a smaller display screen, such as those used with a PDA or a cellular telephone. The automatic re-authoring system and method input a document to be re-authored and re-authoring parameters, such as display screen size, default font and the like. The automatic re-authoring system and method convert the document into a number of pages, where each page is fully displayable with only at most a minimal amount of scrolling on the display screen of the PDA or cellular phone. At each stage of the re-authoring, a number of different transformations are applied to the original document or a selected re-authored page. The selected re-authored page is the best page resulting from the previous re-authoring stage. The best page at each stage is determined based on the re-authoring parameters and the content of the document being re-authored.

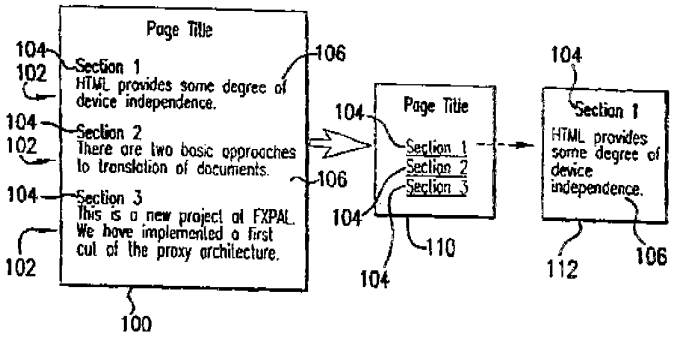



FIG.1

EP 0 949 571 A3

European Patent Office

# EUROPEAN SEARCH REPORT

Application Number
EP 99 30 2718

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (Int.Cl.6) |
|---|---|---|---|
| X | BICKMORE T W ET AL: "Digestor: device-independent access to the World Wide Web" COMPUTER NETWORKS AND ISDN SYSTEMS,NL,NORTH HOLLAND PUBLISHING. AMSTERDAM, vol. 29, no. 8-13, September 1997 (1997-09), page 1075-1082 XP004095305 ISSN: 0169-7552 * the whole document * | 1-10 | G06F17/22 G06F17/30 |
| A | JOHNSON D: "Converting PC GUIs for nonPC devices" CIRCUIT CELLAR INK, FEB. 1998, CIRCUIT CELLAR INC, USA, no. 91, pages 40-42, 44 - 45, XP000852859 ISSN: 0896-8985 * page 40, left-hand column, line 1 - page 42, right-hand column, line 43 * | 1,10 | |

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| THE HAGUE | 15 November 1999 | Fournier, C |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04Q — A2

(11) International Publication Number: **WO 97/49252**

(43) International Publication Date: 24 December 1997 (24.12.97)

(21) International Application Number: PCT/US97/10758

(22) International Filing Date: 20 June 1997 (20.06.97)

(30) Priority Data:
60/020,094 21 June 1996 (21.06.96) US

(71) Applicant *(for all designated States except US)*: INTEGRATED COMPUTING ENGINES, INC. [US/US]; 460 Totten Pond Road, Waltham, MA 02154 (US).

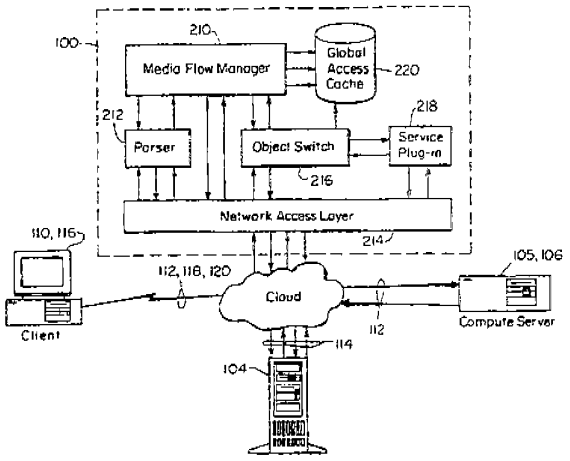
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: SHAH, Ashesh, C. [US/US]; 567 Tremont Avenue, No. 31, Boston, MA 02118 (US). PEDERSEN, Palle [DK/US]; 82 Commonwealth Avenue, No. 10, Boston, MA 02116 (US). RADOVIC, Niksa [HR/US]; 19 Mountain Avenue, Somerville, MA 02143 (US). MANICKAVASAGAM, Senthilkumar [IN/US]; 11 Highland Glen Drive, No. 17, Randolph, MA 02368 (US).

(74) Agents: SMITH, James, M. et al.; Hamilton, Brook, Smith & Reynolds, P.C., Two Militia Drive, Lexington, MA 02173 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

**Published**
*Without international search report and to be republished upon receipt of that report.*

(54) Title: NETWORK BASED PROGRAMMABLE MEDIA MANIPULATOR




(57) Abstract

The media manipulator is a middle layer between the clients, and the remote data servers is the common client-server organization. It transforms the network into a more flexible three-tiered configuration. Requests generated by the clients for media objects from media resources are routed to the media manipulator. It processes the requests and determines if the media objects may be found locally, either cached in the media manipulator itself or in the local data servers. When the media objects are obtained, the media manipulator can be used to perform operations on those objects such as format translations, to apply protective mechanisms for the clients, to speed communications between the remote servers and the clients, or perform compute operations for the clients. In one example, a parser of the manipulator searches for images in the media objects so that service devices can be called to perform data compression or pornography detection on the images. The parser can also search for executable or data files in the media objects and to perform virus scanning or format conversion, respectively.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# NETWORK BASED PROGRAMMABLE MEDIA MANIPULATOR

## RELATED APPLICATIONS

This application claims priority to U.S. Provisional Application No. 60/020,094, filed June 21, 1996, the contents of which is incorporated herein by reference in its entirety.

## BACKGROUND OF THE INVENTION

In a client-server network, on one hand there are clients, typically personal computers, IBM-compatible computers and/or UNIX workstations, for example, equipped with information browsers. On the other hand, there are data servers and compute servers. Data servers are computers with a large storage capacity containing information in different media formats: data records, plain text documents, word processing documents, still pictures, compressed audio and video, and executable files, for example. Compute servers are computers that carry out intensive computational tasks that would typically require too much time for the client to complete. Each compute server might use a single or many processors to complete the given task.

Users interact with their clients in a natural way with a mouse, keyboard, screen, printer, or by some other input/output device. The users need not be concerned about what happens after they make their selection within their clients. Clients then make service requests to geographically dispersed servers. Upon receiving requests from the clients, the servers

perform the desired operations and return the retrieved or computed media stream back to the client for display.

## SUMMARY OF THE INVENTION

The present invention is connected into the ubiquitous two-tiered client-server network of computers. It is designed as a middle layer, middleware, between the clients and the remote data servers. It transforms the network into a more flexible three-tiered configuration. Requests generated by the clients for media objects from media resources are routed to the media manipulator. It processes the requests and determines if the media objects may be found locally, either cached in the media manipulator itself or in local/remote data servers. When the media objects are obtained, the media manipulator can be used to perform operations on those objects such as format translations, to apply protective mechanisms for the clients such as virus scanning, to speed communications between the remote servers and the clients using compression operations, or perform compute operations for the clients.

In general, according to one aspect, the invention features a middle-ware computing system. It includes a network access system that supports communications with media resources and client computers and a media manipulation system that operates on media objects received from the media resources via the network access system prior to forwarding the media objects to the client computers.

In specific embodiments, a parser is used to identify different media types within the media objects

so that service devices may be called to operate on the media types. In one example, the parser searches for images in the media objects and service devices include an image compressor for performing data compression or pornography detection on the images. The parser can also search for executable or data files in the media objects and the service devices then called to perform virus scanning or format conversion, respectively.

In further specifics, a cache is used to store media objects. A media flow manager receives requests for media objects and checks for the presence of the media objects in the cache to preclude the necessity of obtaining the objects from the remote media resources.

The above and other features of the invention including various novel details of construction and combinations of parts, and other advantages, will now be more particularly described with reference to the accompanying drawings and pointed out in the claims. It will be understood that the particular method and device embodying the invention are shown by way of illustration and not as a limitation of the invention. The principles and features of this invention may be employed in various and numerous embodiments without departing from the scope of the invention.

## BRIEF DESCRIPTION OF THE DRAWINGS

In the accompanying drawings, reference characters refer to the same parts throughout the different views. The drawings are not necessarily to scale; emphasis has instead been placed upon illustrating the principles of the invention. Of the drawings:

Fig. 1 is a schematic block diagram illustrating the context in which the inventive media manipulator operates;

Fig. 2 is a block diagram illustrating the interaction between components of the media manipulator according to the invention;

Fig. 3 is an object interaction diagram illustrating the operation of the components of the media manipulator;

Figs. 4A, 4B, and 4C show the message formats for transmitting tasks to compute servers;

Fig. 5 is a block diagram showing the programming of the media manipulator using m-script;

Fig. 6 is another object interaction diagram showing the order of creation of the components of the manipulator; and

Fig. 7 is a block diagram showing another embodiment of the media manipulator.

DETAILED DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates the context in which the media manipulator 100 operates. In many applications, it is important for users to access remote media resources 108 such as the data servers 104 of content providers on the Internet. These users may be at client computers 110 that are inter-connected by a local area network (LAN) 112. The clients 110 access the media resources 108 through a gateway 114 linking the LAN 112 to the Internet. The user's also require access to the media resources 108 remotely at remote clients 116 through, for example, telephone dial-up connections 118 or through cellular/wireless links 120.

The media manipulator 100 is connected into this two-tiered client-server network of computers as a

middle layer between the clients 110, 116 and the remote data servers 104 of the media resources 108.

Fig. 2 is a block diagram illustrating the internal organization of the media manipulator 100. It comprises six basic components: media flow manager 210, media parser 212, network access layer 214, object switch 216, multiple service plugins 218, and global access cache 220. In one embodiment, these components are implemented as separate software objects that run on a common microprocessor or multiprocessor system.

The media flow manager 210 serves as the principle controller for the media manipulator 100. It has access to the various components and can alter their behavior. It specifies the operations to be formed on the received media objects. It is also the storehouse for the information on the media objects as they are received from the media parser. The media flow manager 210 also tracks the physical resources that are functional and available in the media manipulator and on the surrounding LAN in order to determine to which of the resources the media objects flow.

The network access layer 214 makes the media manipulator 100 accessible through many different types of network devices and the protocols running on top of them. In one implementation, the network access layer communicates through the Internet gateway 114 using the TCP/IP protocol, connects to the compute or data server using the protocol of the local area network 112, and communicates with the clients using either the LAN or the protocols necessary to communicate with the remote clients 116 over low-bandwidth connections 118, 120.

When communicating with the remote data servers 104 of the content providers, the network accepts the incoming data streams and assembles them into media objects. These media objects are then made available to the media parser 212, object switch 216, and service plugins 218. The media parser 212 analyses all incoming media objects to extract the relevant media types. These media types include executable files, data files, and images, for example. Information concerning the detected media types is forwarded to the media flow manager 210, which decides what operations should be performed on the media.

The object switch 216 supports a number of incoming and outgoing object gates. Media objects enter into the object switch from the network access layer 214 and from the service plugins' output links. The media objects leaving the object switch 216 go into the network access layer 214 and the service plugins' input links. The object switch routes the objects based on the media manager instructions, either directly or indirectly.

The global access cache is an intelligent mechanism that speeds the operation from the perspective of the user at the clients 110, 116. It determines which media objects are most likely to be used in the future and stores them in the fastest available memory. Media objects that are somewhat less likely to be required again are stored in slower memory or a secondary cache. There can be as many levels of the cache as the physical infrastructure allows, and the caching may take place on data servers that are remote from the main computational resources of the media manipulator. This caching minimizes the time

that different users need to wait for requests to be processed.

The media manipulator 100 is a programmable device. A system administrator can change its behavior by giving it m-script commands. It is also an extendable device. By adding new service plugins, new capabilities can be added to the device. The construction of the components of the media manipulator allows for redundancy and fault tolerance. A hardware failure does not bring the entire system to a halt. The system will keep working and simply notify the administrator that one of its components needs to be replaced.

Fig. 3 is an object flow diagram illustrating the communication between the client 110, 116, content provider's data server 104, and the components of the multimedia manipulator 100.

The first step is the initial connection 1, Connect, between the client 110, 116 and the media manipulator 100 via the network access layer 214. The network access layer 214 accepts this request 2. In one implementation, it accepts by calling a new incidence of itself such that each incidence of the network access layer object supports a single connection outside the media manipulator 100.

After establishing the connection, the client makes a request 4 for a media object. In the typical example, this will be a universal resource locator (URL) to a data server 104 of a content provider on the Internet. The network access layer then calls the media parser 212 and passes the client request 6.

The media parser 212 looks to two sources for the media object simultaneously. The ProcessURL request 8 is passed to the media flow manager 212, which has knowledge of the contents of the global access cache 220. The parser also issues a request 10, GetPage, to the network access layer.

The media flow manager 212 searches for the object in the cache 220. If the cache returns a cache-miss, the request to the provider has not been delayed waiting for the miss status, whereas in the case of a cache-hit, the request to the provider is simply terminated after verifying the validity of the cached page. Using this scheme, there is little increased latency associated with the use of the manipulator 100 in the worst-case cache-miss scenario.

In the illustrated example, a cache-miss occurred. Thus, rather than supplying the object, the cache 220 is prepared 12 to receive the media object, PutInCache. Also, the network access layer 214 connects 14 to the content provider and retrieves 16 the media object or page.

As the media object is being received by the network access layer from the content provider 104, the parser begins to parse 17 the object. As parsing proceeds, the parser also begins to update 18 the global access cache 220 with the parsed portions of the object. Simultaneously, the parser begins the reply 20, 22 to the client via the network access layer.

In one implementation, the parser searches for images in the media objects to perform compression or pornography detection, for example. On encountering an

image, the parser 214 passes a call to the media flow manager to process the image 24 while continuing to parse 26 the media object.

The media flow manager 210 gets the image 28 via the network access layer 214. The fact images are not stored with the page but must be separately requested is an artifact of the HTTP protocol. The network access layer 214 then connects 30 to the content provider 104 and retrieves 32 the image.

When the image is retrieved, the media flow manager 210 places it in the cache with the other portions of the media object and makes a function call to the object switch to process the image 34. The object switch knows the various service plugins that are available and the actions that must be performed on the media types that are discovered by the media parser, which in this example is an image. When called by the object switch 216 to process 36 the media type, the particular service plugin, or multiple plugins when serial operations are required, retrieves 38 the media type, *i.e.*, image, and performs the desired operation on or processes 40 the image. For example, in one instance, this can be compression or thinning to expedite communication to the client. In another case, it can detect the probability of pornography by detecting the percentage of flesh-tone colors in the picture. Once the processing is complete, the new image or revised media object may be placed 42 in the cache 220 or used in a reply to the client 110, 116.

In many instances, the service plugin functionality will be performed by a separate compute

server 105. This computer may be directly accessible by the media manipulator 100 or accessible through the local area network 112. Generally out-sourcing this functionality is desirable, rather than running on the same device with the other components of the media manipulator 100, to avoid depriving those other components of processing bandwidth.

When the plugin does utilize the external compute server, it issues a request message. Fig. 4A illustrates the formatting of the message to the compute server. The message has a number of different fields. It has a version field and a length field defining the length of the content. The type field indicates the type of the message, and the message ID is assigned by the network access layer. The source type indicates the media type. In the context of image files, the type indicates whether the image is in a GIF or JPEG type compression format, for example. The source path is the path to where the image is stored in the global access cache 220, to which the compute server has access. The destination type, path length, path, and parameters define the transformed media type and where it is to be sent.

Fig. 4B illustrates the reply message from the compute server. It again has version, length, type, and message ID fields. The reply code indicates whether or not the service was successful. The destination type, path length, and path indicate the type of the final image after the transform of the compute server has been implemented and where that final image is stored in the global access cache or otherwise.

Fig. 4C shows the error message issued by the compute server when service was unsuccessful or error occurred. To contain this information, the message has a computer server error code identifying the server and a field holding the reason for the error.

As illustrated in Fig. 5, the administrator or Internet application developer specifies the actions of the media manipulator by supplying an m-script language to the media flow manager 210. This is a quasi-configuration, script file which forms a high level programming language of the media manipulator. The following illustrates the general structure of the language with examples showing its use in the media manipulator 100.

name := definition

The name of a rule is simply the name itself (without any enclosing "<" and ">") and is separated from its definition by the colon-equal (":=") character.

"literal"

Quotation marks surround literal text. Unless stated otherwise, the text is case-sensitive.

rule1 | rule2

Elements separated by a bar ("|") are alternatives, e.g.,

"yes | no" will accept "yes" or "no".

{rule1 rule2}

Elements enclosed in parentheses are treated as a single element. Thus, "{elem {foo | bar} elem}" allows the token sequences "elem foo elem" and "elem bar elem".

rule*

The character "*" following an element indicates repetition. For example, "foo bar*", implies, "foo" followed by zero or more of "bars".

[rule]

Square brackets enclose optional elements. For example, "foo [bar]" implies, "foo" followed by zero or one of "bar".

The BNF grammar of the m-script is grouped under three logical groups.

Basic

| | | |
|---|---|---|
| Alphabets | := | {abc...zABC..Z} |
| Variable-chars | := | {abc...zABC...Z-_} |
| Numbers | := | {0...9} |
| Variable-name | := | Variable-chars {Variable-chars \| Numbers}* |
| Host-name | := | Variable-chars {Variable-chars \| Numbers \| "."}* |
| Path-name | := | {Variable-chars \| Numbers \| "." \| "/" \| "\" \| "~" \| ":"}* |
| Others | := | {!@#$%^&*()"{}-=\|\<>,} |
| EOLN | := | "\n" |
| Comment | := | "#" {Variable-chars \| Others \| "." \| "/" \| "~" \| ":" }* EOLN |

This section describes the basic rules used: Alphabets are composed of letters "a" through "z", "A" through "Z"; Numbers are composed of digits zero through nine. Variables-chars are alphabets, dash ("-") and underscore ("_"). A variable name must start with a Variable-char and followed by zero or many variable-chars or numbers. Host-name is similar to variable-name and in-addition can have periods ("."). Path-name is a generic path used for locating files. EOLN is ASCII 13. A comment must start with "#" character and ends with an EOLN.

Generic

m-script := { comment | section }*

Section := section-key "{" section-Desc "}"

Section-Desc := section-line*

Section-line := section-desc-key "=" section-desc-value [EOLN]

Section := server-section | cache-section | service-section | filter-section | action-section

This section describes a generic m-script file. An m-script is a comment or a section. A section must start with a Section-key, followed by a Section-description enclosed in parentheses. The section description is made up of zero or many section lines. A section line starts with a section description key followed by an equal sign ("=") and the section description key's value. There are five types of sections, *viz.*, server, cache, service, filter and action.

Detail

Server-section := "server" "{" server-sec-desc "}"

Server-sec-desc := server-name-line | server-port-line

Server-name-line := "name" "=" host-name [EOLN]

Server-port-line := "port" "=" numbers [EOLN]

A server section starts with the key "server". This section consists of two lines: Name and port lines. The name line specifies the name of the host on which the MM 100 is run. The port line specifies the main port number on which the MM awaits requests from clients.

Cache-section := "cache" "{" cache-sec-desc "}"

Cache-sec-desc := cache-clean-line | cache-direc-line

Cache-clean-line := "cleanup" "=" { number | "no" }

Cache-direc-line := "directory" "=" Path-name

A cache section starts with the key "Cache". This section consists of two lines as well: Cache-clean and directory lines. The cache-clean line specifies the time interval after which the cache cleaning is performed. It takes two values: a positive number (time interval in seconds) or the string "no"(implying never to be cleaned). The directory line specifies the directory in which the cached files need to be stored.

Service-section := "service" "{" Service-sec-desc "}"

Service-sec-desc := Service -id-line | Service-host-line | Service-port-line

Service-id-line := "id" "=" variable-name

Service-host-line := "host" "=" host-name

Service-port-line := "port" "=" numbers

A service section is for service plugins. There must be a service section for each service that has to be used by the MM 100. This section starts with the key "service". The section consists of three lines: Id, Host and port lines. The id line specifies a user defined identifier that can be used in other sections. The host and port lines respectively specify the name of the host and port number on which the service is available.

Filter-sec := "filter" "{" filter-desc "}"

Filter-desc := filter-object-line | filter-action-line

Filter-Object-line := "object" "=" Filter-Object-Name

Filter-Object-Name := "image" | "video" | "java"

Filter-Action-line := "action" "=" variable-name

A filter section starts with the key "filter". This section consists of two lines: object and action line. The object line specifies the name of the object to be identified and filtered. The action line identifies the rule to be applied on the object. Currently, the objects identified are images. In future, objects like video and Java applets can be identified.

Action-section := "action" "{" action-desc "}"

Action-desc := action-id-line | action-cond-line | action-proc-line

Action-id-line := "id" "=" variable-name

Action-cond-line := "cond" "=" Action-cond-exp

Action-cond-exp := [Action-exp-bin-op] Action-exp-var [ Action-cond-exp-op Action-cond-exp ]

Action-exp-bin-op := "!"

Action-cond-exp-op := "&&" | "||" | "==" | "!=" | ">" | "<" | ">=" | "<="

Action-exp-var := { Filter-Object-Name "." Parameter } | { variable-name "." "result" }

Parameter := "any" | "transparent" | "animated"

Action-proc-line := "process" "=" Action-proc-exp

Action-proc-exp := { variable-name| Method-exp } [Action-connect Action-proc-exp ]

Method-exp := Filter-Object-Name "." Method-name "(" Method-Param* ")"

Method-name := "replace"

Method-Param := """ Path-name """

Action-connect := "&" | "|"

The action section is the most complicated section. The action sections can be linked to other action sections forming a list of actions to be applied in tandem. The section starts with the key "action". This section consists of three lines: id, condition and process lines. The id, as before, is a user assigned identifier. The condition line specifies a condition when the process has to be performed. The condition is like a standard "C" expression. It uses object's properties (e.g., image.transparent - image that has a transparent bit), or result of other rules (e.g. rule1.result). The process can be a service identifier or another rule identifier. Several identifiers can be connected using action connectors: "&"(and) or "|"(or). The "&"(and) connector implies both the rules have to be applied in succession (e.g.: rule1 & rule2 - implies apply rule1 and then rule2). The "|" (or) connector implies that apply either of the process (e.g.: compress1 | compress2 - implies, apply compress1 or compress2).

An example is as shown below:

```
1   #m-script for manipulating HTML files
2
3   #listening host name and port
4   server {
5   name = center
6   port = 8001
7   }
8
9   #cache parameters
10  cache{
11  cleanup = no
12  directory = "/opt/mm/cache/images/"
```

```
13      }
14
15      #compress service server 1
16      service{
17      id = compress1
18      host = center
19      port = 7002
20      }
21
22      #compress service server 2
23      service{
24      id = compress2
25      host = center
26      port = 7003
27      }
28
29      filter{
30      object = image
31      action = rule1
32      }
33
34      action{
35      id = rule1
36      cond = image.any && ! image.transparent
37      process = compress1 | compress2
38      }
```

| Line # | Explanation |
|---|---|
| 1. | A comment line starts with a "#" character. Everything to the end of that line is ignored. |
| 5 & 6 | The media manipulator listens on the host "center" and on port "8001" |
| 11 & 12 | The files are cached (global cache) on the server. Keep them longer. Store them in the directory specified. |
| 17-19 | Compute server id is "Compress1". The host address is "center" and is listening on port "7002" |
| 24-26 | Compute server id is "Compress2". The host address is "center" and is listening on port "7003" |
| 30 & 31 | Filter the images and apply rule1 |



IPR2023-00330 Page 00546

```
35      This section is rule1
36      Do the process for any image that is not transparent.
37      Process  images by sending to compress1 or to compress2
```

Example #2

Apart from compressing the images, the images can be tested for pornography. For this a service section has to be added and the action section has to be modified. The following m-script accomplishes this.

```
1       #m-script for manipulating HTML files
2       #This compresses the images and detects them for pornography
3
4       #listening host name and port
5       server {
6       name = center
7       port = 8001
8       }
9
10      #cache parameters
11      cache{
12      cleanup = no
13      directory = "/opt/mm/cache/images/"
14      }
15
16      #compress service server 1
17      service{
18      id = compress1
19      host = center
20      port = 7002
21      }
22
23      #compress service server 2
24      service{
25      id = compress2
26      host = center
27      port = 7003
28      }
29
30      #pornography detect service server 1
31      service{
```

```
32      id = porno1
33      host = center
        port = 7010
34      }
35
36      #pornography detect service server 2
37      service{
38      id = porno2
39      host = center
40      port = 7011
41      }
42
43      filter{
44      object = image
45      action = all_image_rule
46      }
47
48      action{
49      id = all_image_rule
50      cond = image.any
51      process = compress_rule & porno_rule & destroy_rule
52      }
53
54      action{
55      id = compress_rule
56      cond = ! image.transparent
57      process = compress1 | compress2
58      }
59
60      action{
61      id = porno_rule
62      cond = compress_rule.result == 1
63      process = porno1 | porno2
64      }
65
66      action{
67      id = destroy_rule
68      cond = porno_rule.result >= 75
69      process = image.replace("/opt/mm/iib/images/forbidden.gif")
70      }
```

| Line # | Explanation |
|---|---|

5 - 8 Server section
11 - 14 Cache section
16 - 28 Compute servers "Compress1" and "Compress2"
31 - 35 Pornography detection service "porno1" is running in "center" and listening on port 7010
38 - 42 Pornography detection service "porno1" is running in "center" and listening on port 7010
44 - 47 Filter the images and apply all_image_rule
49 - 52 Apply action "all_image_rule" to all images. First apply compress_rule, followed by porno_rule and then by destroy_rule.
55 - 59 Apply action "compress_rule" to non-transparent images. Pass the images to either compress1 or compress2.
61 - 65 Apply action "porno_rule" to images, if compress_rule returned 1. Pass the compressed images to porno1 or porno2.
67 - 71 Apply action "destroy_rule" to images, if porno_rule returned a value greater than or equal to 75(probability of a pornographic image). Replace the image with "forbidden.gif".

Media Flow Manager (MFM)

MFM reads m-script and configures itself and other components based on the m-script. The MFM can be implemented as a multi-threaded object as shown below:

```
class MFM{
private:
        int iPort;
        char *strHostName;
        char *strMFileName;
        MediaParser *pMP;
        GAC *pGAC;
        ObjSw *pOS;
        NAC *pNAC;
        ...
```



IPR2023-00330 Page 00549

```
public:
        MFM();
        ~MFM();
        int Configure(char *strMFileName);
        int ProcessURL(char *strURL, ...);
        int ProcessImage(char *strSrcURL, int iHeight, int iWidth, ...);
        int CheckCacheUpdate(...);
        int CreateInstance();
        ...
};
```

| | |
|---|---|
| Configure(...) | Parses the m-Script file specified and configures the rest of the components. |
| ProcessURL(...) | This is called by the parser, when it encounters a new image. This initiates the cache insertion on GAC. |
| ProcessImage(...) | Mainly invoked by the MediaParser, when it encounters an image tag. This passes the command to the appropriate object switch. |
| CreateInstance(...) | This creates a new instance of the MFM by first copying the internal data structures and then creating a new thread. |

```
Media Parser - HTML Parser
The media parser can be implemented using generic tools
like lex and yacc. The core of the parser can then be
packaged to make parser objects.

class MediaParser {
private:
        MFM *pMFM;
        GAC *pGAC;
        ...
public:
```



IPR2023-00330 Page 00550

```
        MediaParser(MFM *pMFM, GAC *pGAC, ...);
        ~MediaParser();
        int AddFilter(int iObjectType, ...);
        int Parse(...);
        ...
};
```

| | |
|---|---|
| AddFilter(...) | Called by the MFM.Configure, adds to the list of objects that the MediaParser has to look for. |
| Parse(...) | This is called by the NAL, when it successfully establishes a connection with the client. This parses the media. When it encounters the object to be filtered, the parser notifies the MFM by invoking the appropriate function. |

Global Access Cache

The global access cache is a specialized cache system, specifically tuned to keep HTML pages and the images. The images can have multiple versions. These have to be cached separately. The cache is also cleaned regularly as described in the cache section of the m-script.

```
class GAC{
private:
        char *apMainBuckets[MAX_HASH_KEY];
        int Hash(char *strURL);
        ...
public:
        GAC(MFM *pMFM, char *strPath, ...);
        ~GAC();
        int SearchCache(char *strURL, ...);
```

```
        int PutInCache(char *strURL, char *strLocalFilename, FILE *fp, ...);
        int UpdateCache(char *strURL, ...);
        int GetFromCache(char *strURL, int iKey, ...);
        ...
};
```

| | |
|---|---|
| Hash(...) | This is used to create the Hash key based on an URL. |
| SearchCache(...) | This searches the cache for the given URL. |
| PutInCache(...) | First searches the cache(SearchCache()) and if not found, inserts the URL int the cache. |
| UpdateCache(...) | Updates the cache entry with related entries. For example, the URL entry can be updated with image entries that are related to the URL. |
| GetFromCache(...) | Retrieves an URL or an Image. |

```
Network Access Layer
The network access layer for handling HTML pages, primarily
deals with HTTP(Hyper Text Transmission Protocol). It
accepts connection from the clients; makes connection to
the content provider; requests and receives pages and
images from the content provider. In addition to these the
layer also provides connection to compute servers.

class NAL{
private:
        int iPort;
        char *strHostName;
        int iNumCharsRead;
        int iNumCharsWritten;
        char *strURL;
        ...
public:
        NAL(MFM *pMFM, MediaParser *pMP, ...);
```



IPR2023-00330 Page 00552

```
        ~NAL();
        int Listen(char *strHostName, int iPort, ...);
        int Accept(...);
        int Connect(char *strHostName, int iPort, ...);
        int AcceptClients(char *strHostName, int iPort, ...);
        int GetImage(char *strHostName, int iPort, char *strURL, ...);
        int GetURL(char *strHostName, int iPort, char *strURL, ...);
        int SendRequest(...);
        int ReceiveReply(...);
        ...
    };
```

| | |
|---|---|
| Listen(...) | Creates a listening port. |
| Accept(...) | Accepts any client requesting a connect. |
| Connect(...) | Connects to the specified host and port number. Usually called by the GetImage() or GetURL() |
| GetImage(...) | Connects to the ContentProvider and requests the image specified by the URL. This is responsible for building the appropriate request header etc |
| GetURL(...) | Connects to the ContentProvider and requests the page specified by the URL. This is responsible for building the appropriate request header etc. |
| SendRequest(...) | Sends a formatted message to the compute server. The format of the message is shown in the following section. |
| ReceiveReply(...) | Receives a formatted message that is a reply to the message sent earlier. |

```
Service Plugin
The service sections describe the various servers available
for the MM. Each service server is an instance of this
object.
```



IPR2023-00330 Page 00553

```
class ServPlugin{
private:
        char *strId;
        int  iPort;
        char *strHostName;
        ...
public:
        ServPlugin(NAL *pNAL, char *strId, char *strHostName, int iPort, ...);
        ~ServPlugin();
        int Request(char *strSrcPath, char *strDestPath, ...);
        ...
};
```

| | |
|---|---|
| Request(...) | This initiates the request through the NAL. NAL sends the formatted message to the appropriate Compute Server. |

Object Switch

The object switch interfaces the MFM and the service plugins. The object switch mostly implements the rules specified in the action section of the m-script, as instructed by the MFM.

```
class ObjSw{
private:
        MFM *pMFM;
        GAC *pGAC;
        ServPlugin *aSP; //array of service plugins
        ActionList *alAction; //linked list of actions
        –
public:
```



IPR2023-00330 Page 00554

```
        ObjSw(MFM *pMFM, GAC *pGAC, _);
        ~ObjSw();
        int AddServicePlugin(ServPlugin *pSP, _);
        int AddAction(char *strId, char *strCond, char *strProcess, _);
        int ProcessImage(_);
        _
};
```

| | |
|---|---|
| AddServicePlugin(_) | This is invoked by the MFM during configuration phase. This adds the service plugin to its internal list. |
| AddAction(_) | This is also invoked by the MFM during the configuration phase. This adds the actions specified in the m-Script |
| ProcessImage(_) | Invoked by the MFM, this executes the actions in the specified order. |

Compute Server

The compute server executes as a separate processor or on a different machine itself. It can be implemented as an object as well.

```
class CompServ{
private:
        int iPort;
        char *strHostName;
        ...
public:
        CompServ();
        ~CompServ();
        int ReceiveRequest(char *strSrcPath, char *strDestPath, ...);
        int ProcessRequest(...);
        int Reply(...);
```



IPR2023-00330 Page 00555

...

};

| | |
|---|---|
| ReceiveRequest(...) | This receives the formatted message. |
| ProcessRequest(...) | This processes the request. The user can extend the compute server by adding capabilities to this method. |
| Reply(...) | Sends the reply. |

The compute server can also use the NAL to send and receive messages.

Fig. 6 is an object interaction diagram showing the order of creation of the objects/components of the manipulator 100 and the order in which the m-script is processed or read. Of note is the fact that the object switch 216 is called after service plugins 218. This order ensures that the services are all declared. AddAction takes the pointer to those service plugins, and AddServicePlugin identifies the compute server executing the plugin, its host name, and its port. ObjSw ensures the GAC 220 may be updated by the object switch with the results of the service, once executed.

Fig. 7 illustrates another embodiment of the inventive media manipulator 100. The media manipulator described in the previous sections was used as an intermediate processor between the client 106, 116 and the content provider server 104. In this alternative embodiment, an additional, stripped down tunneler version of the manipulator 100' can be used to interact between the client 106, 116 and the media manipulator 100 as described previously. These two instances of the manipulator 100, 100' can now perform in unison to further enhance the user experience.



IPR2023-00330 Page 00556

The tunneler media manipulator 100' and the media manipulator 100 exchange a compressed format suitable for the transmission over a low-bandwidth connection, while the tunneler 100' and the browser(client) exchange information in the client's native format. Apart from these, the client 106, 116 can be inside a firewall f and still use the services of a main media manipulator 100, which may be outside the firewall f. The tunneler 100' can also be used to set various options such as compression quality, specific to the client's need. These options are forwarded to the main media manipulator 100 along with the client's request. The main media manipulator 100 can categorically act on both the tunneler's and client's request.

Apart from compressing images, the tunneler 100' and main media manipulator 100 combination can be used to compress the HTML page itself. The HTML page is a media, and if the service is available to compress it, the m-script can be modified appropriately to send the page to the text-compress-plugin before sending towards the client. The tunneler can intercept this and decompress the page.

The tunneler 100' has following components of the media manipulator: 1) media flow manager 210, 2) media parser 212, 3) object switch 216, 4) network access layer 214, and 5) service plugin 218. It does not the global access cache 220. The service plugin in the tunneler 100' is the compliment of what is used in the media manipulator to decompress the images.



IPR2023-00330 Page 00557

While this invention has been particularly shown and described with references to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the appended claims.



IPR2023-00330 Page 00558

CLAIMS

What is claimed is:

1. A middle-ware computing system comprising:

a network access system that supports communications with media resources and client computers; and

a media manipulation system that operates on media objects received from the media resources via the network access system prior to forwarding the media objects to the client computers.

2. The computing system described in Claim 1, wherein the media manipulation system comprises:

a parser that identifies different media types within the media objects; and

service devices that operate on the media types.

3. The computing system described in Claim 2, wherein the parser searches for images in the media objects and service devices include an image compressor for performing data compression on the images.

4. The computing system described in any of Claims 2-3, wherein the parser searches for executable files in the media objects and service devices include a virus scanner that searches for computer viruses in the files.

5. The computing system described in any of Claims 2-4, wherein the parser searches for images in the media objects and service devices include a pornography detector for assessing a probability that the images are pornographic.

6. The computing system described in any of Claims 2-5, wherein the parser searches for data files in the media objects and service devices include an format converter for changing a format of the data files.

7. The computing system described in any of Claims 2-6, wherein the media manipulation system further comprises an object switch that passes the media types to the service devices to determine operations performed on the different media types.

8. The computing system described in any of Claims 2-7, wherein the media manipulation system further comprises a media flow manager that reassembles the media objects for forwarding to the clients after the manipulation of the media types.

9. The computing system described in Claim 8, further comprising a cache that stores media objects, the media flow manager receiving requests for media objects and checking for the presence of the media objects in the cache to preclude obtaining the objects from the media resources.

10. A middle-ware computing system comprising:

a network access system that supports communications with media resources to obtain media objects from client computers;

a parser that identifies different media types within the media objects;

service devices that manipulate the media types;

an object switch that passes the media types to the service devices to determine operations performed on the different media types; and

a media flow manager that reassembles the media objects for forwarding to the clients after the manipulation of the media types.

11. The computing system described in Claim 10, further comprising a cache that stores media objects, the media flow manager receiving requests for media objects and checking for the presence of the media objects in the cache to preclude obtaining the objects from the media resources.

12. A method for facilitating transmission of media objects between media resources and client computers, the method comprising:

receiving requests for media objects from the client computers to the media resources;

obtaining the media objects;

manipulating the media objects;

forwarding the manipulated media objects to the client computers.

13. The method described in Claim 12, wherein manipulating the media objects comprises:

identifying different media types within the media objects; and

performing separate operations on the different media types.

14. The method described in Claim 13, wherein the step of identifying different media types comprises searching for images in the media objects and the step of performing operations comprises data compressing the images.

15. The method described in any of Claims 13-14, wherein the step of identifying different media types comprises searching for executable files in the media objects and the step of performing operations comprises scanning the files for computer viruses.

16. The method described in any of Claims 13-15, wherein the step of identifying different media types comprises searching for images in the media objects and the step of performing operations comprises assessing a probability that the images are pornographic.

17. The method described in any of Claims 13-16, wherein the step of identifying different media types comprises searching for data files in the media objects and the step of performing operations changing a format of the data files.

18. The method described in any of Claims 13-17, further comprising reassembling the media objects for forwarding to the clients after the manipulation of the media types.

19. The method described in any of Claims 13-18, further comprising routing the media types to form successive operations on the media types.

20. The method described in any of Claims 13-19, further comprising caching media objects that have been received from the media resources and later obtaining the media objects from the cache.

21. The method described in Claim 20, wherein the step of obtaining the media objects comprises requesting the media objects from the media resources while checking for the objects in a cache; and obtaining the media objects from the cache if present.

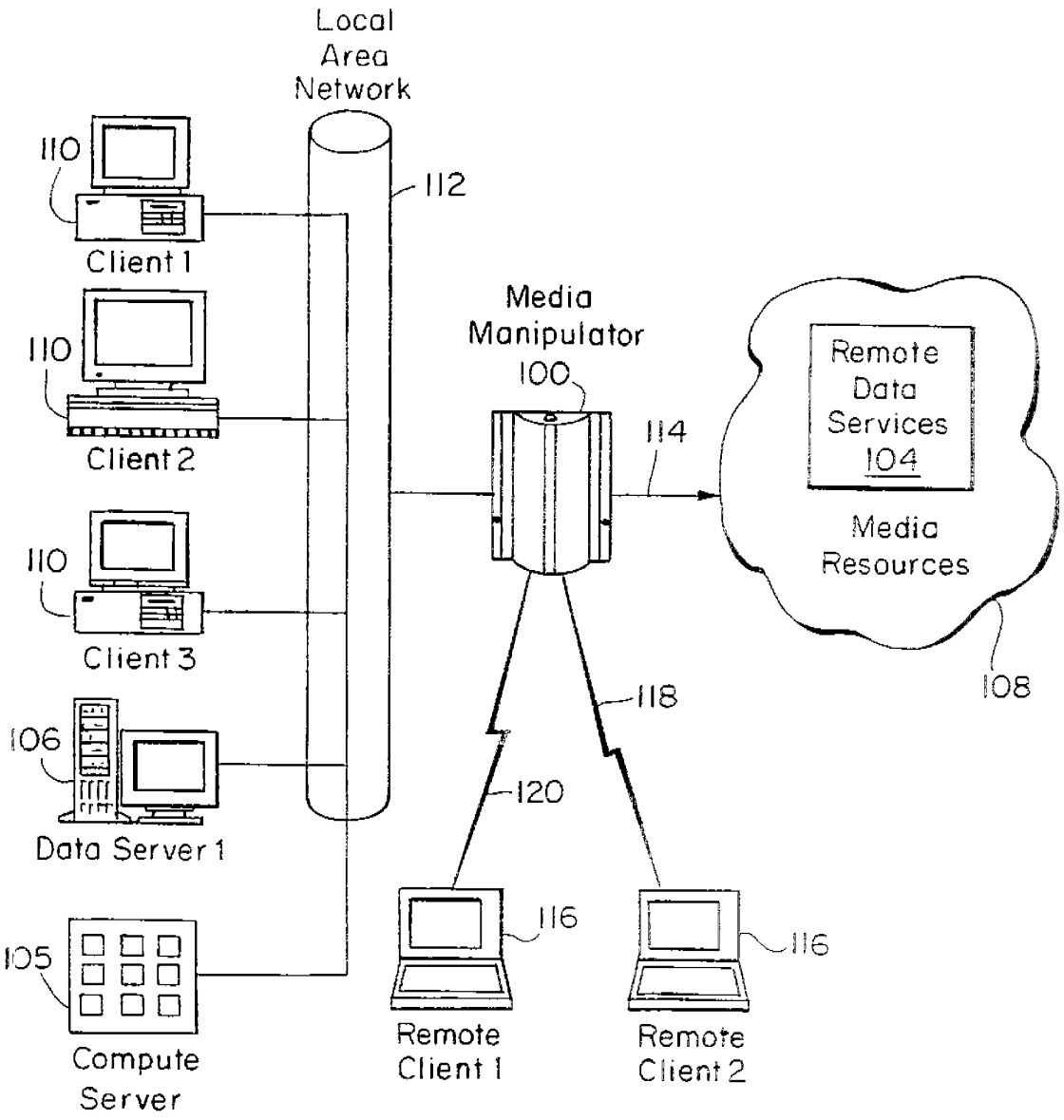

Local
Area
Network
110
112
Client 1
110
Client 2
110
Client 3
106
Data Server 1
105
Compute
Server
Media
Manipulator
100
114
Remote
Data
Services
104
Media
Resources
108
118
120
116
116
Remote
Client 1
Remote
Client 2

FIG. 1

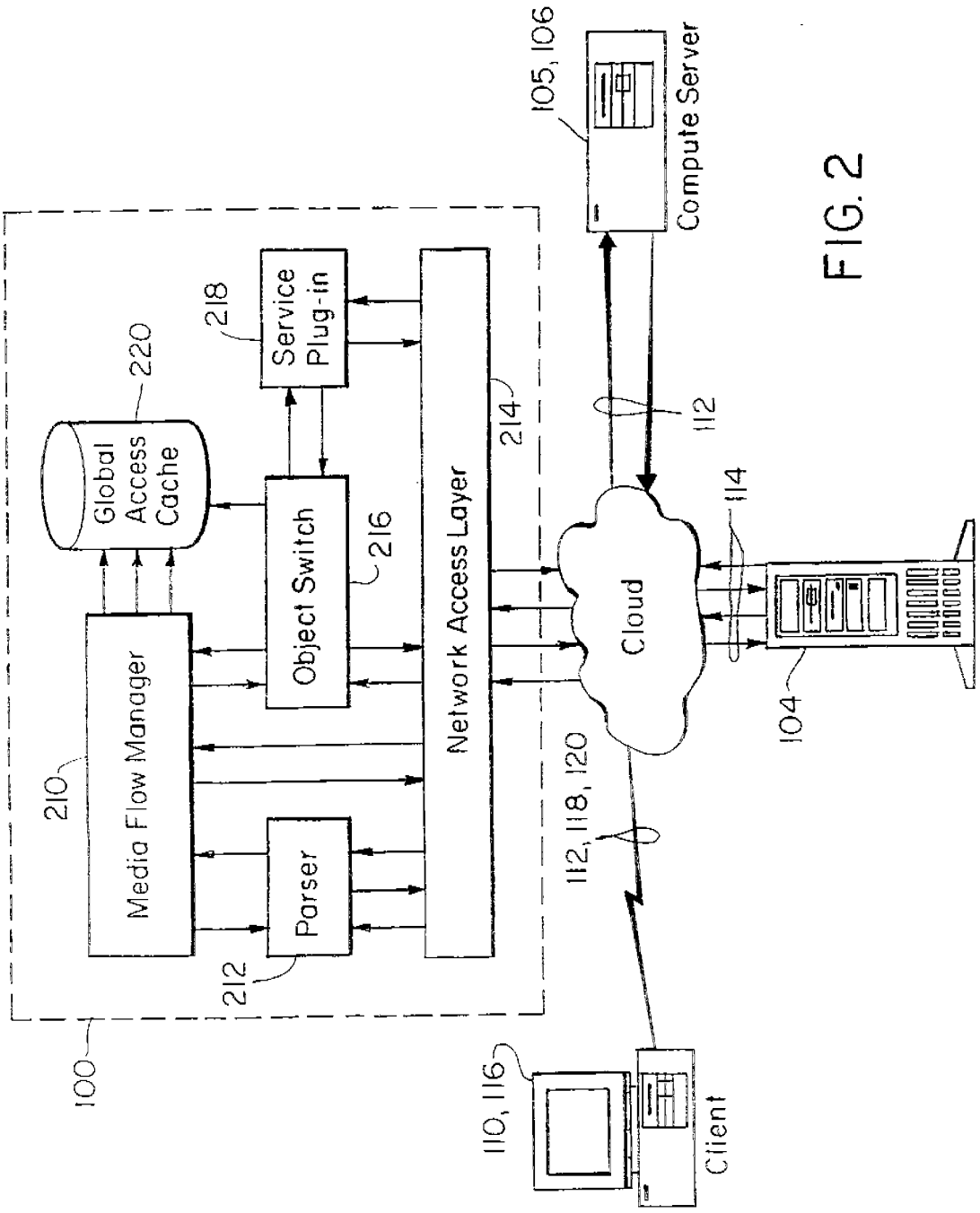

105, 106
Compute Server
218
Service Plug-in
220
Global Access Cache
214
Network Access Layer
216
Object Switch
112
114
Cloud
210
Media Flow Manager
212
Parser
112, 118, 120
104
100
110, 116
Client

FIG. 2

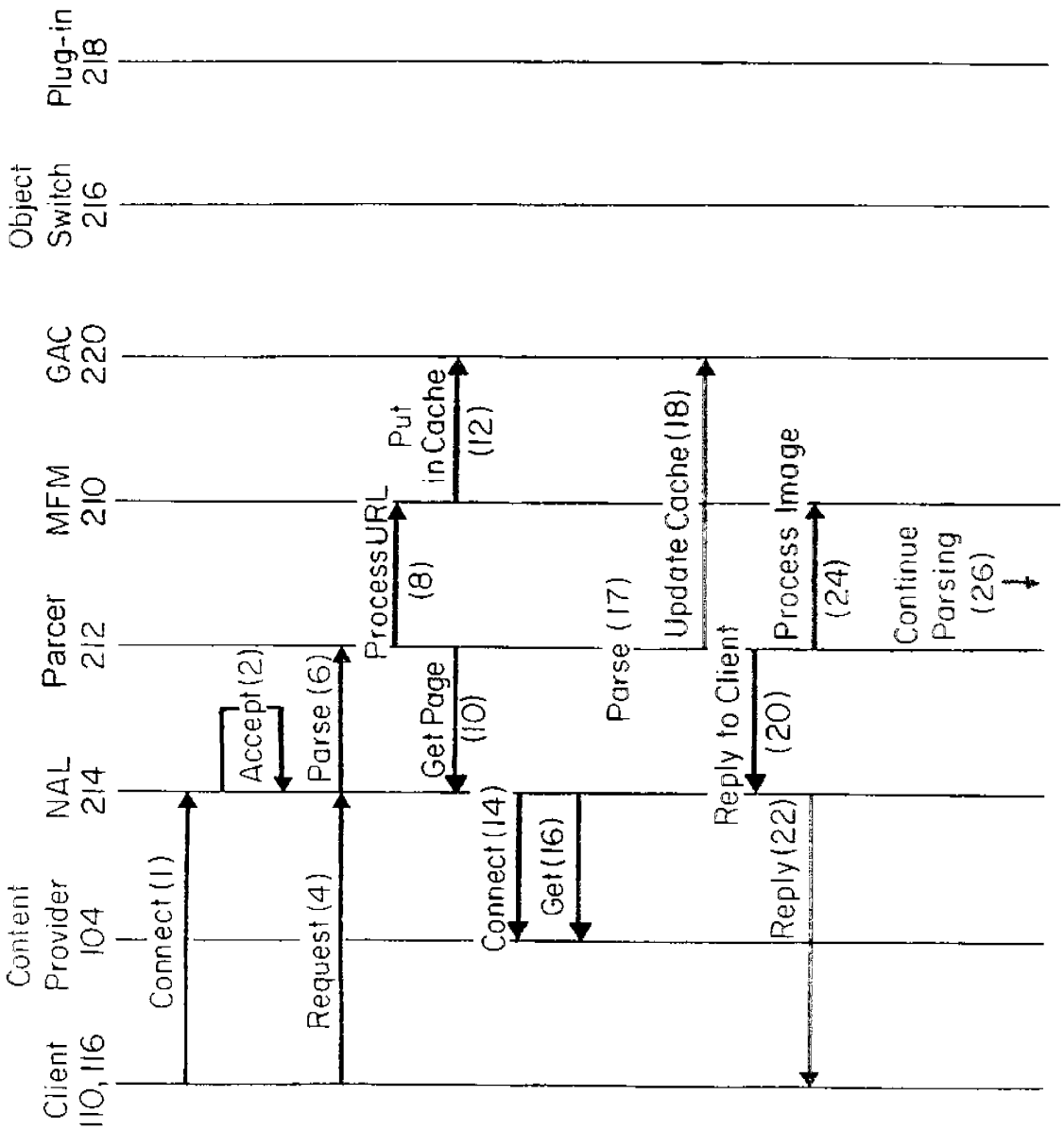



FIG. 3A



IPR2023-00330 Page 00566

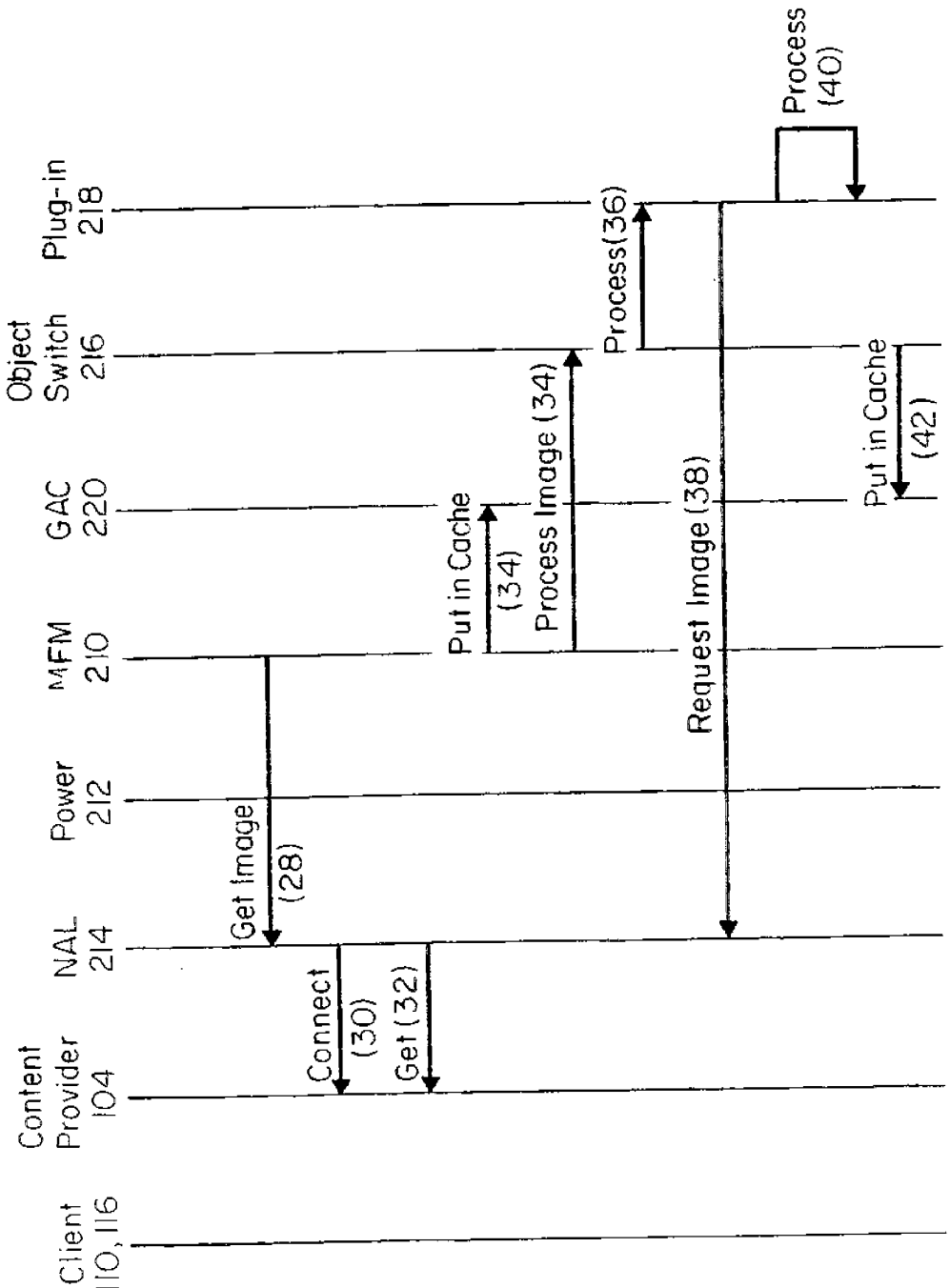



FIG. 3B



IPR2023-00330 Page 00567

| Header | | | | Content | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Version | Length | Type | Message Id | Src Type | Src Path Len | Src Path | Dest Type | Dest Path Len | Dest Path | Dest Param |

| Field | Field Length | Description |
|---|---|---|
| Version | 4 | Message Version Number. E.g. 0100, implies 1.0 |
| Length | 4 | Length of the content |
| Type | 4 | Type of the Message: 1-for request, 2-for reply, 3-for error |
| Message Id | 4 | Numeric ID of the message assigned by the NAL |
| Src Type | 4 | Numeric type of the source image: 1-GIF, 2-JPEG, 3-MM Compress Format 1 |
| Src Path Len | 4 | Length of the Src Path |
| Src Path | - | Path where the image is stored. Can be a network path as well. |
| Dest Type | 4 | Numeric type of the final image: 1-GIF, 2-JPEG, 3-MM Compress Format 1 |
| Dest Path Len | 4 | Length of the Dest Path |
| Dest Path | - | Path where the final image has to be stored |
| Dest Param | 4 | Can be used to set an optional parameter |

FIG. 4A

| Header | | | | Content | | | |
|---|---|---|---|---|---|---|---|
| Version | Length | Type | Message Id | Reply Code | Dest Type | Dest Path Len | Dest Path |

| Field | Field Length | Description |
|---|---|---|
| Version | 4 | Message Version Number. E.g. 0100, implies 1.0 |
| Length | 4 | Length of the content |
| Type | 4 | Type of the Message: 1-for request, 2-for reply, 3-for error |
| Message Id | 4 | Numeric ID of the message assigned by the NAL |
| Reply Code | 4 | The success or failure of the service: 1- success, 0- error |
| Dest Type | 4 | Numeric type of the final image: 1 - GIF, 2 - JPEG, 3 - MM Compress Format 1 |
| Dest Path Len | 4 | Length of the Dest Path |
| Dest Path | - | Path where the final image has to be stored |

FIG. 4B



IPR2023-00330 Page 00568

| Header | | | | Content | | | |
|---|---|---|---|---|---|---|---|
| Version | Length | Type | Message Id | Reply Code | Error Code | Error Reason Len | Error Reason |

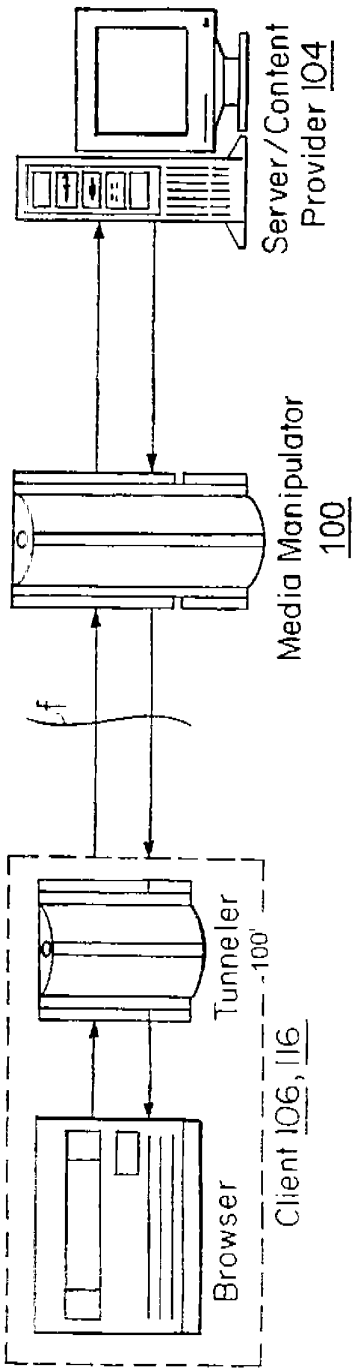
| Field | Field Length | Description |
|---|---|---|
| Version | 4 | Message Version Number. E.g. 0100, implies 1.0 |
| Length | 4 | Length of the content |
| Type | 4 | Type of the Message: 1-for request, 2 for reply, 3 for error |
| Message Id | 4 | Numeric ID of the message assigned by the NAL |
| Reply Code | 4 | The success or failure of the service: 0-error |
| Error Code | 4 | Numeric Error Code assigned by the compute server |
| Error Reason Len | 4 | Length of the reason, the next field |
| Error Reason | - | String describing the error |

FIG. 4C



IPR2023-00330 Page 00569

FIG. 7



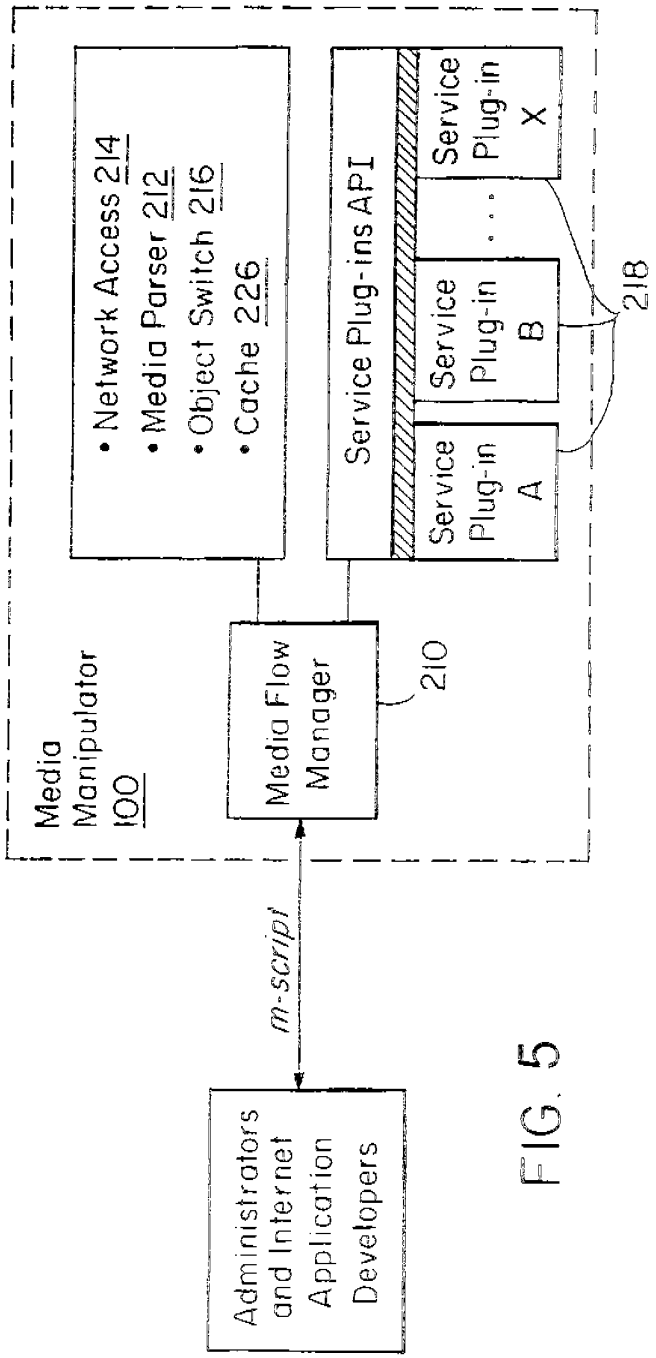




FIG. 5



IPR2023-00330 Page 00570

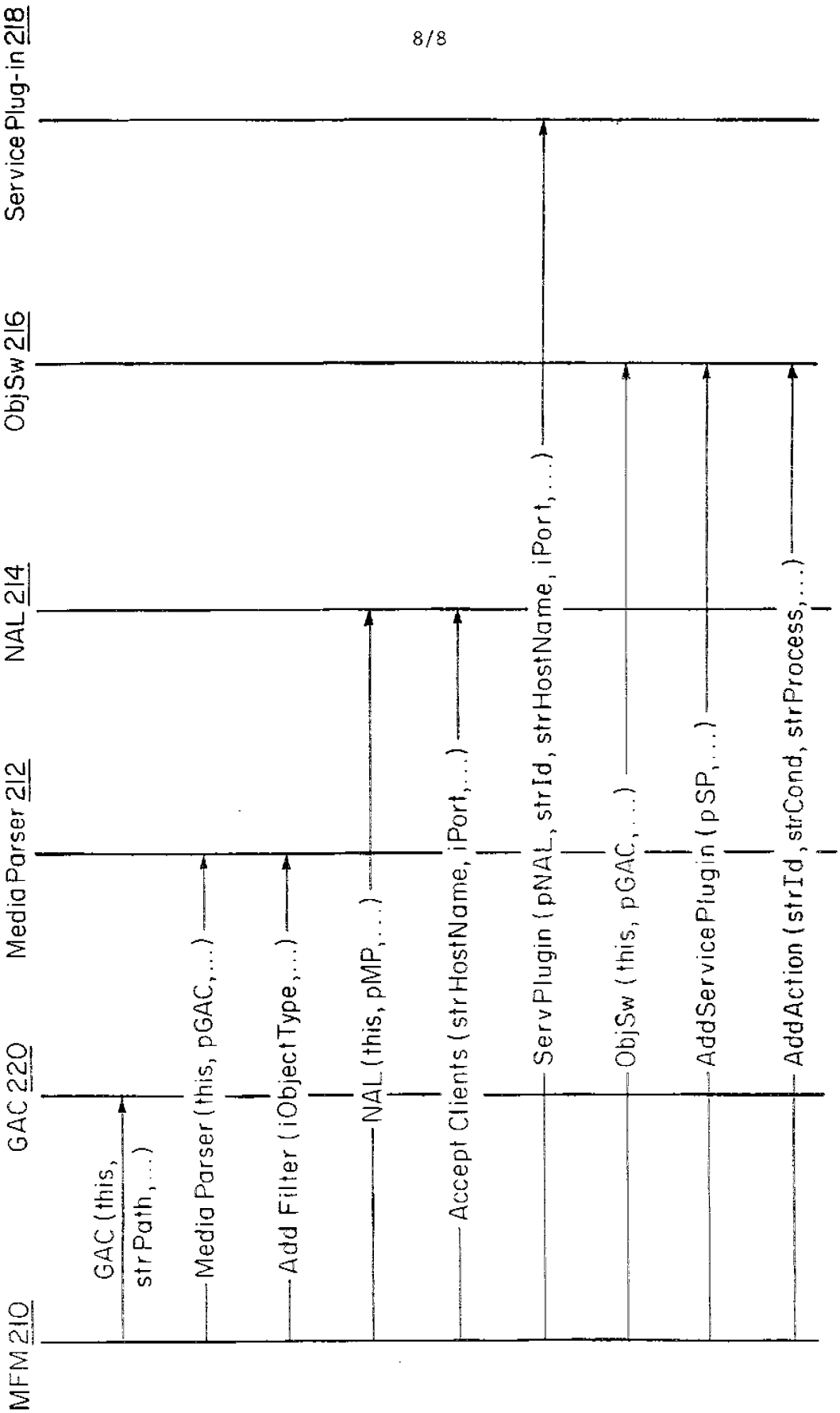



FIG. 6



IPR2023-00330 Page 00571

**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

**(51) International Patent Classification 6 :** H04L 29/06 — A3

**(11) International Publication Number:** WO 97/49252

**(43) International Publication Date:** 24 December 1997 (24.12.97)

**(21) International Application Number:** PCT/US97/10758

**(22) International Filing Date:** 20 June 1997 (20.06.97)

**(30) Priority Data:**
60/020,094 — 21 June 1996 (21.06.96) — US

**(71) Applicant** *(for all designated States except US)*: INTEGRATED COMPUTING ENGINES, INC. [US/US]; 460 Totten Pond Road, Waltham, MA 02154 (US).

**(72) Inventors; and**
**(75) Inventors/Applicants** *(for US only)*: SHAH, Ashesh, C. [US/US]; 567 Tremont Avenue, No. 31, Boston, MA 02118 (US). PEDERSEN, Palle [DK/US]; 82 Commonwealth Avenue, No. 10, Boston, MA 02116 (US). RADOVIC, Niksa [HR/US]; 19 Mountain Avenue, Somerville, MA 02143 (US). MANICKAVASAGAM, Senthilkumar [IN/US]; 11 Highland Glen Drive, No. 17, Randolph, MA 02368 (US).

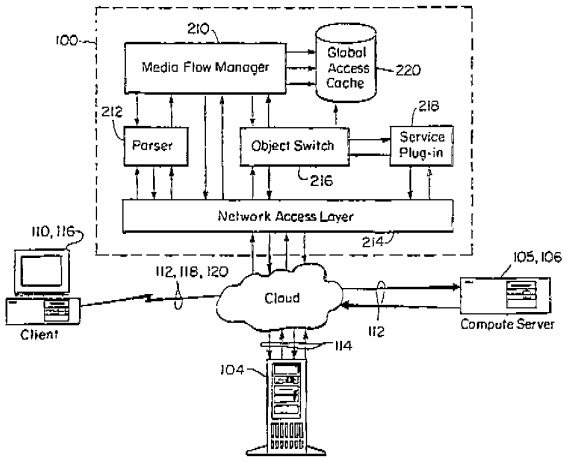
**(74) Agents:** SMITH, James, M. et al.; Hamilton, Brook, Smith & Reynolds, P.C., Two Militia Drive, Lexington, MA 02173 (US).

**(81) Designated States:** AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

**Published**
*With international search report.*

**(88) Date of publication of the international search report:**
30 April 1998 (30.04.98)

**(54) Title:** NETWORK BASED PROGRAMMABLE MEDIA MANIPULATOR




**(57) Abstract**

The media manipulator is a middle layer between the clients (110, 116) and the remote data servers (104) is the common client-server organization. It transforms the network into a more flexible three-tiered configuration. Requests generated by the clients (110) for media objects from media resources are routed to the media manipulator (100). It processes the requests and determines if the media objects may be found locally, either cached (220) in the media manipulator (100) itself or in the local data servers (106). When the media objects are obtained, the media manipulator (100) can be used to perform operations on those objects such as format translations, to apply protective mechanisms for the clients (110), to speed communications between the remote servers (104) and the clients (110), or perform compute operations for the clients (110). In one example, a parser (112) of the manipulator (100) searches for images in the media objects so that service devices (218) can be called to perform data compression or pornography detection on the images. The parser can also search for executable or data files in the media objects and to perform virus scanning or format conversion, respectively.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# INTERNATIONAL SEARCH REPORT

Inter ›nal Application No
PCT/US 97/10758

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | EP 0 669 587 A (AT & T CORP) 30 August 1995 | 1,12 |
| A | see column 4, line 20 - column 5, line 14; figure 1<br>see column 7, line 13-34; figure 2<br>see column 9, line 17-29<br>see column 16, line 39-51<br>--- | 2,3,7,8, 10,13, 14,18 |
| A | THAU R: "Design considerations for the Apache Server API"<br>COMPUTER NETWORKS AND ISDN SYSTEMS, vol. 28, no. 11, May 1996, pages 1113-1122, XP002046988<br>see paragraph 4<br>---<br>-/-- | 2,10,13 |

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 20 November 1997 | 18.12.97 |

| Name and mailing address of the ISA | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,<br>Fax: (+31-70) 340-3016 | Dupuis, H |

# INTERNATIONAL SEARCH REPORT

Inter onal Application No
PCT/US 97/10758

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | TREVOR J ET AL: "Exorcising daemons: a modular and lightweight approach to deploying applications on the Web" COMPUTER NETWORKS AND ISDN SYSTEMS, vol. 28, no. 11, May 1996, pages 1053-1062, XP002046968 see paragraph 3 - paragraph 3.1 --- | 2,3,7,8, 10,13, 14,18 |
| A | WO 96 17306 A (ORACLE CORP) 6 June 1996 see page 8, line 16-21 see page 11, line 29 - page 13, line 22; figure 1 see page 15, line 6-16 see page 17, line 27 - page 18, line 15 see page 34, line 2-28 --- | 1,6, 9-12,17, 20,21 |
| A | HOWLETT D: "Protection on the Web" COMPUTERS AND SECURITY, vol. 15, no. 4, 1996, page 319 XP002046969 see the whole document ----- | 4,15 |



IPR2023-00330 Page 00575

# INTERNATIONAL SEARCH REPORT

Information on patent family members

Inter onal Application No
PCT/US 97/10758

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| EP 0669587 A | 30-08-95 | CA 2140850 A | 25-08-95 |
| WO 9617306 A | 06-06-96 | NONE | |

**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06K 9/00 — A1

(11) International Publication Number: **WO 98/40842**

(43) International Publication Date: 17 September 1998 (17.09.98)

(21) International Application Number: PCT/US98/04700

(22) International Filing Date: 11 March 1998 (11.03.98)

(30) Priority Data:

| | | |
|---|---|---|
| 60/040,241 | 11 March 1997 (11.03.97) | US |
| 09/038,562 | 10 March 1998 (10.03.98) | US |

(71) Applicant: COMPUTER INFORMATION AND SCIENCES, INC. [US/US]; Suite 104, 3401 East University, Denton, TX 76208 (US).

(72) Inventors: CHAO, Hongyang; 1011 Chestnut #10, Denton, TX 76201 (US). HUA, Zeyi; 2197 S. Uecker #357, Denton, TX 76201 (US). FISCHER, Howard, P.; 3106 Saints Circle, Denton, TX 76201 (US). FISCHER, Paul, S.; 34 Timbergreen Circle, Denton, TX 76205 (US).

(74) Agents: SAMPLES, Kenneth, H. et al.; Fitch, Even, Tabin & Flannery, Room 900, 135 S. LaSalle, Chicago, IL 60603 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

**Published**

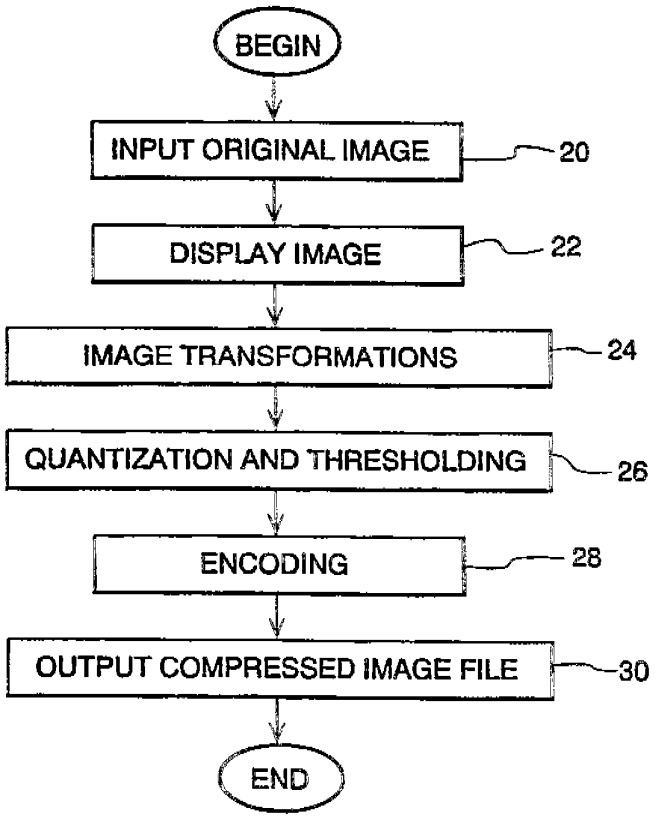
*With international search report.*

*Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.*

(54) Title: SYSTEM AND METHOD FOR IMAGE COMPRESSION AND DECOMPRESSION

(57) Abstract

A wavelet-based image compression system and method are presented (24). Compression is accomplished by performing a wavelet transformation of an input digital image (20). The resulting wavelet coefficients are compared to a threshold value. Coefficients falling below the threshold are discarded. The remaining coefficients are quantized (26). The quantized coefficients are then compressed using an entropy encoding technique (28).



*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# SYSTEM AND METHOD FOR IMAGE COMPRESSION AND DECOMPRESSION

This application claims the benefit of U.S. Provisional Application No. 60/040,241, filed March 11, 1997, System and Method for Still Image Compression, which is incorporated herein by reference.

## Field of the Invention

The present invention relates generally to digital image compression/decompression, and particularly, to a wavelet-based system and method of image compression and decompression.

## Background of the Invention

Nearly every computer user needs to store, transfer, and view images. These images include still images, or pictures, as well as video images, which are sequences of still images displayed in a manner that depicts motion. The enormous size of image files leads to serious file management limitations. For example, a single still image (equivalent to a video frame) displayed by a rectangular array of picture elements (pixels) arranged in 640 rows and 800 columns, with the color of each pixel represented by twenty-four bits, would require over 1.5 megabytes of digital memory to store. One solution to this problem is high-quality data compression technology. Essentially, image compression mathematically transforms a grid of image pixels into a new, much smaller set of digital values holding the information needed to regenerate the original image or data file.

In addition imaging systems, compression technology can be incorporated into "video on demand" systems, such as video servers. Compression technology can also be applied to streaming video, which is the real-time capture and display of video images over a communications link. Applications for streaming video

include video telephones, remote security systems, and other types of monitoring systems.

Several standards for compressing real-time video currently exist. The H.263 standard for real-time video is an industry standard based upon the discrete co-sign transform (DCT). DCT is also the basis for both of the public domain image compression standards, MPEG (Motion Picture Experts Group) and JPEG (Joint Photographic Experts Group). Although the DCT approach performs interframe coding adequately, its compression ratio and speed can be improved upon.

Various other types of data compression have been developed in recent years. Conventional data compression techniques are generally referred to as being either "lossless" or "lossy", depending upon whether data is discarded in the compression process. Examples of conventional lossless compression techniques include Huffman encoding, arithmetic encoding, and Fano-Shannon encoding. With a lossless compression, the decompression process will reproduce all bits of the original image. Lossless compression is important for images found in such applications as medical and space science. In such situations, the designer of the compression algorithm must be very careful to avoid discarding any information that may be required or even useful at some later point.

Lossy compression, in contrast, provides greater efficiency over lossless compression in terms of speed and storage, as some data is discarded. As a result, lossy techniques are employed where some degree of inaccuracy relative to the input data is tolerable. Accordingly, lossy compression is frequently used in video or commercial image processing. Two popular lossy image compression standards are the MPEG and JPEG compression methods.

The wavelet transform has proven to be one of the most powerful tools in the field of data compression. Theoretically, the wavelet transformation is lossless,

but since all computers have only finite precision even when using floating point calculations, most of the transformations are lossy in practice. On the other hand, integer calculations are much faster than floating point for virtually all computers; and integer computations are much easier to implement in hardware, which is more important in some applications. While integers require less memory than real numbers, the direct use of integers in conventional wavelet transforms and their inverses typically causes an unacceptable loss of accuracy. Accordingly, there is a need for a wavelet-based compression technique that permits lossless or near-lossless data compression, yet retains the speed and memory advantages of integer arithmetic.

## Summary of the Invention

It is an advantage of the present invention to provide a system and method of wavelet-based data compression that permits integer computations in a computer without significant loss of accuracy. This is accomplished by using an integer reversible wavelet transform that possesses a property of precision preservation (PPP). The integer reversible transform greatly reduces the computer resources needed to compress and decompress images, as well as the time required to perform the same.

It is an advantage of the present invention to provide a system and method of wavelet-based image compression that is suitable for both still and video images.

It is also an advantage of the present invention to provide a system and method of image compression that is capable of selectively performing lossless and lossy compression of either color or gray-scale images.

According to one aspect of the invention, a wavelet-based image compression method can be implemented using a software program. Compression is accomplished by

performing a wavelet transform on an input digital image. The resulting wavelet components are compared to a threshold value; coefficients falling below the threshold are discarded. The remaining coefficients are quantized. The quantized coefficients are then compressed using an entropy encoding technique, such as arithmetic, run length, or Huffman encoding, or a combination of Huffman and run length encoding. The wavelet transform can be an integer reversible wavelet transform derived using a lifting scheme or correction method, while the quantization scheme can be sub-band oriented. To further enhance the speed of the compression scheme, input color image pixels can be reduced using a color table. In addition, color pixels can be transformed between color spaces prior to wavelet transformation.

According to another aspect of the invention, a corresponding method of decompression is provided.

According to another aspect of the present invention, a compression method is provided that allows user selected portions of an image to compressed to different image qualities, whereby permitting non-uniform image compression.

According to another aspect of the present invention, a compression method is provided that permits compression quality to be based on image specific parameters.

According to another aspect of the present invention, a method of compressing images using a "split and merge" technique is provided.

According to further aspect of the present invention, an image compression system includes a compressor configured to generate a compressed image based on an integer wavelet transform derived using either a lifting scheme or correction method. The compressor can be implemented using one or more electronic components, such as application specific integrated circuits (ASICs), microprocessors, discrete

logic components, or any combination of the aforementioned.

According to another aspect of the present invention, a corresponding image decompression system is provided.

## Brief Description of the Drawings

The invention is pointed out with particularity in the appended claims. However, other features of the invention will become more apparent, and the invention will be best understood by referring to the following detailed description in conjunction with the accompanying drawings, in which:

FIG. 1 illustrates a flow diagram for a method of compressing an image that is in accordance with an embodiment of the present invention;

FIGS. 2-4 depict wavelet coefficients for various levels of decomposition;

FIG. 5 illustrates a flow diagram of a method of decompressing an image that has been compressed using the method of FIG. 1;

FIG. 6 is a block diagram of a system that can incorporate a software program implementing any of the methods shown in FIGS. 1, 5, and 8-13 in accordance with a second embodiment of the present invention;

FIG. 7 is a block diagram of a system for compressing and decompressing an image in accordance with another embodiment of the present invention;

FIG. 8 illustrates a flow diagram of a method compressing an image that is in accordance with a further embodiment of the present invention;

FIG. 9 illustrates a flow diagram of a method for decompressing an image that has been compressed according to the method of FIG. 8;

FIG. 10 illustrates a flow diagram of a method of compressing an image in accordance with a further embodiment of the present invention;

FIG. 11 illustrates a flow diagram of a method of decompressing an image that has been compressed according to the method of FIG. 10;

FIG. 12 illustrates a flow diagram of a method of compressing an image that is in accordance with a further embodiment of the present invention; and

FIG. 13 illustrates a flow diagram of a method for decompressing an image that has been compressed according to the method of FIG. 12.

**Detailed Description of the Preferred Embodiments**

Referring now to the drawings, and in particular to FIG. 1, there is shown a flow diagram of a method for compressing an image that conforms to a first embodiment of the invention. In step 20, a digital image is received from an image source. The digital image consists of a matrix of values representing an array of pixels. Specifically, the array of pixels represents a still image or a frame from a video image. In step 22, the image is optionally displayed on an appropriate viewing device, such as a computer or video display unit having a flat panel or cathode ray tube (CRT). Next, in step 24, color and wavelet transformations of the image take place. The image transformations involved in this step include color transform for color images only, and wavelet transform for both gray level images and color images. In step 26, the values representing the transformed images are quantized and compared to thresholds. Values falling outside the threshold are discarded. In step 28, the remaining quantized values are encoded to remove redundant information, creating a compressed image file. Next, in step 30 the compressed image file is generated as output.

Referring to the color transformation of step 24, digital color images are typically based on an RGB color model, such as is commonly used with TIFF or BMP images. In order to get a higher compression ratio, the

RGB pixels are transformed to other color models, such as YIQ or YUV models. The method can convert RGB inputs into YIQ or YUV color spaces according to the following relationships.

RGB to YIQ:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.596 & -0.275 & 0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

RBG to YUV:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.148 & -0.289 & 0.439 \\ 0.615 & -0.515 & -0.1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

In the YIQ color space, there is one luminescence (Y) and two color planes (I, Q). The Y component is critical, while the I-Q components are less sensitive to error introduced by data compression.

The wavelet transform (also referred to as wavelet decomposition) operates on the converted color space signals. The purpose of the wavelet transform is to represent the original image by a different basis to achieve the objective of decorrelation. There are many different wavelet transforms that can be used in this step. For instance, the reversible integer wavelet transform described herein below is a preferred wavelet transform. However, to develop a better understanding of the preferred transform, the following alternative wavelet transform is first described.

Let $C^0 = [C^0_{jk}]$ (j - 0, ..., M-1; k = 0, ..., N-1) represent the original, uncompressed image, where M and N are integers which have the common factor $2^L$ (L is a positive integer). A one-level wavelet decomposition, where L = 1, results in the four coefficient quadrants as

shown in Figure 2. Each quadrant represents a set of wavelet coefficients.

Quadrant $C^1$ represents the blurred image of the original image $C^0$, where $C^1 = [c^1_{jk}]\ (j=0,\ldots,\frac{M}{2}-1;\ k=0,\ldots,\frac{N}{2}-1)$. $HD^1$ represents the horizontal high frequency part of $C^0$, while $VD^1$ represents the vertical high frequency part of $C^0$, and $DD^1$ represents the diagonal high frequency part of $C^0$. The decomposition can be iteratively repeated L times to obtain different levels of decomposition. For example, for L = 2, $C^0$ is set to equal $C^1$. The iterative formula for computing a decomposition is given as follows:

(1) Let $\bar{C}^0 = rC^0$, r>0 is a factor which can be changed for different needs.

(2) Transform for image columns:

For k=0,...,N-1, calculate

$$\begin{cases} d^1_{0k} = \dfrac{\bar{C}^0_{1k} - \bar{C}^0_{0K}}{2}, \\ d^1_{jk} = \dfrac{1}{4}\left(\bar{C}^{-0}_{2j-1,k} - 2\bar{C}^{-0}_{2j,k} + \bar{C}_{2j+1,k}\right), j=1,\ldots,\dfrac{M}{2}-1. \end{cases} \tag{3.1.1}$$

For k=0,...,N-1, calculate

$$\begin{cases} \tilde{C}^1_{0k} = \bar{C}^0_{1,k} - \dfrac{d_{ok} + d_{1,k}}{2}, \\ \tilde{C}^1_{jk} = \bar{C}_{2j+1,k} - \dfrac{d^1_{jk} + d_{j+1,k}}{2}, j=1,\ldots,\dfrac{M}{2}, \\ \qquad \dfrac{\tilde{C}^{-1}_{N-2}}{2}, k = \tilde{C}^{-0}_{N-1,k} - \dfrac{d_{N-2}}{2}, k. \end{cases} \tag{3.1.2}$$

(3) Transform for rows:

For j = 0, . . . M/2 - 1, computing

$$\begin{cases} hd^1_{j,0}=\dfrac{\tilde{c}_{j1}-\tilde{c}^1_{j0}}{2}, \\ hd^1_{jk}=\dfrac{1}{4}\left(\tilde{c}^1_{j,2k-1}-2\tilde{c}_{j,2k+1},+\tilde{c}_{j,2k+1}\right), k=1\ldots\dfrac{N}{2}-1. \end{cases} \quad (3.1.3)$$

and

$$\begin{cases} c^1_{jo}=\tilde{c}^1_{j,1}-\dfrac{hd^1_{j0}+hd^1_{j1}}{2}, \\ c^1_{jk}=c^1_{j,2k+1}-\dfrac{hd^1_{j,k}+hd^1_{j,k+1},}{2},\ldots,\dfrac{M}{2}-2, \\ c_{j,\frac{N-2}{2}}=\tilde{c}^1_{j,N-1}hd_{j,\frac{N-2}{2}}, \end{cases} \quad (3.1.4)$$

For j = 0, . . . , M/2 - 1, computing

$$\begin{cases} dd^1_{j,0}=\dfrac{\tilde{d}^1_{j,1}-\tilde{d}^1_{j0}}{2}, \\ dd^1_{jk}=\dfrac{1}{4}\left(\tilde{d}^1_{j,2k-1}-2\tilde{d}_{j,2k}+\tilde{d}^1_{j,2k+1}\right), k=1\ldots\dfrac{N}{2}-1. \end{cases} \quad (3.1.5)$$

and

$$\begin{cases} vd^1_{j0}=\tilde{d}^1_{j,1}-\dfrac{dd^1_{j0}+dd^1_{j1}}{2}, \\ vd^1_{jk}=\tilde{d}^1_{j,2k+1}-\dfrac{dd^1_{j,k}-dd^1_{j,k+1}}{2}, k=1,\ldots,\dfrac{M}{2}-2, \\ vd_{j,\frac{N-2}{2}}=\tilde{d}_{N-1,k}-dd^1_{j,\frac{N-2}{2}}, \end{cases} \quad (3.1.6)$$

(4) $C^1=[c^1_{j,k}]$, $HD^1=[hd^1_{j,k}]$, $VD^1[\upsilon d^1_{j,k}]$ and $DD=[dd^1_{j,k}]$, $j=0,\ldots,\dfrac{M}{2}-1$ $k=0,\ldots,\dfrac{M}{2}-1$.

Remark: If it is necessary, we also can use matrix multiply Wavelet Coefficient Image of 1 levels=$W_1 C^{\circ} W_1^T$.

Here, $W^1$ is the transform matrix for $I$ level wavelet decomposition.

FIG. 3 depicts a three-level wavelet decomposition, where L = 3.

In step 26, the first loss in accuracy occurs. Both thresholding and quantization reduce accuracy with which the wavelet coefficients are represented. In step 26, the wavelet coefficients are matched against threshold values, and if the values are less than the established threshold values specified, then the resultant value is set to zero.

An important feature of the invention is that the wavelet coefficients are then quantized to a number of levels depending upon which quadrant is being processed, and the desired compression or quality factor. This can be very important in image compression, as it tends to make many coefficients zeros, especially those for high spatial frequencies, which reduces the size of a compressed image.

A multilevel uniform thresholding method can be used as described below.

Let $T = (t_1, \ldots, t_L, t_{L+1})$ be the chosen thresholds, where $t_t$ is the threshold for 1 the (1=I, ..., L) level and $t_{L+1}$ is a threshold for blurred image $C^L$. Thresholding sets every entry in the blocks $C^L$, $HD^1$, $VD^1$ and $DD^1$ (1 = I, ... L) to be zero if its absolute value is not greater than the corresponding threshold.

For color images, three threshold vectors which correspond three different color planes, such as y, I and Q, are used.

The step of quantization essentially scales the wavelet coefficients and truncates them to a predetermined set of integer values. The quantization table shown in Table 1 can be used.

| | | | | |
|---|---|---|---|---|
| $q^1_{HD}$<br>$q^1_{VD}$<br>$q^1_{DD}$ | $q^2_{HD}$<br>$q^2_{VD}$<br>$q^2_{DD}$ | ...<br>...<br>... | $q^L_{HD}$<br>$q^L_{VD}$<br>$q^L_{DD}$ | $q_c^{L+1}$ |

TABLE 1

In Table 1, the entries $q^1_{HD}$ are quantization factors for blocks $HD^1$ (1 = I, ... L), $q^1_{VD}$ and $q^1_{DD}$ for blocks $VD^1$ and $DD^1$ (1 = I, ..., L) respectively, and the factor $q_c^{L+1}$ is for the most blurred image $C^L$. The factors can be integers between 0 and 255. The quantization scheme for the block $HD^1$ (1 = I, ..., L) is

$$\overline{hd}^1_{j,k}=round\ \frac{hd^1_{j,k}\cdot q^1_{HD}}{\max^1_{HD}},\ j=0,\ldots,\frac{M}{2^1}-1;\ k=0,\ldots,\frac{N}{2^1}-1. \quad (3.2.1)$$

Here, $\overline{hd}^1_{j,k}(j=0,\ldots,\frac{M}{2^1}-1;\ ,k=0,\ldots,\frac{N}{2^1}-1)$ *are quantized wavelet coefficients of block* $HD^1(1=1,\ldots,L)$

$$\max^1_{HD}=\max_{\substack{0\le j\le (M/2^1-1)\\ 0\le k\le (N/2^1-1)}}(|hd^1_{j,k}|),$$

and the function round*(x)* gives the nearest integer of x. Equation (3.2.1) is used for quantization of the other blocks (quadrants).

For color images, there are three separate quantization tables for the different color bands.

In step 28, entropy compression is applied to the resultant coefficients using either Arithmetic, Run Length, or Huffman, or Huffman and Run Length combined. The compression algorithm can be selected at run-time by the user, based on the desired compression ratio and the amount of time required to get the selected level of compression. The encoding step includes the entropy compression as well as coefficient rearranging.

An alternative process to that shown in FIG. 1 includes an optional down sampling of the IQ color planes. This down sampling may be done once or twice to produce two image planes either one-fourth or one-sixteenth the size of the original plane. If the down sampling is done, it will be accomplished prior to the wavelet transform of step 24. The down sampling reduces the compression time and size of the image file.

FIG. 5 shows a corresponding method for decompressing an image compressed using the method of FIG. 1. In step 40, the compressed image file is input. In step 42, the image is decoded. Next, in step 44 the values are de-quantized. Next, in step 46 inverse color and wavelet transformations are performed on the de-quantized data. In step 48, optional image post-processing takes place to refine the decompressed image. In step 50, the decompressed image is displayed.

The decoding of step 42 is the inverse operation of the encoding of step 28. Similarly, it can be divided into two parts: Entropy decoding (Huffman or arithmetic), and coefficient rearranging.

The decoding step produces quantized wavelet coefficients in 3*L+1 blocks. Dequantizing (step 44) uses the same quantization table as quantizing (Table 1), and the scheme as follows: for l = I, ..., L

$$\underline{hd}^{l}_{j,k} = \frac{\overline{hd}^{l}_{j,k} \cdot \max^{l}_{HD}}{q_{HD}}, j=0,\ldots,\frac{M}{2^{l}}-1;\ k=0,\ldots,\frac{N}{2^{l}}-1. \quad \textbf{(4.2.1)}$$

Equation (4.2.1) produces the approximate coefficients for the blocks $HD^{l}$ (l = I, ..., L), which are shown in FIG. 3. The dequantizing scheme for other blocks is similar to 4.1.2).

In step 46, the inverse wavelet transform, also referred to as wavelet reconstruction, is performed prior to the inverse color transformation. FIG. 4 depicts a one-level wavelet reconstruction.

The wavelet reconstruction can be iteratively performed for various levels of decomposition, according to the following equations.

(1) Inverse transform for rows:

For j=0, ....., $\frac{M}{2}$ -1, calculate

$$\begin{cases} \tilde{d}^1_{j,1}=\upsilon d^1_{j,0}+\dfrac{dd^1_{j,0}+dd^1_{j,1}}{2} \\ \tilde{d}^1_{j,2k+1}=\upsilon d^1_{jk}+\dfrac{dd^1_{j,k}-dd^1_{j,k+1}}{2}, k=1,\ldots,\dfrac{N}{2}-2, \\ \tilde{d}_{N-1,k}=\upsilon d_{j,\frac{N-1}{2}}+dd^1_{j,\frac{N-2}{2}}. \end{cases} \quad (4.3.1)$$

and

$$\begin{cases} \tilde{d}^1_{j,0}=d^1_{j,1}-2dd^1_{j,0}, \\ \tilde{d}^1_{j,2k}=\dfrac{\tilde{d}^1_{j,2k-1}+\tilde{d}^1_{j,2k+1}}{2}-2dd^1_{j,k},\ k=1,\ldots,\dfrac{N}{2}-1. \end{cases} \quad (4.3.2)$$

For j=0, ..., M/2 - 1, calculate

$$\begin{cases} \tilde{c}^1_{j,1}=c^1_{j0}+\dfrac{hd^1_{j0}+hd^1_{j1}}{2} \\ \tilde{c}^1_{j,2k+1}=c^1_{jk}+\dfrac{hd^1_{j,k}-hd^1_{j,k+1}}{2}, k=1,\ldots,\dfrac{N}{2}-2, \\ \tilde{c}_{j,N-1}=c_{j,\frac{N-2}{2}}+hd^1_{j,\frac{N-2}{2}}. \end{cases} \quad (4.3.3)$$

and

$$\begin{cases} \tilde{c}^1_{j,0}=c^1_{j,1}-2hd^1_{j,0}, \\ \tilde{c}^1_{j,2k}=\dfrac{1}{2}(\tilde{c}^1_{j,2k-1}+\tilde{c}^1_{j,2k+1})-2hd^1_{jk},\ k=1\ldots,\dfrac{N}{2}-1. \end{cases} \quad (4.3.4)$$

(2) Inverse transform for column:

For k=0, ..., N - 1, calculate

and

$$\begin{cases} \overline{c}^0_{1,k}=\tilde{c}^1_{0k}+\dfrac{\tilde{d}^1_{0k}+\tilde{d}^1_{1,k}}{2}, \\ \overline{c}^0_{2j+1,k}=\tilde{c}^1_{jk}+\dfrac{\tilde{d}^1_{j,k}-\tilde{d}^1_{j+1,k}}{2}, j=1,\ldots,\dfrac{M}{2}-2, \\ \overline{c}^0_{N-1,k}=\tilde{c}^1_{\frac{N-2}{2},k}+\tilde{d}^1_{\frac{N-2}{2},k}. \end{cases} \quad (4.3.5)$$

$$\begin{cases} \overline{c}^0_{0k}=c^0_{1k}-2\tilde{d}^1_{0k}, \\ \overline{c}^0_{2j,k}=\dfrac{1}{2}(\overline{c}^0_{2j-1,k}+\overline{c}^0_{2j+1k},)-2\tilde{d}^1_{jk},\ j=1,\ldots,\dfrac{M}{2}-1. \end{cases} \quad (4.3.6)$$

(3) $c^0_{j,k}=\tilde{c}^0_{j,k}/r,\ j=0,\ldots,M-1;\ k=0,\ldots,N-1.\ C^0=[c^0_{j,k}]^N_1 x^N_2.$

Following the inverse wavelet transformation, an inverse color transform is performed. Equations (5)-(6) give the inverse transforms for the YIQ and YUV color spaces.

For YIQ to RGB:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix} \quad (5)$$

For YUV to RGB:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.000 & 1.140 \\ 1.000 & -0.395 & -0.581 \\ 1.000 & 2.032 & 0.000 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \quad (6)$$

In step 48, a user can optionally apply image filtering to improve the image quality. Filters are known in the art for sharpening, smoothing and brightening images. Users can choose any number of processing filters at compression time. Information defining the selected filters can be stored in the coded image file, in a form such as a one byte flag in a file header. In addition to optionally applying the filters,

the method can also be implemented to automatically detect and apply the selected filters following decompression.

To sharpen an image, a filter is used that weights the eight pixels adjacent to the current pixel, as well as the current pixel, by one or more predetermined values. The weighted values of the nine pixels are then summed to derive a new value for the current pixel. For example, the surrounding eight pixel values can be weighted by the value -35/800, while the current pixel is weighted by 1.35. The sharpening filter is applied to every pixel in the image.

To smooth images, for every pixel, the average of the pixel and the eight adjacent pixels is calculated. Then the pixel value and the average is compared. The smaller of the two replaces the original pixel and is output as the smoothed pixel value.

To brighten images, the weighted sum of each pixel and the correspond eight adjacent pixels is calculated. For example, each of the adjacent pixels can be multiplied by the value 1/90 and the summed with the current pixel to obtain a brighten current pixel.

Another filter that can be used is one that adds a random value between [-12, 12] to each of the pixels in the image.

In FIG. 6 there is displayed a preferred hardware platform that can execute software for implementing an embodiment of the present invention. The computer system of FIG. 3 includes a CPU 62, a main memory 64, an I/O subsystem 66, and a display 68, all coupled to a CPU bus 70. The I/O subsystem 66 communicates with peripheral devices that include an image source 72, an image storage device 74, and a mass storage memory 76. Although shown as three separate devices, peripherals 72-76 can be implemented using a single memory device, such as a hard disk drive commonly found in computers.

The image source 72 may be a digital still image or video source, such as a CD-ROM drive, scanner, or network connection. In addition, the image source 85 can include analog video sources, such as a video camera, VCR, television broadcast or cable receiver. The analog video signals would be converted to a digital form by the image source 85 using conventional conversion techniques. Alternatively, an image source 72 can include a video camera and communications systems for transmitting real-time video to the I/O subsystem 66.

The image storage 74 can be a computer disk, such as a that used by a hard drive, or a portable memory medium, such as a floppy or ZIP disk, or a read/write optical CD.

In operation, a computer program, which implements aspects of the invention, is retrieved from the mass storage memory 76 into the main memory 64 for execution by the CPU 62. Upon execution of the compression aspect of the invention, the compressed image file can be stored in the image storage 74; while upon execution of the decompression aspect of the invention, the decompressed image can be viewed on the display 68. Operating under the control of the computer program, the CPU 62 can process images according to the methods set forth herein, as shown in FIGS. 1-2 and 6-10.

FIG. 7 illustrates an alternative hardware platform implementing a system in accordance with a further embodiment of the present invention. System 80 can be implemented using a variety of different hardware components, such as ASIC (Application Specific Integrated Circuits), or a combination of discrete digital components, such as microprocessors, standard logic components, and other programmable logic devices. The system 80 includes a compression system 81 and a decompression system 82. The compression system 81 can be configured to perform any one or combination of the compression methods set forth in FIGS. 1, 8, 10, and 12;

while the decompression system can be configured to perform any one or combination of the decompression methods set forth in FIGS. 5, 9, 11, and 13.

An image source 85 provides digital pixel values to a color converter 84. The image source 85 can provide the same functionality as described earlier for the image source 72 of FIG. 6.

The color converter 84 performs a color space transformation on the input pixels, such as any of those described herein for FIG. 1. The converter functionality can be provided by conventional integrated circuits that are readily available from various manufacturers. Compressor 86 compresses the transformed pixels, removing redundant data. The compressed image file generated by the compressor 86 can be transferred directly to the decompression system 82 over a transmission medium 91. The transmission medium 91 can be a radio-link, computer network, cable television network, or satellite link. Alternatively, the compressor 86 can transmit its output to a portable storage medium 92, such as an optical, floppy, or ZIP disk; or to a mass storage device 94 such as a computer hard disk or archival system.

The decompressor 88 expands the compressed image file by applying an inverse wavelet transformation, as well as de-quantization and de-encoding functions. The decompressed data is then passed to an inverse color converter 90 that applies an inverse color space transformation to generate pixel values in a color space and format appropriate for the image display 89. Standard electronic components are readily available for performing the function of the inverse color converter 90.

FIG. 8 illustrates a flow diagram of a method of compressing an image in accordance with an alternative embodiment of the present invention. In step 100, a digital image is input. In step 102, a color space transformation is performed on the input image pixels.

In step 104, the pixels are subjected to a wavelet transformation. In step 106, sub-band quantization is performed on the wavelet coefficients. Next, in step 108 the quantized sub-bands are respectively entropy encoded. In step 110, the coded image file is output.

Sub-band oriented quantization and entropy coding are well suited for wavelet-based image compression. The main idea is to take the advantage of different quantizations at different sub-bands (wavelet quadrant) and encode each band accordingly. Quadrants having a high variance in wavelet values can be allocated a finer mesh size for quantization, while those quadrants with smaller variances will be assigned fewer levels of quantization. That is, the number of bits one wishes to allocate to the output could be varied by quadrant. Those quadrants with large variances will utilize more bits, while those with low variants will utilize fewer bits. In this way, the number of bits resulting from quantization will remain the same, but their allocation will differ depending upon the nature of the image. This technique greatly improves image quality while maintaining a high compression ratio.

FIG. 9 illustrates a flow diagram of a method of decompressing an image compressed according to the methods shown in FIG. 8. Step 120, the compressed file is input. In step 122, the input image is entropy decoded. In step 124, de-quantization is performed on the decoded image file. Next, in step 126, an inverse wavelet transform is performed on the image. In step 128, an inverse color transformation is performed. In step 130, post-processing altering is optionally performed. In step 132, the decompressed image file is then displayed.

FIG. 10 illustrates a flow diagram of a method of compressing an image in accordance with another embodiment of the present invention. This method performs color-bit depth compression, which essentially

reduces the number of colors in the image to achieve compression. In step 140, the image is input with its original color. For example, each color pixel could be represented by a standard 24-bit value. Next, in step 142, a color table is created corresponding to the image. The color table is a set of quantized color values. The quantized color values represent a smaller number of colors with correspondingly fewer bits. Each of the input pixels is mapped to the color table. In step 144, an index is calculated for each pixel in the image by dithering the pixel values. Dithering is accomplished by weighting pixels adjacent to the current pixel in a frame and then arithmetically combining the weighted values with the current pixel value to produce the index, which then represents the current pixel. The dithering process is repeated for each pixel in a frame. In step 146, the indexes are wavelet transformed. In step 148, the wavelet coefficients are entropy coded. In step 150, the coded image file is output.

FIG. 11 illustrates a flow diagram of a method of decompressing an image that has been compressed according to the method shown in FIG. 10. In step 160, a compressed image file is received. Next, in step 162, the image file is entropy decoded. In step 164, an inverse wavelet transform is applied to the decoded data. Next, in step 166, post-processing filtering of the image is optionally applied. Next, in step 168, the decompressed image is displayed.

FIG. 12 illustrates another method of compressing an image in accordance with another embodiment of the present invention. In this method, a user can selective vary compression parameters (step 173) to obtain a lossless or near-lossless compressed image at a desired compression ratio. In step 170, the image is input. In step 172, an integer color transform is performed on the input image. In step 173, compression parameters are selected by the user using a software

interface. These parameters can include those described herein below in the subsection title "Peak Signal to Noise Ratio (PSNR) Controlled Compression". In step 174, an integer wavelet transform is performed on the color transformed pixels. In step 176, the wavelet coefficients are entropy coded. Next, in step 178, the compressed image file is then output from the system.

The integer color transformation of step 172 is an integer reversible transform which can be used in color image compression to reduce processing time and image size. Step 172 transforms RGB color components to a set of color components Y-Nb-Nr, which are known.

The RGB to Y-Nb-Nr transform is given by the equations:

$$Y = G + \mathrm{Int}(R/2 + B/2),$$
$$Nb = B - \mathrm{Int}(Y/2),$$
$$Nr = R - \mathrm{Int}(Y/2).$$

The integer wavelet transform of step 174 is described below in detail.

FIG. 13 illustrates a method of decompressing an image file that has been compressed according to the method shown in FIG. 12. In step 180, a compressed image file is input. In step 182, the image is entropy decoded. Next, in step 184, an inverse integer wavelet transform is performed on the decoded data. In step 186, an inverse integer color transform is performed. Next, in step 188 optional post-processing filtering is performed on the image. Next, in step 190, the decompressed image is displayed.

The Y-Nb-Nr to RGB transform of step 186 is given by the equations:

$$R = Nr + \mathrm{Int}(Y/2),$$
$$B = Nb + \mathrm{Int}(Y/2),$$
$$G = Y - \mathrm{Int}\,(R/2 + B/2)$$

The inverse integer wavelet transform of step 184 is described in detail below.

Reversible Integer Wavelet Transform

This method allows a series of transformations which are very close to the corresponding biorthogonal wavelet transforms or some non-orthogonal wavelet transforms, but can be calculated with only integer addition and bit-shift operations. In addition, the integer wavelet transforms created disclosed herein possess a property of precision preservation (PPP). This property is very useful for conserving memory in both compression and decompression, and speed up the whole procedure in some applications. Two general methods from which one can get the integer wavelet transform desired are disclosed.

Basic Integer Wavelet Transformations

Two examples are provided as the starting point for the unique method. For the sake of convenience, length, and simplicity, presented is only the algorithm for a one level decomposition and reconstruction and only for a one dimensional signal. The extension to two dimensions is immediate as the rows and columns can be treated into a sequence of one dimensional signals. For the following examples, assume that $\{C_n^0\}_{n=0}^{N-1}$ is the original signal where the superscript indicates level and the subscript indicates a particular point in the signal. Also, $\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{N_1-1}$ are its decomposition parts at the first level. Here

$$N_1 = \begin{cases} \frac{N}{2}, & \text{if } N \text{ is an even number,} \\ \frac{N+1}{2}, & \text{if } N \text{ is an odd number;} \end{cases}$$

$$M_1 = N - N_1$$

$\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{M_1-1}$ are its low frequency (l) part and high frequency (h) part, respectively. For multi-levels, we just create $\{c_n^1\}_{n=0}^{N_1-1}$ as $\{c_n^0\}_{n=0}^{N_1}$ and repeat the procedure again.

Example 1: A (2,2)-wavelet transform by integer calculation.

This transformation is similar to a variation of the Haar wavelet transform which uses low and high pass analysis (decomposition) filters given as:

| n | 0 | 1 |
|---|---|---|
| $\tilde{h}_n$ | 1/2 | 1/2 |
| $\tilde{g}_n$ | 1/2 | -1/2 |

(1) Compute

$$d_k^1 = c_{2k}^0 - c_{2k+1}^0, \quad k = 0, \ldots, M_1 - 1. \qquad (2.1)$$

(2) Compute

$$c_k^1 = Int\left(\frac{d_k^1}{2}\right) + c_{2k+1}^0, \quad k=0,\ldots,N_1-2,$$

$$c_{N_1-1}^1 = \begin{cases} Int\left(\frac{d_{M_1-1}^1}{2}\right) + C_{2k+1}^0, & \text{if } N \text{ is an even number,} \\ c_{N-1}^0, & \text{if } N \text{ is an odd number.} \end{cases} \qquad (2.2)$$

Here, Int(x) is an arbitrary rounding function which may have different interpretations. For example, Int(x) can be the integer which is nearest to x, or Int(x) may be any integer which satisfies $x-1<Int(x)\leq x$, etc. It is easy to see that all entries in both $\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{M_1-1}$ are integers.

From (2.1)-(2.2), we can easily get the following integer reconstruction algorithm:

(b) Reconstruction

(1) If N is an even number, compute:

$$c_{2k+1}^0 = c_k^1 - Int\left(\frac{d_k^1}{2}\right), \quad k=0,\ldots,N_1-1; \qquad (2.3)$$

or, if N is an odd number, we have

$$c_{2k+1}^0 = c_k^1 - Int\left(\frac{d_k^1}{2}\right). \quad k=0,\ldots,N_1-2; \qquad (2.4)$$
$$c_{N-1}^0 = c_{N_1}^1.$$

(2) Compute

$$c_{2k}^0 = d_k^1 + c_{2k+1}^0 \quad k=0,\ldots,M_1-1. \qquad (2.5)$$

*Remark*. Since (2.1)-(2.6) are not linear because of the rounding operation Int(x), this means the transformation order becomes significant. For instance, if the decomposition was applied first to the columns and then to the rows, the inverse transformation must be applied first to the rows and then to the columns.

Example 2: Lazy wavelet transform.
The lazy wavelet transform is used to illustrate an important concept. The corresponding inverse transform is nothing else but sub-sampling the even and odd indexed samples. Decomposition and reconstruction can use the same formula as follows:

$$c_k^1 = c_{2k}^0, \quad k=0,\ldots,N_1-1;$$
$$d_k^1 = c_{2k+1}^0, \quad k=0,\ldots,M_1-1.$$

Examples 1 and 2 are not good transforms for image compression, but they are simple. Much better transforms can be achieved from these two. As suggested above, they are considered only as a starting point for the integer, reversible, wavelet transform algorithm of the disclosed invention.

It is noted that there is another interesting property in the above two transforms which may not be easily seen. If the values of the signal pixels are represented by a finite number of bits, say one bit or one byte, the same number of bits can be used to represent the result of the forward transform within the computer itself because of the complementary code property. While, from the reconstruction algorithm, the computer will get back the exact original signal through the same complementary code property. This property is called a *Property of Precision Preservation (PPP)* for these wavelets.

It is known that the general values for the high frequency wavelet coefficients are small, and all higher levels of the decomposition also provide generally small

values in the high frequency band. This allows the preservation of precision during the computational stage of the wavelet coefficients. Now, the complementary code property, the other aspect of the *PPP* property is a well known characteristic of integer arithmetic as done by the computer. Consider the computation of the difference of two integers given as $c = b - a$ and the inverse computation of $a = b - c$. The nature of the computation within the computer can be specified as follows:

$$c_m = \begin{cases} b-a \ if \ -2^{q-1} \le b-a < 2^{q-1}-1 \\ -2^q+b-a \ if \ b-a \ge 2^{q-1} \\ 2^q+b-a \ if \ b-a < -2^{q-1} \end{cases}$$

and the inverse is

$$a_m = \begin{cases} b-c_m \ if \ -2^{q-1} \le b-a < 2^{q-1}-1 \\ -2^q+b-c_m \ if \ b-c_m \ge 2^{q-1} \\ 2^q+b-c_m \ if \ b-c_m < -2^{q-1} \end{cases}$$

where the m subscript indicates the internal representation, and the range of the integers a, b, c is $[-2^{q-1}, 2^{q-1}-1]$ . The internal representation of $c_m$ when it is outside the range, its appearance is as a two's complement number, so the representation may not be the same as the external representation of c. However, the same complementary code for the $a_m$ will cause the internal representation to be identical to the external representation of a. For example, if we let b = 2 (00000010) and a = -127 (10000001) then $c_m$ has the internal binary value of (10000001) when q=4. With a value of -127 for $c_m$ the inverse value for $a_m$ will just be *a*.

In fact, for Example 2, this property is obviously true. While for Example 1, if the range of the pixel values is within a finite number of bits, say q, we can only use q bits as the working unit, which means the

value of transform coefficients will also be within the interval with length $2^q$, say $[-2^{q-1}, 2^{q-1} - 1]$. Due to the nature of computation on a machine, most machines will implement (2.1)-(2.2) automatically as follows (the complementary code property):

$$d_k^1=\begin{cases} c_{2k}^0-c_{2k+1}^0, & if\ -2^{q-1}\le c_{2k}^0-c_{2k+1}^0<2^{q-1}, \\ c_{2k}^0-c_{2k+1}^0-2^q, & if\ c_{2k}^0-c_{2k+1}^0\ge 2^{q-1}, \\ 2^q+(c_{2k}^0-c_{2k+1}^0), & if\ c_{2k}^0-c_{2k+1}^0< -2^{q-1}. \end{cases} \tag{2.6}$$

$$c_k^1=\begin{cases} Int(\frac{d_k^1}{2})+c_{2k+1}^0, & if\ -2^{q-1}\le Int(\frac{d_k^1}{2})+c_{2k+1}^0<2^{q-1}, \\ Int(\frac{d_k^1}{2})+c_{2k+1}^0-2^q, & if\ Int(\frac{d_k^1}{2})+c_{2k+1}^0\ge 2^{q-1}, \\ Int(\frac{d_k^1}{2})+c_{2k+1}^0+2^q, & if\ Int(\frac{d_k^1}{2})+c_{2k+1}^0< -2^{q-1}. \end{cases} \tag{2.7}$$

While the reconstruction algorithm (2.3) and (2.5) will be implemented by the computer itself as

$$c_{2k+1}^0=\begin{cases} c_k^1-Int(\frac{d_k^1}{2}), & if\ -2^{q-1}\le c_k^1-Int(\frac{d_k^1}{2})<2^q, \\ 2^q+\left(c_k^1-Int(\frac{d_k^1}{2})\right), & if\ c_k^1-Int(\frac{d_k^1}{2})<-2^{q-1}, \\ \left(c_k^1-Int(\frac{d_k^1}{2})\right)-2^q, & if\ c_k^1-Int(\frac{d_k^1}{2})>-2^{q-1}. \end{cases} \tag{2.8}$$

$$c_{2k}^0=\begin{cases} d_k^1+c_{2k+1}^0, & if\ -2^{q-1}\le d_k^1+c_{2k+1}^0<2^{q-1}, \\ d_k^1+c_{2k+1}^0+2^q, & if\ d_k^1+c_{2k+1}^0<-2^{q-1}, \\ d_k^1+c_{2k+1}^0-2^q, & if\ d_k^1+c_{2k+1}^0\ge -2^{q-1}. \end{cases} \tag{2.9}$$

It is obvious that (2.8)-(2.9) are just the reverse of (2.6)-(2.7). It is also easy to see that if we properly take advantage of the bound in the coefficient size mentioned above, the algorithm can be implemented using a minimal amount of storage.

The following are examples which give motivation for our new approach.

Example 3: A (2.6) wavelet transform by integer calculation (2).

This transformation is similar to using the following analysis filters:

| n | -2 | -1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| $\tilde{h}_n$ | 0 | 0 | 1/2 | 1/2 | 0 | 0 |
| $\tilde{g}_n$ | -1/16 | -1/16 | 1/2 | -1/2 | 1/16 | 1/16 |

(a) Decomposition

Decomposition starts with Example 1 at step (1) and (2), and then upgrades the high frequency component at step (3):

(1) Compute

$$d_k^{1,0}=c_{2k}^0-c_{2k+1}^0, k=0,\ldots,M_1-1.$$

(2) Compute

$$c_k^1=Int(\frac{d_k^{1,0}}{2})+c_{2k+1}^0,\ k=0,\ldots,N_1-2,$$

$$c_{N_1-1}^1=\begin{cases}Int(\frac{d_{M_1-1}^{1,0}}{2}),+c_{N-1}^0, & \text{if } N \text{ is an even number},\\ c_{N-1}^0, & \text{if } N \text{ is an odd number};\end{cases}$$

(3) Compute

$$\begin{cases}d_0^1=Int(\frac{c_0-c_1^1}{4}),+d_0^{1,0}\\ d_k^1,=Int(\frac{c_{k-1}^1-c_{k+1}^1}{4})-d_k^{1,0}, k=1,\ldots,M_1-2,\end{cases}$$

and then, if N is even, calculate

$$d_{M_1-1}^{1}=Int(\frac{C_{N_1-2}^{1}-C_{N_1-1}^{1}}{4})-d_{M_1-1}^{1,0},$$

else, calculate

$$d_{M_1-1}^{1}=Int(\frac{C_{N_1-3}^{1}-C_{N_1-1}^{1}}{4})-d_{M_1-1}^{1,0}.$$

(b) Reconstruction

The reconstruction algorithm is identical to the decomposition algorithm, except it is now running "backwards".

(1) Compute

$$\begin{cases} d_0^{1,0}=Int(\frac{C_0^1-C_1^1}{4})-d_0^1 \\ d_k^{1,0}=Int(\frac{C_{k-1}^1-C_{k+1}^1}{4})-d_k^1, k=1,\ldots,M_1-2, \end{cases}$$

and then, if N is even, calculate

$$d_{M_1-1}^{1,0}=Int(\frac{C_{N_1-2}^{1}-C_{N_1-1}^{1}}{4})-d_{M_1-1}^{1,1},$$

else calculate

$$d_{M_1-1}^{1,0}=Int(\frac{C_{N_1-3}^{1}-C_{N_1-1}^{1}}{4})-d_{M_1-1}^{1,1},$$

(2) If N is an even number, compute

$$c_{2k+1}^{0}=c_k^1-Int(\frac{d_k^{1,0}}{2}),\ k=0,\ldots,N_1-1;$$

or, if N is an odd number, we have

$$c_{2k+1}^{0}=c_k^1-Int(\frac{d_k^{1,0}}{2}),\ k=0,\ldots,N_1-2;$$

$$c_{N-1}^{0}=c_{N_1}^{1}.$$

(3) Compute

$$c_{2k}^0=d_k^{1,0}+c_{2k+1}^0,\ k=0,\ldots,M_1-1.$$

We see in step (2)-(3) above, that they are just the same as shown for the reconstruction of the (2.2)-wavelet transform (Example 1).

Example 4: A (1,3)-wavelet transform by integer calculation.

The following nonlinear transform is a variation of the transform which uses biorthogonal analysis filters:

| n | -1 | 0 | 1 |
|---|---|---|---|
| $\tilde{h}_n$ | 1 | 0 | 0 |
| $\tilde{g}_n$ | 1/4 | -1/2 | 1/4 |

(a) Decomposition

This decomposition starts with the *Lazy wavelet* at step (1) and upgrades the high frequency component at step (2):

(1) Set

$$c_k^1=c_{2k}^0,\ k=0,\ldots,N_1-1;$$

$$d_k^1=c_{2k+1}^0, k=0,\ldots,M_1-1.$$

(2) If N is an even number, calculate

$$\begin{cases} d_k^1=Int\left(\frac{c_{k-}^1-c_{k+1}^1}{2}\right)-d_k^{1,0}, k=0,\ldots,M_1-2, \\ d_{M_1-1}^1,=c_{N_1-1}^1-d_{M_1-1}^{1,0}. \end{cases}$$

Otherwise, if N is an odd number, calculate

$$d_k^1 = Int\left(\frac{c_{2k}^0 + c_{2k+2}^0}{2}\right) - c_{2k+1},\ k=0,\ldots,M_1-1.$$

(b) Reconstruction

(1) Set

$$c_{2k}^0 = c_k^1,\ k=0,\ldots,N_1-1;$$

(2) If N is an even number, calculate

$$\begin{cases} c_{2k+1}^0 = Int\left(\frac{c_{2k}^0 + c_{2k+2}^0}{2}\right) - d_k^1,\ k=0,\ldots,M_1-2, \\ c_{N-1}^0, = c_{N-2}^0 - d_{M_1-1}^1. \end{cases}$$

Otherwise, if N is an odd number, calculate

$$c_{2k+1}^0 = Int\left(\frac{c_{2k}^0 + c_{2k+2}^0}{2}\right) - d_k^1,\ k=0,\ldots,M_1-1.$$

Example 5: (5,3)-wavelet transform by integer calculation.

This transformation is also similar in function to using the biorthogonal analysis filters. It is given by

| n | -2 | -1 | 0 | 1 | 2 |
|---|---|---|---|---|---|
| $\tilde{h}_n$ | -1/8 | 1/4 | 3/4 | 1/4 | -1/8 |
| $\tilde{g}_n$ | 1/4 | -1/2 | 1/4 | 0 | 0 |

(a) Decomposition

This decomposition starts with Example 3 at step (1) and upgrade low frequency components at step (2):

(1) Set

$$c_k^{1,0}=c_{2k}^0,\ k=0,\ldots,N_1-1;$$

If N is an even number, calculate

$$\begin{cases} d_k^1=Int(\frac{c_{2k}^0+c_{2k+2}^0}{2})-c_{2k+1}^0,\ k=0,\ldots,M_1-2, \\ d_{M_1-1}^1=c_{N-2}^0-c_{N-1}^1. \end{cases}$$

Otherwise, if N is an odd number, calculate

$$d_k^1=Int(\frac{c_{2k}^0+c_{2k+2}^0}{2})-c_{2k+1},\ k=0,\ldots,M_1-1.$$

(2) If N is an even number, compute

$$\begin{cases} c_0^1=c_0^{1,0}-Int(\frac{d_0^1}{2}), \\ c_k^1=c_k^{1,0}-Int(\frac{d_{k-1}^1+d_k^1}{4}),\ k=1,\ldots,N_1-2, \\ c_{N-1}^1=c_{N_1-2}^{1,0}-Int(\frac{d_{N_1-2}^1+d_{N_1-1}^1}{4}). \end{cases}$$

Otherwise, if N is an odd number, calculate

$$\begin{cases} c_0^1=c_0^{1,0}-Int(\frac{d_0^1}{2}), \\ c_k^1=c_k^{1,0}-\frac{d_{k-1}^1+d_k^1}{4},\ k=1,\ldots,N_1-2, \\ c_{N_1-1}^1=c_{N_1-1}^{1,0}-Int(\frac{d_{M_1-1}^1}{2}). \end{cases}$$

(b) Reconstruction

(1) Compute

$$c_0^0=c_0^1+Int(\frac{d_0^1}{2}),$$

$$c_{2K}^0=c_k^1+Int(\frac{d_{k-1}^1+d_k^1}{4}),\ k=1,\ldots,N_1-2,$$

Then, if N is even, calculate

$$c_{N-2}^0=c_{N_1-1}^1+Int(\frac{d_{N_1-2}^1+d_{N_1-1}^1}{4}).$$

else calculate

$$c_{N-1}^0=c_{N_1-1}^1+Int(\frac{d_{M_1-1}^1}{2}).$$

(2) Compute

$$c_{2k+1}^0=Int(\frac{c_{2k}^0+c_{2k+2}^0}{2})-d_k^1,\ k=0,\ldots,M_1-2,$$

Then, if N is even, calculate

$$c_{N-1}^0=c_{N-2}^0-d_{M_1-1}^1.$$

The *PPP* property for Examples 1-2 mentioned at the end of the previous section is also applicable for these three examples. It is obvious these three transformations are not really linear, but they are similar to the one using the corresponding filters given above. Especially, the filters in Example 3 and Example 5 belong to, with minor modification, the group of the best biorthogonal filters for image compression.

Also, from the above three examples, we can note that if we begin with integer (linear or nonlinear) wavelet transformations and then use some proper upgrading formulas, we can get other, much better integer, wavelet transformations for image compression.

Lifting Scheme and Integer Biorthogonal Filtering

The *Lifting scheme*, discussed by W. Sweldens in "The Lifting Scheme: A Custom-Designed Construction of Biorthogonal Wavelet", Applied and Computational Harmonic Analysis, Vol. 3, No. 2, April 1996, is a recently developed approach for constructing biorthogonal wavelets with compact support. It can be used, with minor modifications, to create integer biorthogonal wavelet transformations. The following is an adaptation of the lifting scheme.

Definition 1. The set of filters {h, h, g, g}, a set of biorthogonal filters if the following formula is satisfied:

$$\forall \omega \in \mathbf{R} : \tilde{m}(\omega)\overline{M'(\omega)} = 1.$$

where $$m(\omega) = \begin{bmatrix} h(\omega) & h(\omega+\pi) \\ g(\omega) & g(\omega+\pi) \end{bmatrix},$$

and $$h(\omega) = \sum_k h_k e^{-k\omega} \text{ and } g(\omega) = \sum_k g_k e^{-k\omega},$$

and *similarly for* $\tilde{m}(\omega)$, $\tilde{h}(\omega)$ *and* $\tilde{g}(\omega)$.

The following lemma is the main result of the lifting scheme [1] reported as corollary 6 in that paper.

Lemma 1. Take an initial set of finite biorthogonal filters {h, $\tilde{h}^{\circ}$, g, $\tilde{g}^{\circ}$}, then a new set of finite biorthogonal filters {h, $\tilde{h}$, g, $\tilde{g}$} can be found as

$$\tilde{h}(\omega) = \tilde{h}^{0}(\omega) + \tilde{g}(\omega)\overline{s(2\omega)}$$
$$g(\omega) = g^{0}(\omega) - h(\omega)s(2\omega).$$

Similarly if we take {$h^{\circ}$, $\tilde{h}$, g, $\tilde{g}^{\circ}$} as an initial set of biorthogonal filters, a new set of biorthogonal filters {h, $\tilde{h}$, g, $\tilde{g}$} can be found as can be found as

$$h(\omega)=h^0(\omega)+g(\omega)\overline{s(2\omega)}$$
$$\tilde{g}(\omega)=\tilde{g}^0(\omega)-\tilde{h}(\omega)\tilde{s}(2\omega).$$

Here $s(\omega)$ is a trigonometric polynomial and the corresponding filter s is finite, and so is $\tilde{s}(\omega)$. Actually, regarding the filters (4.1) is equivalent to

$$\tilde{h}_k=\tilde{h}_k^0+\sum_l \tilde{g}_{k+2l}\tilde{s}_l$$
$$g_k=g_k^0-\sum_l h_{k-2l}\tilde{s}_l$$

or

$$h_k=h_k^0+\sum_l g_{k+2l}\tilde{s}_l$$
$$\tilde{g}_k=\tilde{g}_k^0-\sum_l \tilde{h}_{k-2l}\tilde{s}_l \qquad (4.26)$$

Next we use the lifting scheme with minor modifications to create an integer, nonlinear, quasi-biorthogonal, wavelet algorithm. Suppose $[c_n^0]$ is a original signal, $[c_n^1]$ and $[d_n^1]$ are again its low and high frequency decomposition parts, obtained by using the filters $\{h, \tilde{h}, g, \tilde{g}\}$.

If we use filters $\{\tilde{h}, \tilde{g}\}$ for decomposition (analysis), the corresponding decomposition algorithm is

$$\begin{cases} c_k^1=\alpha_c\sum_n c_n^0\tilde{h}_{n-2k}, \\ d_k^1=\alpha_d\sum_n c_n^0\tilde{g}_{n-2k}. \end{cases}$$

While the reconstruction algorithm will be

$$c_n^0=2\sum_k\left(\frac{c_k^1 h_{n-2k}}{\alpha_c}+\frac{d_k^1 g_{n-2k}}{\alpha_d}\right),$$

related to the synthesis filter {h,g}. Here, parameters $\alpha_c$ and $\alpha_d$ are positive constants with $\alpha_c \cdot \alpha_d=2$. For example, in the situation of regular biorthogonal decomposition and reconstruction, $\alpha_c=\alpha_d=\sqrt{2}$; and for Example 1 through Example 5 above, $\alpha_c=1$ and $\alpha_d=2$.

If the set of filters {h, $\tilde{h}$, g, $\tilde{g}$} is from {h, $\tilde{h}^0$, $g^0$, $\tilde{g}$} by (4.2b), then decomposition can be accomplished as follows:

1. Calculate

$$\begin{cases} c_k^{1,0}=\alpha_c\sum_n c_n^0 \tilde{h}_{n-2k}^0, \\ d_k^1=\alpha_d\sum_n c_n^0 \tilde{g}_{n-2k}. \end{cases}$$

2. Calculate

$$c_k^1=c_k^{1,0}+\frac{\alpha_c}{\alpha_d}\sum_l d_{k-l}^1 s_l. \qquad (4.4)$$

The relative reconstruction scheme will be:

1. Calculate

$$c_k^{1,0}=c_k^1\frac{\alpha_c}{\alpha_d}\sum_l d_{k-l}^1 s_l \qquad (4.5)$$

2. Calculate

$$c_n^0=2\sum_k\left(\frac{c_k^{1,0} h_{n-2k}}{\alpha_c}+\frac{d_k^1 g_{n-2k}^0}{\alpha_d}\right). \qquad (4.6)$$

Here, equations (4.3) and (4.6) are just the wavelet (inverse) transforms using biorthogonal filters {h, $\tilde{h}^0$, $g^0$,

$\tilde{g}$}. While (4.4) and (4.5) are forward and backward upgrading formulas.

Similarly if the set of filters {h, $\tilde{h}$, g, $\tilde{g}$} is from the initial set of filters {h, $\tilde{h}$, g, $\tilde{g}$} by using (4.2b), the relative decomposition is:

1. Calculate

$$\begin{cases} c_k^1=\alpha_c\sum_n c_n^0 h_{n-2k} \\ d_k^{1,0}=\alpha_d\sum_n c_n^0\overline{g}_{n-2k}^0. \end{cases}$$

2. Calculate

$$d_k^1=d_k^{1,0}\frac{\alpha_c}{\alpha_d}\sum_1 c_{k-1^{a_1}}^1$$

The reconstruction scheme is:

1. Calculate

1. Calculate

$$d_k^{1,0}=d_k^1\frac{\alpha_c}{\alpha_d}\sum_1 c_{k-1^{a_1}}^1$$

2. Calculate

$$c_n^0=2\sum_k\left(\frac{c_k^{1,0}h_{n-2k}}{\alpha_c}+\frac{d_k^1 g_{n-2k}^0}{\alpha_d}\right).$$

Corollary 4.1. *Suppose biorthoganal filters* $\{h,\tilde{h},g,\tilde{g}\}$ *are from initial filters* $\{h,\tilde{h}^0,g^0,\tilde{g}\}$ *by the lifting scheme* (4.1a) *or* (4.2a). *If the decomposition and reconstruction by filters* $\{h,\tilde{h}^0,g^0,\tilde{g}\}$ *can be accomplished only by integer calculation, such as Example* 2, *we also can create a corresponding integer wavelet decomposition and reconstruction scheme which is very "close" to the original one by using filters* $\{h,\tilde{h},g,\tilde{g},\}$. *Here the word "close" means that the difference of the two decompostion schemes is just some rounding error, and this rounding error will l corrected by the integer reconstruction scheme.*

*In fact, if* $\{c_k^{1,0}\}$ *and* $\{d_k^1\}$ *are integer after* (4.3), *we can calculate* $\{c_k^1\}$ *by*

$$c_k^1=c_k^{1,0}+Int\left(\frac{\alpha_c}{\alpha_d}\sum d_{k-l}s_l\right)$$

instead of (4.4). Here Int(x), as described in Section 2, is an arbitrary rounding up function which satisfies $x-1 \le Int(x) \le x+1$. It is obvious that (4.7) is very close to (4.4), and the exact reconstruction scheme can easily be obtained from

$$c_k^{1,0}=c_k^1-Int\left(\frac{\alpha_c}{\alpha_d}\sum_l d_{k-l}s_l\right)$$

and (4.6). There will be a similar result, if the set of biorthogonal filters $\{h, \tilde{h}, g, \tilde{g}\}$ is obtained from the initial set of filters $\{\dot{h}, \tilde{h}, \dot{g}, \tilde{g}^0\}$ by using (4.2b).

Except for the example shown in the *Lazy wavelet* (Example 2), most standard biorthogonal wavelet forms cannot be performed directly by integer, even for one of the simplest wavelets, the *Harr wavelet*. However, if the parameters $\alpha_c$, and $\alpha_d$ are properly chosen and the transform algorithms, such as Example 1 and Example 3, are slightly changed, a variation of the original

biorthogonal wavelet transforms with respect to the set of filters {h, h, g, g} is created. On the other hand, the parameters should be also chosen carefully to guarantee that only addition and shift operations are needed by the algorithm.

If the set of filters $\{h, \tilde{h}, g, \tilde{g}\}$ is obtained from a set of filters $\{h^{\circ}, \tilde{h}, g, \tilde{g}^{\circ}\}$ by the lifting scheme, and the set $\{h^{\circ}, \tilde{h}, g, \tilde{g}^{\circ}\}$ is also obtained from a filter set $\{h^{\circ}, \tilde{h}^{\circ}, g^{\circ}, \tilde{g}^{\circ}\}$, one can repeatedly use Corollary 1 to get a "close" integer wavelet transformation.

## The Correction Method for Creating Integer Wavelet Transforms

Another approach for obtaining integer wavelets is using the so-called *Correction method*. The motivation of this method is from the S+P transform. The lifting scheme for generating biorthogonal wavelets can be considered as a special case of the correction method. From this can be derived complicated filters with fast decomposition and reconstruction algorithms.

Assuming a simple integer wavelet transform, such as Examples 1 through 3, the decomposition and reconstruction scheme of which can be formulated as follows:

Decomposition

$$c_1^{1,0}=df_c\left(\{c_n^0\}\right)$$
$$d_1^{1,0}=df_d\left(\{c_n^0\}\right) \qquad (5.1)$$

Reconstruction

$$c_n^0=rf\left(\{c_1^{1,0}\},\{d_k^{1,0}\}\right) \qquad (5.2)$$

Here, (5.1) and (5.2) can be the same as (4.3) and (4.6) or other algorithms.

In general, after the above decomposition, one may not be satisfied with the result. There may still be some correlation among the high pass components because

of the aliasing from the low pass components, or the low pass components do not carry enough of the expected information from the original signal. Hence, one could make an improvement by putting some correction part on the high pass components or low pass components. There are many ways to accomplish this. However, for the sake of the integer calculation, it is preferable to use following correction method. To make a correction for the high pass part, the corresponding formula would be:

$$d_2^1=d_k^{1,0}-Int(dc_{2,k}^1)k=\cdots,0,1,2\cdots \qquad (5.3)$$

Here, $dc_k^1$ is a correction quantity for $d_k^1$

$$d_k^1=\sum_{j=1}^{s_1}\sigma_i c_{k+1}^{1,0}+\sum_{j=1}^{1}\tau_j d_{k+j}^{1,0}, k=\cdots,0,1,2,\cdots \qquad (5.4)$$

and $\{\sigma_i\}_{j=s_0}^{s_1}$ and $\{\zeta_j\}_{j=1}^{T}$ are given parameters which have been chosen for the user's purpose such as reducing the redundancy among high pass components or some other special requirement. To preserve the integer calculation, any entries in both $\{\sigma_i\}_{i=s_0}^{s_1}$ and $\{\zeta_j\}_{j=1}^{T}$ should be rational numbers with denominators being powers of 2.

From (5.1), (5.3) and (5.4), it is easy to see the perfect reconstruction algorithm can be

$$d_k^{1,0}=d_k^1+Int(dc_k), k=\cdots,m,m-1,m-2\cdots, \qquad (5.5)$$

combined with (5.2).

As mentioned above, the Lifting scheme is a special condition of the correction method. Examples 3 through 5 can also be considered as the examples of this method. We next give an example of the Correction method which cannot be included in the group of Lifting scheme, and also which does not result in a closed form of compact support for biorthogonal filters.

Example 6: S+P transform, which is similar to using following analysis filters.

| n | -2 | -1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| $\tilde{h}_n$ | 0 | 0 | 1/2 | 1/2 | 0 | 0 |
| $\tilde{g}_n$ | -1/16 | -1/16 | 15/32 | -17/32 | 7/32 | -1/32 |

While the synthesis filters do not have compact support, the S+P transform can be implemented as follows:

(a) Decomposition

(1) Take the decomposition step of Example 1, that is, compute

$$d_k^{1,0}=c_{2k}^0-c_{2k+1}^0, k=-0,1,\cdots,M_1-1;$$

and

$$c_k^1=Int\left(\frac{d_k^{1,0}}{2}\right)+c_{2k+1}^0, k=0,\cdots,N_1-2,$$

$$c_{n_1-1}^1=\begin{cases} Int\left(\frac{d_{M_1-1}^{1,0}}{2}\right)+c_{N-1}^0, \\ c_{n-1}^0 \end{cases}$$

(2) Correction Step: Define $S_0 = -1$, $S_1 = 1$, $T = 1$ and

$$\sigma_{-1}=-\frac{1}{4}, \sigma_0=-\frac{1}{6}, \sigma_1=\frac{1}{6};$$

$$\tau_1=\frac{1}{4}.$$

and now compute

$$d_0^1 = d_0^{1,0} - Int\left(\frac{c_0^1 - c_1^1}{4}\right);$$

$$d_k^1 = d_k^{1,0} - Int\left(\frac{2c_{k-1}^1 + c_k^1 - 3c_{k+1}^1 - 2d_{k+1}^{1,0}}{8}\right), k=1,\cdots,M_1-2;$$

$$d_{M_1-1}^1 = d_{M_1-1}^{1,0} - Int\left(\frac{c_{M_1-2}^1 - c_{M_1-1}^1}{4}\right).$$

(b) Reconstruction

(1) Compute

$$d_{M_1-1}^{1,0} = d_{M_1-1}^1 + Int\left(\frac{c_{M_1-2}^1 - c_{M_1-1}^1}{4}\right)$$

$$d_k^{1,0} = d_k^1 + Int\left(\frac{2c_{k-1}^1 + c_k^1 - c_{k+1}^1 - 2d_{k+1}^{1,0}}{8}\right), k=M_1-2,\cdots,1;$$

$$d_0^{1,0} = d_0^1 + Int\left(\frac{c_0^1 - c_1^1}{4}\right).$$

(2) If N is an even number, compute

$$c_{2k+1}^0 = c_k^1 - Int\left(\frac{d_k^1}{2}\right), k=0,\cdots,N_1-1$$

or, if N is an odd number, we have

$$c_{2k+1}^0 = c_k^1 - Int\left(\frac{d_k^1}{2}\right), k=0,\cdots,N_1-2,$$

$$c_{N-1}^0 = c_{N_1}^1.$$

(3) Compute

$$c_{2k}^0 = d_k^1 + c_{2k+1}^0, k=0,\cdots,M_1-1.$$

Boundary Conditions

There are two issues dealing with boundary filtering if the Lifting scheme or the Correction method

is used to generate the integer wavelet transformations. The first is how to process the boundaries which occur in the start-up wavelet transformations. The second is how to deal with the boundaries in the deductive formula. If the boundaries in the start-up wavelet transform have already been established, then those in the upgrading formula are relatively easy to establish. For the *Lifting scheme*, the boundaries in both steps should be processed in the same way. While, for the Correction method, according to (5.3)-(5.4), one has more choices to process boundaries in the second step. Therefore, the process by which the boundaries in the start-up wavelet transformations are established is discussed. Assume compact supported biorthogonal wavelets.

Suppose the original signal is $\{C_n^0\}_{n=0}^{N}$. For creating integer biorthogonal wavelet transformations, use the following symmetric extension:

(1) If current biorthogonal filters have even length, the boundaries of the signal are extended as $C_{-k}^0 = C_{k-1}^0$, $k = 1, 2, \cdots$

(2) If the filters have odd length, the following extension is performed

$C_{-k}^0 = C_k^0$, $k = 1, 2, \cdots$

Examples 1 through 5 use the boundaries given above. In Example 6, the start up wavelet transform uses the above boundaries but in the upgrading step, another boundary filtering is used. In addition, for arbitrarily sized images or signals, one can use the same technique described in the above examples to deal with this condition.

As mentioned earlier, for many applications, lossless image compression is more important than lossy compression. The integer wavelet transforms described above provide the opportunity to compress without loss. It is also obvious that the integer wavelet algorithms

can be used wherever ordinary wavelets are used, especially in signal and image compression. However, for most computers, the integer wavelet transform is much faster than other wavelets and it uses much less memory.

Peak Signal to Noise Ratio (PSNR) Controlled Compression

Peak Signal to Noise Ratio (PSNR) is a widely used quality measurement. PSNR controlled compression allows users to choose their desired PSNR for the compressed image. In each of the compression methods set forth herein, a user can selectively set the PSNR and the desired compression ratio, as well as the initial quantization and threshold levels for each quadrant of wavelet coefficients, to obtain the desired image quality.

For example, the wavelet map of FIG. 3 shows a total of 10 regions (quadrants). Each of these ten quadrants can have two additional parameters associated with them. The parameters define the quantization and threshold values for that particular quadrant. Since there are three planes for color (only one for gray level) the maximum number of parameters that the user can control is 60 -- 10 for quantization and 10 for thresholding for each of the three color layers. In the case of a gray level image, there are only 20 parameters.

If a compression ratio, or a quality factor which indirectly defines a compression ratio, is specified, then the user wants the compression ratio to remain identical over the changes in the parameters. In order to accomplish this, two parameters are monitored: the compression ratio and PSNR (peak signal to noise ratio). The PSNR is defined as PSNR = 20 $\log_{10}$ (X/MSE), where the X is the average absolute value of the pixels in the compressed image and MSE is the mean squared error measured between the compressed and original image. Holding the compression ratio constant, the PSNR needs to

increase to improve image quality. The way to increase the PSNR is to reduce the MSE.

An iterative method can be used to adjust parameters to achieve the desired PSNR. The step are as follows:

(a) Pick an initial parameter setting $P_0$;
(b) Quantize the wavelet coefficients with $P_0$ and calculate the corresponding PSNR;
(c) If the PSNR is close to the desired one, stop and output the coded file; otherwise, get an adjusted vector $\Delta P_0$ and set $P_0 \leftarrow P_0 + \Delta P_0$, go to step (b).

Progressive Decomposition

Progressive decompression allows users to decode images at varying degrees of resolution, starting from the lowest resolution and progressing to the highest resolution. The advantage of this feature is that users can download small pieces of the coded file and view the image at lower resolution to determine if they want to download the whole image. Progressive decomposition can be used with any of the decompression methods previously disclosed herein. Progressive decomposition is accomplished according to the following steps:

(a) Input the lowest bandpass component $C^1$ of the coded file and reconstruct the lowest resolution image $I^0$;
(b) Display image $I^0$;
(c) If the user is not satisfied with the image quality or the resolution is big enough for stop; otherwise, go to step (d);
(d) Input the lowest three band-pass components $HD^1$, $VD^1$, and $DD^1$ successively in the current image file. Reconstruct the new image $I^1$ from $C^1$, $HD^1$, $VD^1$, and $DD^1$. Let $I^0 = I^1$; go to step (b).

Image Map Editor

The image map editor creates an image map over a compressed image file. This permits an image compressed according to one of the methods set forth herein to be easily integrated into a web page using an http link. A user selects one or several areas of compressed image, assigns one or more http links to the areas. The image map editor calculates the coordinates of the areas and outputs the HTML associate with the image. The user can add such information into program source code. Following is an example of such image map:

```
<EMBED SRC="cow.cod" type="image/cis-cod"
WIDTH="257" poly= "44, 45, 103, 78, 103, 86, 54,
86, 54, 78",
href="http://www.infinop.com"></EMBED>
```

Non-Uniform Image Compression

The present invention allows a user to perform non-uniform image compression. Essentially, non-uniform compression is accomplished by dividing an image into one or more rectangles, each representing a matrix of image pixels. Each rectangle can be compressed by any of the methods disclosed herein.

For instance, referring to the compression method of FIG. 8, integrating the non-uniform compression feature with the method allows a user to partition the image into several parts with different interests. The user can then compress these areas with different image and/or compression qualities. The parts can have any shape.

The non-uniform compression feature can be incorporated in to the method of FIG. 8 as follows. Steps 100-102 are performed. Then, the user creates bitmap matrices defining the partitioned areas. Each area is then wavelet transformed. Different quantizations are then applied to the different areas according to the transformed matrices obtained above.

Split and Merge Wavelet Algorithm
for Big Image Compression

This algorithm allows users to compress large images by partitioning them into smaller pieces. The key is to divide the original image into several smaller pieces and compress/decompress them separately by using overlap and de-overlap technique. With this technique, the individually compressed pieces are equivalent to compressed whole image. The user does not see any edge effects in the decompressed image, which normally occur with conventional split and merge methods.

Also, with this algorithm, users can selectively decompress the whole image or choose a specific part to decompress according to an image map created during the compression phase. The algorithm is preferably implemented as a software program executing on a general purpose computer.

There are two ways to compress an image by splitting it: automatically or interactively. The automatic approach is transparent to users since the algorithm will automatically split to the image according to the characteristics of the computer used to perform the compression. Using the automated method, the algorithm first detects the size of the source image and the memory size of the host computer. Next, the image is split into several pieces with a predetermined number of pixels overlapping according to the image size and computer's memory. Overlapping pixels are those that appear in more than one piece of the split image.

Each piece of image is compressed in order according to any of the methods disclosed herein from the image resource.

The split image is decompressed as follows. First, the headers of the compressed image pieces are read to determine their order and compression parameters, such as quantization thresholds and decomposition levels. Next, each piece of the image is decompressed and de-

overlapped. Merge all pieces together in the proper place for display.

Using the interactive method, a user can indicate how many blocks they want to divide the image into and how many pixels they want for overlap. To compress an image according to this approach, the size of the source image is first detected. Then, the user's choice for the number of blocks and number of overlapping pixels is entered. Next, the image is divided into the pieces according to the user's choice and the size of the image. Finally, the individual pieces are compressed according to one of the methods disclosed herein.

The interactively split image is decompressed as follows. First, the header of the coded image is read. Next, an image map is displayed for the user to look at what the image context is about. The user can then choose to display entire image or a specific piece of image. If user chooses to display a single piece of image, the algorithm finds the position of this coded piece and decompresses it. If the user instead chooses to display the entire image, the algorithm decompress each piece of image and de-overlaps it. All pieces are then merged together in the appropriate display location.

Example A, below, shows further technical details related to the present invention.

While specific embodiments of the present invention have been shown and described, it will be apparent to this skilled in the art that the disclosed invention may be modified in numerous ways and may assume many embodiments other than the preferred form specifically set out and described above. Accordingly, it is intended by the appended claims to cover all modifications of the invention which fall within the true spirit and scope of the invention.

EXAMPLE A

## 1.0 Quality Compression Optimization

The following sections deal with the improvement of the compressed image quality. As stated in the report introduction, we believe this is the last significant major problem to be addressed in the still imagery compression system.

## 1.1 Introduction

The issue is the mechanism which can be used to improve the quality of the image based upon the nature of the parameters used in the compression process. We think this is an important issue, and in our experimentation, we have found fairly striking results. For example, if we hold the compression ratio constant, we can compress an image and obtain on a subjective scale of C, where

- A is no observable defect,
- B is observable but not noticeable,
- C is quite noticeable, but not distracting from the image,
- D is very noticeable and detracts from the image,
- E is unacceptable.

If we now change the parameters by a hand optimization, we find we can change the subjective evaluation from a C to a B with the compression ratio left alone. We believe this is significant, in that it guarantees a compression system which is tailored for each image independently, rather than have each image compressed by the same set of parameters irrespective of the image content.

To review the process of compression, consider the following figure, Figure 49.

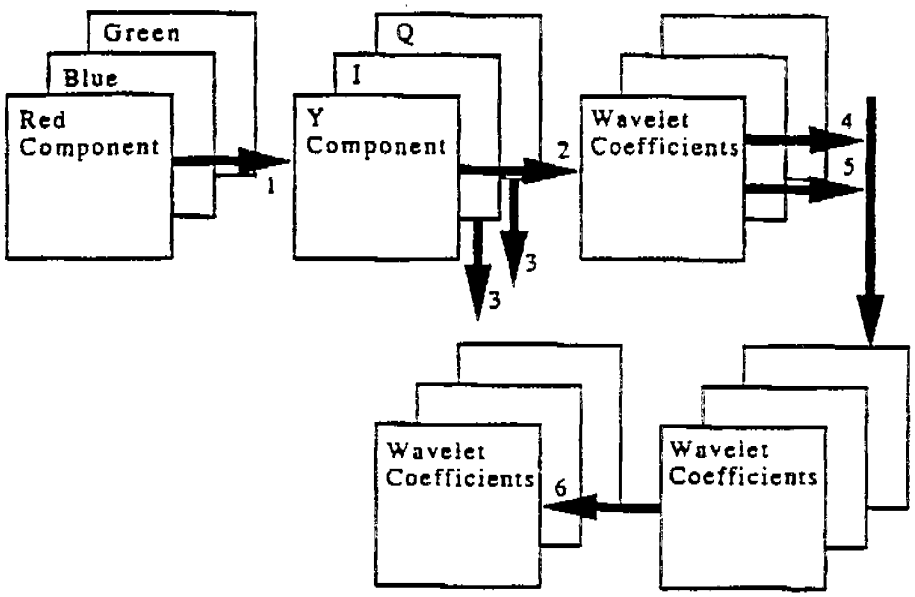



Figure 49
RGB Image in 24 Bit Color Depth Showing Transform to YIQ

Figure 49 shows the relationship between the RGB and YIQ color representation and the processes which take place in the algorithm as this data is transformed into the final lossy set of wavelet coefficients.. The processes which apply to the imagery are shown as heavy arrows with the process identifier shown as part of the arrow. If no process number is present, then the arrow is just a passive link between processes and data. The important point to note is that in color imagery, we deal with three images (one luminescence, and two color planes). The Y component is critical, while the IQ components are less sensitive to error introduced by the compression system.

The processes shown in Figure 49 are as follows:

- (1) Transform the RGB format into YIQ format, using long integers. This format is only an approximation of the YIQ format.

- (2) Transform the YIQ planes into a wavelet decomposition using our own integer wavelet transform. This produces the result shown in Figure 50, but for each plane.

- (3) We have shown an alternative process which is an optional down sampling for the IQ color planes. This down sampling may

be done once or twice to produce two image planes either one fourth or one sixteenth the size of the original plane. If this process is to be done, it will be accomplished prior to the wavelet transform of process (2).

- (4) Here the first step in the loss occurs. The wavelet coefficients are now quantized to a number of levels depending upon which quadrant is being processed, and the desired compression or quality factor.

- (5) Simultaneously with step (4), the wavelet coefficients are being matched against threshold values, and if the values are less than the established threshold values specified, then the resultant value is set to zero.

- (6) The last step in the process is to entropy compress the resultant coefficients using either Arithmetic, Run Length, or Huffman, of Huffman and Run Length combined. The key issue is the amount of compression desired against the invested time required to get that level of compression.

The issue now, irrespective of the down sampling or not of the IQ components, is the fact that we process the three planes into five levels in a decomposition as shown in Figure 50. From this figure, one can see a total of 16 regions (quadrants) which are defined by the numbers 1 through 16. Each of these sixteen quadrants have two additional parameters associated with them. The parameters define the quantization and threshold values for that particular quadrant. Since there are three planes for color (only one for gray level) the maximum number of parameters that the user can control is 96 -- 16 for quantization and 16 for thresholding for each of the three color layers. In the case of a gray level image, there are only parameters for one layer.

Our experience has shown the parameters for an image are very sensitive in some cases, and not in other cases. In order to measure this sensitivity, we generated the variance of the wavelet coefficients in the 16 quadrants. Table 1 provides these values for each of the three planes.

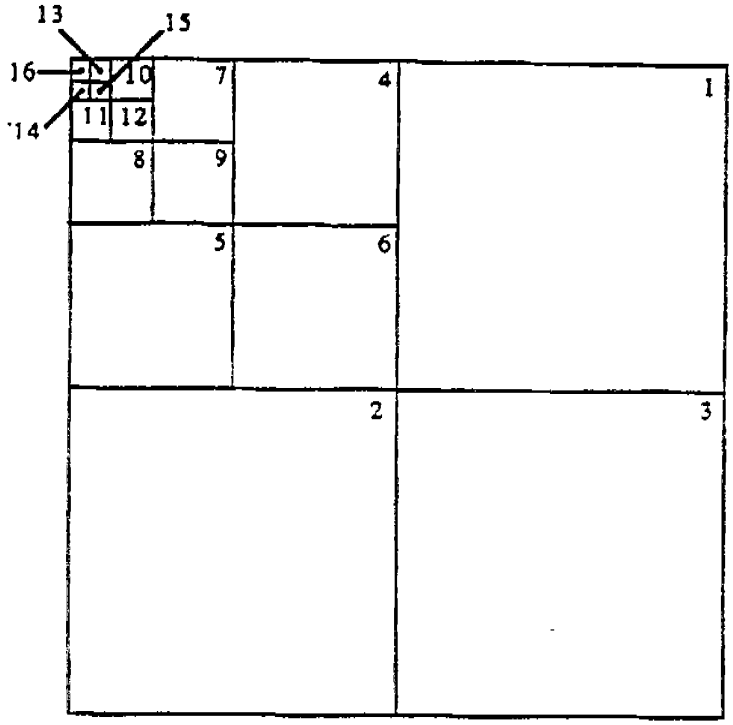



Figure 50
Decomposition of a Plane into Five Levels

| Image | Lenna Wavelet Variance | | | Tulips Wavelet Variance | | |
|---|---|---|---|---|---|---|
| Quadrant | Y | I | Q | Y | I | Q |
| 1 | 13 | 4.7 | 1.4 | 164.2 | 27.3 | 6.6 |
| 2 | 6 | 3.2 | 1.1 | 134.6 | 24.9 | 5.0 |
| 3 | 1 | 1.3 | 0.6 | 28.9 | 4.6 | 1.1 |
| 4 | 56 | 5.8 | 1.7 | 276.1 | 57.7 | 12.9 |
| 5 | 23 | 4.5 | 1.1 | 274.3 | 67.3 | 12.4 |
| 6 | 10 | 2.8 | 0.8 | 124.8 | 22.7 | 5.0 |
| 7 | 128 | 11.4 | 2.6 | 225.7 | 73.2 | 13.7 |
| 8 | 55 | 7.4 | 1.1 | 298.1 | 107.3 | 16.2 |
| 9 | 37 | 3.3 | 1.0 | 114.2 | 35.5 | 6.8 |
| 10 | 253 | 27.5 | 4.1 | 174.8 | 73.9 | 10.7 |
| 11 | 75 | 11.8 | 1.5 | 285.1 | 128.2 | 14.9 |
| 12 | 64 | 6.5 | 1.1 | 107.9 | 47.8 | 6.7 |
| 13 | 499 | 41.7 | 10.1 | 135.4 | 75.6 | 8.7 |
| 14 | 138 | 11.2 | 2.5 | 245.0 | 144.0 | 11.2 |
| 15 | 156 | 12.2 | 2.0 | 89.3 | 49.1 | 5.1 |
| 16 | 1,4161 | 1,281.8 | 295.6 | 7,408.6 | 690.4 | 77.9 |

Table 1
Wavelet Coefficient Variance by Numbered Quadrant for Two Images

The clear information to be gained by the numbers in Table 1 is the images are quite different, and the quantization or threshold levels set for all images will only be an approximate solution at best. The optimum solution would be to have the quantization and threshold values set according to the variance values. Such settings could be found in a table with various ranges, and for each such range, the parameters of interest could be defined. This would give a more optimal solution, but still not the optimal solution. In order to get optimality, one would need to search over the variable space using the near optimal settings to find the actual best values for the parameters.

As the values in Table 1 grow, the implication is a finer mesh size for quantization. That is, the number of bits one wishes to allocate to the output could be varied by quadrant. Those quadrants with large variances will utilize more bits, while those with low variants will utilize fewer bits. In this way, the number of bits resulting from quantization will remain the same, but their allocation will differ depending upon the nature of the image.

## 1.2 Approach

The solution to the problem posed in the previous section is to determine how the ninety-six parameters interact with one another. The problem is a bit more sophisticated than just measuring parameters, however. The issue is with each parameter change, the compression ratio will change. If a compression ratio, or a quality factor which indirectly defines a compression ratio, is specified, then the user wants the compression ratio to remain identical over the changes in the parameters. In order to accomplish this, there are two parameters which we must monitor: PSNR (peak signal to noise ratio which is defined to be PSNR = 20 $\log_{10}$ (X/MSE) where the X is the average absolute value of the pixels in the after image and MSE is the mean squared error measured between the before and after image) and the compression ratio. The compression ratio must be held constant, and the PSNR needs to increase, and the way to increase the PSNR is to reduce the MSE.

The difficulty with this system as described is in many cases, small changes in the parameters introduce significant changes in the MSE. Also, we believe, the parameters are not independent. We have also seen images where the parameters can be changed in one way, then

altered, and the results are exactly the same. This indicates the optimal value is not a single point, but rather something like a plane with little slope.

### 1.3 Status

We have established the rules under which the optimal solution will need to exist, and are at the moment writing software to measure the variance within the Lightning Strike environment. Once this is done, we will being examining many images to see how close we can determine the optimal parameters for a defined set of variances.

# CIS-2 Image Compression Algorithm

HONGYANG CHAO

## Part I: Brief Review of LSIC 3.0

### 1. Introduction

CIS-2 (temporary name), which has been being used in **Lightning Strike 3.0** image compression software, is a wavelet based image compression algorithm. CIS-2 has following inventions:

- Integer reversible wavelet algorithm with Property of Precision Preservation;
- Subband oriented quantization and related entropy coding;
- Wavelet lossless compression for color and gray images;
- Progressive transmission algorithm for color bit compression;
- Progressive transmission and decompression algorithms;
- Non-uniform image compression algorithm;
- Quality based wavelet coefficient quantization tables;
- Attached optional post-processing filters;
- Image map editor;
- Optional peak signal noise ratio controlled compression;
- Special split and merge wavelet compression algorithm for very big image compression without any boundary effects ;
- Image dependent parameter optimization .

### 2. Main steps of the algorithm

In Lsic 3.0, three kinds of different image compression methods are included:

Method 1: Quality controlled wavelet based compression

Method 2: Color bit depth compression

Method 3: Wavelet lossless compression

Section 2.1-2.3 will give brief description of above method. The details will discuss later.

#### 2.1. Main step of method 1

Figure 1 and Figure 2 give the flow charts of the image compression and decompression of method1 respectively. Every step in both compression and decompression has lot of details, which will be described later.

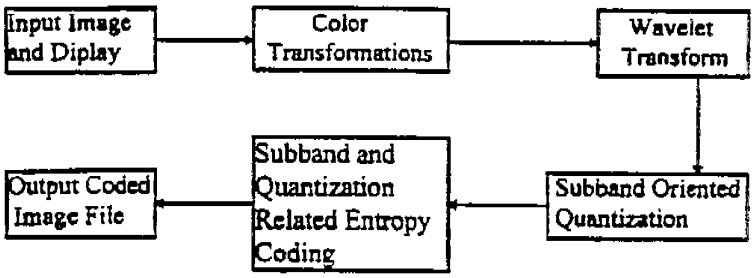



**Figure 1**: Compression flow chart for Method 1

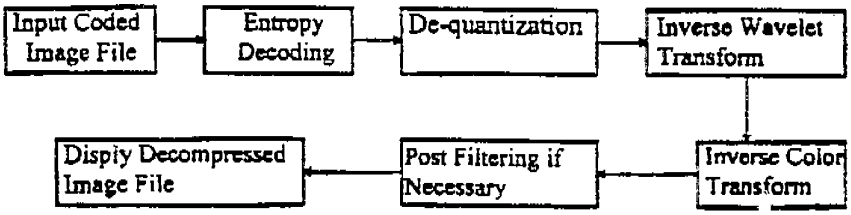



**Figure 2**: Decompression flow chart for Method 1

**2.2. Main step of method 2**

This method based on using less number of colors to approximately represent original images. Following figure gives the main step of the algorithm for the compression and decompression.

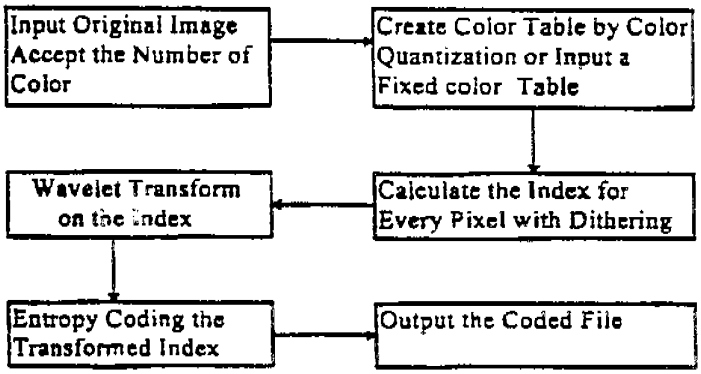



**Figure 3**: Compression flow chart for Method 2

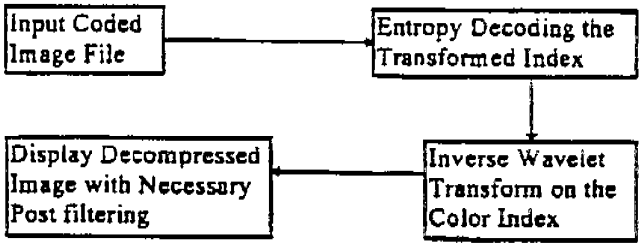



**Figure 4:** Decompression flow chart for Method 2

**2.3. Main steps of method 3**

The main steps of method 3 is almost as same as method 1. However, at the every step we use different methods. Following are its compression and decompression flow charts:

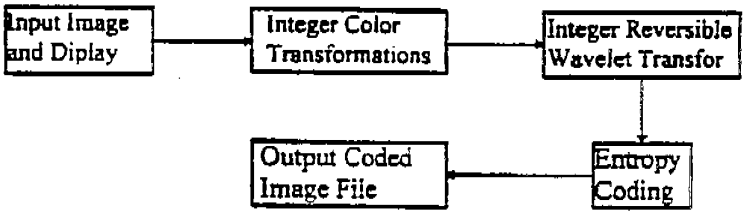



**Figure 5:** Compression flow chart for Method 3

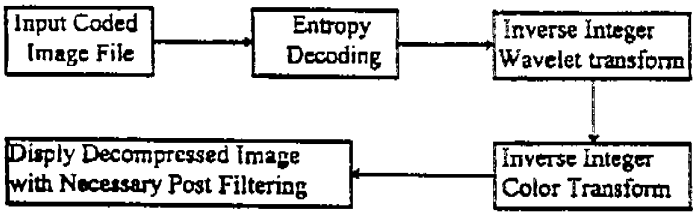



**Figure 6:** Decompression flow chart for Method 3

## 3. Brief Description for Other Features

Most Features (or inventions) are included in above three compression methods. Following is the brief introduction for some inventions list above.

**3.1. Progressive decompression**

This algorithm allows users to decode images from the lowest resolution to highest resolution. The advantage of this feature is that users can download small piece of the coded file and view the image at lower resolution to determine if they want to download the whole image.

Steps:

(a) Input the lowest pass component $LL^0$ of the coded file and reconstruct the lowest resolution image $I^0$;

(b) Display image $I^0$. If the user doesn't like it or the resolution is big enough for , stop; otherwise, go to next step;

(c) Input the lowest three band-pass components $HL^0$, $LH^0$ and $HH^0$ successively in the current coded file. Reconstruct the new image $I^1$ from $LL^0$, $HL^0$, $LH^0$ and $HH^0$. Let $I^0 = I^1$, go to step (b).

**3.2. Non-uniform image compression**

The algorithm allows users to divide the image into several parts with different interests and compress these areas with different qualities. The areas can have any shape. This algorithm is only available for method 1.

Original image goes though all of the procedure except quantization part, which follows the steps below:

(a) Creating the bitmap matrices related to the areas chosen by the user;

(b) Wavelet transform to every bitmap matrix;

(c) Different quantization in different areas according to the transformed matrices obtained above step.

**3.3. Peak Signal Noise Ratio (PSNR) controlled compression**

Peak Signal Noise Ratio (PSNR) is an image quality measurement used by most professional people. PSNR controlled compression allows users to choose their desired PSNR for the compressed image.

The related algorithm is an iterated system:

(a) Picking an initial parameter setting $P_0$;

(b) Quantize the wavelet coefficients with $P_0$ and calculate the corresponding PSNR;

(c) If the PSNR is close to desired one, stop and output the coded file; otherwise, get an adjusted vector $\Delta P_0$ and set $P_0 \leftarrow P_0 + \Delta P_0$, go to step (b);

**3.4. Attached optional post-processing filters**

Users can choose any number of following processing filters at their compressing time. The desired results can stored in the coded image file, and, anyone who decompress the coded file will see the same result immediately.

- Sharpening images
- Smoothing images
- Improving the visual quality
- Brightening the images

**3.5. Image map editor**

Image Map Editor create a image map over Lsic3.0 compressed image file. User selects one or several areas of compressed image, assigns the http links to the areas, then, Image Map Editor calculates the coordinates of the areas and outputs a HTML associate with the image. User can add such information into his/her source code.

Following is an example of such image map:

<EMBED SRC="cow.cod" type = "image/cis-cod" WIDTH= "257" poly= "44, 45, 103, 78, 103, 86, 54, 86, 54, 78", href= "http://www.infinop.com"></EMBED>

**3.6. Split and merge wavelet algorithm for very big image compression**

This algorithm allows users to compress very big image by an ordinary machine. The key is to divide the original image into several smaller pieces and compress/decompress them separately by using overlap and dis-overlap technique. With this technique, the compression/decompression piece by piece is equal to compress/decompress the whole image together, which means users won't see any edge effect at decompressed image which appears at general split method.

Also, with this algorithm, users can either decompress the whole image or choose the specific part to decompress according to a image map we create for the division .

**3.7. Image dependent optimized parameter setting**

This algorithm allows user to get the best (or almost best ) image quality at the desired compression ratio by choosing image related parameter setting.

**3.8. Integer reversible wavelet algorithm with PPP property**

See attached unpublished paper titled as "An Approach to Fast Integer Reversible Wavelet Transformations for Image Compression"

**3.9. Integer color transformation**

This algorithm is a integer reversible transform which has been used in lossless color image compression (Method 3) for Lsic 3.0.

The algorithm transform RGB color components to a new set of color components Y-Nb-Nr:

Forward transform RGB to Y-Nb-Nr:

$$Y = G + \mathrm{Int}\left(\frac{R+B}{2}\right),$$
$$Nb = B - \mathrm{Int}\left(\frac{Y}{2}\right),$$
$$Nr = R - \mathrm{Int}\left(\frac{Y}{2}\right).$$

Inverse transform Y-Nb-Nr to RGB:

$$R = Nr + \mathrm{Int}\left(\frac{Y}{2}\right),$$
$$B = Nb + \mathrm{Int}\left(\frac{Y}{2}\right),$$
$$G = Y - \mathrm{Int}\left(\frac{R+B}{2}\right).$$

**3.10. Subband related Quantization and entropy coding**

This entropy coding method is just designed for wavelet based image compression. The main idea is to take the advantage of different quantization at different subbands and encode each band according to its content. This method reduce the coding cost greatly.

# CIS-1 Image Compression Algorithm

HONGYANG CHAO*
Computer and Information Science Inc.

## 1. Introduction

**CIS-1**, which has been being used in **Lightning Strike** image compression software, is a wavelet based image compression algorithm. CIS-1 has following advantages:

- o Reach almost optimal compression ratio;
- o Keep the major characteristics as more as possible. In other words, it reduce insignificant components gradually according to human visual system, so that people can still accept the image quality at the extremely high compression ratio;
- o Fast.

## 2. Main steps of the algorithm

Figure 1 and Figure 2 give the flow charts of the image compression and decompression respectively. Every st p in both compression and decompression has lot of details, which will be described later.

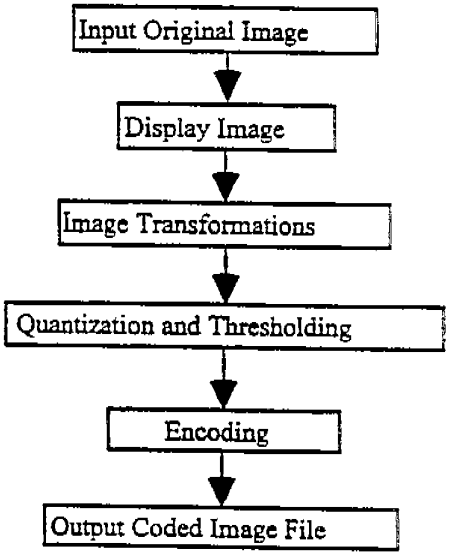



**Figure 1**: Image compression

*

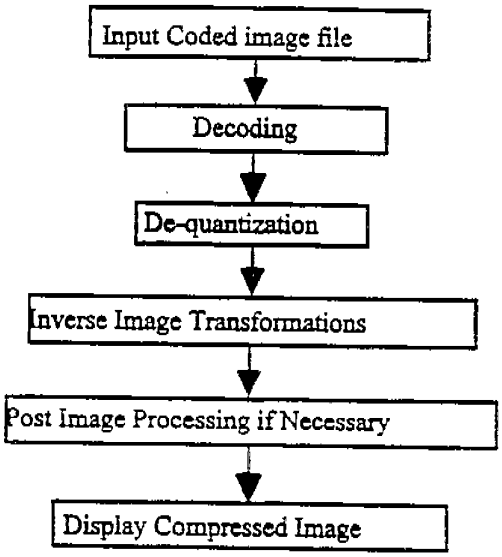



**Figure 2**: Image decompression

### 3. Image Compression

For the compression, we only have to describe three parts: image transformations, quantization and thresholding, and entropy coding.

#### 3.1. Image Transformations

The image transformations involved in this algorithm include color transform (for color images) and wavelet decomposition ( for both gray level images and color images).

(a) Color transform

In general, input color images are based on RGB color model, such as TIFF or BMP images. In order to get high compression ratio, it is better to change RGB color model to other color models, such as YIQ or YUV models.

RGB to YIQ:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

RGB to YUV:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.148 & -0.289 & 0.439 \\ 0.615 & -0.515 & -0.1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

(b) Wavelet decomposition

The purpose of wavelet transform is to represent the original image by different basis to achieve the objective of decorrelation. There are a lot of wavelets which can be used in this step. In CIS-1, we use a wavelet which results in the following algorithm: Suppose $C^0 = \left[c_{jk}^0\right]_{M\times N}$ $(j = 0,\cdots,M-1;\quad k = 0,\cdots,N-1)$ is original image, where $M$ and $N$ are integers which have the common factor $2^L$ ($L$ is a positive integer). After one-level wavelet decomposition, we will get four parts as shown in figure 3.

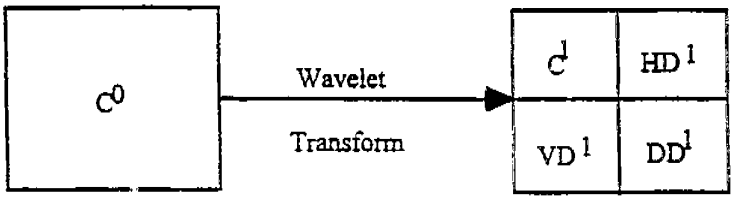



**Figure 3.** Wavelet image decomposition

We call $C^1 = \left[c_{jk}^1\right]$ $(j = 0,\cdots,\frac{M}{2}-1;\ k = 0,\cdots,\frac{N}{2}-1)$ the blurred image of $C^0$, $HD^1$ the horizontal high frequency part of $C^0$, $VD^1$ the vertical high frequency part, and $DD^1$ the diagonal ones. Setting $C^0 = C^1$, we can repeat the same procedure $L$ times or until the size of the new blurred image $C^1$ is small enough. Therefore, we only have to give the algorithm for one level decomposition:

(1) Let $\overline{C}^0 = rC^0$, $r>0$ is a factor which can be changed for different needs.

(2) Transform for image columns:

- For $k = 0,\cdots,N-1$, calculate

$$\begin{cases} \tilde{d}_{0k}^1 = \dfrac{\overline{c}_{1k}^0 - \overline{c}_{0k}^0}{2}, \\ \tilde{d}_{jk}^1 = \dfrac{1}{4}(\overline{c}_{2j-1,k}^0 - 2\overline{c}_{2j,k}^0 + \overline{c}_{2j+1,k}^0), \quad j = 1,\cdots,\frac{M}{2}-1. \end{cases} \tag{3.1.1}$$

- For $k = 0,\cdots,N-1$, calculate

$$\begin{cases} \tilde{c}^1_{0,k} = \tilde{c}^0_{1,k} - \dfrac{\tilde{d}^1_{0,k} + \tilde{d}^1_{1,k}}{2}, \\ \tilde{c}^1_{j,k} = \tilde{c}^0_{2j+1,k} - \dfrac{\tilde{d}^1_{j,k} + \tilde{d}^1_{j+1,k}}{2}, \quad j = 1, \cdots, \frac{M}{2} - 2, \\ \tilde{c}^1_{\frac{M}{2}-1,k} = \tilde{c}^0_{N-1,k} - \tilde{d}^1_{\frac{M}{2}-1,k}. \end{cases} \tag{3.1.2}$$

(3) Transform for rows:

o For $j = 0, \cdots, \frac{M}{2} - 1$, computing

$$\begin{cases} hd^1_{j,0} = \dfrac{\tilde{c}^1_{j1} - \tilde{c}^1_{j0}}{2}, \\ hd^1_{j,k} = \dfrac{1}{4}(\tilde{c}^1_{j,2k-1} - 2\tilde{c}^1_{j,2k} + \tilde{c}^1_{j,2k+1}), \quad k = 1, \cdots \frac{N}{2} - 1. \end{cases} \tag{3.1.3}$$

and

$$\begin{cases} c^1_{j0} = \tilde{c}^1_{j,1} - \dfrac{hd^1_{j0} + hd^1_{j1}}{2}, \\ c^1_{jk} = \tilde{c}^1_{j,2k+1} - \dfrac{hd^1_{j,k} + hd^1_{j,k+1}}{2}, \quad k = 1, \cdots, \frac{M}{2} - 2, \\ c^1_{j,\frac{N}{2}-1} = \tilde{c}^1_{j,N-1} - hd^1_{j,\frac{N}{2}-1}. \end{cases} \tag{3.1.4}$$

o For $j = 0, \cdots, \frac{M}{2} - 1$, computing

$$\begin{cases} dd^1_{j,0} = \dfrac{\tilde{d}^1_{j,1} - \tilde{d}^1_{j,0}}{2}, \\ dd^1_{j,k} = \dfrac{1}{4}(\tilde{d}^1_{j,2k-1} - 2\tilde{d}^1_{j,2k} + \tilde{d}^1_{j,2k+1}), \quad k = 1, \cdots, \frac{M}{2} - 1. \end{cases} \tag{3.1.5}$$

and

$$\begin{cases} vd^1_{j0} = \tilde{d}^1_{j,1} - \dfrac{dd^1_{j0} + dd^1_{j1}}{2}, \\ vd^1_{jk} = \tilde{d}^1_{j,2k+1} - \dfrac{dd^1_{j,k} - dd^1_{j,k+1}}{2}, \quad k = 1, \cdots, \frac{M}{2} - 2, \\ vd^1_{j,\frac{M}{2}-1} = \tilde{d}^1_{N-1,k} - dd^1_{j,\frac{M}{2}-1}. \end{cases} \tag{3.1.6}$$

(4) $\mathbf{C}^1 = \left[c^1_{j,k}\right]$, $\mathbf{HD}^1 = \left[hd^1_{j,k}\right]$, $\mathbf{VD}^1 = \left[vd^1_{j,k}\right]$ and $\mathbf{DD}^1 = \left[dd^1_{j,k}\right]$, $j = 0, \cdots, \frac{M}{2} - 1$; $k = 0, \cdots, \frac{M}{2} - 1$.

*Remark*: If it is necessary, we also can use matrix multiply

Wavelet Coefficient Image of $l$ levels = $\mathbf{W}_l \mathbf{C}^0 \mathbf{W}_l^T$.

Here, $W_l$ is the transform matrix for $l$ level wavelet decomposition.

**3.2. Thresholding and Quantization**

Both thresholding and Quantization allow us to reduce accuracy with which the wavelet coefficients are represented when converting the wavelet decomposition to an integer representation. This can be very important in image compression, as it tends to make many coefficients zeros--especially those for high spatial frequencies.

After $L$ level, for example $L$=3, wavelet decomposition, we get the wavelet coefficients of the original image as plotted in Figure 4:

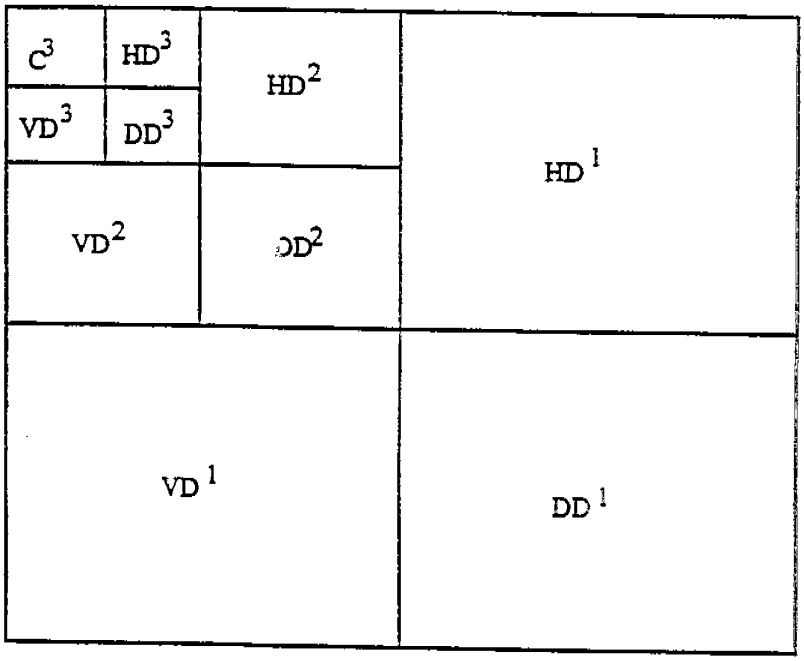



**Figure 4.** L=3 wavelet coefficients distribution

(a) Thresholding

In algorithm CIS-1, we use multilevel uniform thresholding method: Let

$$\mathbf{T} = (t_1, \cdots, t_L, t_{L+1})$$

be the chosen thresholds, where $t_l$ is the threshold for $l$ th $(l = 1, \cdots, L)$ level and $t_{L+1}$ is a threshold for blurred image $\mathbf{C}^L$. Thresholding is to set every entry in the blocks $\mathbf{C}^L$,

$\mathbf{HD}^l$, $\mathbf{VD}^l$ and $\mathbf{DD}^l$ $(l=1,\cdots,L)$ to be zeros if its absolute value is not greater than the corresponding threshold.

*Remark.* For color image, we can have three threshold vectors which correspond three different color bands, such as Y, I and Q.

(b) Quantization

Quantization is to scale the wavelet coefficients and truncate them to integer values. In CIS-1, we use the quantization table shown in Table 1 to implement it.

| | | | | |
|---|---|---|---|---|
| $q^1_{HD}$ | $q^2_{HD}$ | $\bullet\bullet\bullet$ | $q^L_{HD}$ | |
| $q^1_{VD}$ | $q^2_{VD}$ | $\bullet\bullet\bullet$ | $q^L_{VD}$ | $q_C^{L+1}$ |
| $q^1_{DD}$ | $q^2_{DD}$ | $\bullet\bullet\bullet$ | $q^L_{DD}$ | |

**Table 1.** Quantization table

Here, the entries $q^l_{HD}$ are quantization factors for blocks $\mathbf{HD}^l$ $(l=1,\cdots,L)$, $q^l_{VD}$ and $q^l_{DD}$ for blocks $\mathbf{VD}^l$ and $\mathbf{DD}^l$ $(l=1,\cdots,L)$ respectively, and the factor $q_C^{L+1}$ is for the most blurred image $\mathbf{C}^L$. All the factors are integers between 0 and 255. The quantization scheme for the block $\mathbf{HD}^l$ $(l=1,\cdots,L)$ is

$$\overline{hd}^l_{j,k} = round\left(\frac{hd^l_{j,k} \bullet q^l_{HD}}{\max^l_{HD}}\right),\ j=0,\cdots,\tfrac{M}{2^l}-1;\quad k=0,\cdots,\tfrac{N}{2^l}-1. \quad (3.2.1)$$

Here, $\overline{hd}^l_{j,k}$ $(j=0,\cdots,\frac{M}{2^l}-1;\quad k=0,\cdots,\frac{N}{2^l}-1)$ are quantized wavelet coefficients in block $\mathbf{HD}^l$ $(l=1,\cdots,L)$,

$$\max^l_{HD} = \max_{\substack{0\le j\le(M/2^l-1)\\ 0\le k\le(N/2^l-1)}}\left(\left|hd^l_{j,k}\right|\right),$$

and the function *round* $(x)$ gives the nearest integer of $x$. The scheme of quantization for other blocks are the similar to (3.2.1).

*Remark.* For color image, as the same as thresholding, we can have three separate quantization tables for different color bands.

### 3.3. Entropy Coding

Here, the encoding means the lossless compression for the wavelet coefficients. It is divided into two parts: Coefficient arrangement and entropy coding (Huffman or arithmetic).

## 4. Decompression

### 4.1 Decoding

Decoding, just as encoding, can be divided into two parts: Entropy decoding (Huffman or arithmetic), and coefficient rearranging.

### 4.2 Dequantization

After Decoding, we get quantized wavelet coefficients in $3*L+1$ Blocks. Dequantizing uses the same quantization table as quantizing, and the scheme as follow: for $l = 1,\cdots,L$

$$hd^l_{j,k} = \frac{\overline{hd}^l_{j,k} * \max^l_{HD}}{q^l_{HD}},\ j = 0,\cdots,\frac{M}{2^l}-1;\quad k = 0,\cdots,\frac{N}{2^l}-1. \qquad (4.2.1)$$

(4.2.1) allows us to get the approximate coefficients for the blocks $\mathbf{HD}^l$ $(l = 1,\cdots,L)$, which is shown in Figure 4 . The dequantizing scheme for other blocks are similar to (4.1.2).

### 4.3 Inverse Image Transformations

(a) Wavelet reconstruction

We are going to describe the algorithm for one-level reconstruction which is plotted in Figure 5.

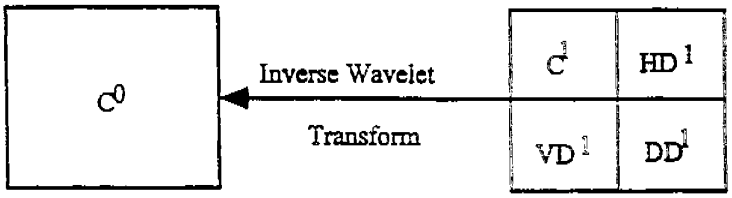



**Figure 5.** One-level wavelet reconstruction

(1) Inverse transform for rows:

- For $j = 0,\cdots,\frac{M}{2}-1$, calculate

$$\begin{cases} \tilde{d}^1_{j,1} = vd^1_{j,0} + \dfrac{dd^1_{j,0} + dd^1_{j,1}}{2} \\ \tilde{d}^1_{j,2k+1} = vd^1_{j,k} + \dfrac{dd^1_{j,k} - dd^1_{j,k+1}}{2}, \quad k = 1,\cdots,\frac{N}{2}-2, \\ \tilde{d}^1_{N-1,k} = vd^1_{j,\frac{N}{2}-1} + dd^1_{j,\frac{N}{2}-1}. \end{cases} \qquad (4.3.1)$$

and

$$\begin{cases} \tilde{d}^1_{j,0} = \tilde{d}^1_{j,1} - 2dd^1_{j,0}, \\ \tilde{d}^1_{j,2k} = \dfrac{(\tilde{d}^1_{j,2k-1} + \tilde{d}^1_{j,2k+1})}{2} - 2dd^1_{j,k}, \quad k = 1, \cdots, \frac{N}{2} - 1. \end{cases} \tag{4.3.2}$$

o For $j = 0, \cdots, \frac{M}{2} - 1$, calculate

$$\begin{cases} \tilde{c}^1_{j,1} = c^1_{j0} + \dfrac{hd^1_{j0} + hd^1_{j1}}{2}, \\ \tilde{c}^1_{j,2k+1} = c^1_{jk} + \dfrac{hd^1_{j,k} + hd^1_{j,k+1}}{2}, \quad k = 1, \cdots, \frac{N}{2} - 2, \\ \tilde{c}^1_{j,N-1} = c^1_{j,\frac{N}{2}-1} + hd^1_{j,\frac{N}{2}-1}. \end{cases} \tag{4.3.3}$$

and

$$\begin{cases} \tilde{c}^1_{j0} = \tilde{c}^1_{j1} - 2hd^1_{j,0}, \\ \tilde{c}^1_{j,2k} = \dfrac{1}{2}(\tilde{c}^1_{j,2k-1} + \tilde{c}^1_{j,2k+1}) - 2hd^1_{jk}, \quad k = 1, \cdots \frac{N}{2} - 1. \end{cases} \tag{4.3.4}$$

(2) Inverse transform for column:

o For $k = 0, \cdots, N - 1$, calculate

$$\begin{cases} \tilde{c}^0_{1,k} = \tilde{c}^1_{0k} + \dfrac{\tilde{d}^1_{0,k} + \tilde{d}^1_{1,k}}{2}, \\ \tilde{c}^0_{2j+1,k} = \tilde{c}^1_{jk} + \dfrac{\tilde{d}^1_{jk} + \tilde{d}^1_{j+1,k}}{2}, \quad j = 1, \cdots, \frac{M}{2} - 2, \\ \tilde{c}^0_{N-1,k} = \tilde{c}^1_{\frac{N}{2}-1,k} + \tilde{d}^1_{\frac{N}{2}-1,k}. \end{cases} \tag{4.3.5}$$

and

$$\begin{cases} \tilde{c}^0_{0,k} = \tilde{c}^0_{1k} - 2\tilde{d}^1_{0k}, \\ \tilde{c}^0_{2j,k} = \dfrac{1}{2}(\tilde{c}^0_{2j-1,k} + \tilde{c}^0_{2j+1,k}) - 2\tilde{d}^1_{jk}, \quad j = 1, \cdots, \frac{M}{2} - 1. \end{cases} \tag{4.3.6}$$

(3) $c^0_{j,k} = \tilde{c}^0_{j,k} / r, \quad j = 0, \cdots, M-1; \quad k = 0, \cdots, N-1.$ $C^0 = \left[c^0_{j,k}\right]_{M \times \frac{N}{2}}$.

(b) Inverse color transform

For color image, we have to do inverse color transform

o YIQ to RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}.$$

$$\begin{cases} \tilde{d}^1_{j,0} = \tilde{d}^1_{j,1} - 2dd^1_{j,0}, \\ \tilde{d}^1_{j,2k} = \dfrac{(\tilde{d}^1_{j,2k-1} + \tilde{d}^1_{j,2k+1})}{2} - 2dd^1_{j,k}, \quad k = 1, \cdots, \frac{N}{2} - 1. \end{cases} \tag{4.3.2}$$

o For $j = 0, \cdots, \frac{M}{2} - 1$, calculate

$$\begin{cases} \tilde{c}^1_{j,1} = c^1_{j0} + \dfrac{hd^1_{j0} + hd^1_{j1}}{2}, \\ \tilde{c}^1_{j,2k+1} = c^1_{jk} + \dfrac{hd^1_{j,k} + hd^1_{j,k+1}}{2}, \quad k = 1, \cdots, \frac{N}{2} - 2, \\ \tilde{c}^1_{j,N-1} = c^1_{j,\frac{N}{2}-1} + hd^1_{j,\frac{N}{2}-1}. \end{cases} \tag{4.3.3}$$

and

$$\begin{cases} \tilde{c}^1_{j0} = \tilde{c}^1_{j1} - 2hd^1_{j,0}, \\ \tilde{c}^1_{j,2k} = \dfrac{1}{2}(\tilde{c}^1_{j,2k-1} + \tilde{c}^1_{j,2k+1}) - 2hd^1_{jk}, \quad k = 1, \cdots \frac{N}{2} - 1. \end{cases} \tag{4.3.4}$$

(2) Inverse transform for column:

o For $k = 0, \cdots, N-1$, calculate

$$\begin{cases} \tilde{c}^0_{1,k} = \tilde{c}^1_{0k} + \dfrac{\tilde{d}^1_{0,k} + \tilde{d}^1_{1,k}}{2}, \\ \tilde{c}^0_{2j+1,k} = \tilde{c}^1_{jk} + \dfrac{\tilde{d}^1_{jk} + \tilde{d}^1_{j+1,k}}{2}, \quad j = 1, \cdots, \frac{M}{2} - 2, \\ \tilde{c}^0_{N-1,k} = \tilde{c}^1_{\frac{M}{2}-1,k} + \tilde{d}^1_{\frac{M}{2}-1,k}. \end{cases} \tag{4.3.5}$$

and

$$\begin{cases} \tilde{c}^0_{0,k} = \tilde{c}^0_{1k} - 2\tilde{d}^1_{0k}, \\ \tilde{c}^0_{2j,k} = \dfrac{1}{2}(\tilde{c}^0_{2j-1,k} + \tilde{c}^0_{2j+1,k}) - 2\tilde{d}^1_{jk}, \quad j = 1, \cdots, \frac{M}{2} - 1. \end{cases} \tag{4.3.6}$$

(3) $c^0_{j,k} = \tilde{c}^0_{j,k} / r, \quad j = 0, \cdots, M-1; \quad k = 0, \cdots, N-1.$ $\mathbf{C}^0 = \left[c^0_{j,k}\right]_{M \times N}$.

(b) Inverse color transform

For color image, we have to do inverse color transform

o YIQ to RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}.$$

o YUV to RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.000 & 1.140 \\ 1.000 & -0.395 & -0.581 \\ 1.000 & 2.032 & 0.000 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix}$$

**4.4 Necessary Post Image Processing**

(a) Color Quatization

# An Approach to Fast Integer Reversible Wavelet Transforms for Image Compression*

Hongyang Chao**
Computer and Information Science Inc.
3401 E. University, Suite 104, Denton, TX 76208
AND
Dept. of Computer Science, Zhongshan Univ.
Guangzhou 510275, P.R. China

Paul Fisher
Computer and Information Science Inc.
3401 E. University, Suite 104, Denton, TX 76208
AND
Dept. of Computer Science, Univ. of North Texas
Denton, TX 76205

## Abstract

In this paper, we propose a general method for creating integer wavelet transforms which can be used in both lossless (reversible) and lossy compression of signals and images with arbitrary size. This method allows us to get a series of transformations which are very close to the corresponding biorthogonal wavelet transforms or some non-orthogonal wavelet transforms, but can be calculated with only integer addition and bit-shift operations. In addition, the integer wavelet transforms created in this paper possess a property of precision preservation (PPP). This property is very useful for conserving memory in both compression and decompression, and speed up the whole procedure in some applications. The motivation of this paper comes from the lifting scheme [1] and S+P transform [3].

* This work has been partially supported by the US Navy under SBIR Contract N00039-94-C-0013.

** The author is partially supported by the National Science Foundation of P. R. China and the Science Foundation of Zhongshan University.

## 1. Introduction

The wavelet transform has proven to be one of the most powerful tools in the field of image compression. Theoretically, the wavelet transformation is lossless, but since all computers have only finite precision, most of transformations are lossy in practice, even when we use floating point calculations. On the other hand, integer calculations are much faster than floating point for virtually all computers; and integer computations are much easier to implement in hardware which is more important in some applications. The memory utilization of integers is also a positive consideration. The difficulty is, if we directly use integers in the wavelet transform and its inverse without some proper considerations, it will cause the loss of accuracy. For some important image applications, the user wants to have complete control of the precision in which the image pixels are represented during the compression process, and thus prefers to have the image compressed from lossless to lossy.

Lossless compression is also very important for images found in such applications as medical and space science. In such situations, the designer of the compression algorithm must be very careful to avoid discarding any information that may be required or even useful at some later point From the academic point of view, it is also very interesting to have a compression scheme which has very fast performance, and which can exactly reconstruct the image when necessary. In addition, it is also very useful to have some wavelet transforms which exhibit the property of precision preservation (PPP), which can be utilized in the computer which has limited precision and limited memory without losing any precision during the computation.

In this paper, we are going to describe two general methods from which one can get the integer wavelet transform desired. All of the wavelet transforms from the methods given in this paper possess the property of precision preservation (PPP). We draw on the work of several other authors who have already contributed to this area [2-3], where some specific examples were developed. However, this paper presents a more general method which allows one to see several new results as well as those presented and acknowledged prior to this work.

This paper is organized as follows: Section 2 and 3 give some examples of integer wavelet transforms. The examples in section 2 are the starting point for our approach, and the examples in Section 3 show the steps and motivation of our general method. Section 4 indicates how one can use the lifting technique to create an integer biorthogonal wavelet transform. The Correction technique to generate more general integer wavelet transforms is described in Section 5. Section 6 describes how to process boundaries in order to apply the integer calculation in finite sized images or signals. In Section 7, we prove the integer wavelet transforms developed by both the lifting and correction method possess the property of precision preservation (PPP). Some example images are also shown in this section. The last section, Section 8, provides the conclusion to this paper.

## 2. Basic integer wavelet transformations

We provide the following two examples as the starting point for our new method. For the sake of convenience, length, and simplicity, we only discuss the algorithm for a one level decomposition and reconstruction and only for a one dimensional signal. The extension to two dimensions is immediate as the rows and columns can be treated into a sequence of one dimensional signals. For the following examples, assume that $\{c_n^0\}_{n=0}^{N-1}$ is the original signal where the superscript indicates level and the subscript indicates a particular point in the signal. Also, $\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{M_1-1}$ are its decomposition parts at the first level. Here

$$\begin{cases} N_1 = \begin{cases} \frac{N}{2}, & \text{if } N \text{ is an even number,} \\ \frac{N+1}{2}, & \text{if } N \text{ is an odd number;} \end{cases} \\ M_1 = N - N_1. \end{cases}$$

$\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{M_1-1}$ are its low frequency ($l$) part and high frequency ($h$) part, respectively. For multi-levels, we just treat $\{c_n^1\}_{n=0}^{N_1-1}$ as $\{c_n^0\}_{n=0}^{N-1}$ and repeat the procedure again.

Example 1: A (2,2)-wavelet transform by integer calculation.

This transformation is similar to a variation of the Haar wavelet transform which uses low and high pass analysis (decomposition ) filters given as:

| n | 0 | 1 |
|---|---|---|
| $\tilde{h}_n$ | 1/2 | 1/2 |
| $\tilde{g}_n$ | 1/2 | -1/2 |

(1) Compute

$$d_k^1 = c_{2k}^0 - c_{2k+1}^0, \quad k = 0, \cdots, M_1 - 1. \tag{2.1}$$

(2) Compute

$$c_k^1 = \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0, \quad k = 0, \cdots, N_1 - 2,$$

$$c_{N_1-1}^1 = \begin{cases} \mathrm{Int}(\frac{d_{N_1-1}^1}{2}) + c_{N-1}^0, & \text{if } N \text{ is an even number,} \\ c_{N-1}^0, & \text{if } N \text{ is an odd number.} \end{cases} \tag{2.2}$$

Here, $\mathrm{Int}(x)$ is an arbitrary rounding function which may have different interpretations. For example, $\mathrm{Int}(x)$ can be the integer which is nearest to $x$, or $\mathrm{Int}(x)$ may be any integer which satisfies $x - 1 < \mathrm{Int}(x) \le x$, etc. It is easy to see that all entries in both $\{c_n^1\}_{n=0}^{N_1-1}$ and $\{d_n^1\}_{n=0}^{M_1-1}$ are integers.



IPR2023-00330 Page 00652

From (2.1)-(2.2), we can easily get the following integer reconstruction algorithm:

(b) Reconstruction

(1) If $N$ is an even number, compute

$$c^0_{2k+1} = c^1_k - \mathrm{Int}(\frac{d^1_k}{2}), \quad k = 0, \cdots, N_1 - 1 \; ; \tag{2.3}$$

or, if $N$ is an odd number, we have

$$\begin{aligned} c^0_{2k+1} &= c^1_k - \mathrm{Int}(\frac{d^1_k}{2}), \quad k = 0, \cdots, N_1 - 2, \\ c^0_{N-1} &= c^1_{N_1}. \end{aligned} \tag{2.4}$$

(2) Compute

$$c^0_{2k} = d^1_k + c^0_{2k+1}, \quad k = 0, \cdots, M_1 - 1. \tag{2.5}$$

*Remark* Since (2.1)-(2.6) are not linear because of the rounding operation $\mathrm{Int}(x)$, this means the transformation order becomes significant. For instance, if the decomposition was applied first to the columns and then to the rows, the inverse transformation must be applied first to the rows and then to the columns.

**Example 2**: Lazy wavelet transform.

The Lazy wavelet transform *does not do anything*. However, this illustrates an important concept. The corresponding inverse transform is nothing else but sub-sampling the even and odd indexed samples. Decomposition and reconstruction can use same formula as follows:

$$\begin{aligned} c^1_k &= c^0_{2k}, \; k = 0, \cdots, N_1 - 1; \\ d^1_k &= c^0_{2k+1}, \; k = 0, \cdots, M_1 - 1. \end{aligned}$$

Examples 1 and 2 are not good transforms for image compression, but they are simple. Much better transforms can be achieved from these two. As suggested above, we consider them only as a starting point for our integer, reversible, wavelet transform algorithm.

We must mention that there is another interesting property in the above two transforms which may not be easily seen. If the values of the signal pixels are represented by a finite number of bits, say one bit or one byte, we can still use the same number of bits to represent the result of the forward transform within the computer itself because of the complementary code property. While, from the reconstruction algorithm, the computer will get back the exact original signal through the same complementary code property. We call this property *a Property of Precision Preservation (PPP)* for these wavelets.

It is known that the general values of the high frequency wavelet coefficients are small, and all higher levels in the decomposition also provide generally small values in the high frequency band. This allows the preservation of precision during the computational stage of the wavelet coefficients. Now, the complementary code property, the other aspect of the *PPP* property is a well known characteristic of integer arithmetic as done by the computer. Consider the computation of the difference of two integers given as $c = b\text{-}a$ and the inverse computation of $a = b - c$. The nature of the computation within the computer can be specified as follows:

$$c_m = \begin{cases} b-a & \text{if } -2^{q-1} \le b-a < 2^{q-1}-1 \\ -2^q + b - a & \text{if } b-a \ge 2^{q-1} \\ 2^q + b - a & \text{if } b-a < -2^{q-1} \end{cases}$$

and the inverse is

$$a_m = \begin{cases} b-c_m & \text{if } -2^{q-1} \le b-c_m < 2^{q-1}-1 \\ -2^q + b - c_m & \text{if } b-c_m \ge 2^{q-1} \\ 2^q + b - c_m & \text{if } b-c_m < -2^{q-1} \end{cases}$$

where the m subscript indicates the internal representation, and the range of the integers a, b, c is $\left[-2^{q-1},\ 2^{q-1}-1\right]$. The internal representation of $c_m$ when it is outside the range, its appearance is as a two's complement number, so the representation may not be the same as the external representation of c. However, the same complementary code for the $a_m$ will cause the internal representation to be identical to the external representation of $a$. For example, if we let $b=2$ (00000010) and $a=-127$ (10000001) then $c_m$ has the internal binary value of (10000001) when $q=4$. With a value of -127 for $c_m$, the inverse value for $a_m$ will just be $a$.

In fact, for Example 2, this property is obviously true. While for Example 1, if the range of the pixel values is within a finite number of bits, say q, we can only use q bits as the working unit, which means the value of transform coefficients will also be within the interval with length $2^q$, say $\left[-2^{q-1},\ 2^{q-1}-1\right]$. Due to the nature of computation on a machine, most machines will implement (2.1)-(2.2) automatically as follows (the complementary code property):

$$d_k^1 = \begin{cases} c_{2k}^0 - c_{2k+1}^0, & \text{if } -2^{q-1} \le c_{2k}^0 - c_{2k+1}^0 < 2^{q-1}, \\ c_{2k}^0 - c_{2k+1}^0 - 2^q, & \text{if } c_{2k}^0 - c_{2k+1}^0 \ge 2^{q-1}, \\ 2^q + \left(c_{2k}^0 - c_{2k+1}^0\right), & \text{if } c_{2k}^0 - c_{2k+1}^0 < -2^{q-1}. \end{cases} \tag{2.6}$$

$$c_k^1 = \begin{cases} \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0, & \text{if } -2^{q-1} \le \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0 < 2^{q-1}, \\ \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0 - 2^q, & \text{if } \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0 \ge 2^{q-1}, \\ \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0 + 2^q, & \text{if } \mathrm{Int}(\frac{d_k^1}{2}) + c_{2k+1}^0 < -2^{q-1}. \end{cases} \tag{2.7}$$

While the reconstruction algorithm (2.3) and (2.5) will be implemented by the computer itself as

$$c_{2k+1}^0 = \begin{cases} c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}), & \text{if } -2^{q-1} \le c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}) < 2^q, \\ 2^q + \left( c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}) \right), & \text{if } c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}) < -2^{q-1}, \\ \left( c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}) \right) - 2^q, & \text{if } c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}) > 2^{q-1}. \end{cases} \tag{2.8}$$

$$c_{2k}^0 = \begin{cases} d_k^1 + c_{2k+1}^0, & \text{if } -2^{q-1} \le d_k^1 + c_{2k+1}^0 < 2^{q-1}, \\ d_k^1 + c_{2k+1}^0 + 2^q, & \text{if } d_k^1 + c_{2k+1}^0 < -2^{q-1}, \\ d_k^1 + c_{2k+1}^0 - 2^q, & \text{if } d_k^1 + c_{2k+1}^0 \ge 2^{q-1}. \end{cases} \tag{2.9}$$

It is obvious that (2.8)-(2.9) are just the reverse of (2.6)-(2.7). It is also easy to see that if we properly take advantage of the bound in the coefficient size mentioned above, the algorithm can be implemented using a minimal amount of storage.

## 3. More Examples and Additional Analysis

In this section we are going to give more examples which will give some motivation for our new approach.

Example 3: A (2,6)-wavelet transform by integer calculation [2].

This transformation is similar to using following analysis filters

| n | -2 | -1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| $h_n$ | 0 | 0 | 1/2 | 1/2 | 0 | 0 |
| $\tilde{h}_n$ | -1/16 | -1/16 | 1/2 | -1/2 | 1/16 | 1/16 |

(a) Decomposition

Decomposition starts with Example 1 at step (1) and (2), and then upgrades the high frequency component at step (3):

(1) Compute

$$d_k^{1,0} = c_{2k}^0 - c_{2k+1}^0, \quad k = 0, \cdots, M_1 - 1.$$

(2) Compute

$$c_k^1 = \mathrm{Int}(\frac{d_k^{1,0}}{2}) + c_{2k+1}^0, \quad k = 0, \cdots, N_1 - 2,$$

$$c_{N_1-1}^1 = \begin{cases} \mathrm{Int}(\frac{d_{M_1-1}^{1,0}}{2}) + c_{N-1}^0, & \text{if } N \text{ is an even number,} \\ c_{N-1}^0, & \text{if } N \text{ is an odd number.} \end{cases}$$

(3) Compute

$$\begin{cases} d_0^1 = \mathrm{Int}(\frac{c_0^1 - c_1^1}{4}) - d_0^{1,0} \\ d_k^1 = \mathrm{Int}(\frac{c_{k-1}^1 - c_{k+1}^1}{4}) - d_k^{1,0}, \quad k = 1, \cdots, M_1 - 2. \end{cases}$$

and then, if $N$ is even, calculate

$$d_{M_1-1}^1 = \mathrm{Int}(\frac{c_{N_1-2}^1 - c_{N_1-1}^1}{4}) - d_{M_1-1}^{1,0},$$

else, calculate

$$d_{M_1-1}^1 = \mathrm{Int}(\frac{c_{N_1-3}^1 - c_{N_1-1}^1}{4}) - d_{M_1-1}^{1,0}.$$

(b) Reconstruction

The reconstruction algorithm is identical to the decomposition algorithm, except it is now running "backwards".

(1) Compute

$$\begin{cases} d_0^{1,0} = \mathrm{Int}(\frac{c_0^1 - c_1^1}{4}) - d_0^1 \\ d_k^{1,0} = \mathrm{Int}(\frac{c_{k-1}^1 - c_{k+1}^1}{4}) - d_k^1, \quad k = 1, \cdots, M_1 - 2, \end{cases}$$

and then, if $N$ is even, calculate

$$d_{M_1-1}^{1,0} = \mathrm{Int}(\frac{c_{N_1-2}^1 - c_{N_1-1}^1}{4}) - d_{M_1-1}^{1,1},$$

else, calculate

$$d_{M_1-1}^{1,0} = \mathrm{Int}(\frac{c_{N_1-3}^1 - c_{N_1-1}^1}{4}) - d_{M_1-1}^{1,1}.$$

(2) If $N$ is an even number, compute

$$c_{2k+1}^0 = c_k^1 - \mathrm{Int}(\frac{d_k^{1,0}}{2}), \quad k = 0, \cdots, N_1 - 1;$$

or, if $N$ is an odd number, we have

$$c^0_{2k+1} = c^1_k - \mathrm{Int}(\frac{d^{1,0}_k}{2}), \quad k = 0, \cdots, N_1 - 2,$$

$$c^0_{N-1} = c^1_{N_1}.$$

(3) Compute

$$c^0_{2k} = d^{1,0}_k + c^0_{2k+1}, \quad k = 0, \cdots, M_1 - 1.$$

We see in step (2)-(3) above, that they are just the same as shown for the reconstruction of the (2,2)-wavelet transform (Example 1).

Example 4: A (1,3)-wavelet transform by integer calculation.
The following nonlinear transform is a variation of the transform which uses biorthogonal analysis filters:

| n | -1 | 0 | 1 |
|---|---|---|---|
| $\bar{h}_n$ | 1 | 0 | 0 |
| $\tilde{g}_n$ | 1/4 | -1/2 | 1/4 |

(a) Decomposition

This decomposition starts with the *Lazy wavelet* at step (1) and upgrades the high frequency component at step (2):

(1) Set
$$c^1_k = c^0_{2k}, \ k = 0, \cdots, N_1 - 1;$$
$$d^1_k = c^0_{2k+1}, \ k = 0, \cdots, M_1 - 1.$$

(2) If $N$ is an even number, calculate

$$\begin{cases} d^1_k = \mathrm{Int}(\frac{c^1_k + c^1_{k+1}}{2}) - d^{1,0}_k, & k = 0, \cdots, M_1 - 2, \\ d^1_{M_1-1} = c^1_{N_1-1} - d^{1,0}_{M_1-1}. \end{cases}$$

Otherwise, if $N$ is an odd number, calculate

$$d^1_k = \mathrm{Int}(\frac{c^0_{2k} + c^0_{2k+2}}{2}) - c^0_{2k+1}, \quad k = 0, \cdots, M_1 - 1.$$

(b) Reconstruction

(1) Set
$$c^0_{2k} = c^1_k, \ k = 0, \cdots, N_1 - 1;$$

(2) If $N$ is an even number, calculate

$$\begin{cases} c^0_{2k+1} = \mathrm{Int}(\frac{c^0_{2k} + c^0_{2k+2}}{2}) - d^1_k, & k = 0, \cdots, M_1 - 2, \\ c^0_{N-1} = c^0_{N-2} - d^1_{M_1-1}. \end{cases}$$



IPR2023-00330 Page 00657

Otherwise, if $N$ is an odd number, calculate

$$c^0_{2k+1} = \mathrm{Int}(\frac{c^0_{2k} + c^0_{2k+2}}{2}) - d^1_k, \quad k = 0, \cdots, M_1 - 1.$$

Example 5: A (5,3)-wavelet transform by integer calculation.

This transformation is also similar in function to using the biorthogonal analysis filters. It is given by

| n | -2 | -1 | 0 | 1 | 2 |
|---|---|---|---|---|---|
| $\tilde{h}_n$ | -1/8 | 1/4 | 3/4 | 1/4 | -1/8 |
| $\tilde{g}_n$ | 1/4 | -1/2 | 1/4 | 0 | 0 |

(a) Decomposition

This decomposition starts with Example 3 at step (1) and upgrade low frequency components at step (2):

(1) Set $c^{1,0}_k = c^0_{2k}, \quad k=0, \cdots, N_1 - 1;$

If $N$ is an even number, calculate

$$\begin{cases} d^1_k = \mathrm{Int}(\frac{c^0_{2k} + c^0_{2k+2}}{2}) - c^0_{2k+1}, & k = 0, \cdots, M_1 - 2, \\ d^1_{M_1-1} = c^0_{N-2} - c^0_{N-1}. \end{cases}$$

Otherwise, if $N$ is an odd number, calculate

$$d^1_k = \mathrm{Int}(\frac{c^0_{2k} + c^0_{2k+2}}{2}) - c^0_{2k+1}, \quad k = 0, \cdots, M_1 - 1.$$

(2) If N is an even number, compute

$$\begin{cases} c^1_0 = c^{1,0}_0 - \mathrm{Int}(\frac{d^1_0}{2}), \\ c^1_k = c^{1,0}_k - \mathrm{Int}(\frac{d^1_{k-1} + d^1_k}{4}), & k = 1, \cdots, N_1 - 2, \\ c^1_{N_1-1} = c^{1,0}_{N_1-2} - \mathrm{Int}(\frac{d^1_{N_1-2} + d^1_{N_1-1}}{4}). \end{cases}$$

Otherwise, if $N$ is an odd number, calculate



IPR2023-00330 Page 00658

$$\begin{cases} c_0^1 = c_0^{1,0} - \mathrm{Int}(\frac{d_0^1}{2}), \\ c_k^1 = c_k^{1,0} - \mathrm{Int}(\frac{d_{k-1}^1 + d_k^1}{4}), \quad k = 1, \cdots, N_1 - 2, \\ c_{N_1-1}^1 = c_{N_1-1}^{1,0} - \mathrm{Int}(\frac{d_{M_1-1}^1}{2}). \end{cases}$$

(b) Reconstruction

(1) Compute

$$c_0^0 = c_0^1 + \mathrm{Int}(\frac{d_0^1}{2}),$$

$$c_{2k}^0 = c_k^1 + \mathrm{Int}(\frac{d_{k-1}^1 + d_k^1}{4}), \quad k = 1, \cdots, N_1 - 2,$$

Then, if $N$ is even, calculate

$$c_{N-2}^0 = c_{N_1-1}^1 + \mathrm{Int}(\frac{d_{N_1-2}^1 + d_{N_1-1}^1}{4}).$$

else calculate $$c_{N-1}^0 = c_{N_1-1}^1 + \mathrm{Int}(\frac{d_{M_1-1}^1}{2}).$$

(2) Compute $$c_{2k+1}^0 = \mathrm{Int}(\frac{c_{2k}^0 + c_{2k+2}^0}{2}) - d_k^1, \quad k = 0, \cdots, M_1 - 2,$$

Then, if $N$ is even, calculate $$c_{N-1}^0 = c_{N-2}^0 - d_{M_1-1}^1.$$

The *PPP* property for Example 1-2 mentioned at the end of the previous section is also applicable for these three examples. It is obvious these three transformations are not really linear, but they are similar to the one using the corresponding filters given above. Especially, the filters in Example 3 and Example 5 belong to, with minor modification, the group of the best biorthogonal filters for image compression according to both our experience and the conclusion of [4].

Also, from the above three examples, we can note that if we begin with integer (linear or nonlinear) wavelet transformations and then use some proper upgrading formulas, we can get other, much better integer, wavelet transformations for image compression. Now, the key problem is: What kind of deductive formulas should be used? We provide an answer to this question in the following two sections, Section 4 and Section 5.

## 4. Lifting Scheme and Integer Biorthogonal Filtering

The *Lifting scheme*, discovered by Sweldens [1], is a new approach for constructing biorthogonal wavelets with compact support. However, the most interesting part of this method

for us is: it can be used, with minor modification, to create integer biorthogonal wavelet transformations. The following is an adaptation of the technique of [1].

Definition 1. The set of filters $\{h, \tilde{h}, g, \tilde{g}\}$ is a set of *biorthogonal filters* if the following formula is satisfied:

$$\forall \omega \in \mathbf{R}: \ \tilde{m}(\omega)\overline{m^t(\omega)} = 1.$$

where
$$m(\omega) = \begin{bmatrix} h(\omega) & h(\omega+\pi) \\ g(\omega) & g(\omega+\pi) \end{bmatrix},$$

and
$$h(\omega) = \sum_k h_k e^{-ik\omega} \text{ and } g(\omega) = \sum_k g_k e^{-ik\omega},$$

and similarly for
$$\tilde{m}(\omega), \tilde{h}(\omega) \text{ and } \tilde{g}(\omega).$$

The following lemma is the main result of the *lifting scheme* [1] reported as corollary 6 in that paper.

Lemma 1. Take an initial set of finite biorthogonal filters $\{h, \tilde{h}^0, g^0, \tilde{g}\}$, then a new set of finite biorthogonal filters $\{h, \tilde{h}, g, \tilde{g}\}$ can be found as

$$\begin{aligned} \tilde{h}(\omega) &= \tilde{h}^0(\omega) + \tilde{g}(\omega)\overline{s(2\omega)} \\ g(\omega) &= g^0(\omega) - h(\omega)s(2\omega). \end{aligned} \tag{4.1a}$$

Similarly, if we take $\{h^0, \tilde{h}, g, \tilde{g}^0\}$ as an initial set of biorthogonal filters, a new set of finite biorthogonal filters $\{h, \tilde{h}, g, \tilde{g}\}$ can be found as

$$\begin{aligned} h(\omega) &= h^0(\omega) + g(\omega)\overline{\tilde{s}(2\omega)} \\ \tilde{g}(\omega) &= \tilde{g}^0(\omega) - \tilde{h}(\omega)\overline{\tilde{s}}(2\omega). \end{aligned} \tag{4.1b}$$

Here, $s(\omega)$ is a trigonometric polynomial and the corresponding filter $s$ is finite, and so is $\overline{\tilde{s}}(\omega)$. Actually, regarding the filters, (4.1) is equivalent to

$$\begin{aligned} \tilde{h}_k &= \tilde{h}_k^0 + \sum_l \tilde{g}_{k+2l} s_l \\ g_k &= g_k^0 - \sum_l h_{k-2l} s_l. \end{aligned} \tag{4.2a}$$

or

$$\begin{aligned} h_k &= h_k^0 + \sum_l g_{k+2l} \tilde{s}_l \\ \tilde{g}_k &= \tilde{g}_k^0 - \sum_l \tilde{h}_{k-2l} \tilde{s}_l. \end{aligned} \tag{4.2b}$$

Next, we use the lifting scheme with minor modification to create an integer, nonlinear, quasi-biorthogonal, wavelet algorithm. Suppose $\{c_k^0\}$ is a original signal, $\{c_k^1\}$ and $\{d_k^1\}$ are again its low and high frequency decomposition parts, obtained by using the filters $\{h, \tilde{h}, g, \tilde{g}\}$.



IPR2023-00330 Page 00660

If we use filters $\{\tilde{h}, \tilde{g}\}$ for decomposition (analysis), the corresponding decomposition algorithm is

$$\begin{cases} c_k^1 = \alpha_c \sum_n c_n^0 \tilde{h}_{n-2k}, \\ d_k^1 = \alpha_d \sum_n c_n^0 \tilde{g}_{n-2k}. \end{cases}$$

While the reconstruction algorithm will be

$$c_n^0 = 2\sum_k \left( \frac{c_k^1 h_{n-2k}}{\alpha_c} + \frac{d_k^1 g_{n-2k}}{\alpha_d} \right),$$

related to the synthesis filter $\{h, g\}$. Here, parameters $\alpha_c$ and $\alpha_d$ are positive constants with $\alpha_c \bullet \alpha_d = 2$. For example, in the situation of regular biorthogonal decomposition and reconstruction, $\alpha_c = \alpha_d = \sqrt{2}$; and for Example 1 through Example 5 above, $\alpha_c = 1$ and $\alpha_d = 2$.

If the set of filters $\{h, \tilde{h}, g, \tilde{g}\}$ is from $\{h, \tilde{h}^0, g^0, \tilde{g}\}$ by (4.2b), then the decomposition can be accomplished as follows:

1. Calculate

$$\begin{cases} c_k^{1,0} = \alpha_c \sum_n c_n^0 \tilde{h}_{n-2k}^0, \\ d_k^1 = \alpha_d \sum_n c_n^0 \tilde{g}_{n-2k}. \end{cases} \tag{4.3}$$

2. Calculate

$$c_k^1 = c_k^{1,0} + \frac{\alpha_c}{\alpha_d} \sum_l d_{k-l}^1 s_l. \tag{4.4}$$

The relative reconstruction scheme will be:

1. Calculate

$$c_k^{1,0} = c_k^1 - \frac{\alpha_c}{\alpha_d} \sum_l d_{k-l}^1 s_l, \tag{4.5}$$

2. Calculate

$$c_n^0 = 2\sum_k \left( \frac{c_k^{1,0} h_{n-2k}}{\alpha_c} + \frac{d_k^1 g_{n-2k}^0}{\alpha_d} \right). \tag{4.6}$$

Here, equations (4.3) and (4.6) are just the wavelet ( inverse ) transforms using biorthogonal filters $\{h, \tilde{h}^0, g^0, \tilde{g}\}$. While (4.4) and (4.5) are forward and backward upgrading formulas.

Similarly, if the set of filters $\{h, \tilde{h}, g, \tilde{g}\}$ is from the initial set of filters $\{h^0, \tilde{h}, g, \tilde{g}^0\}$ by using (4.2b), the relative decomposition is:

1. Calculate

$$\begin{cases} c_k^1 = \alpha_c \sum_n c_n^0 \tilde{h}_{n-2k}, \\ d_k^{1,0} = \alpha_d \sum_n c_n^0 \tilde{g}_{n-2k}^0. \end{cases}$$



IPR2023-00330 Page 00661

2. Calculate $d_k^1 = d_k^{1,0} - \frac{\alpha_d}{\alpha_c}\sum_l c_{k-l}^1 s_l$.

The reconstruction scheme is:

1. Calculate $d_k^{1,0} = d_k^1 + \frac{\alpha_d}{\alpha_c}\sum_l c_{k-l}^1 s_l$

2. Calculate $c_n^0 = 2\sum_k\left(\frac{c_k^1 h_{n-2k}^0}{\alpha_c} + \frac{d_k^1 g_{n-2k}}{\alpha_d}\right)$

For the sake of clarity, we haven't considered the boundary situation, but we will address this later.

Corollary 4.1. Suppose biorthogonal filters $\{h, \bar{h}, g, \bar{g}\}$ are from initial filters $\{h, \bar{h}^0, g^0, \bar{g}\}$ by the lifting scheme (4.1a) or (4.2a). If the decomposition and reconstruction by filters $\{h, \bar{h}^0, g^0, \bar{g}\}$ can be accomplished only by integer calculation, such as Example 2, we also can create a corresponding integer wavelet decomposition and reconstruction scheme which is very "close" to the original one by using filters $\{h, \bar{h}, g, \bar{g}\}$. Here the word "close" means that the difference of the two decomposition schemes is just some rounding error, and this rounding error will be corrected by the integer reconstruction scheme.

In fact, if $\{c_k^{1,0}\}$ and $\{d_k^1\}$ are integer after (4.3), we can calculate $\{c_k^1\}$ by

$$c_k^1 = c_k^{1,0} + \mathrm{Int}\left(\frac{\alpha_c}{\alpha_d}\sum_l d_{k-l}^1 s_l\right). \tag{4.7}$$

instead of (4.4). Here $\mathrm{Int}(x)$, as described in Section 2, is an arbitrary rounding up function which satisfies $x-1 \le \mathrm{Int}(x) \le x+1$. It is obvious that (4.7) is very "close" to (4.4), and the exact reconstruction scheme can easily be obtained from

$$c_k^{1,0} = c_k^1 - \mathrm{Int}\left(\frac{\alpha_c}{\alpha_d}\sum_l d_{k-l}^1 s_l\right) \tag{4.8}$$

and (4.6). There will be a similar result, if the set of biorthogonal filters $\{h, \bar{h}, g, \bar{g}\}$ is obtained from the initial set of filters $\{h^0, \bar{h}, g, \bar{g}^0\}$ by using (4.2b).

We can now note, except for the example shown in the Lazy *wavelet*, (Example 2) most standard biorthogonal wavelet transforms cannot be performed directly by integer, even for one of the simplest wavelets, the *Haar wavelet*. However, if we properly choose the parameters $\alpha_c$ and $\alpha_d$, and slightly change the transform algorithms, such as Example 1 and Example 3, we can have a variation of the original biorthogonal wavelet transforms with respect to the set of filters



IPR2023-00330 Page 00662

$\{h, \tilde{h}^0, g^0, \tilde{g}\}$ (or $\{h^0, \tilde{h}, g, \tilde{g}^0\}$). On the other hand, the parameters $\{s_i\}$ should be also chosen carefully to guarantee that only addition and shift operations are needed by the algorithm.

Another observation: if the set of filters $\{h, \tilde{h}, g, \tilde{g}\}$ is obtained from a set of filters $\{h^0, \tilde{h}, g, \tilde{g}^0\}$ by the lifting scheme, and the set $\{h^0, \tilde{h}, g, \tilde{g}^0\}$ is also obtained from a filter set $\{h^0, \tilde{h}^0, g^0, \tilde{g}^0\}$, we can repeatedly use Corollary 1 to get a "close" integer wavelet transformation.

## 5. The Correction Method for Creating Integer Wavelet Transforms

In this section, we will describe another approach for obtaining integer wavelets by using the so called *Correction method*. The motivation of this method is from the S+P transform, and we will now generalize this approach. Actually, the lifting scheme for generating biorthogonal wavelets can be considered as a special case of the correction method. From this method we can get some even complicated filters with fast decomposition and reconstruction algorithm.

Suppose that we already have a simple integer wavelet transform, such as Examples 1 through 3, the decomposition and reconstruction scheme of which can be formulated as follows:

Decomposition

$$c_i^{1,0} = df_c\left(\{c_n^0\}\right)$$
$$d_i^{1,0} = df_d\left(\{c_n^0\}\right) \tag{5.1}$$

Reconstruction

$$c_n^0 = rf\left(\{c_i^{1,0}\}, \{d_i^{1,0}\}\right) \tag{5.2}$$

Here, (5.1) and (5.2) can be the same as (4.3) and (4.6) or other algorithms.

In general, after the above decomposition, one may not be satisfied with the result. There may still be some correlation among the highpass components because of the aliasing from the lowpass components, or the lowpass components do not carry enough of the expected information from the original signal. Hence, we could make an improvement by putting some correction part on the highpass components or lowpass components. There are many ways to accomplish this. However, for the sake of the integer calculation, we prefer to use following correction method. For example, if we want to make a correction for the highpass part, the corresponding formula would be:

$$d_i^1 = d_i^{1,0} - \mathrm{Int}\left(dc_i^1\right) \quad k = \cdots, 0, 1, 2, \cdots. \tag{5.3}$$

Here, $dc_i^1$ is a correction quantity for $d_i^1$

$$dc_i^1 = \sum_{\alpha=s_1}^{s_2} \sigma_\alpha c_{i+\alpha}^{1,0} + \sum_{\beta=1}^{\tau} \tau_\beta d_{i+\beta}^{1,0}, \quad k = \cdots, 0, 1, 2, \cdots \tag{5.4}$$



IPR2023-00330 Page 00663

and, $\{\sigma_i\}_{i=-S_0}^{S_1}$ and $\{\tau_j\}_{j=1}^{T}$ are given parameters which have been chosen for the user's purpose, such as reducing the redundancy among highpass components or some other special requirement. We are not going to discuss how to choose these parameters, but one can refer to the references [3, 5, 6] for clarification of this process. The only thing we need to mention is, for the sake of the integer calculation, any entries in both $\{\sigma_i\}_{i=-S_0}^{S_1}$ and $\{\tau_j\}_{j=1}^{T}$ should be rational numbers with denominators being powers of 2.

From (5.1), (5.3) and (5.4), it is easy to see the perfect reconstruction algorithm can be

$$d_k^{1,0} = d_k^1 + \mathrm{Int}(dc_k), \ k = \cdots, \ m, \ m-1, \ m-2 \cdots, \tag{5.5}$$

combined with (5.2).

As mentioned above, the Lifting scheme is a special condition of the Correction method. Examples 3 through 5 can also be considered as the examples of this method. We next give an example of the Correction method which cannot be included in the group of Lifting scheme, and also which does not result in a closed form of compact support for biorthogonal filters.

Example 6 S+P transform [3], which is similar to using following analysis filters

| n | -2 | -1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| $h_n$ | 0 | 0 | 1/2 | 1/2 | 0 | 0 |
| $\tilde{h}_n$ | -1/16 | -1/16 | 15/32 | -17/32 | 7/32 | -1/32 |

While, the synthesis filters do not have compact support. However, the S+P transform can be implemented as follows:

(a) Decomposition

(1) Take the decomposition step of Example 1, that is, compute

$$d_k^{1,0} = c_{2k}^0 - c_{2k+1}^0, \quad k = 0, \ 1, \cdots, M_1 - 1;$$

and

$$c_k^1 = \mathrm{Int}\left(\frac{d_k^{1,0}}{2}\right) + c_{2k+1}^0, \quad k = 0, \cdots, N_1 - 2,$$

$$c_{N_1-1}^1 = \begin{cases} \mathrm{Int}\left(\dfrac{d_{M_1-1}^{1,0}}{2}\right) + c_{N-1}^0, & \text{if } N \text{ is an even number,} \\ c_{N-1}^0, & \text{if } N \text{ is an odd number.} \end{cases}$$

(2) Correction Step: Define $S_0 = -1$, $S_1 = 1$, $T = 1$ and

$$\sigma_{-1} = -\tfrac{1}{4}, \ \sigma_0 = -\tfrac{1}{4}, \ \sigma_1 = \tfrac{1}{4};$$

$$\tau_1 = \tfrac{1}{4}.$$

and now compute



IPR2023-00330 Page 00664

In fact, the quantity $b_t$ would have the same value in both (4.7) and (4.8) if we calculate it in the same way. On the other hand, if the working unit for $b_t$ is $q$ bits, the machine will give $b_t$ another value, say $\bar{b}_t$ ( $-2^{q-1} \le \bar{b}_t < 2^{q-1}$ ), where $\bar{b}_t$ is not equal to $b_t$ in the sense of mathematics if the value of $b_t$ is beyond the interval $[-2^{q-1}, 2^{q-1}-1]$. However, $\bar{b}_t$ will be the same in both (4.7) and (4.8). Therefore, the machine will automatically implement (7.1) and (7.2) as

$$c_t^1 = \begin{cases} c_t^{1,0} + \bar{b}_t, & \text{if } -2^{q-1} \le c_t^{1,0} + \bar{b}_t < 2^{q-1}, \\ c_t^{1,0} + \bar{b}_t - 2^q, & \text{if } c_t^{1,0} + \bar{b}_t \ge 2^{q-1}, \\ 2^q + c_t^{1,0} + \bar{b}_t & \text{if } c_t^{1,0} + \bar{b}_t < -2^{q-1}. \end{cases} \tag{7.1m}$$

and

$$c_t^{1,0} = \begin{cases} c_t^1 - \bar{b}_t, & \text{if } -2^{q-1} \le c_t^1 - \bar{b}_t < 2^{q-1}, \\ 2^q + c_t^1 - \bar{b}_t, & \text{if } c_t^1 - \bar{b}_t < -2^{q-1}, \\ c_t^1 - \bar{b}_t - 2^q, & \text{if } c_t^1 - \bar{b}_t \ge 2^{q-1}. \end{cases} \tag{7.2m}$$

It is easy to see that (7.2m) is just the backward operation of (7.1m), which provides the evidence that the conclusion of this lemma is correct.

It should be mentioned that the coefficients $\{c_t^1\}$ obtained by (4.3) and (7.1m) might not be the "real" wavelet coefficients using common sense. However, if we still use the working unit with $q$ bits precision at the reconstruction step, (7.2m) and (4.6) will give the exact original signal back. On the other hand, the coefficients $\{c_t^1\}$ still keep the most continuity of the "real" wavelet coefficients. Therefore, when we repeat the decomposition step on $\{c_t^1\}$, most small coefficients in its high frequency part $\{d_t^2\}$ will be almost the same as the "real" coefficients (within some rounding error), which allows us to still take advantage of the "real" wavelet transform in image compression.

A similar argument can show the same *PPP* property will hold for the integer wavelet transforms generated by the Correction method in Section 5.

As we mentioned before, for many applications, the lossless image compression is as important as lossy compression. The integer wavelet transforms give the opportunity to compress without loss. It is also obvious that the integer wavelet algorithms can be used wherever ordinary wavelets are used, especially in signal and image compression. However, for most computers, the integer wavelet transform is much faster than the ordinary one and it uses much less memory. The following are some applications illustrating these types of transforms.

$$\begin{cases} d_0^1 = d_0^{1,0} - \mathrm{Int}(\frac{c_0^1 - c_1^1}{4}); \\ d_k^1 = d_k^{1,0} - \mathrm{Int}(\frac{2c_{k-1}^1 + c_k^1 - 3c_{k+1}^1 - 2d_{k+1}^{1,0}}{8}), \quad k = 1, \cdots, M_1 - 2; \\ d_{M_1-1}^1 = d_{M_1-1}^{1,0} - \mathrm{Int}(\frac{c_{M_1-2}^1 - c_{M_1-1}^1}{4}). \end{cases}$$

(b) Reconstruction

(1) Compute

$$\begin{cases} d_{M_1-1}^{1,0} = d_{M_1-1}^1 + \mathrm{Int}(\frac{c_{M_1-2}^1 - c_{M_1-1}^1}{4}); \\ d_k^{1,0} = d_k^1 + \mathrm{Int}(\frac{2c_{k-1}^1 + c_k^1 - c_{k+1}^1 - 2d_{k+1}^{1,0}}{8}), k = M_1 - 2, \cdots, 1; \\ d_0^{1,0} = d_0^1 + \mathrm{Int}(\frac{c_0^1 - c_1^1}{4}). \end{cases}$$

(2) If $N$ is an even number, compute

$$c_{2k+1}^0 = c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}), \quad k = 0, \cdots, N_1 - 1$$

or, if $N$ is an odd number, we have

$$c_{2k+1}^0 = c_k^1 - \mathrm{Int}(\frac{d_k^1}{2}), \quad k = 0, \cdots, N_1 - 2,$$

$$c_{N-1}^0 = c_{N_1}^1.$$

(3) Compute

$$c_{2k}^0 = d_k^1 + c_{2k+1}^0, \quad k = 0, \cdots, M_1 - 1.$$

## 6. Boundary Conditions

In the previous two sections, we did not show how to get the integer, wavelet transform at the boundaries of signals. However, for all of the examples given above, the boundaries have been considered. There are two issues dealing with boundary filtering if we use the *Lifting scheme* or the *Correction method* to generate the integer wavelet transformations. The first is how to process the boundaries which occur in the start-up wavelet transformations. The second is how to deal with the boundaries in the deductive formula. If the boundaries in the start-up wavelet transform have already been established, then those in the upgrading formula are easy to establish. In fact, for the *Lifting scheme*, the boundaries in both steps should be processed in the same way. While, for the Correction *method*, it is easy to see from (5.3)-(5.4) that one has more choices to process boundaries in the second step. Therefore, the only thing we need to discuss here is the process by which the boundaries in the start up wavelet transformations are established. Assume we begin with compact supported biorthogonal wavelets.

Suppose the original signal is $\{c_n^0\}_{n=0}^N$. For creating integer biorthogonal wavelet transformations we can use the following symmetric extension [7]:

(1). If current biorthogonal filters have even length, we extend the boundaries of the signal as $c_{-k}^0 = c_{k-1}^0$, $k = 1,2,\cdots$;

(2). If the filters have odd length, we do the extension as $c_{-k}^0 = c_k^0$, $k = 1,2,\cdots$

Example 1 through 5 use the boundaries give above. In Example 6, the start up wavelet transform uses the above boundaries, but in the upgrading step, another boundary filtering is used. In addition, for arbitrarily sized images or signals, one can use the same technique which we described in the above examples to deal with this condition.

## 7. Some Applications

Before talking about any applications of the integer wavelet transform given above, we first prove that a nice *property of precision preservation (PPP)*, which is similar to the one mentioned in Section 2, holds for both the Lifting and Correction upgrading technique. This property is very important for many applications.

Lemma 7.1 Suppose that our integer wavelet transform starts with a pair of biorthogonal filters with the *PPP* property discussed in Section 2, that is, (4.3) and (4.6) possess this property. Then, the same property will be preserved in the whole algorithm if we adopt the Lifting scheme to be the upgrading formula.

In other words, Lemma 7.1 states if we only use the working units with the same precision as the original signal or image to calculate the wavelet transform developed in Section 4, the equations (4.8) and (4.6) are still the backward operations of the equations (4.3) and (4.7).

*Proof.* Assume that we only use $q$ bits to represent images or signals, say, the range of the pixel values is within $[-2^{q-1}, 2^{q-1}-1]$ . According to the hypothesis of the lemma, the equations (4.3) and its inverse (4.6) have the *PPP* property. Therefore, what we have to verify here is that the equation (4.7) and its inverse (4.8) can preserve the same property. We rewrite (4.7) and (4.8) as follow:

$$c_k^1 = c_k^{1,0} + b_k, \tag{7.1}$$

and its inverse

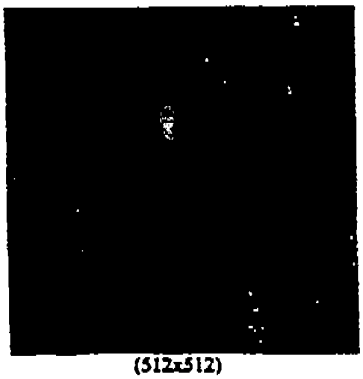
$$c_k^{1,0} = c_k^1 - b_k. \tag{7.2}$$

Here,

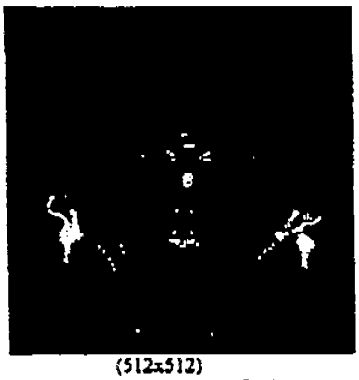
$$b_k = \mathrm{Int}\left(\frac{\alpha_r}{\alpha_d}\sum_i d_{k-i}^1 s_i\right).$$

**Application 1.** Lossless image compression

As mentioned at the beginning of this paper, the integer wavelet transformation established by the techniques described in this paper can always be used for lossless image compression because of the reversible ability. Especially, we can use the *PPP* property discussed in Lemma 7.1. We have used this wavelet lossless technology (WLT) for gray scale lossless image compression, and we have tried several images. For most natural images, the size of wavelet lossless compressed images is much smaller than corresponding GIF images. Figure 1 through 4 give some examples. Figure 1 is a standard image for compression, Figure 2 and 4 are X-ray images and Figure 3 is a two-value image but we treat it as a 8 bit gray scale image in order to compare with the GIF format. In fact, if we convert Figure 2 to a binary image, better result can be obtained by the JBIG technique.



(512x512)
**Figure 1.** Compression Ratio
WLT: 1.9:1/GIF: 1.05:1



(512x512)
**Figure 2.** Compression Ratio:
WLT 4.5:1/GIF: 2.72:1

"Visioneer may have come up with on
against the paper blizzard. . .gets pile
way to others throughout your compar

**Figure 3.** Compression Ratio: WLT: 20.8/ GIF 17.8 (152x794)

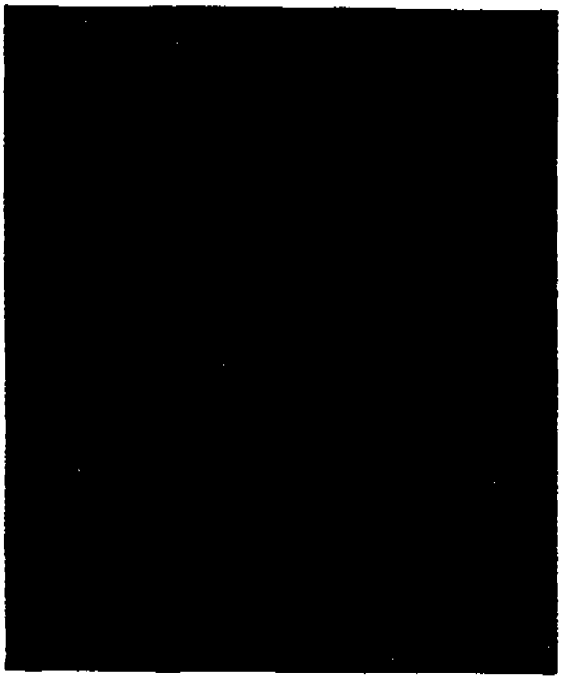


**Figure 4.** Compression Ratio: WLT 3.8:1/GIF 1.98:1 (1232x1024)

**Application 2.** Large scale medical image compression

Usually, 12 bits are used to represent one pixel in medical images. In this situation, the values of the pixels vary from 0 to 4095. Such images require careful treatment when a transform coding method is used for compression. If we use ordinary biorthogonal wavelets, the range of the transform coefficients will expand to $[-2^{16}, 2^{16}]$ when five levels of transform are used. Therefore, a longer working unit has to be employed, which consumes significant computer resources. However, the integer wavelet technique developed in this paper will solve this problem. For example, if we use the transforms given in Example 3, 5 and 6, the values of transform coefficients will be limited to the range of $[-2^{13}, 2^{13}]$. Even if we do not use the *PPP* property for these wavelets, *16* bits for the working unit is sufficient for all computations.

## 8. Conclusion

This paper has shown the processes necessary in order to obtain a non-linear, integer, biorthogonal, or non-biorthogonal reversible wavelet transform suitable for signal or image processing. We have shown how such a transform can be obtained either using the Lifting method, or the Correction method. For example, all interpolation wavelets can be modified to be corresponding integer wavelets without loosing any properties of original wavelets. In addition,

we have shown under certain conditions, the precision of the transform computation on the computer can remain at the same precision of the data, thus reducing the need for additional computer memory during the transform computation. These are extremely powerful techniques when the target data are large images, or the requirements establish a need for speed.

Although this paper establishes the structure for the integer transform based upon the biorthogonal wavelet or some non-biorthogonal wavelet, we do not imply the examples in this paper are necessarily the best wavelets for any particular application. However, we do claim if one is going to use such a technique, the ideas suggested in this paper will provide the best implementation.

**References**

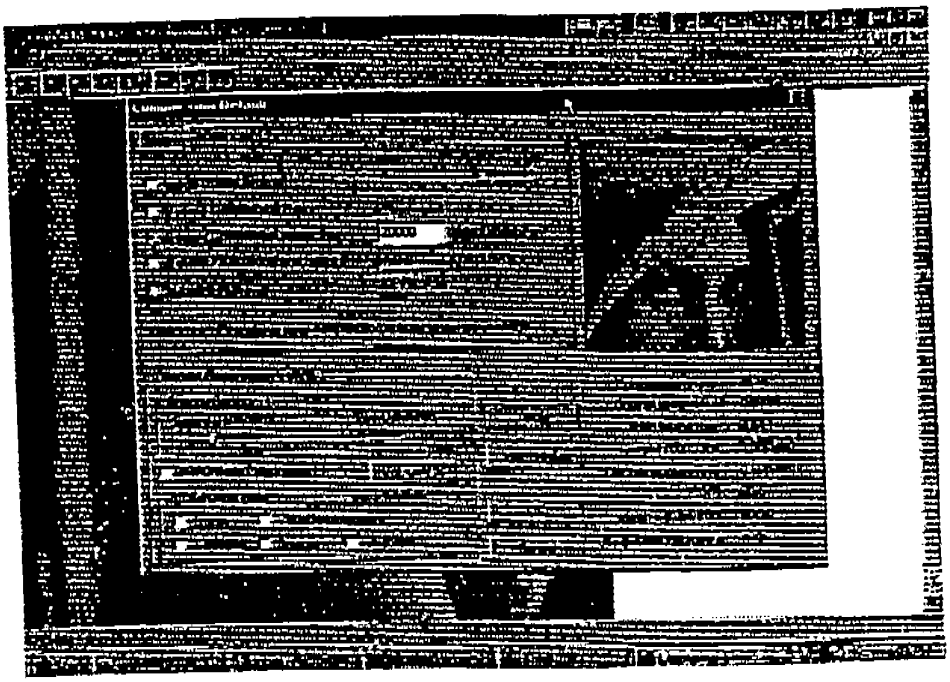
1. Wim Sweldens, *The lifting scheme: A custom-design construction of biorthogonal wavelets*, Applied and Computational Harmonic Analysis, Vol. 3, No. 2, April 1996.
2. A. Zandi, J. Allen, E. Schwartz and M. Boliek, *CREW: Compression with reversible embedded wavelets*, in IEEE Data Compression Conference, (Snowbird, Utah), pp.212-221, March 1995.
3. Amir Said, *An image multiresolution representation for lossless and lossy compression*, Submitted to the IEEE Transactions on Image Processing.
4. J. Villasenor, B. Belzer, and J. Liao, *Wavelet filter evaluation for image compression*, IEEE Trans. Image Processing, Vol. 4, pp.1053-1060.
5. G.R. Kuduvalli and R.M. Rangayyan, *Performance analysis of reversible image compression techniques for high-resolution digital teleradiology*, IEEE Trans. Med. Imaging, Vol. 11, pp. 430-445, Sept. 1992.
6. W.H. Press, B.P Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes: the Art of Science Programming*, Cambridge University Press, Cambridge, New York, 1986.
7. G. Strang and Truong Nguyen, *Wavelets and filter banks*, Wellesley-Cambridge Press, 1996.

# INFINITRON

Windows Image Compressor V3.0

Lightning Strike

**Announcing, Lightning Strike™ Image Compressor (LSIC) version 3.0, a Windows 95 tool that compresses still images from 50:1 to 200:1 using INFINITRON's proprietary wavelet technology. LSIC is a versatile, easy to use tool for Web and Graphic designers that can handle a wide variety of digital image formats, and it includes filters and convenient web tools. Images can be compressed 3 to 5 times more than JPEG, while maintaining similar or better image fidelity. Images can be viewed in 2 to 4 seconds over the Internet rather than 10 to 20 seconds for images compressed under JPEG. This has enormous benefits for reducing bottlenecks on corporate networks and the web, and in addition, requires less storage space.**



**Lightning Strike Features**

**Compression.** Images can be compressed as high as 200:1 using wavelet technology.

**Compression Control.** An EASY mode allows the user to compress images with minimal input, requiring only a decision between more quality or more compression. An ADVANCED mode enables the user to select; 1) image file size, 2) compression ratio, 3) PSNR, or 4) master level. Web designers will like the one step process to control the size of their image files, thus insuring the speed an image may be viewed on a browser.

**Non Uniform Compression.** Regions of an image can be selected for less compression to preserve a higher image quality while the rest of the image is compressed to the specified compression ratio. In this way important parts of a picture maintain crucial details while the over all picture file can be made as small as possible

**Post Reconstruction Filters.** Filters are available to enhance the reconstructed image. At compression time the user can preset a control to have these filters operate automatically during reconstruction. The filters include: quality improvement, sharpen (edge enhancement), smoothing, and brighten.

**Transparencies.** The user will have the ability to set pixels transparent so that a color in the background (already on the page) can be seen through the picture. This is useful for creative web site developers.

**Progressive Compression.** An image can be compressed so that when it is viewed it will appear quickly, first with low resolution, and then progressively building up in detail as it is downloaded. This insures the viewer does not lose interest while the image is downloaded.

INFINITRON

Windows Image Compressor V3.0

Lightning Strike



Cow, Bitmap, No Compression



Cow, Lightning Strike, 85:1 compression



Cow, JPEG 85:1 Compression

**Applications Lightning Strike Image Compression**

- Images on the Web
- Photo Stock
- Data Warehousing
- Catalogues
- Video Games
- CDs (Encyclopedias, Museums, Science, and Medicine)
- Archives (Art, History, Genealogy)
- Medical Imaging

**Performance**

Encode time typically less than 3 seconds for a 320 X 240 pixel, 24 bit color image on a 133 MHz. Pentium with 16 MB RAM.

**Minimum Recommended System**

Windows 95/NT OS
Pentium 100 MHz, 8 MB RAM
2 MB for program files
10 MB plus to swap image files

**Ancillary INFINITRON Products**

- Netscape Navigator Plug-in
- Java Applet
- ActiveX Control
- Web Site Image Converter
- Lightning Strike SDK
- GML Banner Generator

**About INFINITRON, Inc.**

Founded in 1992, INFINITRON, Inc. is a private company that specializes in the design and marketing of high quality image and video compression solutions for a wide array of markets. INFINITRON is based in Vancouver, BC with labs in Regina, Saskatchewan and Denton Texas.

Download a FREE demo version of Lightning Strike Windows Compressor from:
www.infinitron.com

INFINITRON USA Office 3401 East University, #104, Denton, TX. 76208
Tel: 817.484.1165 FAX: 817.484.0588

INFINITRON Canada Office 10th Flr 1199 W. Hastings, Vancouver, BC. V6E 3T5
Tel: 604.688.9789 FAX: 604.688.9798

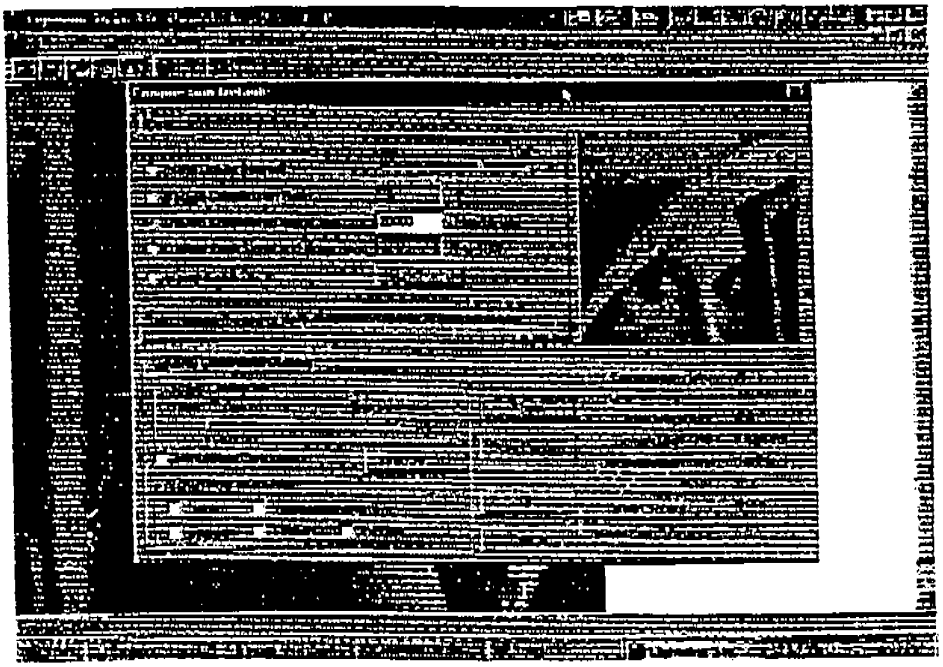
INFINITRON Research International, Inc., All Rights Reserved

# INFINITRON

*windows image Compressor V3.0*

*Lightning Strike*

**Announcing, Lightning Strike Image Compressor (TM) (LSIC) version 3.0, a Windows 95 tool that compresses still images from 50:1 to 200:1 using Infinitron's proprietary Wavelet technology. LSIC is a versatile, easy to use tool for Web and Graphic designers that can handle a wide variety of digital image formats and includes filters, and convenient web tools. Images can be compressed to files 4 times smaller than JPEG, while maintaining similar or better image fidelity. Images can be viewed in 2 to 5 seconds over the internet rather than 10 to 20 seconds for images compressed under JPEG. This has enormous benefits for reducing bottlenecks on corporate networks and the web, and in addition, requires less storage space.**



**Lightning Strike Features**

**Compression.** Images can be compressed to ~~over 100:1~~ using Wavelet Technology.

**Compression Control.** An EASY mode allows the user to compress images with minimal input, requiring only a decision between more quality or more compression. An ADVANCED mode enables the user to select; 1) image file size, 2) compression ratio, 3) PSNR, or 4) master level. Web designers will like the one step process to control the size of their image files, thus insuring the speed an image may be viewed on a browser.

**Non Uniform Compression.** Regions of an image can be selected for less compression to preserve a higher image quality while the rest of the image is compressed to the specified compression ratio. In this way important parts of a picture maintain crucial details while the over all picture file can be made as small as possible

**Post Reconstruction Filters.** Filters are available to enhance the reconstructed image. At compression time the user can preset a control to have these filters operate automatically during reconstruction. The filters include: quality improvement, sharpen (edge enhancement), smoothing, and brighten.

**Transparencies.** The user will have the ability to set pixels transparent so that a color in the background (already on the page) can be seen through the picture. This is useful for creative web site developers.

**Progressive Compression.** An image can be compressed so that when it is viewed it will appear quickly, first with low resolution, and then progressively building up in detail as it is downloaded. This insures the viewer does not loose interest while the image is downloaded.

# INFINITRON

*windows Image Compressor V3.0*



*Lightning Strike*



Lenna, JPEG 115:1 Compression

Lenna .bmp, No Compression



**Applications Lightning Strike Image Compression**

- Images on the Web
- Photo Stock,
- Data Warehousing
- Catalogues.
- Video Games
- CDs (Encyclopedias, Museums, Science, and Medicine)
- Archives (Art, History, Genealogy)
- Medical Imaging

**Performance**

Encode time typically less than 3 seconds for a 320 X 240 pixel, 24 bit color image on a 133 MHz. Pentium with 16 MB RAM.

**Minimum Recommended System**

Windows 95/NT OS
Pentium 100 MHz, 8 MB RAM
2 MB for program files
10 MB plus to swap image files

**Ancillary INFINITRON Products**

- Netscape Navigator Plug-in
- Java Applet
- ActiveX Control
- Web Site Image Converter
- Lightning Strike SDK
- GMI. Banner Generator

Ahout INFINITRON, Inc.

Founded in 1992, INFINITRON, Inc. is a private company that specializes in the design and marketing of high quality image and video compression solutions for a wide array of markets. INFINITRON is based in Vancouver, BC with labs in Regina, Saskatchewan and Denton Texas.



Download a FREE demo version of Lightning Strike Windows Compressor from:
www.infinitron.com

INFINITRON USA Office 3401 East University, #104, Denton, TX. 76208
Tel: 817.484.1165 FAX: 817.484.0588

INFINITRON Canada Office 10th Flr 1199 W. Hastings, Vancouver, BC, V6E 3T5
Tel: 604.688.9789 FAX: 604.688.9798

IPR2023-00330 Page 00674

**INFINITRON**
*Lightning Strike*



**Product Fact Sheet**
*Windows Compressor*

## Compressor Version 3.0

The Lightning Strike Compressor is a Windows tool that compresses still images from a wide variety of digital image formats using Infinitron's proprietary Wavelet algorithm. Images can be compressed to files 5 times smaller than JPEG, while maintaining similar or better image fidelity. Images can be viewed in 1 or 2 seconds over the internet rather than 10 to 20 seconds for images compressed under JPEG. This has enormous benefits for transmitting over corporate networks or the web, and in addition, saves space required for storing all those images.

Lightning Strike is a collection of tools in a user friendly environment. Two levels of user control are offered, one quick and easy for most applications, the other a master level for the advanced user who wishes to control parameters to maximize image quality.

The compression approach used by Lightning Strike is based upon integer wavelets. This technology is acknowledged by leading experts as a superior compression technique as compared to discrete cosine transform used in JPEG.

## Lightning Strike Windows Compressor Features

### Image Compression Options and Control

#### Compression Technique Options

Both Infinitron's Wavelet Compression and other frequently used compression methods are included in the product so users need only have Lightning Strike on their work station to perform all image compressions. Images can be compressed to Wavelet, JPEG, PNG, and GIF.

#### Compression Quality Versus Speed Options

The user can select one of two encoding processes that trade quality for speed of compression and ease of use. With the "Advanced" option selected, the optimum compression parameters are set by the user to give the best possible images for selected compression ratio. With "Easy" selected you get the fastest compression without having to know details of parameter selection.

#### Compression Ratio Control

The compressed image file size or compression ratio may be specified rather than the quality factor. This enables a web designer to control the size of their image files or the speed an image may be viewed, in a one step process.

#### Region of Interest Focusing

Regions of an image can be selected for less compression to preserve a higher image quality while the rest of the image is compressed to the specified compression ratio. In this way important parts of a picture maintain crucial details while the over all picture file can be made as small as possible. This is also known as Non-Uniform Compression.

**INFINITRON**
*Lightning Strike*



**Product Fact Sheet**
*Windows Compressor*

### Split and Merge

Very large images, which could not otherwise be compressed due to their large size, can be split into smaller images and compressed individually. This process has the side advantage of using RAM more effectively speeding time for compression. The split images can be reassembled using the merge aspect of the feature.

### Post Reconstruction Filters

Filters are available to enhance the reconstructed image. At compression time the user can preset a control to have these filters operate automatically during reconstruction. The filters include; quality improvement, sharpen (edge enhancement), smoothing, and brighten.

### Transparencies

The user will have the ability to set pixels transparent so that a color in the background (already on the page) can be seen through the picture. This is useful for creative web site developers. This gives the ability to display pictures other than the rectangular shape allotted on the web page, i.e. circles, polygons etc.. Also, designers often use this feature for shadowing, letters and objects.

### Progressive Decompression

An image can be compressed so that when it is viewed it will appear quickly, first with low resolution, and then progressively building up in detail as it is downloaded. This insures the viewer does not loose interest while the image is downloaded.

INFINITRON
*Lightning Strike*

**Product Fact Sheet**
*Windows Compressor*

**Image Comparisons**

*Picture of Lena, with no Compression (512 X 512 Image)*



INFINITRON
*Lightning Strike*

**Product Fact Sheet**
*Windows Compressor*

*Picture of Lena Compressed 100:1 with Lightning Strike*



**INFINITRON**
*Lightning Strike*



**Product Fact Sheet**
*Windows Compressor*

## File Functions

### PNG File Structure
The compressed images are stored in file format compliant with the PNG standard. In the future this file format will replace the GIF format used today.

### Batch Compress
The user is able to compress many images at once by adding or deleting image files (or paths) to a list box.

### Image Statistics
The compressor stores image statistics on each compressed image which may be viewed by the user. The following information is provided: image dimensions, compression ratio, file sizes, MSE, PSNR, maximum pixel difference, compression and decompression times.

## Performance and System Requirements

### Encoding and Decode Time.
The typical time to encode or decode a 320X240, 24 bit color image is 1 second on a Pentium running at 133 MHz with 16 Meg RAM.

### Minimum Recommended System
The minimum system requirements for an IBM PC Compatible are:

| | |
|---|---|
| Hard Disc Drive | 2 Mbytes free for program files. 10 Mb plus to swap image files. |
| Operating System | MS Windows 3.1(Win32)/ 95/ NT |
| RAM | 8 Mbytes |

This software is also available on the Apple MAC, Solaris, and UNIX platforms.

INFINITRON
*Lightning Strike*



**Product Fact Sheet**
*Windows Compressor*

## Ancillary Infinitron Products

### Netscape Navigator Plug-in
Netscape plug-ins are available for the Mac68k, Mac PPC, Widows 3.1 and 95/NT.

### Java Applet
Java Applets are available for the Mac68k, Mac PPC, Widows 3.1 and 95/NT.

### ActiveX Control
The Lightning Strike decompression software is available for such applications as Microsoft's Internet Explorer, as an ActiveX control.

### Web Site Image Converter
This utility will automate the conversion of web pages from JPEG to the Lightning Strike format. The utility searches an HTML file and replicates it replacing any JPEG image tags with Lightning Strike tags and converting the JPEG image files to Lightning Strike. The utility can follow link tags to recursively convert and replicate an entire web site or sub-section of a web site to the Lightning Strike format. This utility will be available for Windows NT and most flavors of the UNIX operating system.

### Lightning Strike Software Developers Kit
Using the SDK, a developer can integrate the highly efficient Lightning Strike module libraries into their own applications.



Download a FREE demo version of Lightning Strike Windows Compressor from: www.infinitron.com

Infinitron USA Office — 3401 East University, #104, Denton, TX. 76208
Tel: 817.484.1165 FAX: 817.484.0588

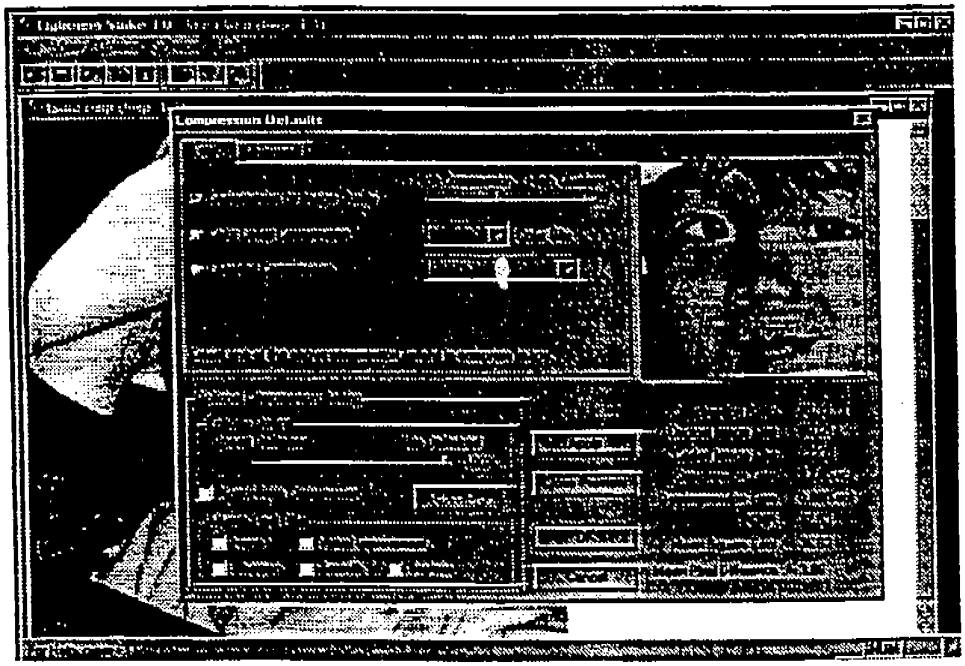
Infinitron Canada Office — 10th Flr 1199 W. Hastings, Vancouver, BC, V6E 3T5
Tel: 604.688.9789 FAX: 604.688.9789

# INFINITRON

## *Windows Image Compressor V3.0*

**Announcing, Lightning Strike™ Image Compressor (LSIC) version 3.0, a Windows 95 tool that compresses still images from 20:1 to 200:1 using INFINITRON's proprietary wavelet technology. LSIC is a versatile, easy to use tool for Web and Graphic designers that can handle a wide variety of digital image formats, and it includes filters and convenient web tools. Images can be compressed at ratios well in excess of present JPEG ratios while maintaining comparable image fidelity. This translates to much shorter image down load time on the web. This has enormous benefits for reducing bottlenecks on corporate networks and the web, and in addition, requires less storage space.**

*Lightning Strike*



**Lightning Strike Features**

**Compression.** Uses a proprietary integer wavelet.

**Compression Control.** An EASY mode allows the user to compress images with minimal input, requiring only a decision between quality and compression. An ADVANCED mode enables the user to select: 1) image file size, 2) compression ratio, 3) PSNR, or 4) master level for professionals where every parameter can be altered. We also provide the highest, wavelet lossless compression for users wishing this capability. Web designers will like the one step process to control the size of their image files allowing control over the delivery time of an image over a network.

**Non Uniform Compression.** Regions can be selected for less compression to preserve image quality while the rest of the image is compressed to the specified compression ratio. In this way, important parts of a picture maintain as small as possible

**Post Reconstruction Filters.** Filters are available to enhance the reconstructed image. At compression time the user can preset a control to have these filters operate automatically during reconstruction. The filters include: visual quality improvement, sharpen (edge enhancement), smoothing, and brighten.

**Transparencies.** The user will have the ability to set pixels transparent, so that a color in the background (already on the page) can be seen through the picture. This is useful for creative web site developers.

**Progressive Compression.** An image can be compressed so that when it is viewed it will appear quickly, first with low resolution, and then progressively building up in detail as it is downloaded. This insures the viewer does

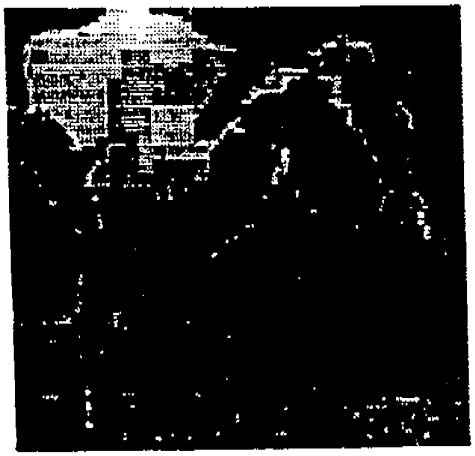
# INFINITRON

## *Windows Image Compressor V3.0*

*Lightning Strike*



Cow. Bitmap. No Compression



Cow. JPEG 85:1 Compression



Cow. Lightning Strike, 85:1 compression

**Applications for Lightning Strike Image Compression**

- Images on the Web
- Photo Stock
- Data Warehousing
- Catalogues
- Video Games
- CDs (Encyclopedias, Museums, Science, and Medicine)
- Archives (Art, History, Genealogy)
- Medical Imaging

**Performance**

Encode time typically less than 2.5 seconds for a 640 X 480 pixel, 24 bit color image on a 133 MHz. Pentium with 16 MB RAM. Decode time is less than .75 seconds.

**Minimum Recommended System**

Windows 95/NT OS
Pentium 100 MHz, 8 MB RAM
2 MB for program files
10 MB plus to swap image files

**Ancillary INFINITRON Products**

- Netscape Navigator Plug-in
- Java Applet
- ActiveX Control
- Web Site Image Converter
- Lightning Strike SDK
- GML Banner Animation/Compression
- Black and White Image Compression

**About INFINITRON, Inc.**

Founded in 1992, INFINITRON, Inc. is a private company that specializes in the design and marketing of high quality image and video compression solutions for a wide array of markets. INFINITRON is based in Vancouver, BC with offices in Regina, Saskatchewan and Denton Texas.



Download a FREE demo version of Lightning Strike Windows Compressor from:
www.infinitron.com

INFINITRON USA Office 3401 East University, #104, Denton, TX. 76208
Tel: 817.484.1165 FAX: 817.484.0586

INFINITRON 10th Flr 1199 W. Hastings, Vancouver, BC, V6E 3T5

**What is claimed is:**

1. A method for compressing an image, comprising the steps of:

performing a wavelet transformation of the image;

quantizing the wavelet transformed image;

applying entropy coding to the quantized image; and

outputting a file that includes the entropy coded image.

2. The method of claim 1, further comprising the following step:

performing a color transformation of the image.

3. The method of claim 1, further comprising the following step:

performing the wavelet transformation using an integer wavelet transform.

4. The method of claim 3, further comprising:

deriving the integer wavelet transform using a lifting scheme.

5. The method of claim 3, further comprising:

deriving the integer wavelet transform using a correction method.

6. The method of claim 1, wherein the step of quantizing includes the sub-step of:

processing the wavelet transformed image using sub-band oriented quantization.

7. The method of claim 1, further comprising:

comparing the wavelet transformed image to at least one predetermined threshold value.

8. A method for wavelet-based image compression using reduced color components, comprising the steps of:

creating a color table for an input image having a plurality of pixels;

calculating an index for each of the pixels, whereby generating a plurality of indices;

performing a wavelet transformation on the indices;

applying entropy coding on the transformed indices; and

outputting a file that includes the entropy coded indices.

9. The method of claim 8, further comprising:

dithering the pixels to generate the indices.

10. The method of claim 8, further comprising:

partitioning a large image into a plurality of small images to produce the input image.

11. The method of claim 10, wherein the large image is selectively partitioned.

12. An image processing system, comprising:

means for performing a wavelet transformation on an input image;

means for quantizing the wavelet transformed image;

means for entropy coding to the quantized image; and

means for outputting the entropy coded image.

13. The image processing system of claim 12, further comprising:

means for receiving the entropy coded image;

means for entropy decoding the received image;

means for de-quantizing the decoded image; and

means for performing an inverse wavelet transformation on the de-quantized image to produce an output image.

14. The image processing system of claim 12, further comprising:

means for displaying the output image.

15. The image processing system of claim 12, further comprising:

means for transmitting the entropy encoded image over a communications medium.

16. An image compression system, comprising:

a compressor configured to generate a compressed image based on an integer wavelet transform derived using a technique selected from a lifting scheme and a correction method.

17. The image compression system of claim 16, wherein the compressor quantizes a wavelet transformed image to produce the compressed image.

18. The image compression system of claim 16, wherein the compressor entropy encodes a quantized image to produce the compressed image.

19. The image compression system of claim 16, wherein the compressor performs a color transformation to produce the compressed image.

20. An image decompression system, comprising:

a decompressor configured to generate a decompressed image based on an integer inverse wavelet transform derived using a technique selected from a lifting scheme and a correction method.

21. A computer-readable memory storing a computer program for directing a computer system to perform image compression, wherein the computer program implements steps for performing a wavelet transformation of an input image, quantizing the wavelet transformed image, applying entropy coding to the quantized image, and outputting a file that includes the entropy coded image.

22. A method of compressing a data file, comprising the steps of:

performing a wavelet transformation of the data file to provide a series of wavelet coefficients;

quantizing those wavelet coefficients which fall above a predetermined threshold value to provide a quantized series of wavelet coefficients; and

compressing the quantized series of wavelet coefficients to provide a compressed data file.

23. The method of claim 22 wherein the compressing step comprises the step of applying an entropy coding to the quantized series of wavelet coefficients.

24. The method of claim 23 wherein the entropy coding is selected from the group of arithmetic, Huffman, run length and Huffman run length combined.

25. The method of claim 23 further comprising the step of performing a color transformation of the data file prior to the wavelet transformation step.

26. The method of claim 25 wherein the quantizing step comprises sub-band orientation quantization.

27. The method of claim 26 wherein the wavelet transformation step comprises integer wavelet transformation.

28. The method of claim 22 further comprising the step of filtering the data file prior to the wavelet transformation step.

29. The method of claim 27 wherein the integer wavelet transformation comprises biorthogonal filter method.

30. The method of claim 27 wherein the integer wavelet transformation comprises the correction method.

31. A compressed data file comprising a wavelet transformation of a data file having a series of compressed, quantized wavelet coefficients, the quantized wavelet coefficients having a value above a predetermined threshold value to provide a quantized series of wavelet coefficients.

32. A program for compressing a data file comprising:

a routine for performing a wavelet transformation of the data file to provide a series of wavelet coefficients;

a routine for quantizing those wavelet coefficients which fall above a predetermined threshold value to provide a quantized series of wavelet coefficients; and

a routine for compressing the quantized series of wavelet coefficients to provide a compressed data file.

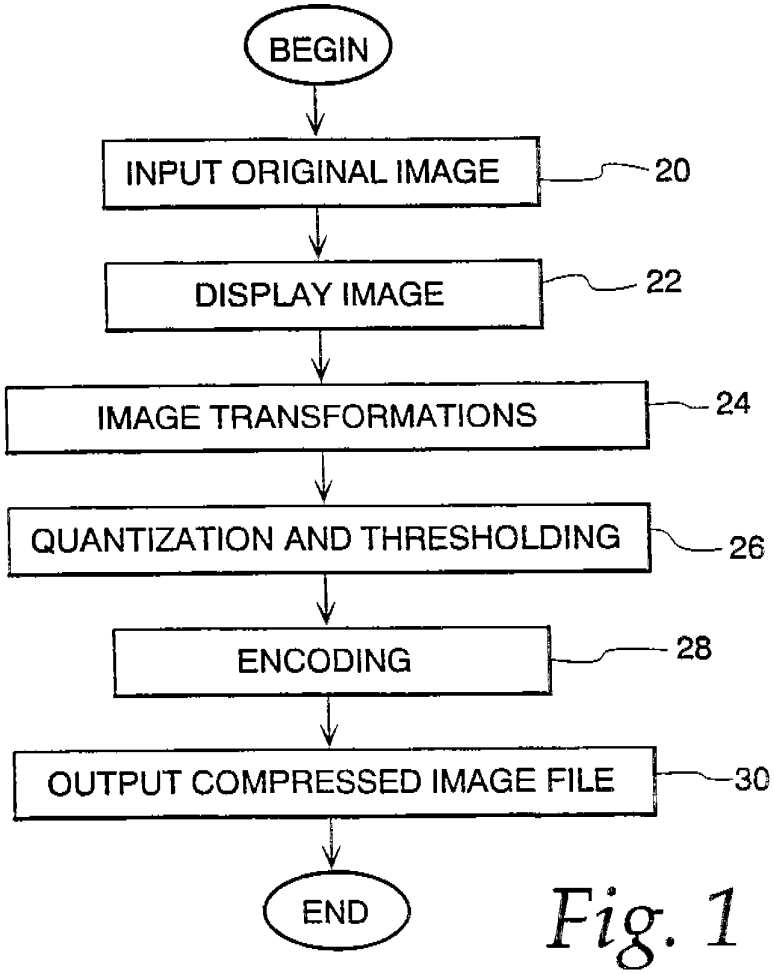

BEGIN
INPUT ORIGINAL IMAGE
20
DISPLAY IMAGE
22
IMAGE TRANSFORMATIONS
24
QUANTIZATION AND THRESHOLDING
26
ENCODING
28
OUTPUT COMPRESSED IMAGE FILE
30
END

*Fig. 1*

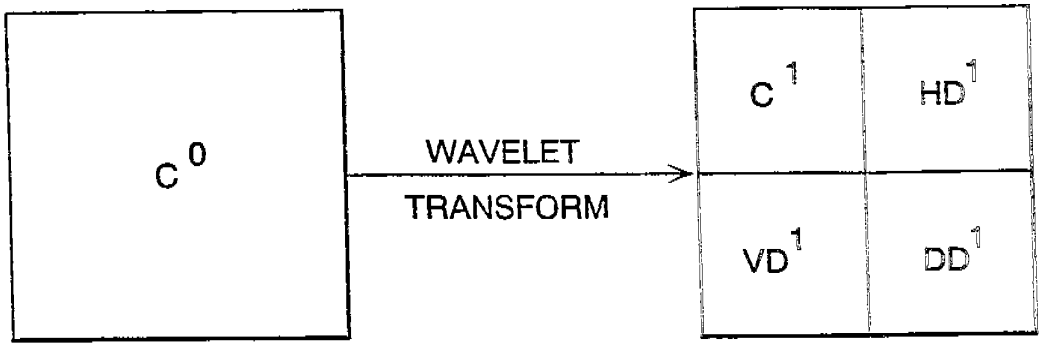

C 0
WAVELET
TRANSFORM
C 1
HD 1
VD 1
DD 1

*Fig. 2*

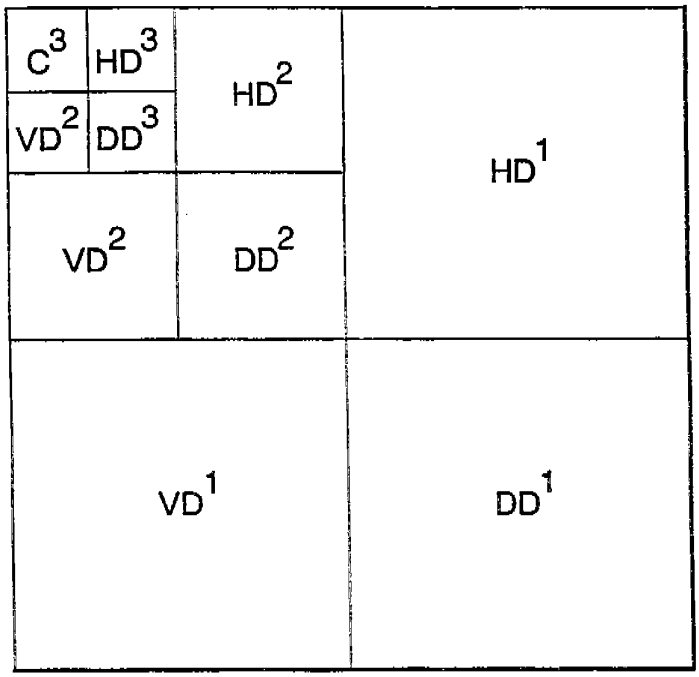

C3
HD3
HD2
VD2
DD3
HD1
VD2
DD2
VD1
DD1

*Fig. 3*

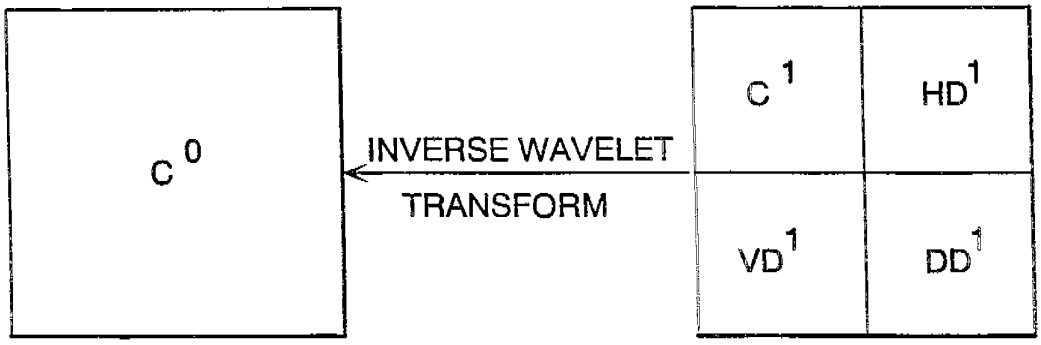

C 0
INVERSE WAVELET
TRANSFORM
C 1
HD1
VD1
DD1

*Fig. 4*

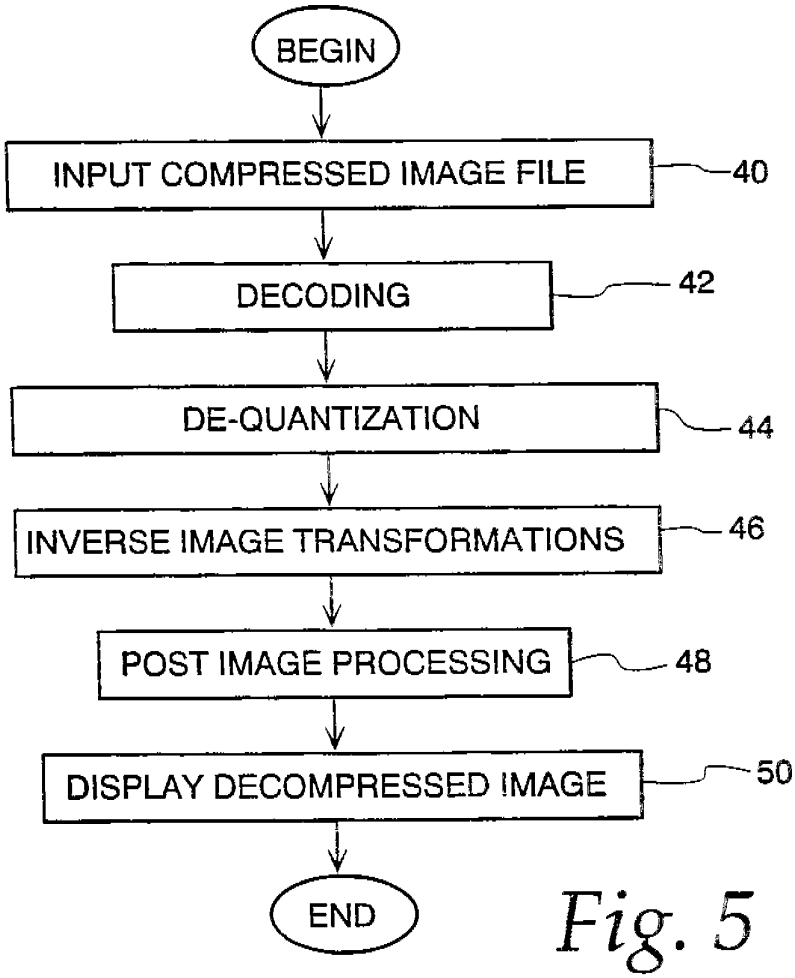

BEGIN
INPUT COMPRESSED IMAGE FILE
40
DECODING
42
DE-QUANTIZATION
44
INVERSE IMAGE TRANSFORMATIONS
46
POST IMAGE PROCESSING
48
DISPLAY DECOMPRESSED IMAGE
50
END

Fig. 5

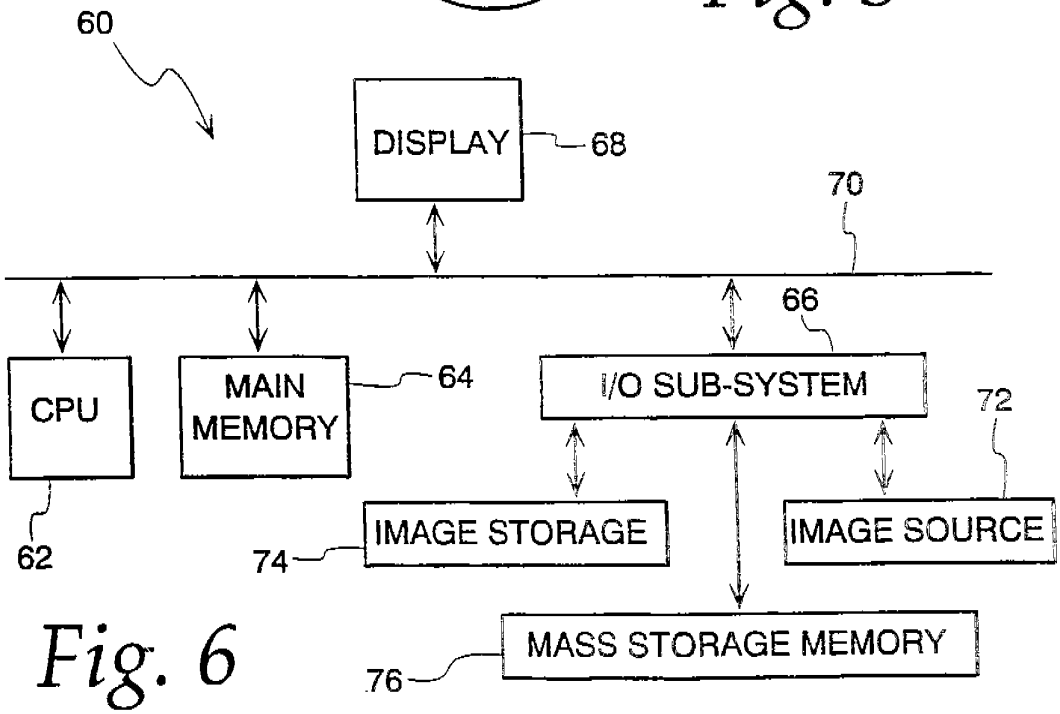

60
DISPLAY
68
70
66
CPU
MAIN MEMORY
64
I/O SUB-SYSTEM
72
62
74
IMAGE STORAGE
IMAGE SOURCE
MASS STORAGE MEMORY
76

Fig. 6

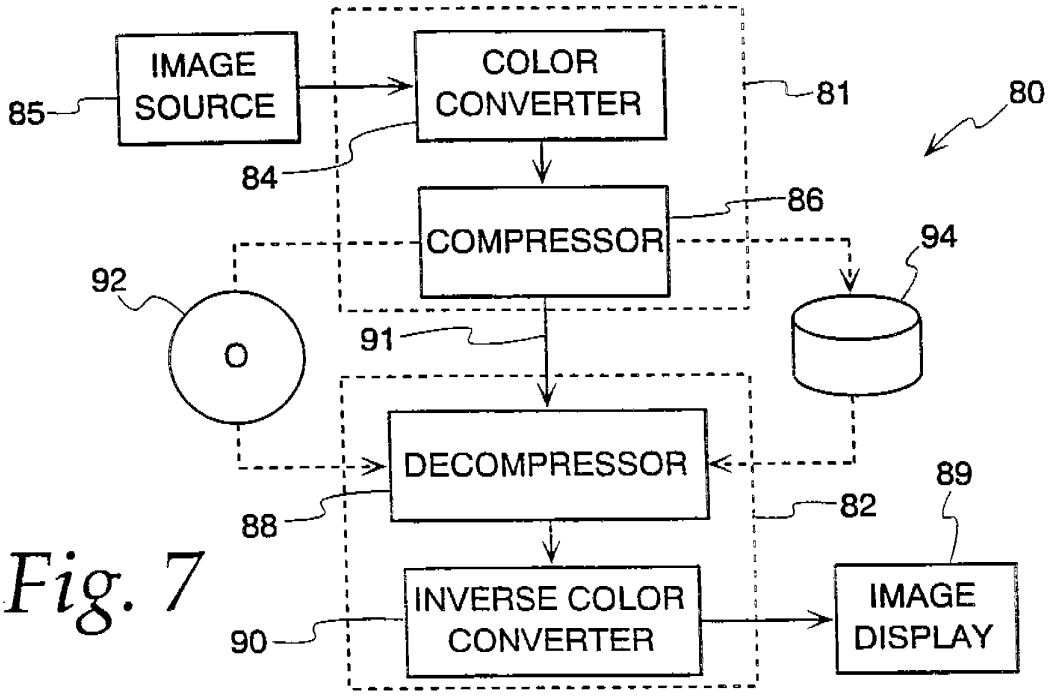

IMAGE SOURCE
85
COLOR CONVERTER
81
80
84
COMPRESSOR
86
94
92
O
91
DECOMPRESSOR
88
82
89
INVERSE COLOR CONVERTER
90
IMAGE DISPLAY

Fig. 7

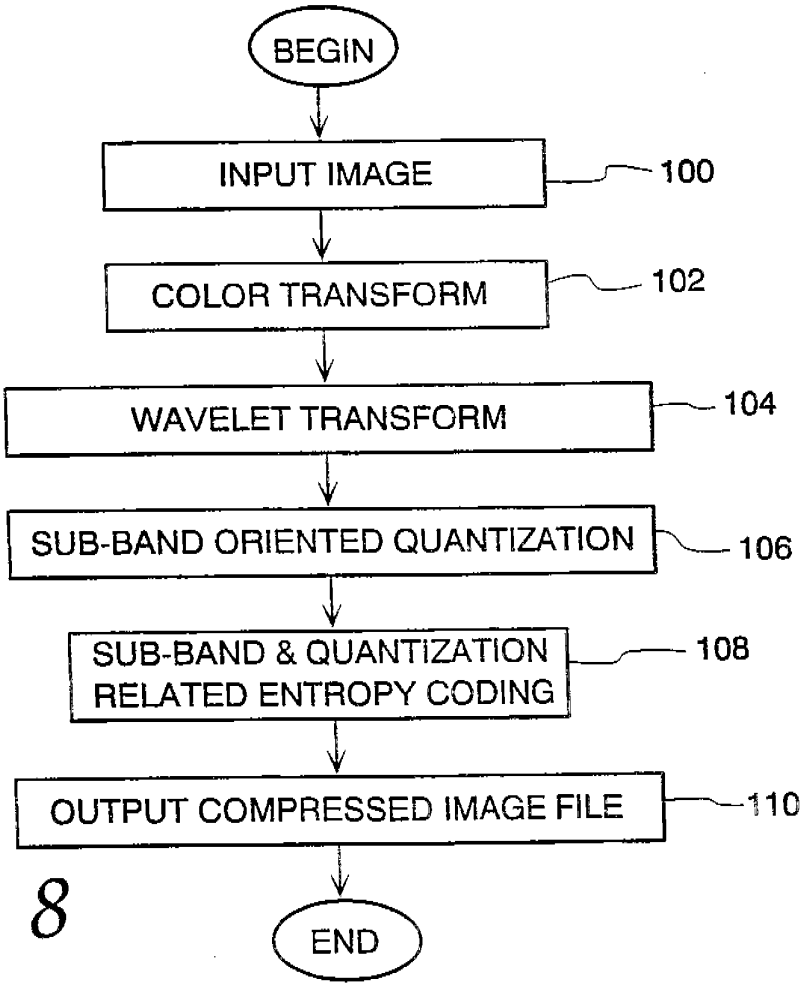

BEGIN
INPUT IMAGE
100
COLOR TRANSFORM
102
WAVELET TRANSFORM
104
SUB-BAND ORIENTED QUANTIZATION
106
SUB-BAND & QUANTIZATION RELATED ENTROPY CODING
108
OUTPUT COMPRESSED IMAGE FILE
110
END

Fig. 8

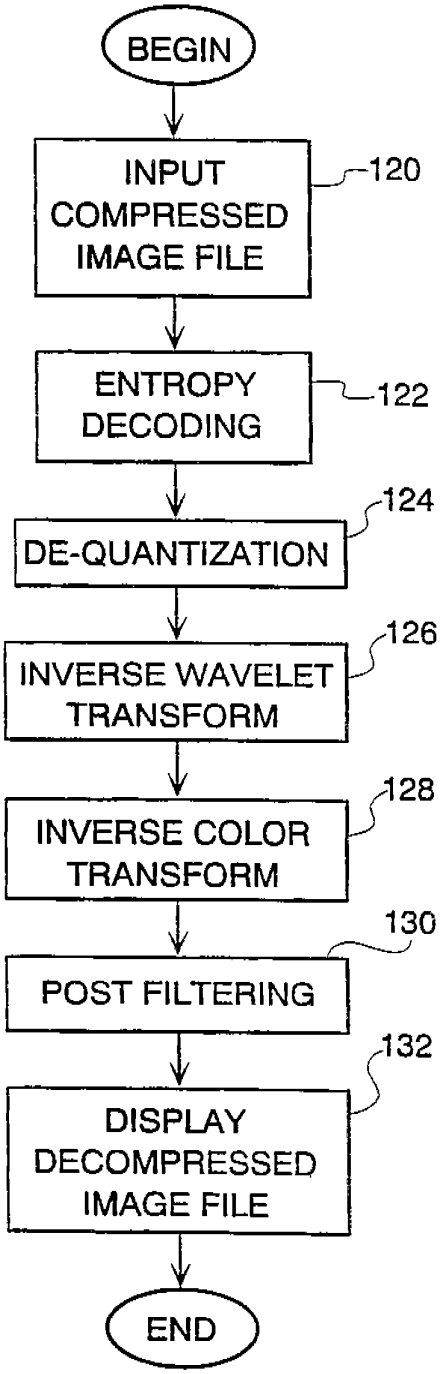

BEGIN
INPUT COMPRESSED IMAGE FILE
120
ENTROPY DECODING
122
DE-QUANTIZATION
124
INVERSE WAVELET TRANSFORM
126
INVERSE COLOR TRANSFORM
128
POST FILTERING
130
DISPLAY DECOMPRESSED IMAGE FILE
132
END

Fig. 9

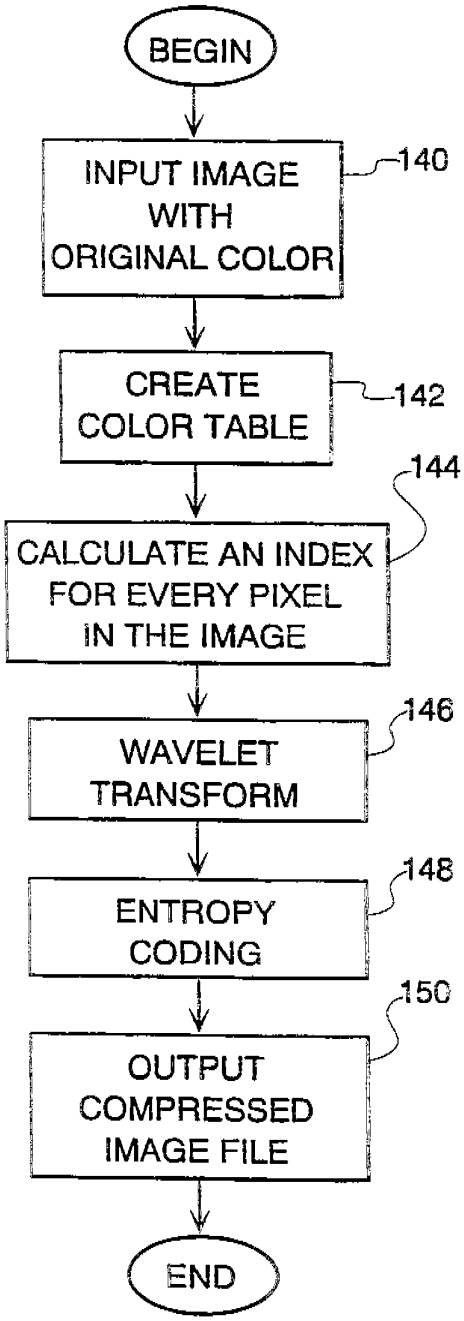

BEGIN
INPUT IMAGE WITH ORIGINAL COLOR
140
CREATE COLOR TABLE
142
CALCULATE AN INDEX FOR EVERY PIXEL IN THE IMAGE
144
WAVELET TRANSFORM
146
ENTROPY CODING
148
OUTPUT COMPRESSED IMAGE FILE
150
END

Fig. 10

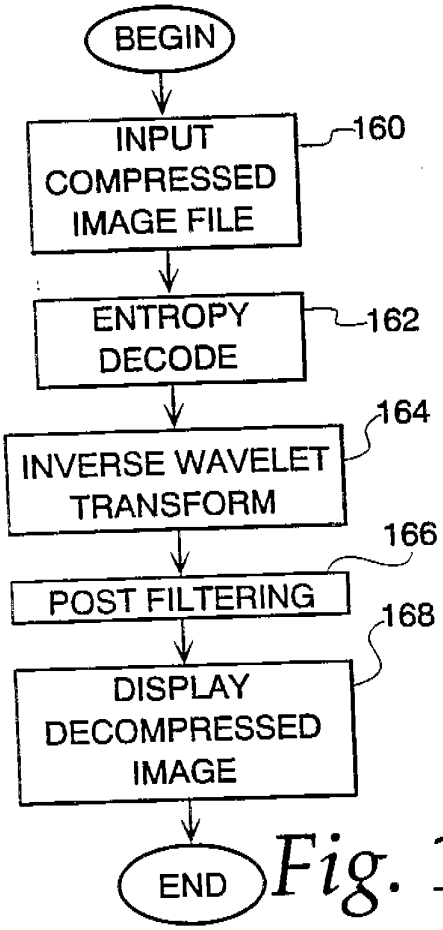



Fig. 11

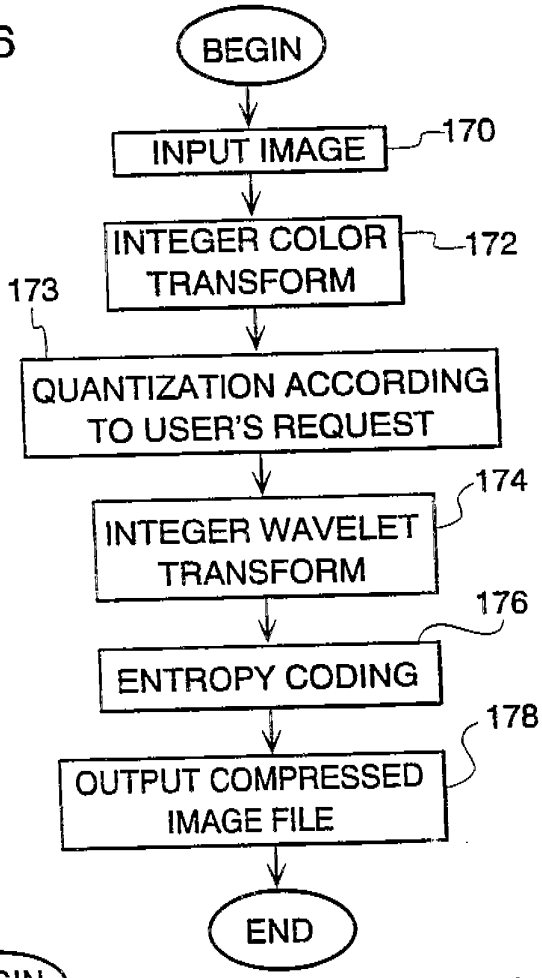



Fig. 12

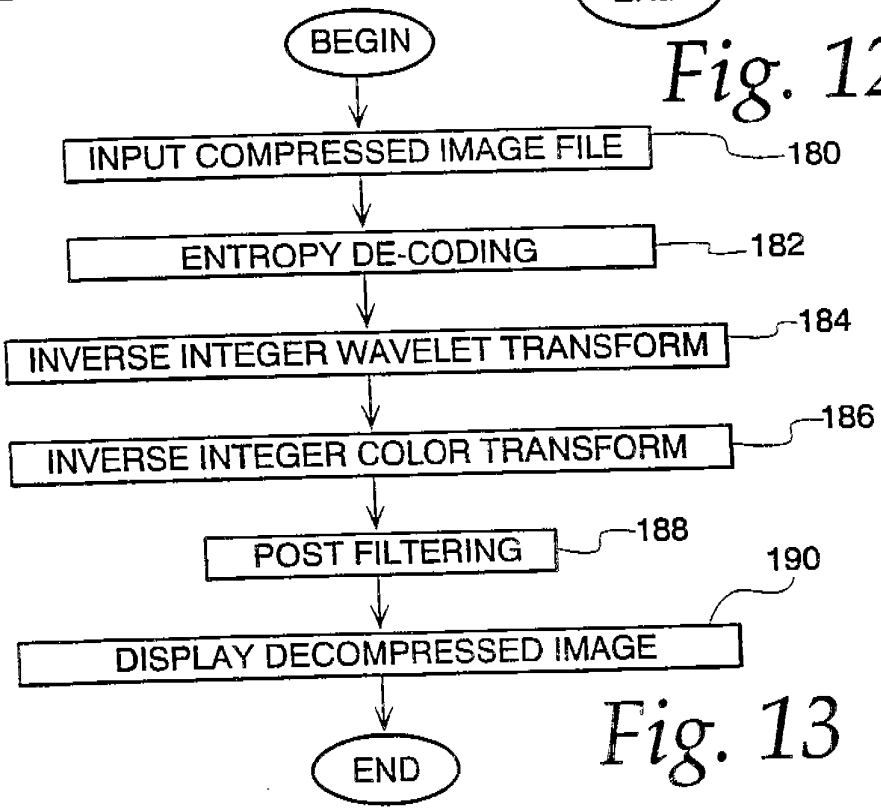



Fig. 13

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/04700

**A. CLASSIFICATION OF SUBJECT MATTER**

IPC(6) :G06K 9/00
US CL :Please See Extra Sheet.
According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 382/232, 233, 236, 238, 239, 240, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | US 5,495,292 A (ZHANG et al) 27 February 1996, col. 3, lines 1-68. | 1-32 |
| Y | US 5,414,780 A (CARNAHAN) 09 May 1995, col. 4, lines 1-68. | 1-32 |

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents: | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "E" | earlier document published on or after the international filing date | "Y" | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | "&" | document member of the same patent family |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 09 JUNE 1998 | 24 AUG 1998 |
| Name and mailing address of the ISA/US<br>Commissioner of Patents and Trademarks<br>Box PCT<br>Washington, D.C. 20231 | Authorized officer<br>LEO BOUDREAU |

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/04700

A. CLASSIFICATION OF SUBJECT MATTER:
US CL :

382/232, 233, 236, 238, 239, 240, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253

**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 13/38, 15/17

A1

(11) International Publication Number: **WO 98/43177**

(43) International Publication Date: 1 October 1998 (01.10.98)

(21) International Application Number: PCT/US98/05304

(22) International Filing Date: 19 March 1998 (19.03.98)

(30) Priority Data:

| | | |
|---|---|---|
| 60/041,366 | 25 March 1997 (25.03.97) | US |
| 08/925,275 | 8 September 1997 (08.09.97) | US |

(71) Applicant: INTEL CORPORATION [US/US]; 2200 Mission College Boulevard, P.O. Box 58119, Santa Clara, CA 95052–8119 (US).

(72) Inventors: TSO, Michael, Man–Hak; 5744 S.E. Preston Court, Hillsboro, OR 97123 (US). WILLIS, Thomas, G.; 619 S.W. Arboretum Circle, Portland, OR 97221 (US). RICHARDSON, John, W.; 2748 N.E. 19th Avenue, Portland, OR 97212 (US). KNAUERHASE, Robert, Conrad; 4926 S.W. Corbett Avenue #108, Portland, OR 97201 (US). MACIELINSKI, Damien; 415 S.W. 121st Place, Portland, OR 97225 (US).

(74) Agents: ALTMILLER, John, C. et al.; Kenyon & Kenyon, 1025 Connecticut Avenue, N.W., Washington, DC 20036 (US).

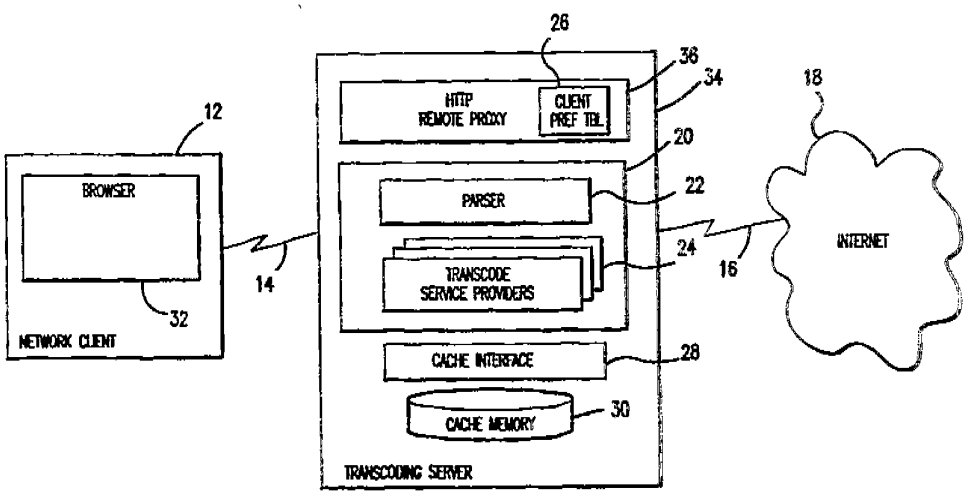
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

**Published**

*With international search report.*

*Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.*

(54) Title: SYSTEM FOR DYNAMICALLY TRANSCODING DATA TRANSMITTED BETWEEN COMPUTERS




(57) Abstract

A system for dynamically transcoding data transmitted between computers is implemented in an apparatus for use in transmitting data between a network server (10) and a network client (12) over a communications link (14). The apparatus includes a parser (22) coupled to a transcode service provider (24). The parser (22) is configured to selectively invoke the transcode service provider (24) in response to a predetermined selection criterion.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# SYSTEM FOR DYNAMICALLY TRANSCODING DATA TRANSMITTED BETWEEN COMPUTERS

## Background of the Invention

This application claims the benefit of U.S. Provisional Application No. 60/041,366, filed March 25, 1997.

## Field of the Invention

The present invention relates generally to the field of data communications for personal computers (PCs), and in particular to a system for dynamically transcoding data transmitted between two computers over a communications link.

## Related Art

The Internet is quickly becoming the preferred data communications medium for a broad class of computer users ranging from private individuals to large multi-national corporations. Such users now routinely employ the Internet to access information, distribute information, correspond electronically, and even conduct personal conferencing. An ever-growing number of individuals, organizations and businesses have established a presence on the Internet through "web pages" on the World-Wide Web (WWW).

For a wide variety of reasons, it may be desirable to manipulate data transmitted between a local client computer and a network server computer. For



IPR2023-00330 Page 00698

example, in certain instances it may be advantageous to dynamically add, modify or delete content retrieved from an Internet server computer before that content is provided to a client computer. Conversely, it may be advantageous to modify a content request from a client computer prior to transmitting the request to an Internet server computer. While such dynamic manipulation of requests and responses is desirable, it is impractical to expect the expansive Internet infrastructure to quickly change to accommodate such a new capability. For this reason, it is desirable to implement such new capabilities in a way that does not require changes to either existing client computers or Internet server computers.

It is known to deploy a proxy server, or network proxy, as an intermediary between one or more client computers and an external network such as the Internet. Network proxies are described generally in Ian S. Graham, HTML Source Book: A Complete Guide to HTML 3.0 403 (2d ed. 1996). One common application for a proxy server is as a so-called "firewall," wherein the proxy server is responsible for all communications with the outside world. In other words, local devices are not permitted to communicate directly with external network computers, such as Internet servers. Instead, each local device directs requests for network-resident data to the proxy server. When the proxy server receives such a request, it forwards the request to the appropriate external computer, receives the response from the external computer, and then forwards the response to the local device. The external computer thus has no knowledge of the local devices. In this way, the local devices are protected from potential dangers such as unauthorized access.

Existing proxy servers do not manipulate the data passing through them. In essence, proxy servers are merely blind conduits for requests and responses. This limitation of existing proxy servers restricts these devices from being used to full advantage when facilitating communications between local devices and network devices. There is therefore a need for a so-called "smart" proxy capable of examining the data passing through it, whether it be a request intended for an external network device or network content being returned to a local device, and dynamically acting



IPR2023-00330 Page 00699

upon that data. Such a device can be used to transparently provide a wide range of services that were heretofore impossible without modifying existing Internet infrastructure.

## Summary of the Invention

Embodiments of the present invention relate to devices, systems and methods for transcoding information transmitted between computers, such as a network server computer and a network client computer.

According to one embodiment, an apparatus for use in transmitting data between a network server and a network client over a communications link includes a parser coupled to a transcode service provider. The parser is configured to selectively invoke the transcode service provider in response to a predetermined selection criterion.

## Brief Description of the Drawings

**Fig. 1** is a schematic diagram illustrating an environment in which embodiments of the present invention may be applied.

**Fig. 2** is a schematic diagram illustrating a transcoder module according to an embodiment of the present invention.

**Fig. 3** is a schematic diagram illustrating an embodiment of the present invention for a non-enabled network client.

**Fig. 4** is a schematic diagram illustrating an example of a user interface for providing a non-enabled network client with control over transcoding functionality.

**Fig. 5** is a schematic diagram illustrating an embodiment of the present invention for an enabled network client.

**Fig. 6** is a schematic diagram illustrating a network client with transcoding functionality integrated in a browser according to an embodiment of the present invention.



IPR2023-00330 Page 00700

**Figs. 7-9** are flow charts illustrating logic for presenting a requested URL object to a network client according to an embodiment of the present invention.

**Detailed Description**

Embodiments of the present invention provide the ability to dynamically transcode information transmitted between, for example, a network server computer and a network client computer. As used herein, the term "transcode" applies to virtually any manipulation of data including, but not limited to, adding, modifying or deleting data.

Referring now to **Fig. 1**, which illustrates an environment in which embodiments of the present invention may be advantageously applied, a network server **10** manages the transfer of data from the Internet **18** to a network client **12**. Network client **12** may be any computer having suitable data communications capability.

Network client **12** communicates requests for information to, and receives information from, network server **10** over a client/server communications link **14**. Client/server communications link **14** may comprise, for example, a so-called "slow network" using, for example, POTS (Plain Old Telephone System) dial-up technology or wireless connections. Alternatively, client/server communications link **14** may comprise a so-called "fast network," such as a LAN or WAN (Wide Area Network), which is capable of operating at much higher speeds than are possible with slow networks. Combinations of these access methods are also possible. For example, network client **12** may use a POTS or wireless dial-up connection to a modem bank maintained by an ISP (Internet Service Provider), which is in turn connected to network server **10** over a LAN. Network server **10** communicates with computers resident on Internet **18** through server/network communications link **16**, which may comprise any suitable communications medium known in the art.



IPR2023-00330 Page 00701

According to a first general embodiment of the present invention, illustrated schematically in **Fig. 2,** a transcoder **20** includes a parser **22** and a plurality of transcode service providers **24**. Parser **22** is configured to act upon data received by transcoder **20**, such as a request for a network object generated by a client device or a reply to such a request provided by a content server device. In this particular embodiment, parser **22** is responsible for selectively invoking one or more of transcode service providers **24** based upon a predetermined selection criterion.

Transcoder **20** may be implemented, for example, as a software module installed in a network proxy, in a client device, in a network server device, or in a content server device. In one particular implementation, illustrated in **Fig. 3**, transcoder **20** is installed in a remote transcoding server **34** arranged between network client **12** and Internet **18**. Transcoding server **34** may comprise, or be a part of, a network server, a stand-alone computer in communication with a network server, or a distributed system of computers. Remote transcoding server **34** may be coupled, for example, to an ISP's network, a corporate network, or anywhere on Internet **18**, and may provide multiple users (i.e., clients) with a means to obtain content on Internet **18**.

In the particular embodiment illustrated in **Fig. 3**, transcoding server **34** includes an HTTP (HyperText Transfer Protocol) remote proxy **36**, capable of accessing Internet **18** over server/network communications link **16**. HTTP remote proxy **36** differs from known network proxies, which generally are little more than a conduit for requests to, and replies from, external Internet resources, in that it is capable not only of examining such requests and replies, but also of acting upon commands in the requests by, for example, determining whether or not to transcode content. Moreover, using transcoder **20**, HTTP remote proxy **36** is capable of changing content received from Internet **18** prior to returning it to a requesting network client **12**, as is explained further below.



IPR2023-00330 Page 00702

Looking more closely at the embodiment in **Fig. 3**, transcoder **20** is coupled to HTTP remote proxy **36**. Parser **22** manages the transcoding of data to be transmitted from transcoding server **34** to network client **12**. To this end, parser **22** controls transcode service providers **24** to selectively transcode content based on a predetermined selection criterion. For example, one or more transcode service providers **24** may provide the capability to compress and/or scale different types of data content, such as image, video, or HTML (HyperText Markup Language). Such uses are described further in co-pending U.S. patent applications Serial No. 08/772,164 entitled "System for Enhancing Data Access Over a Communications Link," filed on December 20, 1996, and Serial No. 08/799,654 entitled "Method and Apparatus for Scaling Image Data," filed on February 11, 1997, both of which are assigned to Intel Corporation. For purposes of illustrating certain features of the present invention, a number of embodiments are described below in terms of content scaling/compression; however, as is explained, transcode service providers **24** may provide a wide variety of transcoding functions.

As shown in **Fig. 3**, transcoding server **34** may also include a server-side cache memory **30** managed by a server-side cache interface **28**. Server-side cache memory **30** may be used to store both original and transcoded versions of content for later transmission to network client **12** without the need to re-retrieve the content from Internet **18** or to re-transcode the content.

Transcoding server **34** is coupled to network client **12** by client/server communications link **14**. Network client **12** includes a browser **32**, such as the Netscape Navigator v.3.0 browser (although the invention is not limited in this respect), which manages the presentation of data to a user. In this embodiment, network client **12** is "non-enabled," meaning no specialized transcoding software is preloaded on network client **12**.

Parser **22** may comprise a relatively simple, uniform interface to



IPR2023-00330 Page 00703

HTTP remote proxy **36**, and may provide an API (Application Programming Interface) for transcoding data received by HTTP remote proxy **36**. Parser **22** manages one or more transcode service providers **24** that are accessed through a common SPI (Service Provider Interface). In this particular embodiment, parser **22** is designed in compliance with the Windows Open Systems Architecture (WOSA), and may be implemented as a Win32 DLL (Dynamic Link Library). The WOSA architecture, described in Readings on Microsoft Windows and WOSA (Microsoft Corp. 1995), enables additional transcode service providers **24** to be dynamically added to the system to provide new features and/or better transcoding algorithms, while at the same time not requiring changing or retesting other software components in the system. This feature is especially beneficial where transcoding server **34** also interacts with "enabled" network clients equipped with specialized transcoding software. It should be noted that some of the features of parser **22** described below may be inapplicable to the non-enabled client embodiment of **Fig. 3**; however, transcoding server **34** may advantageously be configured flexibly enough to process requests from both non-enabled and enabled network clients.

Like parser **22**, server-side cache interface **28** may be modeled after a standard Get/Set interface. Server-side cache memory **30** essentially "owns" all cached objects, in that it manages the properties and storage of the objects and may invalidate any non-locked object at any time; however, the actual format of any given cached object is known only by parser **22** and its associated transcode service providers **24**. Thus, for data integrity and transcoding efficiency purposes, all access to server-side cache memory **30** in this embodiment is through parser **22**.

Server-side cache interface **28** may include the following calls:

```
CreateEntry(URL, &Entry, ...);
GetEntry(URL, &Entry);
CreateStream(Entry, &StreamEntry, ...);
GetStream(Entry, &StreamEntry, ...);
CloseEntry(Entry);
CloseStreamEntry(StreamEntry);
```



IPR2023-00330 Page 00704

```
GetProperties(Entry, &Properties, ...);
SetProperties(Entry, &Properties, ...);
Read(StreamEntry, &OutStream, ...);
Write(StreamEntry, &InStream, ...).
```

Unlike most cache memories, server-side cache interface **28** and server-side cache memory **30** enable maintenance of multiple representations of a given cached object, with descriptive information about each representation included in server-side cache memory **30**. In addition, server-side cache interface **28** and server-side cache memory **30** serve as a synchronization point for multi-threaded accesses to cached objects. It should be noted that the illustrated embodiment does not require any particular configuration for server-side cache interface **28** and/or server-side cache memory **30**. Indeed, functionality attributed to these components in the various embodiments described herein may be readily implemented in other system components.

The CreateEntry() call creates and returns a cache entry for a specified hypertext object. This call also creates an entry stream for an original version of the hypertext object. Similarly, the GetEntry() call obtains a cache entry for a hypertext object already existing in cache memory **30**. Both the CreateEntry() and GetEntry() calls set locks on associated cached objects until a CloseEntry() call is invoked. Once a lock is set, the cached object will not be replaced or invalidated by cache interface **28**, permitting one or more transcode service providers **24** to safely perform any required cache operations, such as object retrieval and/or storage.

After a cache entry is created or opened by a CreateEntry() or GetEntry() call, the CreateStream() or GetStream() calls may respectively create or open an extra stream entry for the cached object. Each extra stream entry is associated with a different transcoded version of the hypertext object, which may be retrieved or appended to by one of transcode service providers **24**. Stream-based processing of cached objects makes it possible for transcoding server **34** to begin transmitting a transcoded version of a hypertext object to a requesting network client



IPR2023-00330 Page 00705

12 even while transcode service provider 24 is appending additional transcoded content to that same version. Advantages of this stream-based processing include reducing user latency through incremental painting of objects and avoiding unnecessary idle time on client/server communications link 14, thereby providing users with a more responsive "feel."

The GetProperties() and SetProperties() calls retrieve and store information about cached objects, including information maintained by transcode service provider 24 used to determine transcoding properties and transcoding status of a cached object. Transcode service provider 24 may use such information, for example, to determine current compression progress for scaled data access and staged refinements.

The Read() call reads data from a specified cached object data stream. For example, transcode service provider 24 may invoke this call and tunnel stream data through HTTP remote proxy 36 directly to network client 12. The Write() call caches data from a new HTTP data stream. This call will append an incoming data stream received from, for example, a Web server or transcode service provider 24, to an opened cache stream which may be concurrently read using the Read() call.

In the present embodiment, parser 22 includes the following calls:

```
GetObject(URL, InParams, &OutParams, &OutStream, ...);
GetScaledObject(URL, InParams, &OutParams, &OutStream, Stage,
...);
PutObject(URL, InParamStruct, &InStream, &OutParams,
&OutStream, ...).
```

As detailed below, parser 22 uses these calls to manage the provision of requested content to network client 12.

The GetObject() call is used to service non-enabled client requests, and returns a non-transcoded (i.e., original) version of a specified hypertext object. In this embodiment, transcoding server 34 assumes that each HTTP request has a unique thread that may be blocked until the request is satisfied. Accordingly, the



IPR2023-00330 Page 00706

GetObject() call will block until it either returns the requested data stream or indicates failure with a cause (e.g., object does not exist). This ability to return a so-called standard hypertext object is advantageous for compatibility reasons, enabling embodiments of the present invention to be used with existing browsers that do not include support for certain transcoding functionality (e.g., advanced data compression), and enabling users to selectively retrieve non-transcoded versions.

The GetScaledObject() call is similar to GetObject(), and is also used to request an object from server-side cache memory **30**; however, it adds support for requesting a particular version of that object, such as a high-quality rendition. Unlike traditional caching proxies, transcode service providers **24** can use server-side cache memory **30** to store several different versions of an object to support clients with different communications and/or presentation capabilities. Thus, an additional "Stage" parameter may be used to indicate which version of the cached object is to be returned to network client **12**. Where transcode service provider **24** is configured to scale network content, it may use this parameter to request a version of a cached object having, for example, a default scaled quality, a refinement to a better-quality version, or the original non-scaled version.

In this embodiment, when network client **12** requests a hypertext object, HTTP remote proxy **36** uses either the GetObject() or GetScaledObject() call (depending on if network client **12** is capable of receiving scaled/transcoded datatypes) to retrieve the hypertext object from parser **22**. If the hypertext object is not found, parser **22** uses the CreateEntry() call to create an entry (in effect, a placeholder) in server-side cache memory **30** for the new object. The new entry is returned to HTTP remote proxy **36**, which requests the hypertext object from Internet **18**. As a data stream for the hypertext object is returned, HTTP remote proxy **36** calls parser **22** using the PutObject() call, passing into this call the new entry and the handle to the data stream to be placed into the entry. Parser **22** selects an appropriate transcode service provider **24** based, for example, on the content type



IPR2023-00330 Page 00707

of the data stream. In this context, the term content type encompasses a datatype, an HTTP MIME (Multipurpose Internet Mail Extensions) type, a content format, and so on. The selected transcode service provider **24** uses a separate thread to read the incoming data stream, transcode it, and place it within the entry of server-side cache memory **30**. The current thread immediately returns to HTTP remote proxy **36**, which once again calls GetScaledObject() (or GetObject()). This case will always result in a cache hit. This thread then works simultaneously with the separate thread in the PutObject() to tunnel data (either original or transcoded) from transcoding server **34** to network client **12**.

Multiple-thread processing improves the efficiency of the present embodiment by not waiting for a hypertext object to be received in its entirety by HTTP remote proxy **36,** or added in its entirety to server-side cache memory **30**, before beginning to send the object to network client **12**. Another benefit of multiple-thread processing is that parser **22** may efficiently process requests for the same hypertext object from multiple network clients **12**. The hypertext object need only be retrieved from Internet **18** once, and appropriate versions may be transmitted to such multiple network clients **12** concurrently. It should be noted, however, that embodiments of the present invention may be implemented without multiple-thread processing.

As noted above, parser **22** may selectively invoke one of transcode service providers **24** based upon satisfaction of a predetermined selection criterion. Such selection criterion may comprise, for example, information contained in a header portion of a data packet received by transcoding server **34**, such as a MIME type, a URL (Uniform Resource Locator), a last modified time indicator and so on. Alternatively, the predetermined selection criterion may comprise information contained in a data portion of such a data packet, such as particular content, key words, structures (for example, heading levels), and so on. Still further, the predetermined selection criterion may comprise a condition of the device on which



IPR2023-00330 Page 00708

transcoding server **34** is installed (for example, a current processing load), a condition of a device to which transcoding server **34** is coupled, or a condition of a communications link. Transcoding server **34** may provide the ability to dynamically update such predetermined selection criteria.

The following discussion provides still more examples of the types of information which may be used to dictate which of transcode service providers **24** are invoked. It should be noted, however, that these examples are provided by way of illustration only, and are not intended to limit in any way the scope of the invention claimed herein. The predetermined selection criterion may comprise: (1) network client **12**, such as a display dimension, resolution, number of colors, processor type, memory/disk configuration, modem or network interface type, installed add-in boards (for example, hardware compression/decompression), software configuration (for example, availability of pre-installed software decompression modules), physical location/proximity (for example, as determined by a telephone area code), and user identity; (2) characteristics of transcoding server **34** or some other network server, including system load and identification information (for example, the owner of the server); (3) content characteristics, such as its data type, type of encoding/compression, size, and dimension; (4) network characteristics, including best-case, worst-case and average latency, bandwidth and/or error rates (for example, for wireless communications) between network client **12** and a proxy, and/or between a proxy and a server (this may be predetermined for guaranteed bandwidth links like ATM (Asynchronous Transfer Mode), or dynamically measured/predicted for so-called "best effort" links like many IP (Internet Protocol) links); (5) proxy characteristics, including system load, available storage, physical location/proximity, and identity (owner); (6) user preferences, including preferred content quality/speed tradeoff, language, content rating, exclusion list, inclusion list, data type-specific preferences (for example, "never download" images), include/exclude advertising, amount of advertising desired, offensive language removal, whether the user's defined or learned preferences may be disclosed (and to whom), custom rules or



IPR2023-00330 Page 00709

programs for filtering/transcoding/processing data, and shared preferences with either another user or a group of users (any of the foregoing user preferences may be explicitly defined or system predicated, such as based on usage statistics compiled over time); (7) group preferences, including results from collaborative rating systems, whether manual (for example, a prior user manually assigned a rating to a Web page after viewing it) or automatic (for example, given a large number of users who accessed a link on a given page, the probability of any given user subsequently following that link); (8) content provider preferences, including the degree of alteration desired for its content, the prioritization for download and display of different content types, cache restriction or prioritization parameters such as update frequency or replacement preferences, the types of users to target, rules or programs to run for customizing content (for example, news or advertising, custom language translation software) based on user or client characteristics, desire for receiving certain types of user or group data collected (for example, demographics or access patterns), and type of payment/reward offered in exchange for such information; and (9) other preferences, including software vendor rules or programs for dynamically checking content created or distributed using unauthorized software and companies' desire to enforce correct usage of certain types of content (for example, trademarks and logos).

Applying the above-listed selection criteria, or combinations thereof, embodiments of the present invention may be used to provide a virtually limitless range of dynamic transcoding services. For example, client and/or proxy physical proximity, in combination with demographic data, may be used for extremely targeted advertising. Such advertising may be added to any content passing through a proxy, for example, or some other mechanism. This can in turn be tailored even further based upon the user's willingness to tolerate advertising or share demographic information, as well as the advertiser's ability/willingness to subsidize or otherwise reward the user for participation.



IPR2023-00330 Page 00710

Embodiments of the present invention may be advantageously used to reduce the amount of data that is transmitted to network client **12**, thereby promoting faster downloading and rendering of content. Suitable transcoding techniques include lossy compression and transcoding to a more efficient (and perhaps not widely supported) format specifically for the transmission. Similarly, HTTP remote proxy **36** may be configured to "predigest" Web sites or groups of sites to produce extremely condensed overviews of large amounts of content (for example, a tree structure, pages with only first-level or first- and second-level headings, thumbnails of pages, or only parts of a page or site that have changed since the user's last visit). Such applications may be especially advantageous for poorly-connected or computationally limited devices such as PDAs (Personal Digital Assistant), since this predigestion can be performed on a well-connected proxy server with an abundance of computational power, and the concise result can be easily downloaded and rendered on the more limited device.

Embodiments of the present invention may alternatively be used for dynamic translation of data, such as Web pages, to a user's native language (determined by user preference or automatically by the physical location of network client **12** or transcoding server **34**). Such a capability greatly simplifies the task of making content truly global, as well as reduces storage and maintenance required at the content provider (that is, only one copy of the content need be maintained, rather than different copies for each of a plurality of different languages).

Embodiments of the present invention may be used to block certain types of content or to automatically censor offensive language (similar to a "beep" used for television broadcasts). Only the particular offensive parts of the content (for example, obscene words) may be removed, or entire Web sites may be blocked. Similarly, transcoding server **34** may be configured to scan content for certain words or phrases to ensure that trademarks or logos are used correctly (for example, as a source identifier rather than a generic product designation). This feature may be offered as a service to companies or organizations, who would supply a list of words



IPR2023-00330 Page 00711

or phrases to flag. A similar capability could be used to automatically insert links into the content upon detection of certain words or phrases. For example, Intel Corporation might want to automatically add a link to its corporate Website whenever the name "Intel" is used in a Web page. Using an embodiment of the present invention, such links can be dynamically added to the content before it is displayed to a user. In a similar vein, an embodiment of the present invention may be used to scan for content that was created or distributed using unlicensed software. This feature may be implemented using special keys (binary bit patterns) embedded in the content or headers put in by the content creation or distribution software. The scanning logic and logic for taking a predetermined responsive action, such as denying service or posting a warning, may optionally be supplied by the vendor of the software in question or configured into transcoding server **34**.

Embodiments of the present invention may also be used to scan content for computer viruses prior to sending such content to network client **12**. For example, an existing virus scanning routine may be installed on transcoding server **34**, possibly as a plug-in module Transcoding server **34** may then be configured to invoke the virus scanning routine to ensure any content transmitted to network client **12** is free of viruses. A significant advantage provided by such an embodiment is that virus scanning software need only be maintained on transcoding server **34**, rather than on a plurality of network clients **12**. In this way, the benefit of upgrades to the virus checking software may be efficiently and timely provided to large numbers of users, thus avoiding the problem of any particular user relying on outdated virus scanning software.

Embodiments of the present invention may also be used to produce custom content on demand in accordance with user-specific preferences and/or associations with collaborative rating systems. In a variation on such an embodiment, transcoding server **34** can collect preferences and append them as part of a client request transmitted to a content provider so that the dynamic content generation can



IPR2023-00330 Page 00712

be done at the content server. Likewise, a proxy provider (for example, an Internet Service Provider (ISP)), can collect and make available to content providers information such as user preferences and data access statistics, as well as content provider specific statistics (for example, how many users from a given region or a given profile accessed a particular Web site, and at what time, in the past month). Such information may be used for applications such as targeted advertising.

Embodiments of the present invention may further be used to automatically check the validity of links in an object, and correct or remove invalid links, prior to transmitting the object to network client **12**. This capability may be provided, for example, as a service to content providers who may not have the most up-to-date information on Websites they are linked to which have moved or been deleted.

To further illustrate the general operation of the embodiment illustrated in **Fig. 3**, assume a user of network client **12** wishes to access a particular Web page, or URL (Uniform Resource Locator), on Internet **18**. Further assume that the desired URL resides on, or is accessible through, transcoding server **34**. Network client **12**, via browser **32**, transmits an HTTP request for the hypertext object to transcoding server **34** over client/server communications link **14**. Where browser **32** normally accesses Internet **18** through a proxy, browser **32** is configured to pass user requests through transcoding server **34** via browser's **32** standard proxy configuration procedures. As is well known in the art, browser **32** may actually transmit a plurality of additional HTTP requests corresponding to each of various distinct hypertext objects that may be embedded in the Web page. In such a case, transcoding server **34** may process each such request in the manner described below.

According to this embodiment, HTTP remote proxy **36** is capable of distinguishing between a non-enabled network client **12** and an enabled network client **12**. This may be accomplished, for example, using a private protocol to



IPR2023-00330 Page 00713

transmit content requests from an enabled network client to transcoding server **34**, so that the use of some other communications protocol indicates network client **12** is non-enabled. This method of sending a private protocol in each request to HTTP remote proxy **36** is an improvement over a registration type process. The overhead involved in making the enabled/non-enabled determination on a per request basis is relatively small, while providing a significant advantage because it addresses the situation for HTTP remote proxy **36** where a first network client disconnects and a second network client, likely with different communications and/or presentation capabilities, reconnects using the same IP address.

Upon determining that network client **12** is non-enabled, HTTP remote proxy **36** may record the IP address of network client **12** in a client preference table **26** maintained in a local data store (client preference table **26** may improve performance of this or other embodiments, but is not required). HTTP remote proxy **36** then passes the hypertext object to parser **22**. HTTP remote proxy **36** may also inform parser **22** of any applicable user preferences (e.g., from client preference table **26**). Upon being invoked, parser **22** first calls cache interface **28** with the requested hypertext object to determine whether a copy of the required version already resides in server-side cache memory **30**. For purposes of illustration, assume no entry exists in server-side cache memory **30** for the requested hypertext object. HTTP remote proxy **36** then invokes a call to retrieve the hypertext object from Internet **18** over server/network communications link **16**. Assuming the requested hypertext object is found, HTTP remote proxy **36** begins receiving an HTTP data stream representing the hypertext object. HTTP remote proxy **36** passes the handle for this incoming data stream to parser **22**.

Parser **22** dynamically determines whether the data stream satisfies any applicable predetermined selection criteria. For example, where transcode service providers **24** are configured to scale data of different types, parser **22** may determine the content type for the data stream (e.g., image/jpeg, image/gif,



IPR2023-00330 Page 00714

video/mpeg) by interrogating a MIME type in the content-type header record that appears at the beginning of the incoming HTTP data stream. If parser **22** detects a match for a predetermined selection criterion, the HTTP stream handle is given to the appropriate transcode service provider **24**. Transcode service provider **24** then transcodes the data stream appropriately, and HTTP remote proxy **26** transmits the transcoded data stream to network client **12**.

A non-enabled network client **12** may optionally be provided with the ability to actively control aspects of the transcoding process, or indeed whether or not to transcode requested content at all. To provide this ability, HTTP remote proxy **36** may embed additional instructions at the beginning of the HTML header for the requested URL prior to transmitting the associated data stream to network client **12**. These embedded instructions may be implemented, for example, as JavaScript codes, VB Script codes or Java Applet codes. As browser **32** of network client **12** receives the data stream, the embedded instructions will automatically execute so long as browser **32** is equipped to support them. For example, if the embedded instructions are implemented as JavaScript codes, browser **32** may be a JavaScript-enabled browser such as a Netscape Navigator v.2.0 or above browser, or an Internet Explorer v.3.0 or above browser. If browser **32** is not equipped for such HTML scripting, the embedded instructions will not interfere with the browser's **32** normal processing, as such browsers **32** are typically configured to ignore any data they cannot interpret.

The embedded instructions transmitted to network client **12** may enable the user to manipulate some of the transcoding capabilities of transcoding server **34**. As illustrated in **Fig. 4**, the embedded instructions may drive a user interface in the form of a pop-up window **40** that is displayed at the top of a browser window **38**. Pop-up window **40** includes a three-state switch **42** having "ON," "OFF" and "AUTO" settings, and may also include a hypertext link **44** which the user may follow to download specialized client software supporting, for example, more



IPR2023-00330 Page 00715

sophisticated transcoding functionality (i.e., become "enabled"). The initial setting of three-state switch **42** may be based upon a prior determination by HTTP remote proxy **36** as to whether network client **12** has an established preference for reception of transcoded content. If so, three-state switch **42** may be set to "ON;" if not, three-state switch **42** may be set to "OFF." A goal of this feature is to provide the user with some means for communicating a preference to HTTP remote proxy **36** with regard to aspects of particular transcoding features, such as a content quality/latency tradeoff where the transcoding comprises data compression/scaling. Persons skilled in the art will recognize that many other means for providing this capability are possible, and such other means could enable the user to communicate preferences beyond simply a yes/no indication for transcoding.

In the illustrated in **Fig. 4**, pop-up window **40** enables the user to change his or her preference as to whether transcoded or original content is desired, and communicates such changes to HTTP remote proxy **36**. Pop-up window **40** may or may not interact with browser **32**, meaning the user's preference will only take effect after setting three-state switch **42** and clicking on the browser's **32** "RELOAD" button **46** to cause browser **32** to request the (transcoded or un-transcoded) content for presentation to the user. Subsequent pages in the current session may then be rendered in accordance with the new setting of three-state switch **42** without further user intervention. Upon receipt, HTTP remote proxy **36** may update user preference table **26** accordingly. As an alternative, pop-up window **40** may be configured to automatically invoke the "RELOAD" operation when the user indicates a change (such as by flipping three-state switch **42**). Where browser **32** is a JavaScript-enabled browser, JavaScript instructions inserted by HTTP remote proxy **36** in the HTML document may "POST" the state of three-state switch **42** to HTTP remote proxy **36** and also cause browser **32** to "RELOAD" the current URL.

It is possible to allow a non-enabled network client **12** to save the state of three-state switch **42** on network client **12** across multiple sessions of



IPR2023-00330 Page 00716

browser **32** using what is known in the art as a "cookie." In other words, a cookie may be used to store the state of three-state switch **42** persistently. When a new session of browser **32** is initiated by a user, this state information may be read from network client **12** and "POSTed" by the JavaScript code (inserted at the beginning of the HTML document) to HTTP remote proxy **36** before any content for the requested hypertext object is actually sent to network client **12**. This will allow HTTP remote proxy **36** to update user preference table **26** with the correct state of three-state switch **42**, and hence send correctly-transcoded content to network client **12**. In such an embodiment, the state information may be "POSTed" to HTTP remote proxy **36** each time a given URL is requested by browser **32**. This will allow network client **12** to receive the correctly-transcoded content even if the HTTP remote proxy **36** to which it is coupled changes due to, for example, a change in geographical location of network client **12** or network load-balancing procedures.

The embodiment shown in **Fig. 3** may also be used for network clients **12** that already access Internet **18** through a standard proxy. JavaScript-enabled browsers **32** may query the local IP address of network client **12** and "POST" this information to HTTP remote proxy **36**. The HTTP header of this "POST" message will contain the IP address of the standard proxy, which will now be different from the IP address of network client **12** (which is included in the contents of the message). A comparison of the two IP addresses will determine whether network client **12** resides behind a standard proxy. HTTP remote proxy may then use this information to update transcoding information about network client **12** in user preference table **26**.

According to another embodiment of the present invention, illustrated in **Fig. 5**, network client **12** may be "enabled," containing specialized software to support, for example, more sophisticated transcoding features than are provided by the above-described embodiments, or to perform some or all of the transcoding functions on the client side. As illustrated, network client **12** includes an HTTP local



IPR2023-00330 Page 00717

proxy **48** coupled to a client-side parser **50** which, similar to parser **22** of transcoding server **34**, controls one or more client-side transcode service providers **52**. Each transcode service provider **52** may be configured, for example, to transcode content before it is rendered to a user or to perform a counterpart transcoding function (e.g., decoding, decompression) with respect to a function performed by a corresponding transcode service provider **24** of transcoding server **34**. As in transcoding server **34**, network client **12** may include a client-side cache memory **56** managed by a client-side cache interface **54**. Client-side cache interface **54** may be an already-existing facility supported by the operating system, such as WININET. Using an existing caching facility reduces the amount of software that is to be downloaded to network client **12** to implement this embodiment, and also allows other applications, such as disconnected browsers, to share client-side cache memory **56**.

HTTP local proxy **48**, client-side parser **50** and client-side transcode service providers **52** (collectively, the client software) may be downloaded to network client **12** on demand, such as by clicking on hypertext link **44** presented by pop-up window **38** illustrated in **Fig. 4**. Alternatively, the client software could be distributed to users on a portable storage medium, such as a diskette or CD-ROM, or it may be preloaded on an off-the-shelf personal computer. In the embodiment of **Fig. 5**, the client software is separate from browser **32**; however, in yet another embodiment the client software may be integrated in browser **32** (see **Fig. 6**).

The enabled client embodiments provide network client **12** with expanded flexibility for rendering hypertext objects. As in the non-enabled client embodiments described above, enabled network client **12** may receive a transcoded data stream from HTTP remote proxy **36** in a format that is already supported by the standard internal rendering software of browser **32** (e.g., JPG, GIF). This would be the case where, for example, the transcoding process involved adding or deleting text to the hypertext object. In addition, HTTP remote proxy **36** may transcode a hypertext object to a data stream having a new MIME type, such as where the



IPR2023-00330 Page 00718

transcoding process comprised scaling or data compression, in which case a client-side transcode service provider **52** could be provided to convert the data stream back to a MIME type supported by browser **32**. For example, HTTP remote proxy **36** could transmit a file compressed using a non-standard, not well-supported but leading-edge compression algorithm to network client **12**, and client-side transcode service provider **52** could uncompress the file back to its original format. This approach has the benefit of relieving HTTP local proxy **48** from having to provide a user interface, and eliminates restrictions imposed by limitations as to the data types supported by browser **32**. In this way, the transcoding process can remain transparent to users, browsers and Web servers even when it involves changing content to different datatypes.

Yet another possibility is that enabled network client **12** includes one or more add-ins **46** specifically configured to transcode, render or playback content received by network client **12**. Add-ins **46** may be implemented, for example, using Netscape plug-ins or ActiveX controls. Moreover, add-ins **46** may be installed as part of the client software, as illustrated in **Fig. 5**, or integrated with browser **32**. Such add-ins **46** are beneficial in that they generally may be configured to permit a user to click on a specific object to obtain a different version (e.g., higher quality) representation. Add-ins **46** are also beneficial in that they appear to a user to be well-integrated with browser **32**, and are easily upgradeable. Combinations of the above-described presentation facilities are also possible.

In an advantageous optional application of add-ins **46**, network client **12** may be configured to request that an appropriate add-in **46** be downloaded from HTTP remote proxy **36** in the event that network client **12** determines it is unable to transcode a particular received data stream. HTTP remote proxy **36** could then download the necessary add-in **46** or, alternatively, resend the data stream in a different format. This facility provides for automatic extension of the system, ensuring that client software is as current as possible.



IPR2023-00330 Page 00719

In the embodiment of **Fig. 5**, browser **32** is configured to send all HTTP requests through HTTP local proxy **48**, thus allowing HTTP local proxy **48** to improve retrieval and rendering of requested hypertext objects. For example, when HTTP local proxy **48** receives an HTTP request from browser **32** for a hypertext object associated with a Web page, it passes the URL to client-side cache interface **54** to check whether a copy of the hypertext object already exists in client-side cache memory **56**. If the hypertext object is cached, HTTP local proxy **48** passes the cached object to browser **32** for rendering. If the requested hypertext object is not cached, HTTP local proxy **48** transmits an HTTP request to transcoding server **34** for processing. HTTP local proxy **48** may use a custom Get() request for this purpose to enable transcoding server **34** to identify network client **12** as enabled. Performing the processing described above with reference to other embodiments, transcoding server **34** will return a data stream for the hypertext object to HTTP local proxy **48**.

To further illustrate the features and benefits of embodiments of the present invention, the flow charts provided in **Figs. 7-9** illustrate the logic for an embodiment of a method by which an enabled network client may render a hypertext object resident on the Internet. The flow charts are not intended to be comprehensive of all processing that is performed, but rather are intended to describe the overall flow of the method. Detailed descriptions of the various processes have been provided above with reference to various disclosed embodiments. Where practical, the following description includes reference numbers for previously-described structural elements, although the method is not limited to those structures.

Referring now to **Fig. 7**, processing begins when a user on network client **12** requests a hypertext object from browser **32 (Step 100)**. This could be in the form of a request for a specific Web page, in which case a plurality of hypertext objects will likely be displayed to the user, or in the form of a click on an image already being displayed to the user. Browser **32** may be configured to pass all HTTP



IPR2023-00330 Page 00720

requests through HTTP local proxy **48**, so HTTP local proxy **48** may intercept the HTTP(URL) request from browser **32** (**Step 110**).

In this particular embodiment, HTTP local proxy **48** first checks whether the requested hypertext object exists in client-side cache memory **56** (**Step 120**). To do this, HTTP local proxy **48** may invoke client-side parser **50** using a GetScaledObject(URL) call, which in turn issues a GetEntry call to client-side cache interface **54** to open a stream for the cached object. This effectively "retrieves" the cached object from client-side cache memory **56** if it exists (**Step 140**). HTTP local proxy **48** then passes the stream to browser **32**, which displays the cached object to the user (**Step 150**).

Referring now to **Fig. 8**, if the requested URL object is not found in client-side cache memory **56**, HTTP local proxy **48** transmits a request for the object to transcoding server **34** using, for example, a Post of a GetStage(URL, Stage=0) call (**Step 160**). Upon receipt of this call, HTTP remote proxy **36** invokes parser **22**, which ir turn issues a GetScaledObject() call to server-side cache interface **28** to determine whether a non-transcoded version of the requested hypertext object already exists in the server-side cache memory **30** (**Step 170**). If the hypertext object is cached, server-side cache interface **28** issues a GetEntry call to open a stream for the cached object (**Step 200**). In addition, parser **22** may issue a GetProperties(URL, ...) call to server-side cache interface **28** to retrieve information about the transcoding properties and transcoded status (such as the refinement level) of the cached object.

If parser **22** determines that the requested hypertext object does not exist in the server-side cache memory **30**, HTTP remote proxy **36** issues an HTTP request to retrieve the hypertext object from Internet **18** (**Step 190**). If the object is not found, HTTP remote proxy **36** returns an error to network client **12** which browser **32** will communicate to the user (**Step 220**); if the object is found, HTTP remote proxy **36** passes the handle for the incoming data stream to parser **22**, which



IPR2023-00330 Page 00721

in turn initiates caching of an original version of the retrieved hypertext object (**Step 230**).

Referring now to **Fig. 9**, once the requested hypertext object has started to be obtained, parser **22** determines whether (and how) to transcode the object before transmitting it to network client **12** (**Step 240**). Both this decision-making process and exemplary transcoding processes are described in detail above. For purposes of the present illustration, assume parser **22** determined that transcoding was appropriate and therefore generated a transcoded version of the requested hypertext object (**Step 250**). HTTP remote proxy **36** transmits a data stream for the transcoded hypertext object to network client **12** (**Step 260**). Upon receipt, HTTP local proxy **48** initiates caching of the transcoded hypertext object (**Step 270**). In addition, client-side parser **50** determines whether any further processing is required before the hypertext object is rendered (e.g., a new MIME type has been established by transcoding server **34**) (**Step 280**).

If no additional transcoding is required, HTTP local proxy **48** passes the handle for the received data stream to browser **32** for display to the user (**Step 290**). If additional transcoding is required, client-side parser **50** passes the handle to an appropriate transcode service provider **52** (**Step 300**). The result of this latter processing may be a hypertext object which browser **32** can readily display to the user (**Step 320**), or the result may be a hypertext object having a non-standard MIME type, in which case browser **32** may invoke add-in **46** to display the object (**Step 330**).

According to another embodiment of the present invention, additional data or programs need not necessarily be inserted as part of a response to a client request. Rather, data and programs may be transparently "pushed" to network client **12** without the user or the browser **32** software's detection or intervention. One advantage of this approach is that transcoding server **34** is able to detect when



IPR2023-00330 Page 00722

client/server communications link **14** is underutilized, and can thus push data to network client **12** with limited risk of interfering with other transactions. An especially advantageous implementation uses at least a local proxy, which could issue its own requests (rather than being user-driven) to content providers or networked proxy servers, or receive unsolicited data pushed to it from the network. The local proxy may store the data in a client-side cache, install it as a program, or prompt the user to take some further action. Many potential uses for such an embodiment are possible. For example, an advertiser of software products or music can preload network client **12** with trial versions of products before prompting the user with an advertisement, thus enabling instant playback capability without the user having to wait for a demo to be downloaded (and possibly losing interest in the meantime).

A number of different configurations are possible for implementing embodiments of the present invention. In a first configuration, the only additional apparatus required is a remote proxy. That is, no new software needs to be installed on network client **12**. The remote proxy may reside anywhere on a suitable network, such as the Internet, including at particular content provider sites. Alternatively, the remote proxy may be located at ISP local POPs (Point of Presence), for example, if location-specific characteristics are to be used as predetermined selection criteria. Of course, such information can be gathered by other methods as well, such as user-preference settings or assigning location-specific domain names to proxies. In a second configuration, a new piece of client software acting as a local proxy may be installed, for example, on a client device. The user would then point the client application's proxy to the local host. Combinations of these exemplary configurations are likewise possible, as well as simultaneously having multiple modes active (for example, a local proxy acting as a pass-through for some requests and a non-pass-through for others that require the use of a remote proxy).

Where network client **12** connects to a remote proxy over a relatively slow communications link, it may be particularly advantageous to implement



IPR2023-00330 Page 00723

transcoding and link validity checking on remote proxies. Combinations of remote and local proxies can sometimes give more efficient implementations of certain applications, such as automatic data/program download and interactively displaying predigested content. Other applications, such as translation and trademark enforcement, can be done efficiently on local proxies alone, but may be more advantageously done on remote proxies because the results can be cached for use by others, thereby saving resources for future requests. Still other applications, such as clickstream analysis, are generally better implemented on a local proxy because there are more resources available locally to the individual user, and also for privacy reasons.

In view of the foregoing description, it should be apparent that it is possible for there to be more than one so-called "smart" proxy arranged between a client device and a content server device. If left unchecked, such a condition can result in content being altered excessively (for example, too many ads inserted, multiple lossy compressions resulting in indecipherable images). To address this problem, an embodiment of the present invention may use a special proxy-to-proxy protocol that extends the existing request/response structures to indicate whether and what sort of transcoding has already been performed on the content. Such a specialized protocol, in addition to other proxy-to-proxy messages which may be implemented on an as-needed basis, enables multiple proxies to work collaboratively, yet still transparently to users, client software, existing "standard" proxies and content servers.

According to yet another embodiment of the present invention, a proxy server may be used to provide certain Internet proxy or server users with so-called "VIP" treatment, identifying users who are entitled (either through payment or based on some other selection criterion, such as extent of usage) to have a higher priority when competing with other users for proxy resources. By contrast, with existing Internet proxies and servers, users are serviced either on a random or first-come/first-served basis.



IPR2023-00330 Page 00724

In one particular implementation of such an embodiment, transcoding server **34** may be configured to extract user IP addresses from requests it processes and maintain information such as how frequently, or for what duration, a user is browsing a particular Web site. Such information could be used to determine "frequent browser miles" at particular Web sites. Users can then be rewarded with faster response times for subsequent visits to the site, or the site owner could choose to reward the user with improved performance on all sites reached through the same proxy. Still another possibility is that users may pay for such preferred service, being assigned a password which may be provided to transcoding server **34**. Yet another possibility is that a Web site owner can pay a proxy provider to improve the performance of all users while visiting the owner's site.

In another particular implementation, information identifying users to be given VIP treatment may be passed to transcoding server **34** in the form of a Web page. Upon receipt of such a Web page, the proxy may subsequently allow servicing threads to perform work for requests generated by VIP users first. To do this, transcoding server **34** may boost thread scheduling priorities (within the operating system) for the VIP service, while ensuring there is no starvation of any thread (that is, no user should be denied access entirely by VIP users). In addition, transcoding server **34** may permit preferential caching for particular Web sites and more aggressive pre-fetching for VIP users. Still further, transcoding server **34** may use more resource-intensive compression algorithms, for example, to provide better quality content for the same latency at the expense of slowing down access for non-VIP users.

It is possible that certain content providers or users will not wish to have their content dynamically altered in any manner. Accordingly, embodiments of the present invention may be implemented in such a way that either content providers or users are given the capability to override any potentially content-altering service.



IPR2023-00330 Page 00725

This may be accomplished, for example, using a pass-through technique triggered by a special tag embedded within the content.

As the foregoing description demonstrates, embodiments of the present invention may be used to provide a system for improving the communications capabilities of computers accessing networks such as the Internet. Embodiments of the invention may be advantageously applied to computers having limited communications bandwidth available, such as mobile computers or personal computers accessing a network over a modem connection. The unique features of such embodiments enhance the ability of these computers to access data on the network in a timely fashion with reduced user-visible latencies, thereby enabling content authors to produce rich content without fear that only users with highly-sophisticated data communications and display capabilities are able to enjoy it. Embodiments of the present invention may also be advantageously used for purposes other than, or in addition to, reducing latency. Such purposes include, for example, converting color images to greyscale images for users lacking a color display; filtering and/or deleting undesired content, such as pornography; adding content, such as advertising; and language translation.

Although the present invention has been described with reference to embodiments for accessing data from the Internet, persons skilled in the art will recognize that it is equally applicable to other networking environments. For example, embodiments of the present invention may be used to enhance data communications between a network client computer and an "intranet." An intranet typically is a secure corporate network modeled after the Internet architecture, and generally includes mechanisms for communicating with external networks such as the Internet.

The foregoing is a detailed description of particular embodiments of the present invention. The invention embraces all alternatives, modifications and variations that fall within the letter and spirit of the claims, as well as all equivalents



IPR2023-00330 Page 00726

of the claimed subject matter. For example, some or all of the features described above as being provided by a remote proxy may be implemented in a content server. Likewise, some or all of the features described above as being provided by a local proxy may be implemented in a browser application. Persons skilled in the art will recognize from the foregoing detailed description that many other alternatives, modifications and variations are possible.



IPR2023-00330 Page 00727

**What Is Claimed Is:**

1. An apparatus for use in transmitting data between a network server and a network client over a communications link, said apparatus comprising a parser coupled to a transcode service provider, said parser being configured to selectively invoke said transcode service provider in response to a predetermined selection criterion.

2. The apparatus of claim 1, wherein said predetermined selection criterion comprises a characteristic of the data being transmitted.

3. The apparatus of claim 1, wherein said predetermined selection criterion comprises a characteristic of the communications link.

4. The apparatus of claim 1, wherein said predetermined selection criterion comprises a characteristic of the network server.

5. The apparatus of claim 1, wherein said predetermined selection criterion comprises a characteristic of the network client.

6. The apparatus of claim 1, wherein said predetermined selection criterion comprises a user preference.

7. The apparatus of claim 1, wherein data is transmitted from the network server to the network client in response to a request by the network client, said predetermined selection criterion being included in said request.

8. A method for providing a network client with a data object residing on a network server, wherein the network client and the network server are coupled by a communications link, said method comprising the steps of:



IPR2023-00330 Page 00728

receiving a data object from the network server;

selectively transcoding the data object according to a predetermined selection criterion; and

providing the data object to the network client.

9. The method of claim 8, wherein said transcoding step further comprises comparing a characteristic of the received data object to the predetermined selection criterion.

10. The method of claim 8, wherein said step of selectively transcoding the data object further comprises determining whether the data object includes content created with an unregistered software product.

11. The method of claim 10, wherein said step of selectively transcoding the data object further comprises adding a message to the data object corresponding to said detection of content created with an unregistered software product.

12. The method of claim 8, wherein said step of selectively transcoding the data object comprises compressing a portion of the data object.

13. The method of claim 8, wherein said step of selectively transcoding the data object comprises translating a portion of the data object from a first language to a second language.

14. The method of claim 8, wherein said step of selectively transcoding the data object further comprises determining whether the data object includes offensive content.

15. The method of claim 14, wherein said step of selectively transcoding the data object further comprises modifying the data object to prevent offensive content from being rendered by the network client.



IPR2023-00330 Page 00729

16. The method of claim 8, wherein said step of selectively transcoding the data object further comprises adding advertising information into the data object.

17. The method of claim 16, wherein said advertising information is selected in accordance with user profile information.

18. The method of claim 8, wherein said step of selectively transcoding the data object further comprises determining whether the data object includes a link to a second data object.

19. The method of claim 18, further comprising the step of validating the link to a second data object.

20. The method of claim 19, wherein said step of selectively transcoding the data object further comprises correcting an invalid link.

21. The method of claim 8, wherein said step of selectively transcoding the data object further comprises communicating information relating to said transcoding to the network server.

22. The method of claim 8, wherein said step of selectively transcoding the data object further comprises determining whether the network client is preconfigured to receive preferential treatment of requests.

23. A set of instructions residing on a storage medium for execution by a computer, the computer being coupled to a device for rendering a data object to a user, said set of instructions comprising instructions for:

parsing a data object to be rendered to detect content corresponding to a predetermined selection criterion;



IPR2023-00330 Page 00730

selectively transcoding the data object in response to said detection prior to rendering the data object.

24. The set of instructions of claim 23, wherein the storage medium comprises a magnetic storage device.

25. The set of instructions of claim 23, wherein the storage medium comprises a memory installed in a computer.



IPR2023-00330 Page 00731

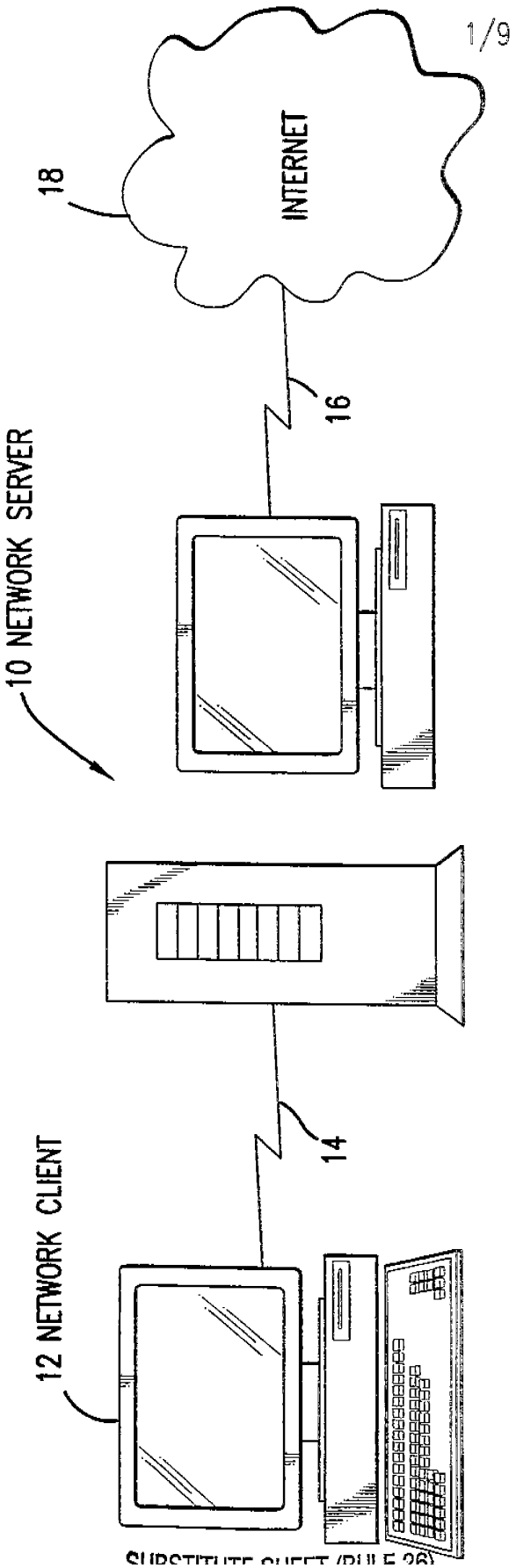



FIG. 1

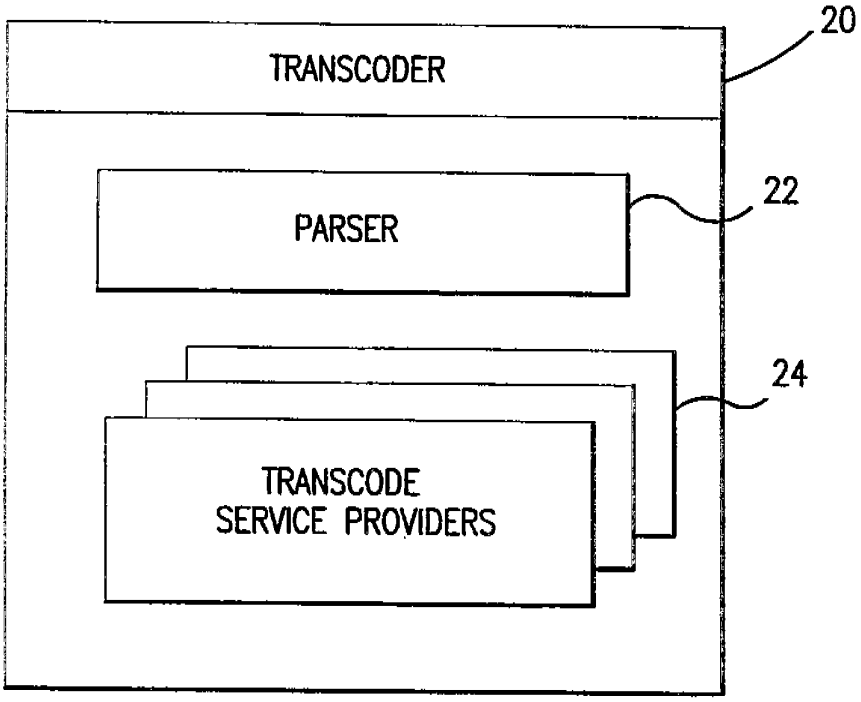



FIG.2



IPR2023-00330 Page 00733

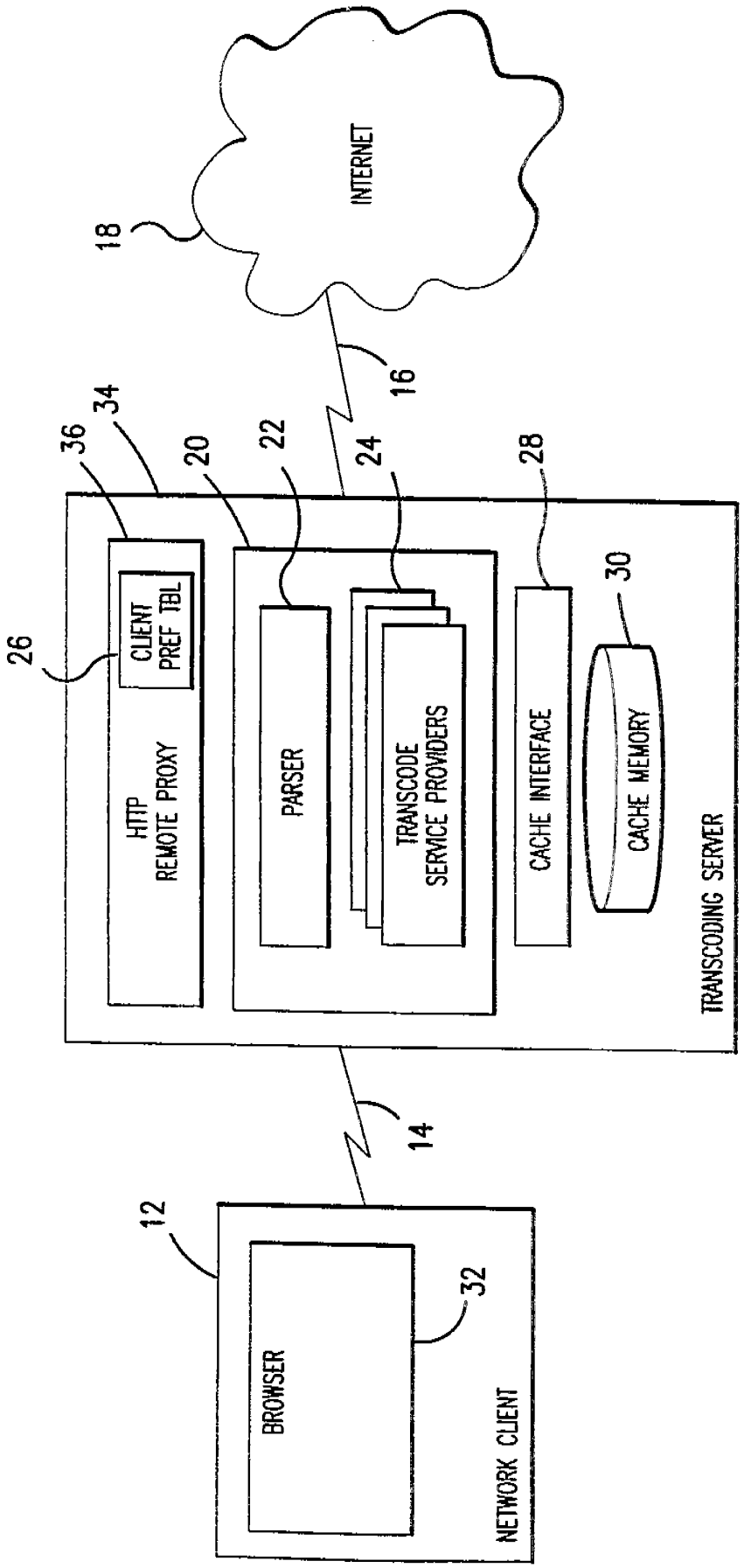

INTERNET
18
16
36
34
20
22
24
28
26
CLIENT PREF TBL
HTTP REMOTE PROXY
PARSER
TRANSCODE SERVICE PROVIDERS
CACHE INTERFACE
30
CACHE MEMORY
TRANSCODING SERVER
14
12
BROWSER
32
NETWORK CLIENT

FIG.3

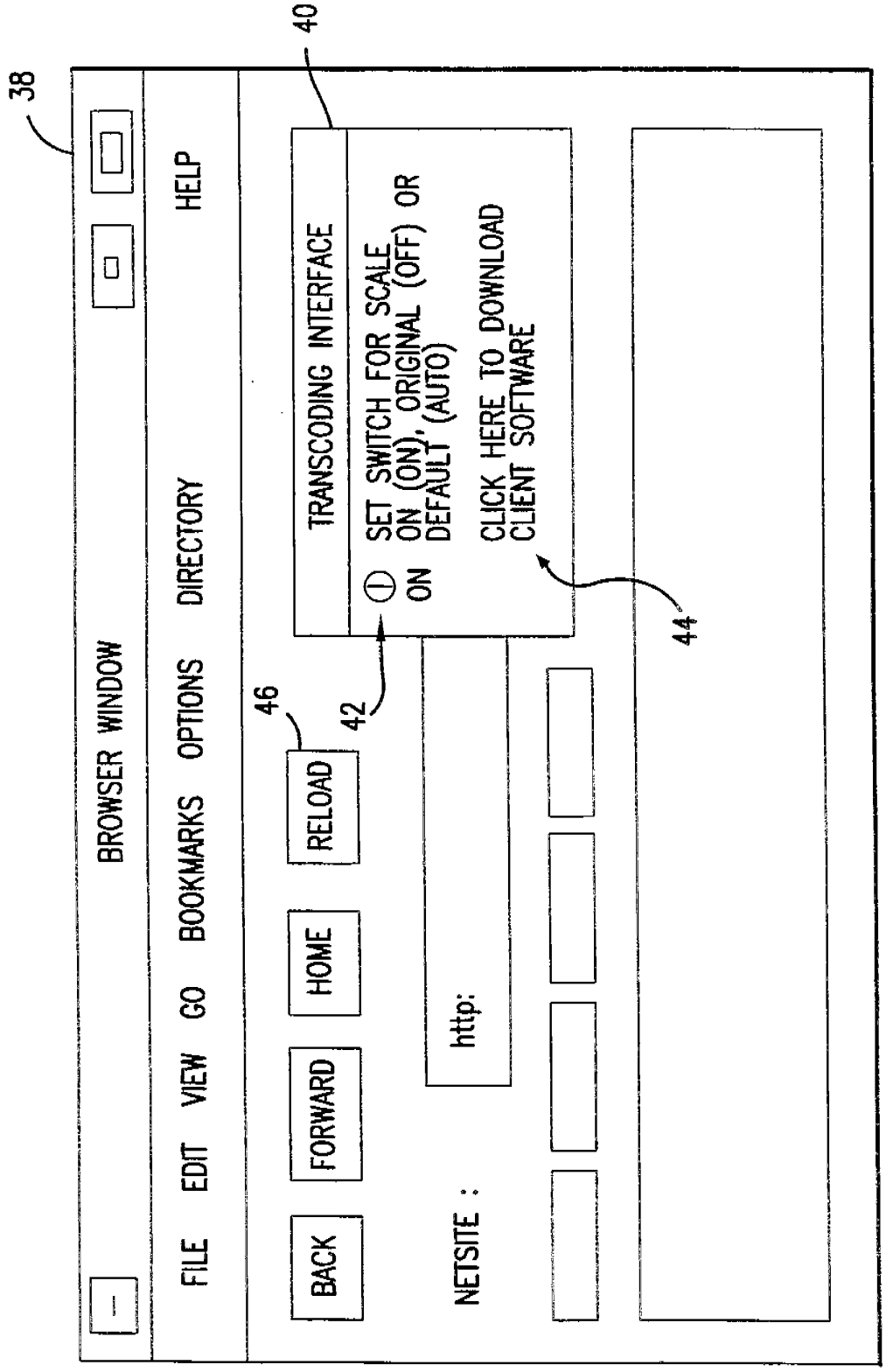



FIG.4



IPR2023-00330 Page 00735

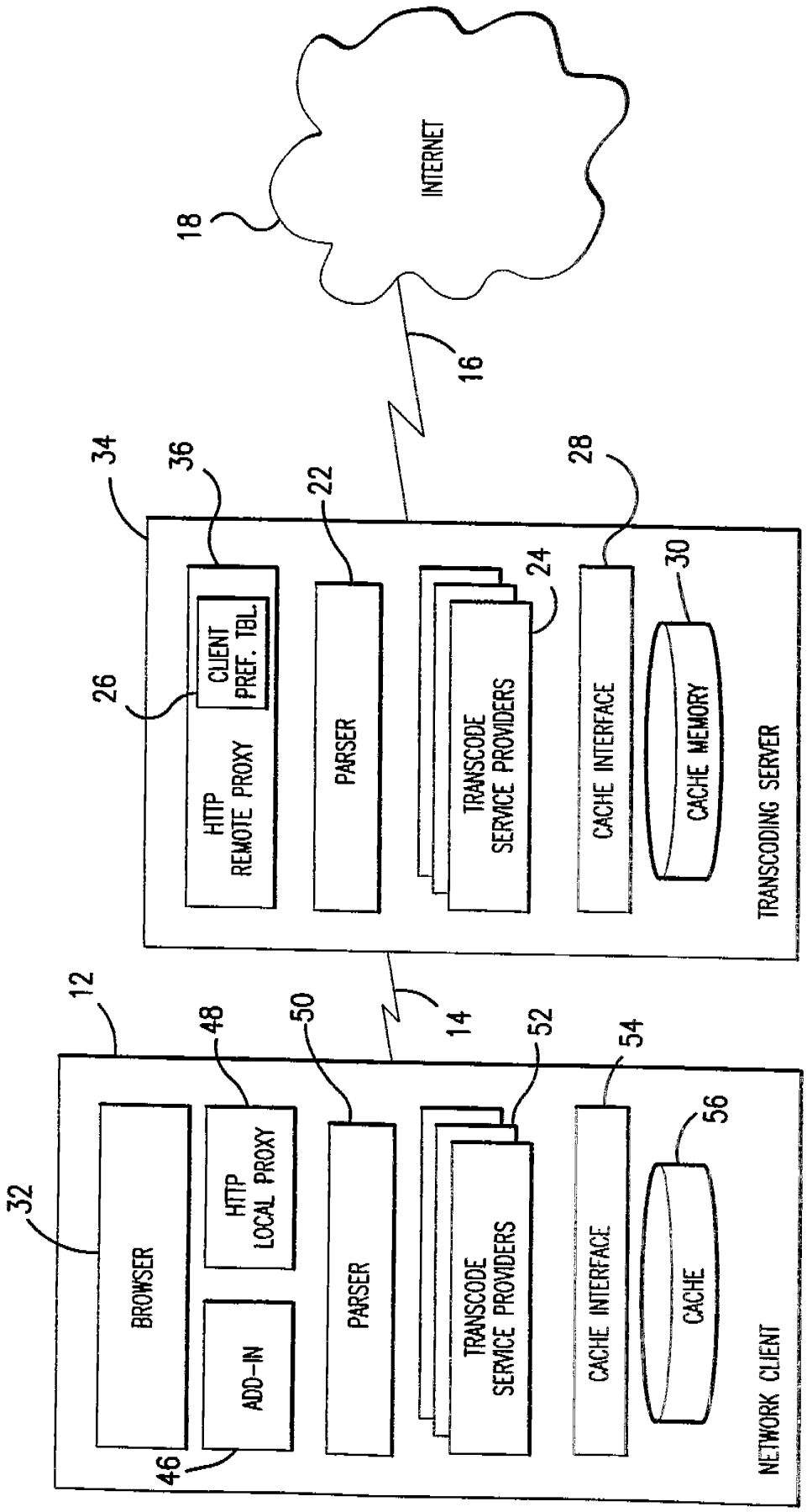

INTERNET
18
16
34
36
22
28
26
CLIENT PREF. TBL.
HTTP REMOTE PROXY
PARSER
TRANSCODE SERVICE PROVIDERS
24
CACHE INTERFACE
30
CACHE MEMORY
TRANSCODING SERVER
12
14
48
50
52
54
32
BROWSER
HTTP LOCAL PROXY
ADD-IN
PARSER
TRANSCODE SERVICE PROVIDERS
CACHE INTERFACE
56
CACHE
46
NETWORK CLIENT

FIG.5

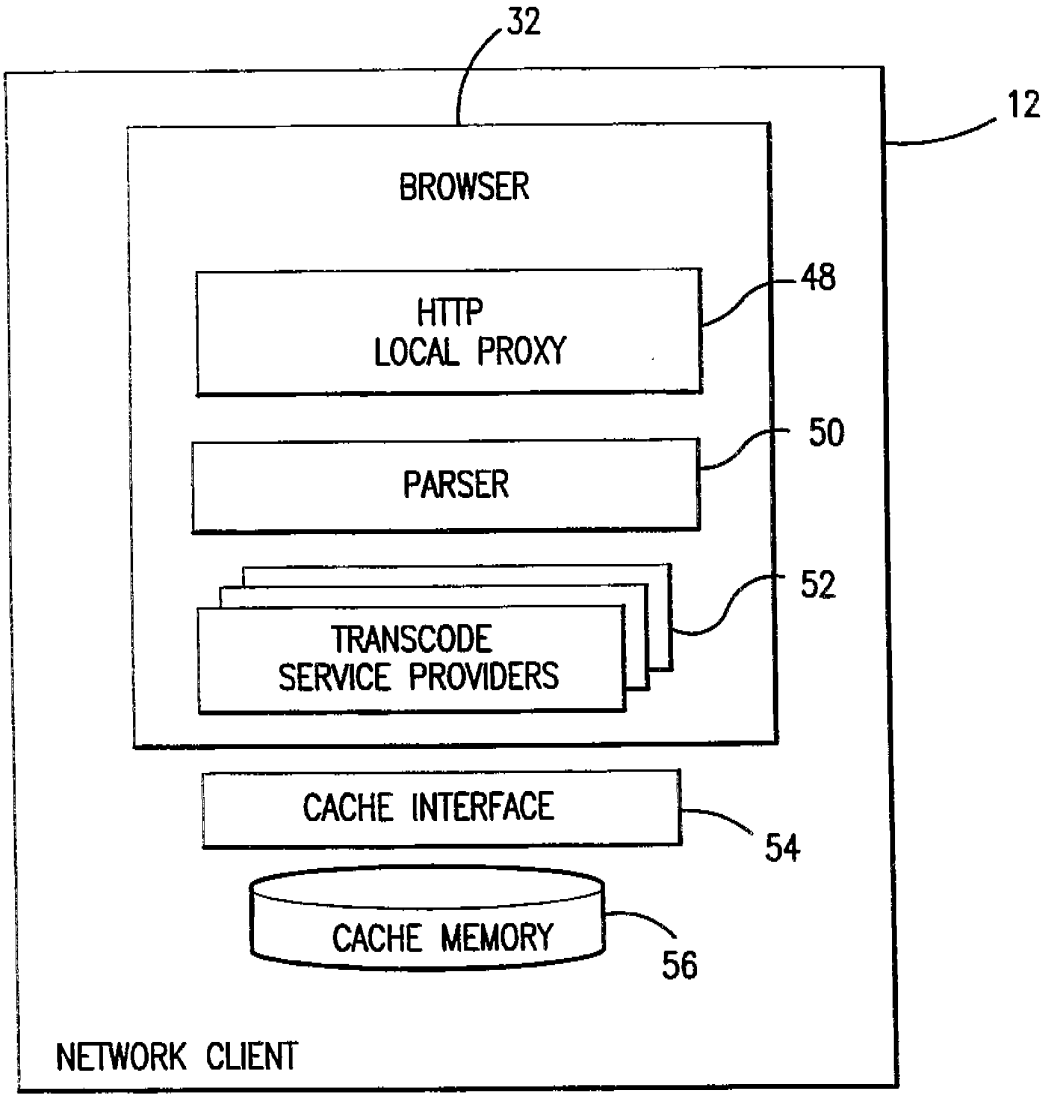

32
12
BROWSER
48
HTTP
LOCAL PROXY
50
PARSER
52
TRANSCODE
SERVICE PROVIDERS
CACHE INTERFACE
54
CACHE MEMORY
56
NETWORK CLIENT

FIG.6

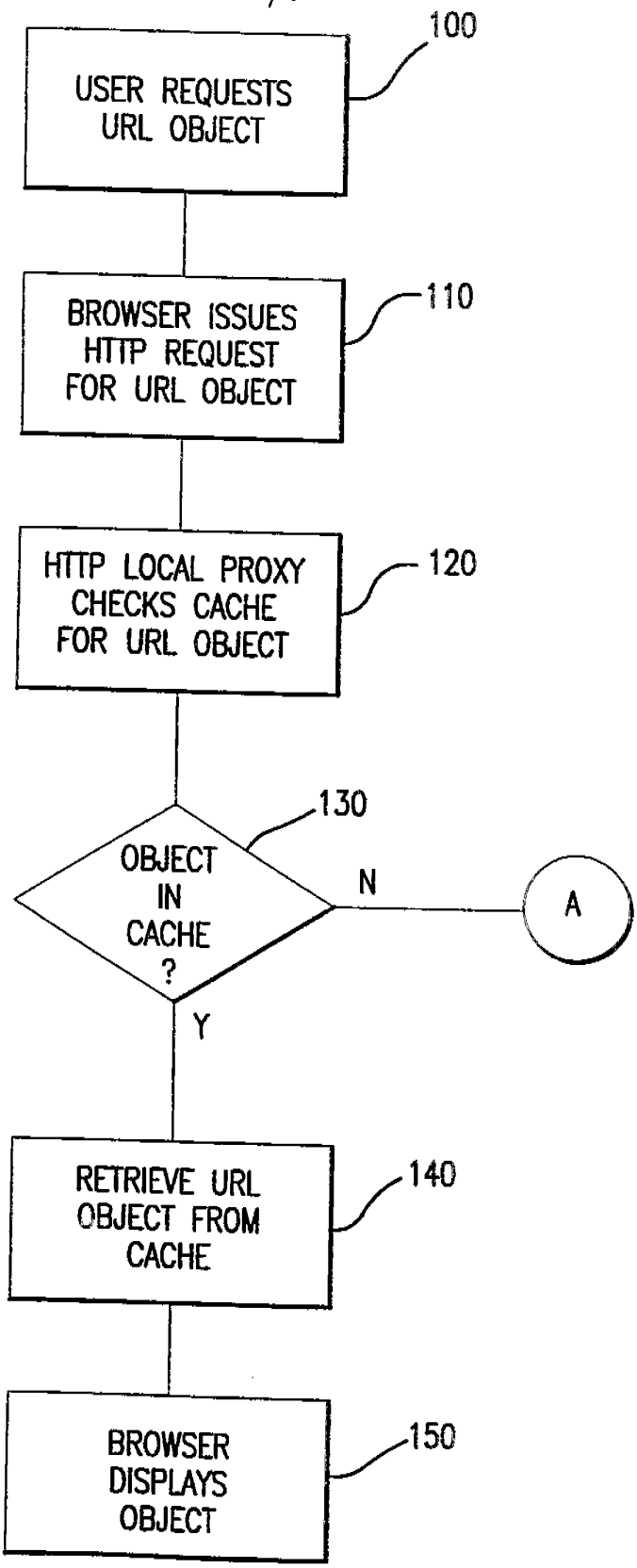



FIG.7



IPR2023-00330 Page 00738

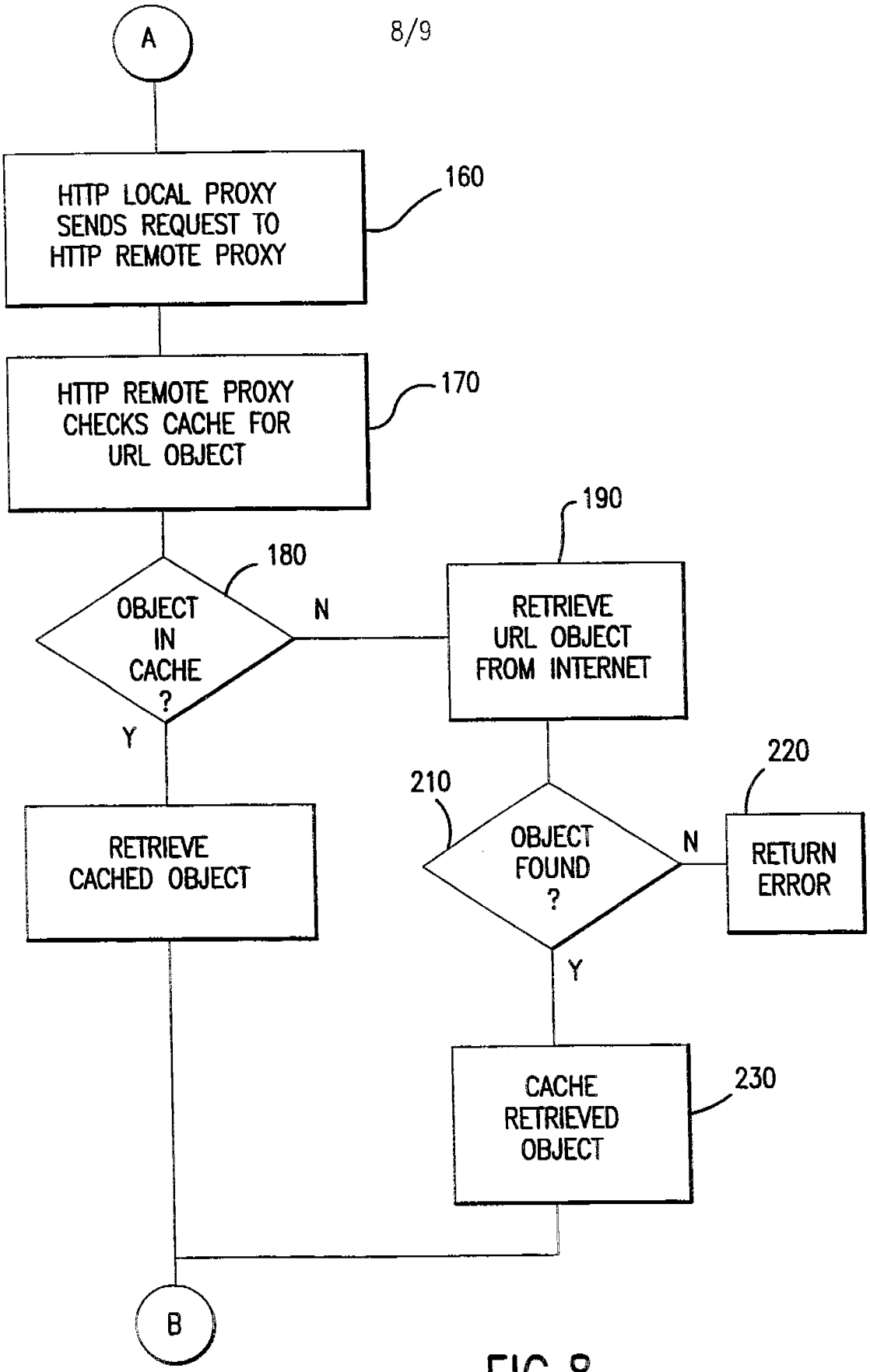

A
HTTP LOCAL PROXY SENDS REQUEST TO HTTP REMOTE PROXY
160
HTTP REMOTE PROXY CHECKS CACHE FOR URL OBJECT
170
190
180
OBJECT IN CACHE ?
N
RETRIEVE URL OBJECT FROM INTERNET
Y
220
210
RETRIEVE CACHED OBJECT
OBJECT FOUND ?
N
RETURN ERROR
Y
CACHE RETRIEVED OBJECT
230
B

FIG.8

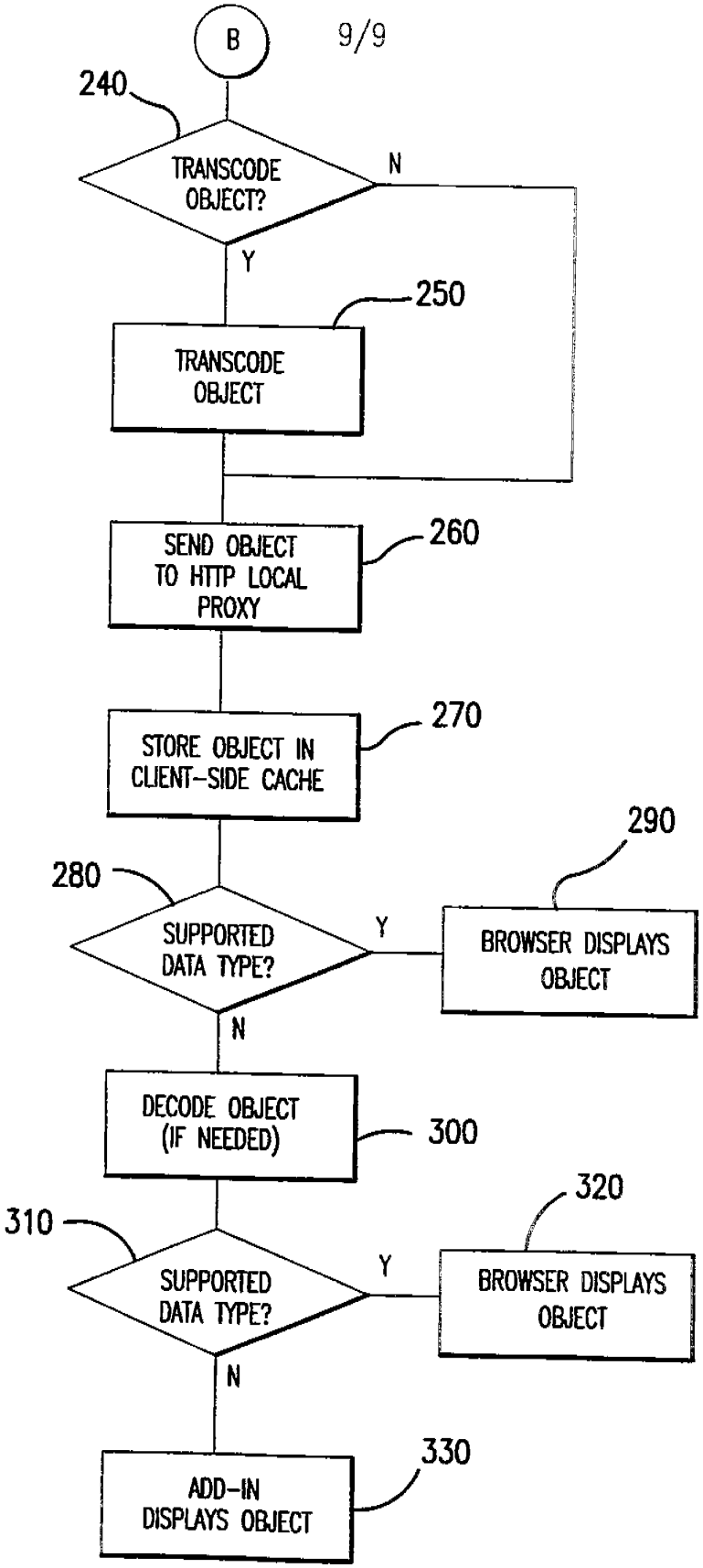

B
240
TRANSCODE OBJECT?
N
Y
250
TRANSCODE OBJECT
SEND OBJECT TO HTTP LOCAL PROXY
260
STORE OBJECT IN CLIENT-SIDE CACHE
270
290
280
SUPPORTED DATA TYPE?
Y
BROWSER DISPLAYS OBJECT
N
DECODE OBJECT (IF NEEDED)
300
320
310
SUPPORTED DATA TYPE?
Y
BROWSER DISPLAYS OBJECT
N
330
ADD-IN DISPLAYS OBJECT

FIG.9

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/05304

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 13/38, 15/17

US CL :395/200.76, 200.48, 200.59; 345/335

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/200.76, 200.48, 200.59, 200.33, 200.47, 200.32, 200.58, 200.77; 345/335

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

NONE

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

USPTO APS

search terms: WWW, URL, HTML, HTTP, parser, configure, data object, selection criteria, forward

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | US 5,544,320 A (KONRAD) 06 August 1996<br>Abstract; Fig. 2; C16:14-C17:18 | 1-25 |
| Y, P | US 5,706,434 A (KREMEN et al.) 06 January 1998<br>Abstract; Fig. 1, 5, 6; C5:21-59; C10:23-C11:50 | 1-25 |
| A, P | US 5,724,556 A (SOUDER et al.) 03 March 1998<br>Abstract; Fig. 6, 9 | 1-25 |
| X, E | US 5,768,510 A (GISH) 16 June 1998<br>Abstract; Fig. 6-7, 11-13, 26; C5:24-50; C18:60-C20:18; C22:49-C23:38; C25:66-C26:20 | 1-25 |

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents: | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "E" | earlier document published on or after the international filing date | "Y" | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | "&" | document member of the same patent family |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| | |
|---|---|
| Date of the actual completion of the international search<br>03 JULY 1998 | Date of mailing of the international search report<br>13 AUG 1998 |
| Name and mailing address of the ISA/US<br>Commissioner of Patents and Trademarks<br>Box PCT<br>Washington, D.C. 20231 | Authorized officer<br>MARK H. RINEHART |

# IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

| | | |
|---|---|---|
| First Named Inventor | : | Sean BARGER |
| Serial No. | : | 12/713,637 |
| Filed | : | February 26, 2010 |
| Art Unit | : | 2455 |
| Confirmation Number | : | 5103 |
| Examiner | : | Charles C. Murphy |
| Title | : | Automated Media Delivery System |
| Attorney Docket No. | : | EQUI0016D2 |

July 1, 2011

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

## INFORMATION DISCLOSURE STATEMENT

Examiner:

This Information Disclosure Statement is submitted:

(X) under 37 CFR 1.97(b), or
(Within three months of filing national application; or date of entry of international application; or before mailing date of first office action on the merits; whichever occurs last)

( ) under 37 CFR 1.97(c) together with either a:
- ( ) Certification under 37 CFR 1.97(e), or
- ( ) a $180.00 fee under 37 CFR 1.17(p), or

(After the CFR 1.97(b) time period, but before final action or notice of allowance, whichever occurs first)

( ) under 37 CFR 1.97(d) together with a:
- ( ) Certification under 37 CFR 1.97(e), and
- ( ) a $180.00 fee under 37 CFR 1.17(p).

(Filed after final action or notice of allowance, whichever occurs first, but before payment of the issue fee)

(X) The Commissioner is authorized to charge any additional fees or credit any overpayment to Deposit Account No. 07-1445 (Order No. EQUI0016D2).

(X) Applicant(s) submit herewith PTO Form 1449 (Modified) -- Information Disclosure Citation together with copies of patents, publications or other information of which applicant(s) are aware, which applicant(s) believe(s) may be material to the examination of this application and for which there may be a duty to disclose in accordance with 37 CFR 1.56.

It is requested that the information disclosed herein be made of record in this application.

Respectfully Submitted,



Michael A. Glenn
Reg. No. 30,176

Customer No. 22862

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | Sean BARGER | EQUI0016D2 | 5103 |

22862 7590 06/01/2012

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

| EXAMINER | |
|---|---|
| MURPHY, CHARLES C | |
| ART UNIT | PAPER NUMBER |
| 2455 | |

| NOTIFICATION DATE | DELIVERY MODE |
|---|---|
| 06/01/2012 | ELECTRONIC |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

ptomatters@glenn-law.com

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 12/713,637 | BARGER ET AL. | |
| | Examiner | Art Unit | |
| | CHARLES MURPHY | 2455 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *30 June 2011*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

4)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

5)☒ Claim(s) *1 and 3-23* is/are pending in the application.

5a) Of the above claim(s) _____ is/are withdrawn from consideration.

6)☐ Claim(s) _____ is/are allowed.

7)☒ Claim(s) *1 and 3-23* is/are rejected.

8)☐ Claim(s) _____ is/are objected to.

9)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

10)☐ The specification is objected to by the Examiner.

11)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

12)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *07/01/2011*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

## *Detailed Action*

**1.** This Office Action is responsive the Amendment filed on 06/30/2011. Claims 1, 3-11 have been amended. Claims 1, 3-23 remain pending for examination. Claims 2 have been cancelled from examination by applicant.

### Information Disclosure Statement

The information disclosure statement (IDS) submitted was filed on 07/01/2011. The submission is in compliance with the provisions of 37 CFR 1.97. Accordingly, the information disclosure statement is being considered by the examiner

## *Claim Objections*

Claims 13-14 are objected to because of the following informalities: Claims 13-14 are dependent on claim 2 instead of claim 12, but claim 2 has been cancelled by the Applicant. Appropriate correction is required.

### Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claim 12-23 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

As per claim 12, the claim limitation recites "computer program product". However, the usage of the phrase "computer program product" is broad enough to include both "non-transitory" and "transitory" (moving electrons, etc) media. The specification does not clearly limit the utilization of a non-transitory computer readable medium and, thus does not constitute functional descriptive material. Therefore, when the broadest reasonable interpretation of a claim covers a signal per se, the claim must be rejected under 35 U.S.C. § 101 as covering non-statutory subject matter. See *In re Nuijten,* 500 F.3d 1346, 1356-57 (Fed. Cir. 2007) (transitory embodiments are not directed to statutory subject matter).

The USPTO recognizes that applicants may have claims directed to computer readable media that cover signals *per se*, which the USPTO must reject under 35 U.S.C. § 101 as covering both non-statutory subject matter and statutory subject matter. In an effort to assist the patent community in overcoming a rejection or potential rejection under 35 U.S.C. § 101 in this situation, the USPTO suggests the following approach. A claim drawn to such a computer readable medium that covers both ***transitory*** and ***non-transitory*** embodiments may be amended to narrow the claim to cover only statutory embodiments to avoid a rejection under 35 U.S.C. § 101 by adding the limitation "***non-transitory***" to the claim. *Cf. Animals - Patentability*, 1077 *Off. Gaz. Pat. Office* 24 (April 21, 1987) (suggesting that applicants add the limitation "non-human" to a claim covering a multi-cellular organism to avoid a rejection under 35 U.S.C. § 101). Such an amendment would typically not raise the issue of new matter,

even when the specification is silent because the broadest reasonable interpretation relies on the ordinary and customary meaning that includes signals *per se*. The limited situations in which such an amendment could raise issues of new matter occur, for example, when the specification does not support a non-transitory embodiment because a signal *per se* is the only viable embodiment such that the amended claim is impermissibly broadened beyond the supporting disclosure. *See*, *e.g.*, *Gentry Gallery, Inc. v. Berkline Corp.*, 134 F.3d 1473 (Fed. Cir. 1998).

The dependent claims included in the statement of rejection but not specifically addressed in the body of the rejection have inherited the deficiencies of their parent claim and have not resolved the deficiencies. Therefore, they are rejected based on the same rationale as applied to their parent claims above.

***Claim Rejections - 35 USC § 103***

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

1. **Claims 1 and 3-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over Suzuki et al. US Patent 6,463,445 (hereinafter referred to as Suzuki) in view of Bhagwat et al. US Patent 6,563,517 (hereinafter referred to as Bhagwat)**

**in further view of Krishnaswamy et al. US Patent 6,909,708 (hereinafter referred to as Krishnaswamy)**

**2.** Regarding claim 1, **Suzuki** teaches an non transitory computer program product comprising:

one or more computer-readable physical and/or memory storage media, the one or more computer-readable memory media having stored thereon computer executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following ***(Suzuki, col. 15, lines 6-50)***

an act of receiving a request for media content to be delivered to a client presentation system;*(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 55-67)***

at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system; *(**Suzuki** teaches a transcoding system using client identification information received at the time of request to determine the transcoding parameters required to*

*properly display requested content on a client device. )* ***(Suzuki, col. 7, lines 35-67)***

an act of causing the transcoded media content to be delivered to the client presentation system; and *(**Suzuki** delivering transcoded content to a requesting client )* ***(Suzuki, col. 7, lines 23-36)***

**Suzuki** does not teach:

an act of caching the transcoded media content.

the act of receiving a request for media content wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

In an analogous art, **Bhagwat** teaches:

an act of caching the transcoded media content. *(**Bhagwat** teaches caching a transcoded media file)****(Bhagwat, col. 5, lines 15-25 and 50-57; cool 6, lines 45-50)***

the act of receiving a request for media content wherein the requested media content has a limited number of base transcoding profiles

associated therewith, each base transcoding profile corresponding to a cached version of the requested media content; *(**Bhagwat** teaches a multi resolution cache in a transcoding proxy. This multi resolution cache stores the original (base) file that is to be transcoded as well as various reduced resolution versions of the original file that have been transcoded for specific devices. The parameters used to transcode this base file into a specified media can be considered transcoding profiles used to transcode the base file. )**(Bhagwat, col. 5, lines 10-25 and 50-57; cool 6, lines 45-50, col. 13, lines 20-40)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**Suzuki-Bhagwat** does not teach:

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile;

In an analogous art, **Krishnaswamy** teaches:

> an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile; *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)**(Krishnaswamy, col. 135, lines 25-35)***

> At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat** and **Krishnaswamy** before him or her, to modify **Suzuki-Bhagwat** to include the teachings of **Krishnaswamy** because this would create a system where users can manage and control a network while a network operator can maintain network quality and routing selection (**Krishnaswamy, col. 2, lines 60-63)**

**3.** Regarding claim 3, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, wherein:

the act of receiving a request for media content is performed by receiving the request over the Internet. *(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 15-20 and 55-67)***

**4.** Regarding claim 4, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, wherein:

an act of receiving a request for the media content to be delivered to a second client presentation system; *(**Bhagwat** teaches a client computer and a web server communicating with each other over the internet)* ***(Bhagwat, col. 3, lines 9-15)***

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the second client presentation system are the same as the transcoding parameters that were applied to the requested media content prior to delivery to the first client presentation system; and*(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media object is provided to the requesting client.)****(Bhagwat, col. 5, lines 50-67)***

an act of delivering the cached transcoded media content, rather than separately transcoding the requested media content again for the second client presentation system. *(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media object is provided to the requesting client.)**(Bhagwat, col. 5, lines 50-67)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**5.** Regarding claim 5, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, wherein:

the requested media content is video media. *(**Suzuki** teaches a user sending a request for a video from a client to a server system) **(Suzuki, col. 6, lines 35-55)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat**

because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**6.** Regarding claim 6, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, wherein:

the identified transcoding parameters includes at least video size parameters. *(**Suzuki** teaches a user sending a request for content from a client to a server system with transcoding parameters including the display size of a requesting client. The display size of a requesting client is used by a transcoder to alter the video size of a requested video so it is displayed correctly on the client screen. As such, the display size is considered the video size parameter for transcoding by the Examiner)* ***(Suzuki, col. 6, lines 50-55)***

**7.** Regarding claim 7, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, comprising:

prior to the act of transcoding, an act of determining that a transcoded representation of the requested media content transcoded using the identified transcoding parameters does not already exist. *(**Bhagwat** teaches checking if a request media file is cached at the requested*

*resolution or transcoding level. These parameters are what is used to transcode the media file.)**(Bhagwat, col. 5, lines 50-60)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**8.** Regarding claim 8, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, wherein:

the act of identifying transcoding parameters comprises an act of selecting a transcoding profile from a set of pre-existing transcoding profiles. *(**Bhagwat** teaches choosing a set of transcoding parameters based on user's preferences.)**(Bhagwat, col. 13, lines30-40)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**9.** Regarding claim 9, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 1, comprising:

an act of reading one or more transcoding parameters from the request; and *(**Suzuki** teaches a user sending a request for content from a client to a server system with transcoding parameters of a requesting client This is encoded in the Client Information Signal)* ***(Suzuki, col. 6, lines 35-55)***

an act of identifying the one or more transcoding parameters from the request as being the identified transcoding parameters to be applied. *(**Suzuki** teaches a user sending a request for content from a client to a server system with transcoding parameters of a requesting client This is encoded in the Client Information Signal)* ***(Suzuki, col. 6, lines 35-55)***

**10.** Regarding claim 10, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 2, comprising:

an act of selecting a pre-existing base transcoded version of the requested media content that has been transcoded in accordance with only a portion of the identified transcoding parameters; and *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)****(Krishnaswamy, col. 135, lines 25-35)***

an act of incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters. *(**Bhagwat** teaches incrementally transcoding a file)**(Bhagwat, col. 12, lines 14-16)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat** and **Krishnaswamy** before him or her, to modify **Suzuki-Bhagwat** to include the teachings of **Krishnaswamy** because this would create a system where users can manage and control a network while a network operator can maintain network quality and routing selection (**Krishnaswamy, col. 2, lines 60-63)**

**11.** Regarding claim 12, **Suzuki** teaches an non transitory computer program product comprising:

one or more computer-readable physical and/or memory storage media, the one or more computer-readable memory media having stored thereon computer executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following ***(Suzuki, col. 15, lines 6-50)***

an act of receiving a request for media content to be delivered to a client presentation system,*(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 55-67)***

an act of identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system; *(**Suzuki** teaches a transcoding system using client identification information received at the time of request to determine the transcoding parameters required to properly display requested content on a client device. )* ***(Suzuki, col. 7, lines 35-67)***

an act of delivering the transcoded media content to the client presentation system; *(**Suzuki** delivering transcoded content to a requesting client )* ***(Suzuki, col. 7, lines 23-36)***

**Suzuki** does not teach:

maintaining a transcoding profile library, the transcoding profile library comprising a plurality of transcoding profiles for media content;

wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

determining whether an existing base transcoding profile exists within the transcoding profile library corresponds to the requested media content transcoded for delivery to the client presentation system;

adding a new transcoding profile to the transcoding profile library corresponding to the identified transcoding parameters and the base transcoding profile; and

an act of caching the transcoded media content.

In an analogous art, **Bhagwat** teaches:

maintaining a transcoding profile library, the transcoding profile library comprising a plurality of transcoding profiles for media content; *(**Bhagwat** teaches a multi resolution cache in a transcoding proxy. This multi resolution cache stores the original (base) file that is to be transcoded as well as various reduced resolution versions of the original file that have been transcoded for specific devices. The parameters used to transcode this base file into a specified media can be considered transcoding profiles*

*used to transcode the base file. )****(Bhagwat, col. 5, lines 10-25 and 50-57; col 6. lines 45-50, col. 13, lines 20-40)***

wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content; *(****Bhagwat*** *teaches a multi resolution cache in a transcoding proxy. This multi resolution cache stores the original (base) file that is to be transcoded as well as various reduced resolution versions of the original file that have been transcoded for specific devices. The parameters used to transcode this base file into a specified media can be considered transcoding profiles used to transcode the base file. )****(Bhagwat, col. 5, lines 10-25 and 50-57; cool 6, lines 45-50, col. 13, lines 20-40)***

determining whether an existing base transcoding profile exists within the transcoding profile library corresponds to the requested media content transcoded for delivery to the client presentation system; *(****Bhagwat*** *teaches determining if an up to date transcoded version of a requested file is available. If not it also determines if an up to date base version of a file is available. )****(Bhagwat, col. 5, lines 50-67)***

adding a new transcoding profile to the transcoding profile library corresponding to the identified transcoding parameters and the base transcoding profile; and an act of caching the transcoded media content. *(**Bhagwat** teaches is an up to dated transcoded version of a base file is not available, the base file is transcoded into the requested format and the newly transcoded file is sent to the requested client as stored on the server with is version number and transcoding characteristics. )**(Bhagwat, col. 5, lines 50-67, col. 6, lines 1-15)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**Suzuki-Bhagwat** does not teach:

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding based on multiple transcoding parameters on top of a base transcoding profile;

In an analogous art, **Krishnaswamy** teaches:

> an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding based on multiple transcoding parameters on top of a base transcoding profile; *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)**(Krishnaswamy, col. 135, lines 25-35)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat** and **Krishnaswamy** before him or her, to modify **Suzuki-Bhagwat** to include the teachings of **Krishnaswamy** because this would create a system where users can manage and control a network while a network operator can maintain network quality and routing selection (**Krishnaswamy, col. 2, lines 60-63)**

**12.** Regarding claim 13, **Suzuki-Bhagwat**-**Krishnaswamy** further teaches the computer program product of claim 12, wherein:

the act of receiving a request for media content is performed by receiving the request over the Internet. *(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 15-20 and 55-67)***

**13.** Regarding claim 14, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 12, wherein:

an act of receiving a request for the media content to be delivered to a second client presentation system; *(**Bhagwat** teaches a client computer and a web server communicating with each other over the internet)* ***(Bhagwat, col. 3, lines 9-15)***

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the second client presentation system are the same as the transcoding parameters that were applied to the requested media content prior to delivery to the first client presentation system; and*(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media*

an act of delivering the cached transcoded media content, rather than separately transcoding the requested media content again for the second

client presentation system. *(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media object is provided to the requesting client.)**(Bhagwat, col. 5, lines 50-67)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**14.** Regarding claim 15, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 12, wherein:

act of caching the transcoded media content comprises an act of first caching a transcoded stream of the transcoded media content while the act of transcoding is still occurring, and further caching a transcoded file of the transcoded media content after the act of transcoding has completed.*(The Applicants Specification teaches generating intermediate images during the transcoding processes. These intermediate images are then cached so they not need be generated again **(Specification, page 30, lines 5-16)** **Bhagwat** teaches a similar method using he Bit Map of a decoded JPEG image. This Bit Map is cached after the JPEG decoding*

*and then further used to generate other resource and derivative images based on a client's transcoding parameters.)**(Bhagwat, col. 5, lines 30-50)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**15.** Regarding claim 16, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 15, wherein:

an act of providing the transcoded stream to the second client presentation system. *(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media object is provided to the requesting client.)**(Bhagwat, col. 5, lines 50-67)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding

system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**16.** Regarding claim 17, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 15, wherein:

an act of providing the transcoded file to the second client presentation system. *(**Bhagwat** teaches first checking a cache after a request is received by a client for a media object. If the media object transcoded with the required parameters is cached then the cached media object is provided to the requesting client.)**(Bhagwat, col. 5, lines 50-67)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**17.** Regarding claim 18, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

the requested media content is video media. *(**Suzuki** teaches a user sending a request for a video from a client to a server system) **(Suzuki, col. 6, lines 35-55)***

**18.** Regarding claim 19, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

> the identified transcoding parameters includes at least video size parameters. *(**Suzuki** teaches a user sending a request for content from a client to a server system with transcoding parameters including the display size of a requesting client. The display size of a requesting client is used by a transcoder to alter the video size of a requested video so it is displayed correctly on the client screen. As such, the display size is considered the video size parameter for transcoding by the Examiner)* ***(Suzuki, col. 6, lines 50-55)***

**19.** Regarding claim 20, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

> prior to the act of transcoding, an act of determining that a transcoded representation of the requested media content transcoded using the identified transcoding parameters does not already exist. *(**Bhagwat** teaches checking if a request media file is cached at the requested resolution or transcoding level. These parameters are what is used to transcode the media file.)* ***(Bhagwat, col. 5, lines 50-60)***
>
> At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki**

before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**20.** Regarding claim 21, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

the act of identifying transcoding parameters comprises an act of selecting a transcoding profile from a set of pre-existing transcoding profiles. *(**Bhagwat** teaches choosing a set of transcoding parameters based on user's preferences.)**(Bhagwat, col. 13, lines30-40)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat** because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**21.** Regarding claim 22, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

an act of reading one or more transcoding parameters from the request; and *(**Suzuki** teaches a user sending a request for content from a client to*

*a server system with transcoding parameters of a requesting client This is encoded in the Client Information Signal)* ***(Suzuki, col. 6, lines 35-55)***

an act of identifying the one or more transcoding parameters from the request as being the identified transcoding parameters to be applied. *(**Suzuki** teaches a user sending a request for content from a client to a server system with transcoding parameters of a requesting client This is encoded in the Client Information Signal)* ***(Suzuki, col. 6, lines 35-55)***

**22.** Regarding claim 23, **Suzuki-Bhagwat-Krishnaswamy** further teaches the computer program product of claim 13, wherein:

an act of selecting a pre-existing base transcoded version of the requested media content that has been transcoded in accordance with only a portion of the identified transcoding parameters; and *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)****(Krishnaswamy, col. 135, lines 25-35)***

an act of incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters. *(**Bhagwat** teaches incrementally transcoding a file)****(Bhagwat, col. 12, lines 14-16)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat** and **Krishnaswamy** before him or her, to modify **Suzuki-Bhagwat** to include the teachings of **Krishnaswamy** because this would create a system where users can manage and control a network while a network operator can maintain network quality and routing selection (**Krishnaswamy, col. 2, lines 60-63)**

**23. Claim 11 rejected under 35 U.S.C. 103(a) as being unpatentable Suzuki et al. US Patent 6,463,445 (hereinafter referred to as Suzuki) in view of Bhagwat et al. US Patent 6,563,517 (hereinafter referred to as Bhagwat) in further view of Krishnaswamy et al. US Patent 6,909,708 (hereinafter referred to as Krishnaswamy) in further view of Garofalakis et al. US Patent 5,845,279 (hereinafter referred to as Garofalakis)**

**24.** Regarding claim 11, **Suzuki-Bhagwat-Krishnaswamy** teaches a method comprising:

an act of receiving a request for media content to be delivered to a client presentation system for media content; *(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 55-67)***

at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system; *(**Suzuki** teaches a transcoding system using client identification information received at the time of request to determine the transcoding parameters required to properly display requested content on a client device. )* ***(Suzuki, col. 7, lines 35-67)***

**Suzuki** does not teach:

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system are the same as transcoding parameters that are being applied to the requested media content prior to delivery to another client presentation system; and

wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

In an analogous art, **Bhagwat** teaches:

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system are the same as transcoding parameters that are being applied to the requested media content prior to delivery to another client presentation system; and ***Bhagwat*** *teaches checking if a request media file is cached at the requested resolution or transcoding level. These parameters are what is used to transcode the media file.)****(Bhagwat, col. 5, lines 50-60)***

wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content; *(****Bhagwat*** *teaches a multi resolution cache in a transcoding proxy. This multi resolution cache stores the original (base) file that is to be transcoded as well as various reduced resolution versions of the original file that have been transcoded for specific devices. The parameters used to transcode this base file into a specified media can be considered transcoding profiles used to transcode the base file. )****(Bhagwat, col. 5, lines 10-25 and 50-57; cool 6, lines 45-50, col. 13, lines 20-40)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Bhagwat** and **Suzuki** before him or her, to modify **Suzuki** to include the teachings of **Bhagwat**

because this would create a more dynamic and flexible transcoding system by allowing the dynamic adjustment of transcoding parameters (**Bhagwat, col. 3, lines 25-30)**

**Suzuki-Bhagwat** does not teach:

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile;

In an analogous art, **Krishnaswamy** teaches:

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile; *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)**(Krishnaswamy, col. 135, lines 25-35)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat** and **Krishnaswamy** before him or her, to modify **Suzuki-Bhagwat** to include the teachings of **Krishnaswamy** because this would create a system

where users can manage and control a network while a network operator can maintain network quality and routing selection (**Krishnaswamy, col. 2, lines 60-63)**

**Suzuki-Bhagwat-Krishnaswamy** does not teach:

an act of delivering the transcoded media content to both client presentation systems concurrently.

In an analogous art, **Garfalakis** teaches:

an act of delivering the transcoded media content to both client presentation systems concurrently.*(**Garofalakis** teaches the ability to deliver media concurrently to multiple systems)**(Garofalakis, col. 2, lines 50-55)***

At the time of the invention, it would have been obvious to one of ordinary skill in the art, having the teachings of **Suzuki-Bhagwat-Krishnaswamy** and **Garfalakis** before him or her, to modify **Suzuki-Bhagwat-Krishnaswamy** to include the teachings of **Garfalakis** because this would create a system that would schedule and transmit various media segments of different popularity and lengths at different times. (**Garfalakis, col. 2, lines 60-63)**

### *Response to Arguments*

Applicant's arguments have been considered but are moot because the arguments do not apply to any of the references being used in the current rejection.

### *Conclusion*

Any inquiry concerning this communication or earlier communications from the examiner should be directed to CHARLES MURPHY whose telephone number is (571)270-5444. The examiner can normally be reached on 8AM - 5PM Monday - Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Emmanuel Moise can be reached on (571) 272-3865. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/CHARLES MURPHY/
Examiner, Art Unit 2455

/EMMANUEL L. MOISE/
Supervisory Patent Examiner, Art Unit 2455

| Notice of References Cited | Application/Control No. 12/713,637 | Applicant(s)/Patent Under Reexamination BARGER ET AL. | |
|---|---|---|---|
| | Examiner CHARLES MURPHY | Art Unit 2455 | Page 1 of 1 |

**U.S. PATENT DOCUMENTS**

| * | | Document Number Country Code-Number-Kind Code | Date MM-YYYY | Name | Classification |
|---|---|---|---|---|---|
| * | A | US-6,563,517 | 05-2003 | Bhagwat et al. | 715/735 |
| * | B | US-6,909,708 | 06-2005 | Krishnaswamy et al. | 370/352 |
| * | C | US-5,845,279 | 12-1998 | Garofalakis et al. | 1/1 |
| * | D | US-6,463,445 | 10-2002 | Suzuki et al. | 1/1 |
| | E | US- | | | |
| | F | US- | | | |
| | G | US- | | | |
| | H | US- | | | |
| | I | US- | | | |
| | J | US- | | | |
| | K | US- | | | |
| | L | US- | | | |
| | M | US- | | | |

**FOREIGN PATENT DOCUMENTS**

| * | | Document Number Country Code-Number-Kind Code | Date MM-YYYY | Country | Name | Classification |
|---|---|---|---|---|---|---|
| | N | | | | | |
| | O | | | | | |
| | P | | | | | |
| | Q | | | | | |
| | R | | | | | |
| | S | | | | | |
| | T | | | | | |

**NON-PATENT DOCUMENTS**

| * | | Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages) |
|---|---|---|
| | U | |
| | V | |
| | W | |
| | X | |

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

| Form 1449 (Modified) | Serial No.: 12/713,637 |
|---|---|
| | Atty. Docket No.: EQUI0016D2 |
| Information Disclosure | First Named Inventor: Sean BARGER |
| Statement By Applicant | Art Unit: 2455 |
| | Confirmation No.: 5103 |
| (Use Several Sheets if Necessary) | Filing Date: February 26, 2010 |

**U.S. Patent Documents**

| Examiner Initials | No. | Patent No. | Issue Date | Patentee | Filing Date |
|---|---|---|---|---|---|
| /C.M./ | 1 | 5,088,052 | 02/1992 | Spielman, et al. | |
| /C.M./ | 2 | 5,355,472 | 10/1994 | Lewis | |
| /C.M./ | 3 | 5,442,771 | 08/1995 | Filepp, et al. | |
| /C.M./ | 4 | 5,530,852 | 06/1996 | Meske, Jr., et al. | |
| /C.M./ | 5 | 5,701,451 | 12/1997 | Rogers, et al. | |
| /C.M./ | 6 | 5,708,845 | 01/1998 | Wistendahl, et al. | |
| /C.M./ | 7 | 5,710,918 | 01/1998 | Lagarde, et al. | |
| /C.M./ | 8 | 5,737,619 | 04/1998 | Judson | |
| /C.M./ | 9 | 5,745,908 | 04/1998 | Anderson, et al. | |
| /C.M./ | 10 | 5,758,110 | 05/1998 | Boss, et al. | |
| /C.M./ | 11 | 5,761,655 | 06/1998 | Hoffman, Michael T. | |
| /C.M./ | 12 | 5,793,964 | 08/1998 | Rogers, et al. | |
| /C.M./ | 13 | 5,819,261 | 10/1998 | Takahashi, et al. | |
| /C.M./ | 14 | 5,822,436 | 10/1998 | Rhoads | |
| /C.M./ | 15 | 5,845,084 | 12/1998 | Cordell, et al. | |
| /C.M./ | 16 | 5,845,279 | 12/1998 | Garofalakis, et al. | |
| /C.M./ | 17 | 5,845,299 | 12/1998 | Arora, et al. | |
| /C.M./ | 18 | 5,860,068 | 01/1999 | Cook | |
| /C.M./ | 19 | 5,860,073 | 01/1999 | Ferrel, et al. | |
| /C.M./ | 20 | 5,861,881 | 01/1999 | Freeman, et al. | |
| /C.M./ | 21 | 5,862,325 | 01/1999 | Reed, et al. | |
| /C.M./ | 22 | 5,864,337 | 01/1999 | Marvin, John | |
| /C.M./ | 23 | 5,870,552 | 02/1999 | Dozier, et al. | |
| /C.M./ | 24 | 5,880,740 | 03/1999 | Halliday, et al. | |
| /C.M./ | 25 | 5,890,170 | 03/1999 | Sidana | |
| /C.M./ | 26 | 5,895,476 | 04/1999 | Orr, et al. | |
| /C.M./ | 27 | 5,895,477 | 04/1999 | Orr, et al. | |
| /C.M./ | 28 | 5,903,892 | 05/1999 | Hoffert, et al. | |
| /C.M./ | 29 | 5,937,160 | 08/1999 | Davis, et al. | |
| /C.M./ | 30 | 5,943,680 | 08/1999 | Ohga, et al. | |
| /C.M./ | 31 | 5,956,737 | 09/1999 | King, et al. | |
| /C.M./ | 32 | 6,009,436 | 12/1999 | Motoyama, et al. | |
| /C.M./ | 33 | 6,456,305 | 09/2002 | Qureshi, et al. | |
| /C.M./ | 34 | 6,483,851 | 11/2002 | Neogi | |
| /C.M./ | 35 | 6,484,149 | 11/2002 | Jemmes, et al. | |

| | | | | | |
|---|---|---|---|---|---|
| /C.M./ | 36 | 6,563,517 | 05/2003 | Bhagwat, et al. | |
| /C.M./ | 37 | 6,591,280 | 07/2003 | Orr | |
| /C.M./ | 38 | 6,623,529 | 09/2003 | Lakritz | |
| /C.M./ | 39 | 6,909,708 | 06/2005 | Krishnaswamy, et al. | |
| /C.M./ | 40 | 6,938,073 | 08/2005 | Mendhekar, et al. | |
| /C.M./ | 41 | 7,284,201 | 10/2007 | Cohen-Solal, Eric | |
| /C.M./ | 42 | 7,477,688 | 01/2009 | Zhang, et al. | |
| /C.M./ | 43 | 7,673,063 | 03/2010 | Xie, et al. | |

**Published U.S, Patent Application**

| Examiner Initials | No. | Publication No. | Publication Date | Applicant |
|---|---|---|---|---|
| /C.M./ | 41 | 2006/0015580 | 01/2006 | Gabriel, et al. |
| /C.M./ | 42 | 2008/0155230 | 06/2008 | Robbins et al |
| /C.M./ | 43 | 2009/0254672 | 10/2009 | Zhang |
| /C.M./ | 44 | 2009/0070485 | 03/2009 | Barger, et al. |
| /C.M./ | 45 | 2009/0089422 | 04/2009 | Barger, et al. |

**Foreign Patent or Published Foreign Patent Application**

| Examiner Initials | No. | Document No. | Publication Date | Applicant |
|---|---|---|---|---|
| /C.M./ | 46 | AU-A-53031/98 | 08/1998 | Dudley, John Mills |
| /C.M./ | 47 | EP 0747842 | 12/1996 | International Business Machines Corp. |
| /C.M./ | 48 | EP 0782085 | 07/1997 | International Business Machines Corp. |
| /C.M./ | 49 | EP 0818907 | 01/1998 | AT&T Corp |
| /C.M./ | 50 | EP 0843276 | 05/1998 | Canon Information Systems Inc. |
| /C.M./ | 51 | EP 0876034 | 11/1998 | International Business Machines Corp. |
| /C.M./ | 52 | EP 0883068 | 12/1998 | Home Information Services, Inc. |
| /C.M./ | 53 | EP 0886409 | 12/1998 | Digital Vision Laboratories Corporation |
| /C.M./ | 54 | EP 0895171 | 02/1999 | Neoforma, Inc. |
| /C.M./ | 55 | EP 0926607 | 06/1999 | Ricoh KK |
| /C.M./ | 56 | EP 0949571 | 10/1999 | Xerox Corp |
| /C.M./ | 57 | WO 97/49252 | 12/1997 | Senthilkumar, et al. |
| /C.M./ | 58 | WO 98/40842 | 09/1998 | Computer Information and Sciences, Inc. |
| /C.M./ | 59 | WO 98/43177 | 10/1998 | Intel Corp |

**Non-Patent Literature Documents**

| Examiner Initials | No. | Author, Title, Date, Place (e.g. Journal) of Publication |
|---|---|---|
| /C.M./ | A | BERINSTEIN, Paula; "The Big Picture; Text and Graphics on UMI's ProQuest Direct: The Best (Yet) of Both Words"; March 1997; retrieved on 3/23/04 from website: http://www.infotoday.com/online/MarOL97/picture3.html, 11 pages. |
| /C.M./ | B | BULTERMAN, DICK.C.A.; Models, Media and Motion: Using the Web to Support Multimedia Documents; Proceedings of 1997 Int'l Conf. on Multimedia Modeling; p. 17-20; November 1997; SINGAPORE |
| /C.M./ | C | DOBSON, R.; Animating Your Web Pages with Direct Animation; Web Techniques; vol.3, no. 6, p. 49-52; June 1998 |

| | | |
|---|---|---|
| /C.M./ | D | McNeil, Sara; Research Interests; retrieved on March 18, 2004 from website: http//www.coe.uh.edu/`smcneil/research.htm, 3 pages. |
| /C.M./ | E | MOHLER, J.L.; Migrating Course Materials to the World Wide Web: A Case Study of the Department of Technical Graphics at Purdue University; Computer Networks and ISDN Systems; Vol. 30, Issues 20-21, pp.1981-1990; November 12, 1988 |
| /C.M./ | F | Sakaguchi, et al.; " A browsing tool for multi-lingual documents for users without multi-lingual fonts"; 1996; ACM International Conference On Digital Libraries, pp. 63-71 |
| /C.M./ | G | Tables of Contents service for Computers & Geosciences; Copyright 1997; Computers and GeoSciences, Volume 23, Issue 5, retrieved on 3/18/04 from website: http://library.iem.ac.ru/comp&geo/00983004/sz977014.html, 2 pages. |
| /C.M./ | H | Zaiane, et al.; "Mining multimedia data"; Nov. 1998; ACM Conference of the Center for Advanced Studies on Collaborative research, pp. 1-18 |

Examiner's Signature: /Charles Murphy/ Date: 05/19/2012

Examiner: Initial citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

| Search Notes | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | Examiner | Art Unit |
| | CHARLES MURPHY | 2455 |

| SEARCHED | | | |
|---|---|---|---|
| Class | Subclass | Date | Examiner |
| 709 | 263 234 231 226 219 217 204 203 200 | 10/7/2010 | Charles Murphy |
| 370 | 552 468 431 353 352 330 280 241 | 10/7/2010 | Charles Murphy |
| 709 | 263 231 226 219 217 204 203 | 3/22/2011 | Charles Murphy |
| 370 | 468 431 353 352 241 | 3/22/2011 | Charles Murphy |
| 709 | 263 234 231 226 219 217 200 | 5/19/2012 | Charles Murphy |

| SEARCH NOTES | | |
|---|---|---|
| Search Notes | Date | Examiner |
| Searched East | 10/7/2010 | Charles Murphy |
| Searched East | 3/22/2011 | Charles Murphy |
| Consulted Wing Chan for potential 101 issue for claims 1-10 | 3/28/2011 | Charles Murphy |
| Searched East | 5/19/2012 | Charles Murphy |

| INTERFERENCE SEARCH | | | |
|---|---|---|---|
| Class | Subclass | Date | Examiner |
| | | | |

| /CHARLES MURPHY/<br>Examiner.Art Unit 2455 | |
|---|---|

**EAST Search History**

**EAST Search History (Prior Art)**

| Ref # | Hits | Search Query | DBs | Default Operator | Plurals | Time Stamp |
|---|---|---|---|---|---|---|
| S1 | 1 | ("20090254672").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2010/10/07 14:44 |
| S2 | 74 | ("5088052" \| "5355472" \| "5442771" \| "5530852" \| "5701451" \| "5708845" \| "5710918" \| "5737619" \| "5745908" \| "5758110" \| "5761555" \| "5761655" \| "5793964" \| "5819261" \| "5822436" \| "5845084" \| "5845299" \| "5860068" \| "5860073" \| "5861881" \| "5862325" \| "5864337" \| "5870552" \| "5880740" \| "5884337" \| "5890170" \| "5890170" \| "5895476" \| "5895476" \| "5895477" \| "5903892" \| "5937160" \| "5943680" \| "5943680" \| "5956737" \| "5956737" \| "6009436" \| "6456305" \| "6484149" \| "6563517" \| "6591280" \| "6591280" \| "6623529" \| "6623529").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:48 |
| S3 | 2 | S2 (transcod$4) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:49 |
| S4 | 74 | ("5088052" \| "5355472" \| "5442771" \| "5530852" \| "5701451" \| "5708845" \| "5710918" \| "5737619" \| "5745908" \| "5758110" \| "5761555" \| "5761655" \| "5793964" \| "5819261" \| "5822436" \| "5845084" \| "5845299" \| "5860068" \| "5860073" \| "5861881" \| "5862325" \| "5864337" \| "5870552" \| "5880740" \| "5884337" \| "5890170" \| "5890170" \| "5895476" \| "5895476" \| "5895477" \| "5903892" \| "5937160" \| "5943680" \| "5943680" \| "5956737" \| "5956737" \| "6009436" \| "6456305" \| "6484149" \| "6563517" \| "6591280" \| "6591280" \| "6623529" \| "6623529").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:51 |
| S5 | 1 | S4 (transcod$4 same cache) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:51 |
| S6 | 34 | (transcod$4) same (content or media or file) near4 (concurrent$3) | US-PGPUB; USPAT; USOCR; FPRS; | AND | ON | 2010/10/07 17:40 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | EPO; JPO; DERWENT; IBM_TDB | | | |
| S7 | 0 | (transcod$4) same (content or media or file) near4 (concurrent$3) @ad<"19991021" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S8 | 0 | (transcod$4) same (content or media or file) near4 (concurrent$3) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S9 | 15 | (transcod$4) (content or media or file) near4 (concurrent$3) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S10 | 10 | (transcod$4) (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:43 |
| S11 | 0 | (transcod$4 same parameter) (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:18 |
| S12 | 2709 | (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S13 | 763 | (content or media or file) near5 (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S14 | 184 | (content or media or file) near5 (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S15 | 41 | S14 (transcod$3 or convert) | US-PGPUB; USPAT; | AND | ON | 2010/10/07 18:26 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | | | |
| S16 | 1 | S14 (transcod$3) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:26 |
| S17 | 25 | S14 (media.ti.) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:27 |
| S18 | 3 | (content or media or file) near5 (transcod$3 or convert$3) same (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:30 |
| S19 | 157 | (content or media or file) near5 (transcod$3 or convert$3) (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:30 |
| S20 | 6 | S19 (media.ti.) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:30 |
| S21 | 22 | (content or media or file or audio or video or movie) near5 (transcod$3 or convert$3) same (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:31 |
| S22 | 1 | ("5845279").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2010/10/07 18:35 |
| S23 | 0 | S22 concurent$3 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:35 |
| S24 | 0 | S22 concurent$4 | US-PGPUB; USPAT; USOCR; FPRS; | AND | ON | 2010/10/07 18:35 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | EPO; JPO; DERWENT; IBM_TDB | | | |
| S25 | 1 | S22 concurrent$4 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:35 |
| S26 | 13 | (transcod$4) same (content or media or file) near4 (base or primary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:10 |
| S27 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:10 |
| S28 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation or prilimnary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:11 |
| S29 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation or preliminary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:12 |
| S30 | 0 | (transcod$4) near5 (profile) same (content or media or file) near4 (base or primary or foundation or preliminary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:13 |
| S31 | 1 | (transcod$4) near5 (profile) same (content or media or file) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:13 |
| S32 | 1 | ("6359902").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/17 17:35 |
| S33 | 0 | S32 cache | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; | AND | ON | 2011/03/17 17:35 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | IBM_TDB | | | |
| S34 | 1 | ("6563517").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:22 |
| S35 | 1 | ("5845279").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:23 |
| S36 | 1 | ("6909708").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:23 |
| S37 | 3 | S34 or S35 or S36 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:23 |
| S38 | 0 | S37 cache same media | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:23 |
| S39 | 2 | S37 cache | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:23 |
| S40 | 1 | S37 cache same transcode$4 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:24 |
| S41 | 1 | ("6563517").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 16:33 |
| S42 | 1 | ("6909708").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 16:33 |
| S43 | 1 | S41 preferences same request | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 16:50 |
| S44 | 0 | (transcod$4 adj profile) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:46 |
| S45 | 2 | (transcod$4 near5 profile) @ay<"1999" | US-PGPUB; USPAT; | AND | ON | 2011/11/18 20:57 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | | | |
| S46 | 106 | US-1182225-$.DID. OR US-6147977-$.DID. OR US-7149893-$.DID. OR US-6829591-$.DID. OR US-6442589-$.DID. OR US-5327486-$.DID. OR US-6892230-$.DID. OR US-6748569-$.DID. OR US-7221658-$.DID. OR US-5400020-$.DID. OR US-5623260-$.DID. OR US-5657010-$.DID. OR US-5668543-$.DID. OR US-5708780-$.DID. OR US-5796394-$.DID. OR US-5870549-$.DID. OR US-5875437-$.DID. OR US-5937161-$.DID. OR US-6029148-$.DID. OR US-6052673-$.DID. OR US-6073166-$.DID. OR US-6151624-$.DID. OR US-6154738-$.DID. OR US-6157945-$.DID. OR US-6185603-$.DID. OR US-6212550-$.DID. OR US-6216165-$.DID. OR US-6219714-$.DID. OR US-6278936-$.DID. OR US-6314402-$.DID. OR US-6313760-$.DID. OR US-6317060-$.DID. OR US-6363254-$.DID. OR US-6363323-$.DID. OR US-6411891-$.DID. OR US-6415207-$.DID. OR US-5899995-$.DID. OR US-6314434-$.DID. OR US-5742763-$.DID. OR US-6513019-$.DID. OR US-5073852-$.DID. OR US-6542515-$.DID. OR US-5805810-$.DID. OR US-5819046-$.DID. OR US-5579472-$.DID. OR US-5828314-$.DID. OR US-6167409-$.DID. OR US-6347340-$.DID. OR US-6430272-$.DID. OR US-6442591-$.DID. OR US-6438583-$.DID. OR US-6836792-$.DID. OR WO-0176119-$.DID. OR WO-0175604-$.DID. OR WO-0176264-$.DID. OR WO-0180130-$.DID. | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:59 |
| S47 | 18 | S46 profile | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:59 |
| S48 | 139 | (transcoding near5 profile) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:00 |
| S49 | 3 | (transcoding near5 profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:01 |
| S50 | 1 | ("7117361").PN. | US-PGPUB; USPAT; | OR | OFF | 2011/11/18 21:02 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | USOCR | | | |
| S51 | 1 | S50 profile | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:02 |
| S52 | 355 | ((transcod$4 or translation) near5 profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:04 |
| S53 | 49 | ((transcod$4 or translation) near profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:04 |
| S54 | 94 | US-5088052-$.DID. OR US-5355472-$.DID. OR US-5442771-$.DID. OR US-5530852-$.DID. OR US-5701451-$.DID. OR US-5708845-$.DID. OR US-5710918-$.DID. OR US-5737619-$.DID. OR US-5745908-$.DID. OR US-5758110-$.DID. OR US-5761655-$.DID. OR US-5793964-$.DID. OR US-5819261-$.DID. OR US-5822436-$.DID. OR US-5845084-$.DID. OR US-5845279-$.DID. OR US-5845299-$.DID. OR US-5860068-$.DID. OR US-5860073-$.DID. OR US-5861881-$.DID. OR US-5862325-$.DID. OR US-5864337-$.DID. OR US-5870552-$.DID. OR US-5880740-$.DID. OR US-5890170-$.DID. OR US-5895476-$.DID. OR US-5895477-$.DID. OR US-5903892-$.DID. OR US-5937160-$.DID. OR US-5943680-$.DID. OR US-5956737-$.DID. OR US-6009436-$.DID. OR US-6456305-$.DID. OR US-6483851-$.DID. OR US-6484149-$.DID. OR US-6563517-$.DID. OR US-6591280-$.DID. OR US-6623529-$.DID. OR US-6909708-$.DID. OR US-7284201-$.DID. OR US-7477688-$.DID. OR US-7673063-$.DID. OR US-20080155230-$.DID. OR US-20090254672-$.DID. OR US-5303198-$.DID. OR EP-0747842-$.DID. OR EP-0747842121996-$.DID. OR EP-0782085-$.DID. OR EP-0782085071997-$.DID. OR EP-0818907-$.DID. OR EP-0843276-$.DID. OR EP-0843276051998-$.DID. OR EP-0876034-$.DID. OR EP-0876034111998-$.DID. OR EP-0883068-$.DID. OR US-1211998-$.DID. OR EP-0886409-$.DID. OR EP-0895171-$.DID. OR EP-0926607-$.DID. OR EP-0926607061999-$.DID. OR EP-0949571-$.DID. OR EP-0949571101999-$.DID. OR WO-97149252-$.DID. OR US-1211997-$.DID. OR WO-9840842-$.DID. OR WO-9840842091998- | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:18 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | $.DID. OR WO-9843177-$.DID. OR WO-9843177101998-$.DID. OR US-9830041-$.DID. | | | | |
| S55 | 0 | S54 (transcode same automatic$5 near5 client) | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:21 |
| S56 | 1 | S54 (transcode same automatic$5 ) | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:21 |
| S57 | 42 | (transcod$4 near5 automatic$4) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:24 |
| S58 | 1 | S57 (parameter near5 client) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:24 |
| S59 | 2 | S57 (parameter near5 (client or device or user)) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:26 |
| S60 | 7 | S57 (ident$6 near5 (client or device or user)) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:45 |
| S61 | 2 | ("6563517").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | OR | OFF | 2012/05/18 23:06 |

**EAST Search History (Interference)**

<This search history is empty>

**5/19/2012 8:20:53 PM**
**C:\Users\cmurphy1\Documents\EAST\Workspaces\12713637.wsp**

| Index of Claims | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | Examiner | Art Unit |
| | CHARLES MURPHY | 2455 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ✓ | Rejected | - | Cancelled | N | Non-Elected | A | Appeal |
| = | Allowed | ÷ | Restricted | I | Interference | O | Objected |

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

| CLAIM | | DATE | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Final | Original | 10/07/2010 | 03/22/2011 | 05/19/2012 | | | | | | |
| | 1 | ✓ | ✓ | ✓ | | | | | | |
| | 2 | ✓ | - | - | | | | | | |
| | 3 | ✓ | ✓ | ✓ | | | | | | |
| | 4 | ✓ | ✓ | ✓ | | | | | | |
| | 5 | ✓ | ✓ | ✓ | | | | | | |
| | 6 | ✓ | ✓ | ✓ | | | | | | |
| | 7 | ✓ | ✓ | ✓ | | | | | | |
| | 8 | ✓ | ✓ | ✓ | | | | | | |
| | 9 | ✓ | ✓ | ✓ | | | | | | |
| | 10 | ✓ | ✓ | ✓ | | | | | | |
| | 11 | ✓ | ✓ | ✓ | | | | | | |
| | 12 | | | ✓ | | | | | | |
| | 13 | | | ✓ | | | | | | |
| | 14 | | | ✓ | | | | | | |
| | 15 | | | ✓ | | | | | | |
| | 16 | | | ✓ | | | | | | |
| | 17 | | | ✓ | | | | | | |
| | 18 | | | ✓ | | | | | | |
| | 19 | | | ✓ | | | | | | |
| | 20 | | | ✓ | | | | | | |
| | 21 | | | ✓ | | | | | | |
| | 22 | | | ✓ | | | | | | |
| | 23 | | | ✓ | | | | | | |

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | | |
|---|---|---|
| First Named Inventor | : | Sean Barger |
| Serial No. | : | 12/713,637 |
| Filed | : | February 26, 2010 |
| Art Unit | : | 2455 |
| Confirmation Number | : | 5103 |
| Examiner | : | Charles C. Murphy |
| Title | : | Automated Media Delivery System |
| Attorney Docket No. | : | EQUI0016D2 |

November 1, 2012

Mail Stop AMENDMENT
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

**RESPONSE TO OFFICE ACTION**

Applicant submits this Response to the Office Action dated June 1, 2012 in connection with the above-identified patent application.

A **Listing of Claims** begins on Page 2 of this paper, and

**Remarks** begin on Page 11 of this paper.

The Commissioner is authorized to charge any fees that may be due and to credit any overpayments to Deposit Account 07-1445 (Order No. EQUI0016D2). Applicant considers this document to be filed in a timely manner.

## LISTING OF CLAIMS

1. (Currently Amended) A non-transitory computer program product for causing a computing system to dynamically transcode media content to be presented on a client presentation system, the computer program product comprising:

one or more computer-readable media, the one or more computer-readable media having stored thereon computer-executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following:

an act of receiving a request for media content to be delivered to a client presentation system, wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system;

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile;

wherein the act of act of transcoding the requested media content in accordance with the identified transcoding parameters comprises:

an act of selecting a pre-existing base transcoded version of the requested media content comprising intermediate derivative media that has been transcoded in accordance with only a portion of the identified transcoding parameters; and

an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters;

an act of causing the transcoded media content to be delivered to the client presentation system; and

an act of caching the transcoded media content.

2. (Cancelled).

3. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the act of receiving a request for media content is performed by receiving the request over the Internet.

4. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the client presentation system is a first client presentation system, the method further comprising:

an act of receiving a request for the media content to be delivered to a second client presentation system;

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the second client presentation system are the same as the transcoding parameters that were applied to the requested media content prior to delivery to the first client presentation system; and

an act of delivering the cached transcoded media content, rather than separately transcoding the requested media content again for the second client presentation system.

5. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the requested media content is video media.

6. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the identified transcoding parameters includes at least video size parameters.

7. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the method further comprises:

prior to the act of transcoding, an act of determining that a transcoded representation of the requested media content transcoded using the identified transcoding parameters does not already exist.

8. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the act of identifying transcoding parameters comprises an act of selecting a transcoding profile from a set of pre-existing transcoding profiles.

9. (Previously Presented) A non-transitory computer program product in accordance with claim 1, wherein the act of identifying transcoding parameters comprises:

an act of reading one or more transcoding parameters from the request; and

an act of identifying the one or more transcoding parameters from the request as being the identified transcoding parameters to be applied.

10. (Cancelled)

11. (Currently Amended) A method for accessing dynamically transcoding media content, the method comprising:

an act of receiving a request for media content to be delivered to a client presentation system for media content, wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system;

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system are the same as transcoding parameters that are being applied to the requested media content prior to delivery to another client presentation system;

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile;

wherein the act of act of transcoding the requested media content in accordance with the identified transcoding parameters comprises:

an act of selecting a pre-existing base transcoded version of the requested media content comprising intermediate derivative media that has been transcoded in accordance with only a portion of the identified transcoding parameters; and

an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters; and

an act of delivering the transcoded media content to both client presentation systems concurrently.

12. (Currently Amended) A non-transitory computer program product for causing a computing system to dynamically transcode media content to be presented on a client presentation system, the computer program product comprising:

one or more computer-readable physical and/or memory storage media, the one or more computer-readable memory media having stored thereon computer-executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following:

maintaining a transcoding profile library, the transcoding profile library comprising a plurality of transcoding profiles for media content;

an act of receiving a request for media content to be delivered to a client presentation system, wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;

an act of identifying transcoding parameters to be applied to the requested media content prior to delivery to the client presentation system;

determining whether an existing base transcoding profile exists within the transcoding profile library corresponds to the requested media content transcoded for delivery to the client presentation system;

an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding based on multiple transcoding parameters on top of a base transcoding profile;

wherein the act of act of transcoding the requested media content in accordance with the identified transcoding parameters comprises:

an act of selecting a pre-existing base transcoded version of the requested media content comprising intermediate derivative media that

has been transcoded in accordance with only a portion of the identified transcoding parameters; and

an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters;

an act of delivering the transcoded media content to the client presentation system;

adding a new transcoding profile to the transcoding profile library corresponding to the identified transcoding parameters and the base transcoding profile; and

an act of caching the transcoded media content.

13. (Currently Amended) A computer program product in accordance with Claim 12, wherein the act of receiving a request for media content is performed by receiving the request over the Internet.

14. (Currently Amended) A computer program product in accordance with Claim 12, wherein the client presentation system is a first client presentation system, the method further comprising:

an act of receiving a request for the media content to be delivered to a second client presentation system;

an act of determining that the transcoding parameters to be applied to the requested media content prior to delivery to the second client presentation system are

the same as the transcoding parameters that were applied to the requested media content prior to delivery to the first client presentation system; and

an act of delivering the cached transcoded media content, rather than separately transcoding the requested media content again for the second client presentation system.

15. (Previously Presented) A computer program product in accordance with Claim 14, wherein act of caching the transcoded media content comprises an act of first caching a transcoded stream of the transcoded media content while the act of transcoding is still occurring, and further caching a transcoded file of the transcoded media content after the act of transcoding has completed.

16. (Previously Presented) A computer program product in accordance with Claim 15, wherein the act of delivery the cached transcoded media content comprises:

an act of providing the transcoded stream to the second client presentation system.

17. (Previously Presented) A computer program product in accordance with Claim 15, wherein the act of delivery the cached transcoded media content comprises:

an act of providing the transcoded file to the second client presentation system.

18. (Previously Presented) A computer program product in accordance with Claim 13, wherein the requested media content is video media.

19. (Previously Presented) A computer program product in accordance with Claim 13, wherein the identified transcoding parameters includes at least video size parameters.

20. (Previously Presented) A computer program product in accordance with Claim 13, wherein the method further comprises:

prior to the act of transcoding, an act of determining that a transcoded representation of the requested media content transcoded using the identified transcoding parameters does not already exist.

21. (Previously Presented) A computer program product in accordance with Claim 13, wherein the act of identifying transcoding parameters comprises an act of selecting a transcoding profile from a set of pre-existing transcoding profiles.

22. (Previously Presented) A computer program product in accordance with Claim 13, wherein the act of identifying transcoding parameters comprises:

an act of reading one or more transcoding parameters from the request; and

an act of identifying the one or more transcoding parameters from the request as being the identified transcoding parameters to be applied.

23. (Cancelled)

## REMARKS

Applicants respectfully request further examination and reconsideration in view of the above amendments and arguments set forth fully below.

### Claim Objections

Claims 13 and 14 are objected to because they depend from cancelled Claim 2. Applicant has amended Claims 13 and 14 herein to correct their dependency to that of Claim 12, thereby rendering the objections moot.

### Claim Rejections under 35 U.S.C. § 101

Claims 12-23 rejected under 35 USC 101 for allegedly being directed to non-statutory subject matter. Applicant has amended Claim 12 herein to recite that the claim is drawn to a non-transitory computer program product, thereby rendering the rejection moot.

### Claim Rejections under 35 U.S.C. § 103

Claims 1 and 3-23 (noted in the Office Action on page 4 as Claims 1 and 3-10, but noted correctly in the Office Action Summary) are rejected under 35 U.S.C. § 103(a) as being unpatentable over United States Patent No. 6,463,445 to Suzuki *et al* in view of United States Patent No. 6,563,517 to Bhagwat (hereinafter referred to as "Bhagwat") in

further view of United States Patent No. 6,909,708 to Krishnaswamy *et al* (hereinafter referred to as "Krishnaswamy").

Applicant thanks the Examiner for his courtesy and assistance during an interview that was held on August 24, 2012. During the interview, Applicant discussed the subject matter of Claim 10. It was agreed that this subject matter, if expressed in connection with the creation of a further version, such as an appended version or a new version, would overcome the present rejection thereof.

Applicant has amended Claims 1, 11, and 12 to include limitations similar in scope to those of originally submitted Claim 10, modified in accordance with the discussion held with the Examiner during the above-identified interview to recite an intermediate form of the media being transcoded and a final version (highlighted below). Thus, Applicant now claims that:

"the act of act of transcoding the requested media content in accordance with the identified transcoding parameters comprises:

an act of selecting a pre-existing base transcoded version of the requested media content comprising intermediate derivative media that has been transcoded in accordance with only a portion of the identified transcoding parameters; and

an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters..."

Accordingly, Applicant respectfully submits that the present rejections under 35 USC § 103 are traversed.

Support for Applicant's claim amendments can be found in the Specification at least at page 31, line 16 to page 32, line 11.

Applicant has endeavored to summarize the interview accurately and fairly. Any errors in such summary are entirely unintentional.

**CONCLUSION**

Applicant respectfully posits that the pending claims have been distinguished from the art of record, and that all objections to and rejections of the claims have been overcome. Accordingly, Applicant respectfully requests allowance. Should the Examiner deem it helpful he is encouraged to contact Applicant's attorney at (650) 474-8400.

Respectfully submitted,

Michael A. Glenn
Reg. No. 30,176

Customer No. 22862

PTO/SB/22 (10-12)
Approved for use through 1/31/2013. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

# PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a)

Docket Number (Optional): EQUI0016D2

Application Number: 12713637

Filed: 02-26-2010

For: AUTOMATED MEDIA DELIVERY SYSTEM

Art Unit: 2455

Examiner: Charles C. Murphy

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a reply in the above-identified application.

The requested extension and fee are as follows (check time period desired and enter the appropriate fee below):

| | Fee | Small Entity Fee | |
|---|---|---|---|
| ☐ One month (37 CFR 1.17(a)(1)) | $150 | $75 | $ |
| ■ Two months (37 CFR 1.17(a)(2)) | $570 | $285 | $ 285 |
| ☐ Three months (37 CFR 1.17(a)(3)) | $1,290 | $645 | $ |
| ☐ Four months (37 CFR 1.17(a)(4)) | $2,010 | $1,005 | $ |
| ☐ Five months (37 CFR 1.17(a)(5)) | $2,730 | $1,365 | $ |

■ Applicant claims small entity status. See 37 CFR 1.27.

☐ A check in the amount of the fee is enclosed.

☐ Payment by credit card. Form PTO-2038 is attached.

☐ The Director has already been authorized to charge fees in this application to a Deposit Account.

■ The Director is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 071445.

■ Payment made via EFS-Web.

**WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.**

I am the

☐ applicant/inventor.

☐ assignee of record of the entire interest. See 37 CFR 3.71. 37 CFR 3.73(b) statement is enclosed (Form PTO/SB/96).

■ attorney or agent of record. Registration number 30176.

☐ attorney or agent acting under 37 CFR 1.34. Registration number _____.

| | |
|---|---|
| Signature | 11/01/2012 (Date) |
| Michael A. Glenn (Typed or printed name) | 650-474-8400 (Telephone Number) |

**NOTE:** This form must be signed in accordance with 37 CFR 1.33. See 37 CFR 1.4 for signature requirements and certifications. Submit multiple forms if more than one signature is required, see below*.

■ * Total of 1 forms are submitted.

This collection of information is required by 37 CFR 1.136(a). The information is required to obtain or retain a benefit by the public, which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 6 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Mail Stop PCT, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| | 1 | US-6311185 | | Oct 2001 | Markowitz et al. | |
| | 2 | US-7313361 | | Dec 2007 | Steelberg et al. | |
| | 3 | US-7406434 | | Jul 2008 | Chang et al. | |

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| | 4 | US20030225568 | | Dec 2003 | Salmonsen | |
| | 5 | US20040025176 | | Feb 2004 | Franklin et al. | |
| | 6 | US20050255852 | | Nov 2005 | Steelberg et al. | |
| | 7 | US20050278794 | | Dec 2005 | Leinonen et al. | |
| | 8 | US20060127059 | | Jun 2006 | Fanning | |

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 9 | US20070061198 | | Mar 2007 | Ramer et al. | |
| | 10 | US20080195938 | | Aug 2008 | Tischer et al. | |
| | 11 | US20080207182 | | Aug 2008 | Maharajh et al. | |
| | 12 | US20080205389 | | Aug 2008 | Fang et al. | |
| | 13 | US20090013347 | | Jan 2009 | Ahanger et al. | |
| | 14 | US20090240569 | | Sep 2009 | Ramer et al. | |
| | 15 | US20100153495 | | Jun 2010 | Barger, S et al. | |

**FOREIGN PATENT DOCUMENTS**

| Examiner Initial* | Cite No | Foreign Document Number[3] | Country Code[2] | Kind Code[4] | Publication Date | Name of Patentee or Applicant of cited Document | Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear | T[5] |
|---|---|---|---|---|---|---|---|---|

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| **INFORMATION DISCLOSURE STATEMENT BY APPLICANT** (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

| **EXAMINER SIGNATURE** | | | |
|---|---|---|---|
| Examiner Signature | | Date Considered | |

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

## CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

☐ That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (1).

**OR**

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (2).

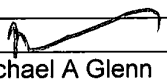
☐ See attached certification statement.

☒ Fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ No certification statement submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

| | | | |
|---|---|---|---|
| Signature |  | Date (YYYY-MM-DD) | 2012-11-01 |
| Name/Print | Michael A Glenn | Registration Number | 30176 |

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

# Electronic Patent Application Fee Transmittal

| | |
|---|---|
| **Application Number:** | 12713637 |
| **Filing Date:** | 26-Feb-2010 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Attorney Docket Number:** | EQUI0016D2 |

Filed as Small Entity

## Utility under 35 USC 111(a) Filing Fees

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Basic Filing:** | | | | |
| **Pages:** | | | | |
| **Claims:** | | | | |
| **Miscellaneous-Filing:** | | | | |
| **Petition:** | | | | |
| **Patent-Appeals-and-Interference:** | | | | |
| **Post-Allowance-and-Post-Issuance:** | | | | |
| **Extension-of-Time:** | | | | |
| Extension - 2 months with $0 paid | 2252 | 1 | 285 | 285 |

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Miscellaneous:** | | | | |
| Submission- Information Disclosure Stmt | 1806 | 1 | 180 | 180 |
| | | **Total in USD ($)** | | **465** |

# Electronic Acknowledgement Receipt

| | |
|---|---|
| **EFS ID:** | 14125746 |
| **Application Number:** | 12713637 |
| **International Application Number:** | |
| **Confirmation Number:** | 5103 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Customer Number:** | 22862 |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Filer Authorized By:** | Michael Glenn |
| **Attorney Docket Number:** | EQUI0016D2 |
| **Receipt Date:** | 01-NOV-2012 |
| **Filing Date:** | 26-FEB-2010 |
| **Time Stamp:** | 14:31:14 |
| **Application Type:** | Utility under 35 USC 111(a) |

## Payment information:

| | |
|---|---|
| Submitted with Payment | yes |
| Payment Type | Deposit Account |
| Payment was successfully received in RAM | $465 |
| RAM confirmation Number | 904 |
| Deposit Account | 071445 |
| Authorized User | |

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

## File Listing:

| Document Number | Document Description | File Name | File Size(Bytes)/ Message Digest | Multi Part /.zip | Pages (if appl.) |
|---|---|---|---|---|---|
| 1 | | EQUI0016D2_Response_2012-11-01.pdf | 1239248<br>d49a4d16f8d2f463884b04c61619fa50cd909b2a | yes | 18 |

**Multipart Description/PDF files in .zip description**

| Document Description | Start | End |
|---|---|---|
| Amendment/Req. Reconsideration-After Non-Final Reject | 1 | 1 |
| Claims | 2 | 10 |
| Applicant Arguments/Remarks Made in an Amendment | 11 | 13 |
| Extension of Time | 14 | 14 |
| Information Disclosure Statement (IDS) Form (SB08) | 15 | 17 |
| Transmittal Letter | 18 | 18 |

**Warnings:**

**Information:**

| | | | | | |
|---|---|---|---|---|---|
| 2 | Fee Worksheet (SB06) | fee-info.pdf | 31897<br>80a7791133b41681f3b8cd958cd32a20a4cb15c3 | no | 2 |

**Warnings:**

**Information:**

| Total Files Size (in bytes): | 1271145 |
|---|---|

**This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.**

**New Applications Under 35 U.S.C. 111**
**If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.**

**National Stage of an International Application under 35 U.S.C. 371**
**If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.**

**New International Application Filed with the USPTO as a Receiving Office**
**If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.**

PTO/SB/06 (07-06)
Approved for use through 1/31/2007. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

# PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

| Application or Docket Number | Filing Date | ☐ To be Mailed |
|---|---|---|
| 12/713,637 | 02/26/2010 | |

## APPLICATION AS FILED – PART I

| | (Column 1) | (Column 2) | SMALL ENTITY ☒ | | OR | OTHER THAN SMALL ENTITY | |
|---|---|---|---|---|---|---|---|
| FOR | NUMBER FILED | NUMBER EXTRA | RATE ($) | FEE ($) | | RATE ($) | FEE ($) |
| ☐ BASIC FEE (37 CFR 1.16(a), (b), or (c)) | N/A | N/A | N/A | | | N/A | |
| ☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m)) | N/A | N/A | N/A | | | N/A | |
| ☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q)) | N/A | N/A | N/A | | | N/A | |
| TOTAL CLAIMS (37 CFR 1.16(i)) | minus 20 = | * | X $ = | | OR | X $ = | |
| INDEPENDENT CLAIMS (37 CFR 1.16(h)) | minus 3 = | * | X $ = | | | X $ = | |
| ☐ APPLICATION SIZE FEE (37 CFR 1.16(s)) | If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s). | | | | | | |
| ☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j)) | | | | | | | |
| * If the difference in column 1 is less than zero, enter "0" in column 2. | | | TOTAL | | | TOTAL | |

## APPLICATION AS AMENDED – PART II

| AMENDMENT | 11/01/2012 | (Column 1) CLAIMS REMAINING AFTER AMENDMENT | | (Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR | (Column 3) PRESENT EXTRA | SMALL ENTITY RATE ($) | ADDITIONAL FEE ($) | OR | OTHER THAN SMALL ENTITY RATE ($) | ADDITIONAL FEE ($) |
|---|---|---|---|---|---|---|---|---|---|---|
| | Total (37 CFR 1.16(i)) | * 20 | Minus | ** 22 | = 0 | X $31 = | 0 | OR | X $ = | |
| | Independent (37 CFR 1.16(h)) | * 3 | Minus | ***3 | = 0 | X $125 = | 0 | OR | X $ = | |
| | ☐ Application Size Fee (37 CFR 1.16(s)) | | | | | | | | | |
| | ☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j)) | | | | | | | OR | | |
| | | | | | | TOTAL ADD'L FEE | 0 | OR | TOTAL ADD'L FEE | |

| AMENDMENT | | (Column 1) CLAIMS REMAINING AFTER AMENDMENT | | (Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR | (Column 3) PRESENT EXTRA | RATE ($) | ADDITIONAL FEE ($) | OR | RATE ($) | ADDITIONAL FEE ($) |
|---|---|---|---|---|---|---|---|---|---|---|
| | Total (37 CFR 1.16(i)) | * | Minus | ** | = | X $ = | | OR | X $ = | |
| | Independent (37 CFR 1.16(h)) | * | Minus | *** | = | X $ = | | OR | X $ = | |
| | ☐ Application Size Fee (37 CFR 1.16(s)) | | | | | | | | | |
| | ☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j)) | | | | | | | OR | | |
| | | | | | | TOTAL ADD'L FEE | | OR | TOTAL ADD'L FEE | |

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.

** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".

*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".

The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Legal Instrument Examiner:
/BRENDA J. DENNY/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

*If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.*

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

# NOTICE OF ALLOWANCE AND FEE(S) DUE

22862 7590 03/20/2013

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

| EXAMINER |
|---|
| MURPHY, CHARLES C |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2455 | |

DATE MAILED: 03/20/2013

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | INV001Sean BARGER | EQUI0016D2 | 5103 |

TITLE OF INVENTION: Automated Media Delivery System

| APPLN. TYPE | ENTITY STATUS | ISSUE FEE DUE | PUBLICATION FEE DUE | PREV. PAID ISSUE FEE | TOTAL FEE(S) DUE | DATE DUE |
|---|---|---|---|---|---|---|
| nonprovisional | SMALL | $890 | $300 | $0 | $1190 | 06/20/2013 |

**THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.**

**THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.**

**HOW TO REPLY TO THIS NOTICE:**

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

**IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.**

## PART B - FEE(S) TRANSMITTAL

**Complete and send this form, together with applicable fee(s), to: Mail** **Mail Stop ISSUE FEE**
**Commissioner for Patents**
**P.O. Box 1450**
**Alexandria, Virginia 22313-1450**
**or Fax (571)-273-2885**

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22862 7590 03/20/2013
GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

**Certificate of Mailing or Transmission**
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

| |
|---|
| (Depositor's name) |
| (Signature) |
| (Date) |

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | INV001Sean BARGER | EQUI0016D2 | 5103 |

TITLE OF INVENTION: Automated Media Delivery System

| APPLN. TYPE | ENTITY STATUS | ISSUE FEE DUE | PUBLICATION FEE DUE | PREV. PAID ISSUE FEE | TOTAL FEE(S) DUE | DATE DUE |
|---|---|---|---|---|---|---|
| nonprovisional | SMALL | $890 | $300 | $0 | $1190 | 06/20/2013 |

| EXAMINER | ART UNIT | CLASS-SUBCLASS |
|---|---|---|
| MURPHY, CHARLES C | 2455 | 709-236000 |

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
- ❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
- ❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.
1 _____
2 _____
3 _____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. The following fee(s) are submitted:
- ❑ Issue Fee
- ❑ Publication Fee (No small entity discount permitted)
- ❑ Advance Order - # of Copies _____

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**
- ❑ A check is enclosed.
- ❑ Payment by credit card. Form PTO-2038 is attached.
- ❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number _____ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

NOTE: Absent a valid certification of Micro Entity Status (see form PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.

❑ Applicant asserting small entity status. See 37 CFR 1.27

NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.

❑ Applicant changing to regular undiscounted fee status.

NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature _____ Date _____

Typed or printed name _____ Registration No. _____

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | INV001Sean BARGER | EQUI0016D2 | 5103 |

22862 7590 03/20/2013

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

| EXAMINER | |
|---|---|
| MURPHY, CHARLES C | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2455 | |

DATE MAILED: 03/20/2013

**Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)**
(application filed on or after May 29, 2000)

The Patent Term Adjustment to date is 173 day(s). If the issue fee is paid on the date that is three months after the mailing date of this notice and the patent issues on the Tuesday before the date that is 28 weeks (six and a half months) after the mailing date of this notice, the Patent Term Adjustment will be 173 day(s).

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

## Privacy Act Statement

**The Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

| Notice of Allowability | Application No. | Applicant(s) | |
|---|---|---|---|
| | 12/713,637 | BARGER ET AL. | |
| | Examiner | Art Unit | |
| | CHARLES MURPHY | 2455 | |

***-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--***

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *the amendment filed 11/01/2012*.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☒ The allowed claim(s) is/are *1,3-9 and 11-22*. As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see http://www.uspto.gov/patents/init_events/pph/index.jsp or send an inquiry to PPHfeedback@uspto.gov .

4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
      1. ☐ Certified copies of the priority documents have been received.
      2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
      3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
**THIS THREE-MONTH PERIOD IS NOT EXTENDABLE**.

5. ☐ CORRECTED DRAWINGS ( as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   **Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).**

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

**Attachment(s)**

1. ☐ Notice of References Cited (PTO-892)
2. ☒ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 11/01/2012
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date _____ .
5. ☐ Examiner's Amendment/Comment
6. ☒ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

| /CHARLES MURPHY/<br>Examiner, Art Unit 2455 | /David Lazaro/<br>Primary Examiner, Art Unit 2455 |
|---|---|

**Information Disclosure Statement**

The information disclosure statement (IDS) submitted was filed on 11/01/2012. The submission is in compliance with the provisions of 37 CFR 1.97. Accordingly, the information disclosure statement is being considered by the examiner

**Reasons for Allowance**

1. Claims 1, 3-9, 11-22 are allowed

2. The following is an Examiner's statement of reasons for allowance:

In interpreting the currently amended claims, in light of the specification, the Examiner finds the claimed invention to be patentably distinct from the prior art of records.

**Suzuki et al. US Patent 6,463,445 (hereinafter referred to as Suzuki)** teaches one or more computer-readable physical and/or memory storage media, the one or more computer-readable memory media having stored thereon computer executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following ***(Suzuki, col. 15, lines 6-50)***

an act of receiving a request for media content to be delivered to a client presentation system;*(**Suzuki** teaches a user sending a request for content from a client to a server system)* ***(Suzuki, col. 6, lines 55-67)***

at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media

content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system; *(**Suzuki** teaches a transcoding system using client identification information received at the time of request to determine the transcoding parameters required to properly display requested content on a client device. )* ***(Suzuki, col. 7, lines 35-67)***

an act of causing the transcoded media content to be delivered to the client presentation system; and *(**Suzuki** delivering transcoded content to a requesting client )* ***(Suzuki, col. 7, lines 23-36)***

**Bhagwat et al. US Patent 6,563,517 (hereinafter referred to as Bhagwat)** teaches an act of caching the transcoded media content. *(**Bhagwat** teaches caching a transcoded media file)****(Bhagwat, col. 5, lines 15-25 and 50-57; cool 6, lines 45-50)***

the act of receiving a request for media content wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content; *(**Bhagwat** teaches a multi resolution cache in a transcoding proxy. This multi resolution cache stores the original (base) file that is to be transcoded as well as various reduced resolution versions of the original file that have been transcoded for specific devices. The parameters used to transcode this base file into a specified media can be considered transcoding profiles used to transcode the base file. )****(Bhagwat, col. 5, lines 10-25 and 50-57; cool 6, lines 45-50, col. 13, lines 20-40)***

**Krishnaswamy et al. US Patent 6,909,708 (hereinafter referred to as Krishnaswamy)** teaches an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile; *(**Krishnaswamy** teaches to use a base file that has already been transcoded, and convert the base file into various formats.)**(Krishnaswamy, col. 135, lines 25-35)***

However, the prior art of records, individually or in combination, fail to explicitly teach or render obvious a computer program product comprising:

one or more computer-readable media, the one or more computer-readable media having stored thereon computer-executable instructions that, when executed by one or more processors of the computing system, cause the computing system to perform the following:

*an act of receiving a request for media content to be delivered to a client presentation system, wherein the requested media content has a limited number of base transcoding profiles associated therewith, each base transcoding profile corresponding to a cached version of the requested media content;*

*at the time of the request, and without input by a network administrator, an act of automatically identifying transcoding parameters to be applied to the requested media*

*content prior to delivery to the client presentation system, wherein identification of transcoding parameters is based on one or more formats of any client presentation system;*

*an act of transcoding the requested media content in accordance with the identified transcoding parameters, such that the identified transcoding parameters are used to perform additional incremental transcoding on top of the base transcoding profile;*

*wherein the act of act of transcoding the requested media content in accordance with the identified transcoding parameters comprises:*

***an act of selecting a pre-existing base transcoded version of the requested media content comprising intermediate derivative media that has been transcoded in accordance with only a portion of the identified transcoding parameters; and***

***an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters;***

*an act of causing the transcoded media content to be delivered to the client presentation system; and*

*an act of caching the transcoded media content.*

The Examiner finds the combination of prior art of records fail to disclose the features of the invention including ***an act of selecting a pre-existing base transcoded***

***version of the requested media content comprising intermediate derivative media that has been transcoded in accordance with only a portion of the identified transcoding parameters; and an act of creating a final version by incrementally performing further transcoding of the pre-existing base transcoded version in accordance with a remaining portion of the identified transcoding parameters;*** as claimed in the invention to reduce the system need for a web author to create different versions of a web site, while also generating media content only if requested by a client browser **(Specification, page 9, lines 7-20)**

Dependent claims 3-9 and 13-22 further limit the allowed independent claims 1, 11, and 12; therefore, they are allowed.

Any comments considered necessary by the applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should clearly be labeled “Comments on Examiner’s Amendment”

Any inquiry concerning this communication or earlier communications from the examiner should be directed to CHARLES MURPHY whose telephone number is (571)270-5444. The examiner can normally be reached on 8AM - 5PM Monday - Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Emmanuel Moise can be reached on (571) 272-3865. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/CHARLES MURPHY/
Examiner, Art Unit 2455

/David Lazaro/
Primary Examiner, Art Unit 2455

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| /C.M./ | 1 | US-6311185 | | Oct 2001 | Markowitz et al. | |
| /C.M./ | 2 | US-7313361 | | Dec 2007 | Steelberg et al. | |
| /C.M./ | 3 | US-7406434 | | Jul 2008 | Chang et al. | |

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| /C.M./ | 4 | US20030225568 | | Dec 2003 | Salmonsen | |
| /C.M./ | 5 | US20040025176 | | Feb 2004 | Franklin et al. | |
| /C.M./ | 6 | US20050255852 | | Nov 2005 | Steelberg et al. | |
| /C.M./ | 7 | US20050278794 | | Dec 2005 | Leinonen et al. | |
| /C.M./ | 8 | US20060127059 | | Jun 2006 | Fanning | |

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

| | | | | | | |
|---|---|---|---|---|---|---|
| /C.M./ | 9 | US20070061198 | | Mar 2007 | Ramer et al. | |
| /C.M./ | 10 | US20080195938 | | Aug 2008 | Tischer et al. | |
| /C.M./ | 11 | US20080207182 | | Aug 2008 | Maharajh et al. | |
| /C.M./ | 12 | US20080205389 | | Aug 2008 | Fang et al. | |
| /C.M./ | 13 | US20090013347 | | Jan 2009 | Ahanger et al. | |
| /C.M./ | 14 | US20090240569 | | Sep 2009 | Ramer et al. | |
| /C.M./ | 15 | US20100153495 | | Jun 2010 | Barger, S et al. | |

**FOREIGN PATENT DOCUMENTS**

| Examiner Initial* | Cite No | Foreign Document Number[3] | Country Code[2] | Kind Code[4] | Publication Date | Name of Patentee or Applicant of cited Document | Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear | T[5] |
|---|---|---|---|---|---|---|---|---|

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| **INFORMATION DISCLOSURE STATEMENT BY APPLICANT** (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Charles C. Murphy |
| | **Attorney Docket Number** | EQUI0016D2 |

| **EXAMINER SIGNATURE** | | | |
|---|---|---|---|
| Examiner Signature | /Charles Murphy/ | Date Considered | 03/07/2013 |

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

| Index of Claims | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | Examiner | Art Unit |
| | CHARLES MURPHY | 2455 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ✓ | Rejected | - | Cancelled | N | Non-Elected | A | Appeal |
| = | Allowed | ÷ | Restricted | I | Interference | O | Objected |

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

| CLAIM | | DATE | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Final | Original | 10/07/2010 | 03/22/2011 | 05/19/2012 | 03/07/2013 | | | | | |
| 1 | 1 | ✓ | ✓ | ✓ | ✓ | | | | | |
| | 2 | ✓ | - | - | - | | | | | |
| 2 | 3 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 3 | 4 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 4 | 5 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 5 | 6 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 6 | 7 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 7 | 8 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 8 | 9 | ✓ | ✓ | ✓ | ✓ | | | | | |
| | 10 | ✓ | ✓ | ✓ | - | | | | | |
| 10 | 11 | ✓ | ✓ | ✓ | ✓ | | | | | |
| 11 | 12 | | | ✓ | ✓ | | | | | |
| 12 | 13 | | | ✓ | ✓ | | | | | |
| 13 | 14 | | | ✓ | ✓ | | | | | |
| 14 | 15 | | | ✓ | ✓ | | | | | |
| 15 | 16 | | | ✓ | ✓ | | | | | |
| 16 | 17 | | | ✓ | ✓ | | | | | |
| 17 | 18 | | | ✓ | ✓ | | | | | |
| 18 | 19 | | | ✓ | ✓ | | | | | |
| 19 | 20 | | | ✓ | ✓ | | | | | |
| 20 | 21 | | | ✓ | ✓ | | | | | |
| 21 | 22 | | | ✓ | ✓ | | | | | |
| | 23 | | | ✓ | - | | | | | |

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

# BIB DATA SHEET

**CONFIRMATION NO. 5103**

| SERIAL NUMBER | FILING or 371(c) DATE<br>RULE | CLASS | GROUP ART UNIT | ATTORNEY DOCKET NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | 709 | 2455 | EQUI0016D2 |

**APPLICANTS**

Sean BARGER, Mill Valley, CA;
Steve Johnson, Mill Valley, CA;
Matt Butler, Beaverton, OR;
Jerry Destremps, Sausalito, CA;
David Pochron, Cambridge, WI;
Trent Brown, San Anselmo, CA;

**** CONTINUING DATA ***************************

This application is a DIV of 12/173,747 07/15/2008
which is a DIV of 11/269,916 11/07/2005 ABN
which is a CIP of 09/929,904 08/14/2001 PAT 6,964,009
which is a CON of 09/425,326 10/21/1999 PAT 6,792,575

**** FOREIGN APPLICATIONS ***************************

**** IF REQUIRED, FOREIGN FILING LICENSE GRANTED **** ** SMALL ENTITY **
03/09/2010

| Foreign Priority claimed | ☐ Yes ☑ No | ☐ Met after Allowance | STATE OR COUNTRY | SHEETS DRAWINGS | TOTAL CLAIMS | INDEPENDENT CLAIMS |
|---|---|---|---|---|---|---|
| 35 USC 119(a-d) conditions met | ☐ Yes ☑ No | | CA | 23 | 11 | 2 |
| Verified and Acknowledged | /CHARLES C MURPHY/ Examiner's Signature | Initials | | | | |

**ADDRESS**

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025
UNITED STATES

**TITLE**

Automated Media Delivery System

| FILING FEE RECEIVED | FEES: Authority has been given in Paper | ☐ All Fees |
|---|---|---|
| 514 | No._____ to charge/credit DEPOSIT ACCOUNT | ☐ 1.16 Fees (Filing) |
| | No._____ for following: | ☐ 1.17 Fees (Processing Ext. of time) |
| | | ☐ 1.18 Fees (Issue) |
| | | ☐ Other _____ |
| | | ☐ Credit |

| Search Notes | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | Examiner | Art Unit |
| | CHARLES MURPHY | 2455 |

| CPC- SEARCHED | | |
|---|---|---|
| Symbol | Date | Examiner |
| | | |

| CPC COMBINATION SETS - SEARCHED | | |
|---|---|---|
| Symbol | Date | Examiner |
| | | |

| US CLASSIFICATION SEARCHED | | | |
|---|---|---|---|
| Class | Subclass | Date | Examiner |
| 709 | 263 234 231 226 219 217 204 203 200 | 10/7/2010 | Charles Murphy |
| 370 | 552 468 431 353 352 330 280 241 | 10/7/2010 | Charles Murphy |
| 709 | 263 231 226 219 217 204 203 | 3/22/2011 | Charles Murphy |
| 370 | 468 431 353 352 241 | 3/22/2011 | Charles Murphy |
| 709 | 263 234 231 226 219 217 200 | 5/19/2012 | Charles Murphy |
| 709 | 263 231 234 226 200 | 3/7/2013 | Charles Murphy |
| 341 | 50 | 3/7/2013 | Charles Murphy |
| 708 | 204 | 3/7/2013 | Charles Murphy |

| SEARCH NOTES | | |
|---|---|---|
| Search Notes | Date | Examiner |
| Searched East | 10/7/2010 | Charles Murphy |
| Searched East | 3/22/2011 | Charles Murphy |
| Consulted Wing Chan for potential 101 issue for claims 1-10 | 3/28/2011 | Charles Murphy |
| Searched East | 5/19/2012 | Charles Murphy |
| Performed Inventors name search | 3/7/2013 | Charles Murphy |
| Consulted David Lazaro on allowability | 3/7/2013 | Charles Murphy |
| Consulted 2400 101 help desk on possible 101 issues | 3/7/2013 | Charles Murphy |

| INTERFERENCE SEARCH |
|---|

| /CHARLES MURPHY/<br>Examiner.Art Unit 2455 | |
|---|---|

| US Class/ CPC Symbol | US Subclass / CPC Group | Date | Examiner |
|---|---|---|---|
| Searched East | | 3/7/2013 | Charles Murphy |

| | |
|---|---|
| /CHARLES MURPHY/<br>Examiner.Art Unit 2455 | |

| Issue Classification | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | **Examiner** | **Art Unit** |
| | CHARLES MURPHY | 2455 |

| CPC | | |
|---|---|---|
| **Symbol** | **Type** | **Version** |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| CPC Combination Sets | | | | |
|---|---|---|---|---|
| **Symbol** | **Type** | **Set** | **Ranking** | **Version** |
| | | | | |
| | | | | |

| US ORIGINAL CLASSIFICATION | | INTERNATIONAL CLASSIFICATION | |
|---|---|---|---|
| **CLASS** | **SUBCLASS** | **CLAIMED** | **NON-CLAIMED** |
| 709 | 236 | G 0 6 F 15 / 16 (2006.0) | |

| CROSS REFERENCE(S) | | | | | | |
|---|---|---|---|---|---|---|
| **CLASS** | **SUBCLASS (ONE SUBCLASS PER BLOCK)** | | | | | |
| 709 | 246 | | | | | |
| 341 | 50 | | | | | |

| | | | |
|---|---|---|---|
| /CHARLES MURPHY/<br>Examiner.Art Unit 2455<br><br>(Assistant Examiner) | (Date) | **Total Claims Allowed:**<br>20 | |
| /DAVID LAZARO/<br>Primary Examiner.Art Unit 2455<br><br>(Primary Examiner) | 03/09/2013<br>(Date) | O.G. Print Claim(s)<br>1 | O.G. Print Figure<br>1 |

| Issue Classification | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | **Examiner** | **Art Unit** |
| | CHARLES MURPHY | 2455 |

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 708 | 204 | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |

| | | | |
|---|---|---|---|
| /CHARLES MURPHY/<br>Examiner.Art Unit 2455<br><br>(Assistant Examiner) | (Date) | **Total Claims Allowed:**<br>20 | |
| /DAVID LAZARO/<br>Primary Examiner.Art Unit 2455<br><br>(Primary Examiner) | 03/09/2013<br><br>(Date) | O.G. Print Claim(s)<br>1 | O.G. Print Figure<br>1 |

| Issue Classification | Application/Control No. | Applicant(s)/Patent Under Reexamination |
|---|---|---|
| | 12713637 | BARGER ET AL. |
| | **Examiner** | **Art Unit** |
| | CHARLES MURPHY | 2455 |

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

| Final | Original | Final | Original | Final | Original | Final | Original | Final | Original | Final | Original | Final | Original | Final | Original |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 15 | 17 | | | | | | | | | | | | |
| | 2 | 16 | 18 | | | | | | | | | | | | |
| 2 | 3 | 17 | 19 | | | | | | | | | | | | |
| 3 | 4 | 18 | 20 | | | | | | | | | | | | |
| 4 | 5 | 19 | 21 | | | | | | | | | | | | |
| 5 | 6 | 20 | 22 | | | | | | | | | | | | |
| 6 | 7 | | 23 | | | | | | | | | | | | |
| 7 | 8 | | | | | | | | | | | | | | |
| 8 | 9 | | | | | | | | | | | | | | |
| | 10 | | | | | | | | | | | | | | |
| 9 | 11 | | | | | | | | | | | | | | |
| 10 | 12 | | | | | | | | | | | | | | |
| 11 | 13 | | | | | | | | | | | | | | |
| 12 | 14 | | | | | | | | | | | | | | |
| 13 | 15 | | | | | | | | | | | | | | |
| 14 | 16 | | | | | | | | | | | | | | |

| | | | |
|---|---|---|---|
| /CHARLES MURPHY/<br>Examiner.Art Unit 2455<br><br>(Assistant Examiner) | (Date) | **Total Claims Allowed:**<br>20 | |
| /DAVID LAZARO/<br>Primary Examiner.Art Unit 2455<br><br>(Primary Examiner) | 03/09/2013<br><br>(Date) | O.G. Print Claim(s)<br>1 | O.G. Print Figure<br>1 |

**EAST Search History**

**EAST Search History (Prior Art)**

| Ref # | Hits | Search Query | DBs | Default Operator | Plurals | Time Stamp |
|---|---|---|---|---|---|---|
| L1 | 27 | US-6311185-$.DID. OR US-7313361-$.DID. OR US-7406434-$.DID. OR US-20030225568-$.DID. OR US-20040025176-$.DID. OR US-20050255852-$.DID. OR US-20050278794-$.DID. OR US-20060127059-$.DID. OR US-20070061198-$.DID. OR US-20080195938-$.DID. OR US-20080207182-$.DID. OR US-20080205389-$.DID. OR US-20090013347-$.DID. OR US-20090240569-$.DID. | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2013/03/07 16:31 |
| L2 | 7 | ((SEAN) near2 (BARGER)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| L3 | 131 | ((STEVE) near2 (JOHNSON)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| L4 | 6 | ((MATT) near2 (BUTLER)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| L5 | 5 | ((JERRY) near2 (DESTREMPS)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| L6 | 5 | ((DAVID) near2 (POCHRON)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| L7 | 10 | ((TRENT) near2 (BROWN)).INV. | US-PGPUB; USPAT; USOCR | AND | ON | 2013/03/07 16:39 |
| S1 | 1 | ("20090254672").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2010/10/07 14:44 |
| S2 | 74 | ("5088052" \| "5355472" \| "5442771" \| "5530852" \| "5701451" \| "5708845" \| "5710918" \| "5737619" \| "5745908" \| "5758110" \| "5761555" \| "5761655" \| "5793964" \| "5819261" \| "5822436" \| "5845084" \| "5845299" \| "5860068" \| "5860073" \| "5861881" \| "5862325" \| "5864337" \| "5870552" \| "5880740" \| "5884337" \| "5890170" \| "5890170" \| "5895476" \| "5895476" \| "5895477" \| "5903892" \| "5937160" \| "5943680" \| "5943680" \| "5956737" \| "5956737" \| "6009436" \| "6456305" \| "6484149" \| "6563517" \| "6591280" \| "6591280" \| "6623529" \| "6623529").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:48 |
| S3 | 2 | S2 (transcod$4) | US-PGPUB; | AND | ON | 2010/10/07 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | | | 14:49 |
| S4 | 74 | ("5088052" \| "5355472" \| "5442771" \| "5530852" \| "5701451" \| "5708845" \| "5710918" \| "5737619" \| "5745908" \| "5758110" \| "5761555" \| "5761655" \| "5793964" \| "5819261" \| "5822436" \| "5845084" \| "5845299" \| "5860068" \| "5860073" \| "5861881" \| "5862325" \| "5864337" \| "5870552" \| "5880740" \| "5884337" \| "5890170" \| "5890170" \| "5895476" \| "5895476" \| "5895477" \| "5903892" \| "5937160" \| "5943680" \| "5943680" \| "5956737" \| "5956737" \| "6009436" \| "6456305" \| "6484149" \| "6563517" \| "6591280" \| "6591280" \| "6623529" \| "6623529").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:51 |
| S5 | 1 | S4 (transcod$4 same cache) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 14:51 |
| S6 | 34 | (transcod$4) same (content or media or file) near4 (concurrent$3) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:40 |
| S7 | 0 | (transcod$4) same (content or media or file) near4 (concurrent$3) @ad<"19991021" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S8 | 0 | (transcod$4) same (content or media or file) near4 (concurrent$3) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S9 | 15 | (transcod$4) (content or media or file) near4 (concurrent$3) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 17:42 |
| S10 | 10 | (transcod$4) (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; | AND | ON | 2010/10/07 17:43 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | IBM_TDB | | | |
| S11 | 0 | (transcod$4 same parameter) (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:18 |
| S12 | 2709 | (content or media or file) same (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S13 | 763 | (content or media or file) near5 (deliver$3 or send$3 or transmit$3) same (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S14 | 184 | (content or media or file) near5 (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:25 |
| S15 | 41 | S14 (transcod$3 or convert) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:26 |
| S16 | 1 | S14 (transcod$3) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:26 |
| S17 | 25 | S14 (media.ti.) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:27 |
| S18 | 3 | (content or media or file) near5 (transcod$3 or convert$3) same (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:30 |
| S19 | 157 | (content or media or file) near5 (transcod$3 or convert$3) (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; | AND | ON | 2010/10/07 18:30 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | EPO; JPO; DERWENT; IBM_TDB | | | |
| S20 | 6 | S19 (media.ti.) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:30 |
| S21 | 22 | (content or media or file or audio or video or movie) near5 (transcod$3 or convert$3) same (deliver$3 or send$3 or transmit$3) near5 (concurrent$3 or simultan$4) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:31 |
| S22 | 1 | ("5845279").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2010/10/07 18:35 |
| S23 | 0 | S22 concurent$3 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:35 |
| S24 | 0 | S22 concurent$4 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:35 |
| S25 | 1 | S22 concurrent$4 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2010/10/07 18:35 |
| S26 | 13 | (transcod$4) same (content or media or file) near4 (base or primary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:10 |
| S27 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:10 |
| S28 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation or prilimnary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; | AND | ON | 2011/03/16 16:11 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | IBM_TDB | | | |
| S29 | 13 | (transcod$4) same (content or media or file) near4 (base or primary or foundation or preliminary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:12 |
| S30 | 0 | (transcod$4) near5 (profile) same (content or media or file) near4 (base or primary or foundation or preliminary) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:13 |
| S31 | 1 | (transcod$4) near5 (profile) same (content or media or file) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/16 16:13 |
| S32 | 1 | ("6359902").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/17 17:35 |
| S33 | 0 | S32 cache | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/17 17:35 |
| S34 | 1 | ("6563517").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:22 |
| S35 | 1 | ("5845279").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:23 |
| S36 | 1 | ("6909708").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 15:23 |
| S37 | 3 | S34 or S35 or S36 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:23 |
| S38 | 0 | S37 cache same media | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:23 |
| S39 | 2 | S37 cache | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; | AND | ON | 2011/03/22 15:23 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | DERWENT; IBM_TDB | | | |
| S40 | 1 | S37 cache same transcode$4 | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 15:24 |
| S41 | 1 | ("6563517").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 16:33 |
| S42 | 1 | ("6909708").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/03/22 16:33 |
| S43 | 1 | S41 preferences same request | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/03/22 16:50 |
| S44 | 0 | (transcod$4 adj profile) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:46 |
| S45 | 2 | (transcod$4 near5 profile) @ay<"1999" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:57 |
| S46 | 106 | US-1182225-$.DID. OR US-6147977-$.DID. OR US-7149893-$.DID. OR US-6829591-$.DID. OR US-6442589-$.DID. OR US-5327486-$.DID. OR US-6892230-$.DID. OR US-6748569-$.DID. OR US-7221658-$.DID. OR US-5400020-$.DID. OR US-5623260-$.DID. OR US-5657010-$.DID. OR US-5668543-$.DID. OR US-5708780-$.DID. OR US-5796394-$.DID. OR US-5870549-$.DID. OR US-5875437-$.DID. OR US-5937161-$.DID. OR US-6029148-$.DID. OR US-6052673-$.DID. OR US-6073166-$.DID. OR US-6151624-$.DID. OR US-6154738-$.DID. OR US-6157945-$.DID. OR US-6185603-$.DID. OR US-6212550-$.DID. OR US-6216165-$.DID. OR US-6219714-$.DID. OR US-6278936-$.DID. OR US-6314402-$.DID. OR US-6313760-$.DID. OR US-6317060-$.DID. OR US-6363254-$.DID. OR US-6363323-$.DID. OR US-6411891-$.DID. OR US-6415207-$.DID. OR US-5899995-$.DID. OR US-6314434-$.DID. OR US-5742763-$.DID. OR US-6513019-$.DID. OR US-5073852-$.DID. OR US-6542515-$.DID. OR US-5805810-$.DID. OR US- | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:59 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | 5819046-$.DID. OR US-5579472-$.DID. OR US-5828314-$.DID. OR US-6167409-$.DID. OR US-6347340-$.DID. OR US-6430272-$.DID. OR US-6442591-$.DID. OR US-6438583-$.DID. OR US-6836792-$.DID. OR WO-0176119-$.DID. OR WO-0175604-$.DID. OR WO-0176264-$.DID. OR WO-0180130-$.DID. | | | | |
| S47 | 18 | S46 profile | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 20:59 |
| S48 | 139 | (transcoding near5 profile) | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:00 |
| S49 | 3 | (transcoding near5 profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:01 |
| S50 | 1 | ("7117361").PN. | US-PGPUB; USPAT; USOCR | OR | OFF | 2011/11/18 21:02 |
| S51 | 1 | S50 profile | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:02 |
| S52 | 355 | ((transcod$4 or translation) near5 profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:04 |
| S53 | 49 | ((transcod$4 or translation) near profile) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2011/11/18 21:04 |
| S54 | 94 | US-5088052-$.DID. OR US-5355472-$.DID. OR US-5442771-$.DID. OR US-5530852-$.DID. OR US-5701451-$.DID. OR US-5708845-$.DID. OR US-5710918-$.DID. OR US-5737619-$.DID. OR US-5745908-$.DID. OR US-5758110-$.DID. OR US-5761655-$.DID. OR US-5793964-$.DID. OR US-5819261-$.DID. OR US- | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:18 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | 5822436-$.DID. OR US-5845084-$.DID. OR US-5845279-$.DID. OR US-5845299-$.DID. OR US-5860068-$.DID. OR US-5860073-$.DID. OR US-5861881-$.DID. OR US-5862325-$.DID. OR US-5864337-$.DID. OR US-5870552-$.DID. OR US-5880740-$.DID. OR US-5890170-$.DID. OR US-5895476-$.DID. OR US-5895477-$.DID. OR US-5903892-$.DID. OR US-5937160-$.DID. OR US-5943680-$.DID. OR US-5956737-$.DID. OR US-6009436-$.DID. OR US-6456305-$.DID. OR US-6483851-$.DID. OR US-6484149-$.DID. OR US-6563517-$.DID. OR US-6591280-$.DID. OR US-6623529-$.DID. OR US-6909708-$.DID. OR US-7284201-$.DID. OR US-7477688-$.DID. OR US-7673063-$.DID. OR US-20080155230-$.DID. OR US-20090254672-$.DID. OR US-5303198-$.DID. OR EP-0747842-$.DID. OR EP-0747842121996-$.DID. OR EP-0782085-$.DID. OR EP-0782085071997-$.DID. OR EP-0818907-$.DID. OR EP-0843276-$.DID. OR EP-0843276051998-$.DID. OR EP-0876034-$.DID. OR EP-0876034111998-$.DID. OR EP-0883068-$.DID. OR US-1211998-$.DID. OR EP-0886409-$.DID. OR EP-0895171-$.DID. OR EP-0926607-$.DID. OR EP-0926607061999-$.DID. OR EP-0949571-$.DID. OR EP-0949571101999-$.DID. OR WO-97149252-$.DID. OR US-1211997-$.DID. OR WO-9840842-$.DID. OR WO-9840842091998-$.DID. OR WO-9843177-$.DID. OR WO-9843177101998-$.DID. OR US-9830041-$.DID. | | | | |
| S55 | 0 | S54 (transcode same automatic$5 near5 client) | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:21 |
| S56 | 1 | S54 (transcode same automatic$5 ) | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:21 |
| S57 | 42 | (transcod$4 near5 automatic$4) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:24 |
| S58 | 1 | S57 (parameter near5 client) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:24 |

| | | | | | | |
|---|---|---|---|---|---|---|
| S59 | 2 | S57 (parameter near5 (client or device or user)) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:26 |
| S60 | 7 | S57 (ident$6 near5 (client or device or user)) @ay<"2000" | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | AND | ON | 2012/05/18 22:45 |
| S61 | 2 | ("6563517").PN. | US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB | OR | OFF | 2012/05/18 23:06 |
| S62 | 27 | US-6311185-$.DID. OR US-7313361-$.DID. OR US-7406434-$.DID. OR US-20030225568-$.DID. OR US-20040025176-$.DID. OR US-20050255852-$.DID. OR US-20050278794-$.DID. OR US-20060127059-$.DID. OR US-20070061198-$.DID. OR US-20080195938-$.DID. OR US-20080207182-$.DID. OR US-20080205389-$.DID. OR US-20090013347-$.DID. OR US-20090240569-$.DID. | US-PGPUB; USPAT; USOCR; FPRS; JPO; DERWENT; IBM_TDB | AND | ON | 2013/02/26 15:03 |

**EAST Search History (Interference)**

| Ref # | Hits | Search Query | DBs | Default Operator | Plurals | Time Stamp |
|---|---|---|---|---|---|---|
| L8 | 0 | ((request near5 media adj content near5 client) same ( transcod$4 near4 profile) same (transcod$4 near5 automatic$4) same (transcod$4 near5 incremental) same (derivative near5 media intermediate near5 media)).clm. | US-PGPUB; USPAT; UPAD | AND | ON | 2013/03/07 16:44 |
| L9 | 0 | ((request near5 media adj content near5 client) same ( transcod$4 near4 profile) same (transcod$4 near5 automatic$4) same (transcod$4 near5 incremental) ).clm. | US-PGPUB; USPAT; UPAD | AND | ON | 2013/03/07 16:44 |
| L10 | 0 | ((request near5 media adj content near5 client) same ( transcod$4 near4 profile) same (transcod$4 near5 automatic$4) ).clm. | US-PGPUB; USPAT; UPAD | AND | ON | 2013/03/07 16:44 |

**3/7/2013 4:44:39 PM**
**C:\Users\cmurphy1\Documents\EAST\Workspaces\12713637.wsp**

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| **INFORMATION DISCLOSURE STATEMENT BY APPLICANT** (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| | | | | | | |

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| | 1 | US-20050091311 | | Apr 28, 2005 | Lund, C et al. | |

**EXAMINER SIGNATURE**

| Examiner Signature | | Date Considered | |
|---|---|---|---|

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Field

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

## CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

☐ That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (1).

**OR**

☒ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (2).

☐ See attached certification statement.

☒ Fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☐ No certification statement submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

| | | | |
|---|---|---|---|
| Signature | | Date (YYYY-MM-DD) | 2013-04-18 |
| Name/Print | Michael A Glenn | Registration Number | 30176 |

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

| Electronic Patent Application Fee Transmittal | |
|---|---|
| **Application Number:** | 12713637 |
| **Filing Date:** | 26-Feb-2010 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Attorney Docket Number:** | EQUI0016D2 |
| Filed as Small Entity | |

**Utility under 35 USC 111(a) Filing Fees**

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Basic Filing:** | | | | |
| **Pages:** | | | | |
| **Claims:** | | | | |
| **Miscellaneous-Filing:** | | | | |
| **Petition:** | | | | |
| **Patent-Appeals-and-Interference:** | | | | |
| **Post-Allowance-and-Post-Issuance:** | | | | |
| **Extension-of-Time:** | | | | |

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Miscellaneous:** | | | | |
| Submission- Information Disclosure Stmt | 2806 | 1 | 90 | 90 |
| | | **Total in USD ($)** | | **90** |

| Electronic Acknowledgement Receipt | |
|---|---|
| **EFS ID:** | 15558981 |
| **Application Number:** | 12713637 |
| **International Application Number:** | |
| **Confirmation Number:** | 5103 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Customer Number:** | 22862 |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Filer Authorized By:** | Michael Glenn |
| **Attorney Docket Number:** | EQUI0016D2 |
| **Receipt Date:** | 18-APR-2013 |
| **Filing Date:** | 26-FEB-2010 |
| **Time Stamp:** | 18:32:49 |
| **Application Type:** | Utility under 35 USC 111(a) |

## Payment information:

| | |
|---|---|
| Submitted with Payment | yes |
| Payment Type | Deposit Account |
| Payment was successfully received in RAM | $90 |
| RAM confirmation Number | 5265 |
| Deposit Account | 071445 |
| Authorized User | |
| The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:<br>Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)<br>Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees) | |

| Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges) | | | | | |
|---|---|---|---|---|---|
| **File Listing:** | | | | | |
| **Document Number** | **Document Description** | **File Name** | **File Size(Bytes)/ Message Digest** | **Multi Part /.zip** | **Pages (if appl.)** |
| 1 | | EQUI0016D2_IDS_US_CERTIFICATION_2013-04-18.pdf | 184867<br>4ec676cb65b666c2237ab4910ee37d31dcb f4f3a | yes | 2 |
| | **Multipart Description/PDF files in .zip description** | | | | |
| | **Document Description** | | **Start** | **End** | |
| | Information Disclosure Statement (IDS) Form (SB08) | | 1 | 1 | |
| | Transmittal Letter | | 2 | 2 | |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| 2 | Fee Worksheet (SB06) | fee-info.pdf | 29937<br>fcdafe0c760c1e1f369def40b1ed8e8decd8 4706 | no | 2 |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| | | **Total Files Size (in bytes):** | 214804 | | |

**This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.**

**New Applications Under 35 U.S.C. 111**
**If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.**

**National Stage of an International Application under 35 U.S.C. 371**
**If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.**

**New International Application Filed with the USPTO as a Receiving Office**
**If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.**

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| | 1 | US-20070234213 | | Oct 4, 2007 | Krikorian et al. | |
| | 2 | US-20080186377 | | Aug 7, 2008 | Eriksson et al. | |
| | 3 | US-20100046842 | | Feb 25, 2010 | Conwell | |
| | 4 | US-20110221745 | | Sep 15, 2011 | Goldman et al. | |
| | 5 | US-20110279638 | | Nov 17, 2011 | Periyannan et al. | |
| | 6 | US-20120016858 | | Jan 19, 2012 | Rathod | |

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Field

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

## FOREIGN PATENT DOCUMENTS

| Examiner Initial* | Cite No | Foreign Document Number[3] | Country Code[2] | Kind Code[4] | Publication Date | Name of Patentee or Applicant of cited Document | Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear | T[5] |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

## NON-PATENT LITERATURE DOCUMENTS

| Examiner Initials* | Cite No | Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published. | T[5] |
|---|---|---|---|
| | | | |

## EXAMINER SIGNATURE

| Examiner Signature | | Date Considered | |
|---|---|---|---|

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Field

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

## CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

☒ That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (1).

**OR**

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e) (2).

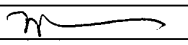
☐ See attached certification statement.

☒ Fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☐ No certification statement submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

| | | | |
|---|---|---|---|
| Signature |  | Date (YYYY-MM-DD) | 2013-04-18 |
| Name/Print | Michael A. Glenn | Registration Number | 30176 |

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

| Electronic Patent Application Fee Transmittal | |
|---|---|
| **Application Number:** | 12713637 |
| **Filing Date:** | 26-Feb-2010 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Attorney Docket Number:** | EQUI0016D2 |
| Filed as Small Entity | |

**Utility under 35 USC 111(a) Filing Fees**

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Basic Filing:** | | | | |
| **Pages:** | | | | |
| **Claims:** | | | | |
| **Miscellaneous-Filing:** | | | | |
| **Petition:** | | | | |
| **Patent-Appeals-and-Interference:** | | | | |
| **Post-Allowance-and-Post-Issuance:** | | | | |
| **Extension-of-Time:** | | | | |

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Miscellaneous:** | | | | |
| Submission- Information Disclosure Stmt | 2806 | 1 | 90 | 90 |
| | | **Total in USD ($)** | | **90** |

| Electronic Acknowledgement Receipt | |
|---|---|
| **EFS ID:** | 15559004 |
| **Application Number:** | 12713637 |
| **International Application Number:** | |
| **Confirmation Number:** | 5103 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Customer Number:** | 22862 |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Filer Authorized By:** | Michael Glenn |
| **Attorney Docket Number:** | EQUI0016D2 |
| **Receipt Date:** | 18-APR-2013 |
| **Filing Date:** | 26-FEB-2010 |
| **Time Stamp:** | 18:34:48 |
| **Application Type:** | Utility under 35 USC 111(a) |

## Payment information:

| | |
|---|---|
| Submitted with Payment | yes |
| Payment Type | Deposit Account |
| Payment was successfully received in RAM | $90 |
| RAM confirmation Number | 5285 |
| Deposit Account | 071445 |
| Authorized User | |
| The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:<br>Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)<br>Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges) | |

## File Listing:

| Document Number | Document Description | File Name | File Size(Bytes)/ Message Digest | Multi Part /.zip | Pages (if appl.) |
|---|---|---|---|---|---|
| 1 | | EQUI0016D2_IDS_FOREIGN_CERTIFICATION_2013-04-18.pdf | 252454<br>9c410d67dd26c5e0770084d02d5d8a1155271251 | yes | 3 |
| | **Multipart Description/PDF files in .zip description** | | | | |
| | **Document Description** | | **Start** | **End** | |
| | Information Disclosure Statement (IDS) Form (SB08) | | 1 | 2 | |
| | Transmittal Letter | | 3 | 3 | |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| 2 | Fee Worksheet (SB06) | fee-info.pdf | 29937<br>ef923be048622e42032b1b75c7014baff7ef9c9c | no | 2 |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| | | **Total Files Size (in bytes):** | 282391 | | |

**This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.**

**New Applications Under 35 U.S.C. 111**
**If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.**

**National Stage of an International Application under 35 U.S.C. 371**
**If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.**

**New International Application Filed with the USPTO as a Receiving Office**
**If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.**

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | | |
|---|---|---|
| First Named Inventor | : | Sean BARGER |
| Serial No. | : | 12/713,637 |
| Filed | : | February 26, 2010 |
| Art Unit | : | 2455 |
| Confirmation Number | : | 5103 |
| Examiner | : | Charles C. Murphy |
| Title | : | Automated Media Delivery System |
| Attorney Docket No. | : | EQUI0016D2 |

May 14, 2013

Mail Stop: ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

**Amendment Under Rule 37 CFR 1.312**

Applicant submits this amendment to the Specification to correct the related application data for the above-referenced Application.

Applicant does not believe any fees are due with this submission. However, the Commissioner is authorized to charge any fees that may be due and to credit any overpayments to Deposit Account 07-1445 (Order No. EQUI0016D2).

Should the Examiner deem it helpful he is encouraged to contact Applicant's attorney at (650) 474-8400.

Respectfully submitted,

Michael A. Glenn
Reg. No. 30,176

Customer No. 22862

Please replace paragraph one on page one of the Specification with the below paragraph.

## CROSS REFERENCE TO RELATED APPLICATIONS

This application is a Divisional of U.S. Serial No. 12/173,747, filed July 15, 2008, which is a Divisional of U.S. Serial No. 11/269,916, filed November 7, 2005, which is a Continuation-in-Part of U.S. Serial No. 09/929,904, filed August 14, 2001, now U.S. Patent No. 6,964,009 granted on November 8, 2005, which is a Continuation-in-Part of U.S. Serial No., 09/425,326, filed October 21, 1999, now U.S. Patent No. 6,792,575, granted on September 14, 2004, each of which is hereby incorporated in its entirety by this reference thereto.

# Electronic Acknowledgement Receipt

| | |
|---|---|
| **EFS ID:** | 15775672 |
| **Application Number:** | 12713637 |
| **International Application Number:** | |
| **Confirmation Number:** | 5103 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Customer Number:** | 22862 |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Filer Authorized By:** | Michael Glenn |
| **Attorney Docket Number:** | EQUI0016D2 |
| **Receipt Date:** | 14-MAY-2013 |
| **Filing Date:** | 26-FEB-2010 |
| **Time Stamp:** | 17:09:31 |
| **Application Type:** | Utility under 35 USC 111(a) |

## Payment information:

| | |
|---|---|
| Submitted with Payment | no |

## File Listing:

| Document Number | Document Description | File Name | File Size(Bytes)/ Message Digest | Multi Part /.zip | Pages (if appl.) |
|---|---|---|---|---|---|
| 1 | | EQUI0016D2-Rule312Amdt-2013-05-14.pdf | 89202<br>6c1ea72ca27c36eb6fc30824dd749d02326a6393 | yes | 3 |

| | Multipart Description/PDF files in .zip description | | |
|---|---|---|---|
| | **Document Description** | **Start** | **End** |
| | Amendment after Notice of Allowance (Rule 312) | 1 | 2 |
| | Specification | 3 | 3 |
| **Warnings:** | | | |
| **Information:** | | | |
| | **Total Files Size (in bytes):** | 89202 | |

**This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.**

**New Applications Under 35 U.S.C. 111**
**If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.**

**National Stage of an International Application under 35 U.S.C. 371**
**If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.**

**New International Application Filed with the USPTO as a Receiving Office**
**If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.**

## PART B - FEE(S) TRANSMITTAL

**Complete and send this form, together with applicable fee(s), to: Mail** Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
**or Fax** (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22862 7590 03/20/2013
GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

**Certificate of Mailing or Transmission**
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

| | |
|---|---|
| Christine Ortt | (Depositor's name) |
| | (Signature) |
| 06/20/2013 | (Date) |

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | INV001Sean BARGER | EQUI0016D2 | 5103 |

TITLE OF INVENTION: Automated Media Delivery System

| APPLN. TYPE | ENTITY STATUS | ISSUE FEE DUE | PUBLICATION FEE DUE | PREV. PAID ISSUE FEE | TOTAL FEE(S) DUE | DATE DUE |
|---|---|---|---|---|---|---|
| nonprovisional | SMALL | $890 | $300 | $0 | $1190 | 06/20/2013 |

| EXAMINER | ART UNIT | CLASS-SUBCLASS |
|---|---|---|
| MURPHY, CHARLES C | 2455 | 709-236000 |

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
- ☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
- ☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Michael A. Glenn
2 Glenn Patent Group
3

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE
AUTOMATED MEDIA PROCESSING SOLUTIONS, INC.
DBA EQUILIBRIUM, AMPS, INC.

(B) RESIDENCE: (CITY and STATE OR COUNTRY)
SAUSALITO, CALIFORNIA

Please check the appropriate assignee category or categories (will not be printed on the patent): ☐ Individual ☒ Corporation or other private group entity ☐ Government

4a. The following fee(s) are submitted:
- ☒ Issue Fee
- ☒ Publication Fee (No small entity discount permitted)
- ☐ Advance Order - # of Copies _____

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**
- ☐ A check is enclosed.
- ☐ Payment by credit card. Form PTO-2038 is attached.
- ☒ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number 071445 (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

| | |
|---|---|
| ☐ Applicant certifying micro entity status. See 37 CFR 1.29 | NOTE: Absent a valid certification of Micro Entity Status (see form PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment. |
| ☐ Applicant asserting small entity status. See 37 CFR 1.27 | NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status. |
| ☐ Applicant changing to regular undiscounted fee status. | NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable. |

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature _____ Date 06/20/2013

Typed or printed name Michael A. Glenn Registration No. 30176

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

## IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

| | | |
|---|---|---|
| First Named Inventor | : | Sean BARGER |
| Serial No. | : | 12/713,637 |
| Filed | : | February 26, 2010 |
| Art Unit | : | 2455 |
| Confirmation Number | : | 5103 |
| Examiner | : | Murphy, Charles C. |
| Title | : | Automated Media Delivery System |
| Attorney Docket No. | : | EQUI0016D2 |

June 20, 2013

Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

**COMMENTS ON STATEMENT OF REASONS FOR ALLOWANCE**

Dear Sir/Madam:

In the Notice of Allowance, mailed March 20, 2013, the Examiner provided Reasons for Allowance (hereinafter referred to as "Reasons"). Applicant believes the Reasons, to the extent understood, may be misconstrued and, as such, are incomplete. Applicant submits that indeed, the claims of the instant application, individually or in combination with other claims (via dependency), describe the patentable subject matter of Applicant's invention(s). Accordingly, the Reasons in no way bind or affect the interpretation, infringement, validity and/or enforceability of any claims(s) or patent(s) resulting from, or relating to this application.

Applicant does not believe any fees are due with this submission. However, the Commissioner is hereby authorized to charge any fees due or credit any overpayments to the Glenn Patent Group Deposit Account 071445 (Order No. EQUI0016D2).

Respectfully Submitted,



Michael A. Glenn
Reg. No. 30,176

Customer No. 22862

# Electronic Patent Application Fee Transmittal

| | |
|---|---|
| **Application Number:** | 12713637 |
| **Filing Date:** | 26-Feb-2010 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Attorney Docket Number:** | EQUI0016D2 |

Filed as Small Entity

## Utility under 35 USC 111(a) Filing Fees

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Basic Filing:** | | | | |
| **Pages:** | | | | |
| **Claims:** | | | | |
| **Miscellaneous-Filing:** | | | | |
| **Petition:** | | | | |
| **Patent-Appeals-and-Interference:** | | | | |
| **Post-Allowance-and-Post-Issuance:** | | | | |
| Utility Appl Issue Fee | 2501 | 1 | 890 | 890 |
| Publ. Fee- Early, Voluntary, or Normal | 1504 | 1 | 300 | 300 |

| Description | Fee Code | Quantity | Amount | Sub-Total in USD($) |
|---|---|---|---|---|
| **Extension-of-Time:** | | | | |
| **Miscellaneous:** | | | | |
| | | **Total in USD ($)** | | **1190** |

# Electronic Acknowledgement Receipt

| | |
|---|---|
| **EFS ID:** | 16106834 |
| **Application Number:** | 12713637 |
| **International Application Number:** | |
| **Confirmation Number:** | 5103 |
| **Title of Invention:** | Automated Media Delivery System |
| **First Named Inventor/Applicant Name:** | Sean BARGER |
| **Customer Number:** | 22862 |
| **Filer:** | Michael Glenn/Christine Ortt |
| **Filer Authorized By:** | Michael Glenn |
| **Attorney Docket Number:** | EQUI0016D2 |
| **Receipt Date:** | 20-JUN-2013 |
| **Filing Date:** | 26-FEB-2010 |
| **Time Stamp:** | 18:00:05 |
| **Application Type:** | Utility under 35 USC 111(a) |

## Payment information:

| | |
|---|---|
| Submitted with Payment | yes |
| Payment Type | Deposit Account |
| Payment was successfully received in RAM | $1190 |
| RAM confirmation Number | 5312 |
| Deposit Account | 071445 |
| Authorized User | |
| The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:<br>Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)<br>Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees) | |

| Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges) | | | | | |
|---|---|---|---|---|---|
| **File Listing:** | | | | | |
| **Document Number** | **Document Description** | **File Name** | **File Size(Bytes)/ Message Digest** | **Multi Part /.zip** | **Pages (if appl.)** |
| 1 | | EQUI0016D2_IssueFee_Comments_2013-06-20.pdf | 136343<br>317aac27384b821ed8f65dfc8bd4897cf918e22d | yes | 4 |
| | **Multipart Description/PDF files in .zip description** | | | | |
| | **Document Description** | | **Start** | **End** | |
| | Issue Fee Payment (PTO-85B) | | 1 | 2 | |
| | Post Allowance Communication - Incoming | | 3 | 4 | |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| 2 | Fee Worksheet (SB06) | fee-info.pdf | 31856<br>7288bb627219c6aba6fa161d304519e5e5e3aff5 | no | 2 |
| **Warnings:** | | | | | |
| **Information:** | | | | | |
| | | **Total Files Size (in bytes):** | 168199 | | |

**This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.**

**New Applications Under 35 U.S.C. 111**
**If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.**

**National Stage of an International Application under 35 U.S.C. 371**
**If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.**

**New International Application Filed with the USPTO as a Receiving Office**
**If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.**

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Field

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| /C.M./ | 1 | US-20070234213 | | Oct 4, 2007 | Krikorian et al. | |
| /C.M./ | 2 | US-20080186377 | | Aug 7, 2008 | Eriksson et al. | |
| /C.M./ | 3 | US-20100046842 | | Feb 25, 2010 | Conwell | |
| /C.M./ | 4 | US-20110221745 | | Sep 15, 2011 | Goldman et al. | |
| /C.M./ | 5 | US-20110279638 | | Nov 17, 2011 | Periyannan et al. | |
| /C.M./ | 6 | US-20120016858 | | Jan 19, 2012 | Rathod | |

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Field

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

## FOREIGN PATENT DOCUMENTS

| Examiner Initial* | Cite No | Foreign Document Number[3] | Country Code[2] | Kind Code[4] | Publication Date | Name of Patentee or Applicant of cited Document | Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear | T[5] |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

## NON-PATENT LITERATURE DOCUMENTS

| Examiner Initials* | Cite No | Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published. | T[5] |
|---|---|---|---|
| | | | |

## EXAMINER SIGNATURE

| Examiner Signature | /Charles Murphy/ | Date Considered | 06/17/2013 |
|---|---|---|---|

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 02/26/2010 | Sean BARGER | EQUI0016D2 | 5103 |

22862 7590 06/21/2013

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

| EXAMINER | |
|---|---|
| MURPHY, CHARLES C | |
| ART UNIT | PAPER NUMBER |
| 2455 | |

| NOTIFICATION DATE | DELIVERY MODE |
|---|---|
| 06/21/2013 | ELECTRONIC |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

ptomatters@glenn-law.com

| Response to Rule 312 Communication | Application No. | Applicant(s) |
|---|---|---|
| | 12/713,637 | BARGER ET AL. |
| | Examiner | Art Unit |
| | CHARLES MURPHY | 2455 |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address –*

1. ☒ The amendment filed on *14 May 2013* under 37 CFR 1.312 has been considered, and has been:

a) ☐ entered.

b) ☒ entered as directed to matters of form not affecting the scope of the invention.

c) ☐ disapproved because the amendment was filed after the payment of the issue fee.
Any amendment filed after the date the issue fee is paid must be accompanied by a petition under 37 CFR 1.313(c)(1) and the required fee to withdraw the application from issue.

d) ☐ disapproved. See explanation below.

e) ☐ entered in part. See explanation below.

Attached are copies of signed IDSs (2) filed 04/18/2013.

| | |
|---|---|
| /EMMANUEL L. MOISE/<br>Supervisory Patent Examiner, Art Unit 2455 | /CHARLES MURPHY/<br>Examiner, Art Unit 2455 |

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

| **INFORMATION DISCLOSURE STATEMENT BY APPLICANT** (Not for submission under 37 CFR 1.99) | | |
|---|---|---|
| | **Application Number** | 12/713,637 |
| | **Filing Date** | Feb 26, 2010 |
| | **First Named Inventor** | Sean Barger |
| | **Art Unit** | 2455 |
| | **Examiner Name** | Murphy, Charles C. |
| | **Attorney Docket Number** | EQUI0016D2 |

**U.S. PATENTS**

| Examiner Initial* | Cite No | Patent Number | Kind Code [1] | Issue Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|

**U.S. PATENT APPLICATION PUBLICATIONS**

| Examiner Initial* | Cite No | Publication Number | Kind Code [1] | Publication Date | Name of Patentee or Applicant of cited Document | Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear |
|---|---|---|---|---|---|---|
| /C.M./ | 1 | US-20050091311 | | Apr 28, 2005 | Lund, C et al. | |

**EXAMINER SIGNATURE**

| Examiner Signature | /Charles Murphy/ | Date Considered | 06/17/2013 |
|---|---|---|---|

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document.
[4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | ISSUE DATE | PATENT NO. | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 12/713,637 | 07/23/2013 | 8495242 | EQUI0016D2 | 5103 |

22862 7590 07/03/2013

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

# ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

**Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)**
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 76 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Sean BARGER, Mill Valley, CA;
Steve Johnson, Mill Valley, CA;
Matt Butler, Beaverton, OR;
Jerry Destremps, Sausalito, CA;
David Pochron, Cambridge, WI;
Trent Brown, San Anselmo, CA;

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The USA offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to encourage and facilitate business investment. To learn more about why the USA is the best country in the world to develop technology, manufacture products, and grow your business, visit SelectUSA.gov.

AO 120 (Rev. 08/10)

| TO: | **Mail Stop 8**<br>**Director of the U.S. Patent and Trademark Office**<br>**P.O. Box 1450**<br>**Alexandria, VA 22313-1450** | **REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK** |
|---|---|---|

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. ( ☐ the patent action involves 35 U.S.C. § 292.):

| DOCKET NO. | DATE FILED | U.S. DISTRICT COURT |
|---|---|---|
| | 5/24/2022 | for the District of Delaware |

| PLAINTIFF | DEFENDANT |
|---|---|
| Equil IP Holdings LLC | Akamai Technologies Inc. |

| | PATENT OR TRADEMARK NO. | DATE OF PATENT OR TRADEMARK | HOLDER OF PATENT OR TRADEMARK |
|---|---|---|---|
| 1 | 9,158,745 | 10/13/2015 | Equil IP Holdings LLC |
| 2 | 8,495,242 | 7/23/2013 | Equil IP Holdings LLC |
| 3 | | | |
| 4 | | | |
| 5 | | | |

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

| DATE INCLUDED | INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading | |
|---|---|---|
| PATENT OR TRADEMARK NO. | DATE OF PATENT OR TRADEMARK | HOLDER OF PATENT OR TRADEMARK |
| 1 | | |
| 2 | | |
| 3 | | |
| 4 | | |
| 5 | | |

In the above—entitled case, the following decision has been rendered or judgement issued:

| DECISION/JUDGEMENT |
|---|
| |

| CLERK | (BY) DEPUTY CLERK | DATE |
|---|---|---|
| | | |

**Copy 1—Upon initiation of action, mail this copy to Director** **Copy 3—Upon termination of action, mail this copy to Director**
**Copy 2—Upon filing document adding patent(s), mail this copy to Director** **Copy 4—Case file copy**