

environment assumes a web server and a web browser connected through a TCP/IP network, such as over the public Internet or a private Intranet. Possible web servers could include Microsoft Internet Information Server, Netscape Enterprise Server or Apache. Possible web browsers include Microsoft Internet Explorer or Netscape Navigator. The client (i.e. web browser) makes a request **1001** to the server (i.e. web server) for a particular web page. This is usually accomplished by a user clicking on a button or a link within a web page. The web server gets the layout and content preferences **1002** for that particular user, with the request to the database keyed off of a unique user id stored in the client (i.e. web browser) and the User profile database **1003**. The web server then retrieves the content **1004** for the page that has been requested from the content database **1005**. The relevant user-centric content, such as calendar, email, contact list, and task list items are then retrieved **1006**. (See FIG. 11 for a more detailed description of this process.) The query to the database utilizes the user content preferences stored as part of the user profile in the User profile database **1003** to filter the content that is returned. The content that is returned is then formatted into a web page **1007** according to the layout preferences defined in the user profile. The web page is then returned to the client and displayed to the user **1008**.

FIG. 11 describes the process of retrieving user-centric content to add to a web page. This process describes **1006** in FIG. 10B in a more detailed fashion. It assumes that the server already has obtained the user profile and the existing content that is going to be integrated into this page. The server parses **1110** the filtered content, looking for instances of events, contact names and email addresses. If any of these are found, they are tagged and stored in a temporary holding space. Then, the server tries to find any user-centric content **1120** stored in various databases. This involves matching the tagged items in the temporary storage space with calendar items **1130** in the Calendar Database **1140**; email items **1115** in the Email Database **1114**; contact items **1117** in the Contact Database **1168**; task list items **1119** in the Task List Database **1118**; and news items **1121** in the News Database **1120**. After retrieving any relevant user-centric content, it is compiled together and returned **1122**.

User Persona

The system allows the user to create a number of different personas that aggregate profile information into sets that are useful in different contexts. A user may create one persona when making purchases for his home. This persona may contain his home address and may indicate that this user is looking to find a good bargain when shopping. The same user may create a second persona that can be used when he is in a work context. This persona may store the user's work address and may indicate that the user prefers certain vendors or works for a certain company that has a discount program in place. When shopping for work-related items, the user may use this persona. A persona may also contain rules and restrictions. For instance, the work persona may restrict the user to making airline reservations with only one travel agent and utilizing booking rules set up by his employer.

FIG. 12 describes the relationship between a user, his multiple personas and his multiple profiles. At the User Level is the User Profile **1200**. This profile describes the user and his account information. There is one unique record in the database for each user who has an account. Attached to each user are multiple Personas **1220**, **1230** & **1240**. These Personas are used to group multiple Profiles into useful

contexts. For instance, consider a user who lives in San Francisco and works in Palo Alto, but has a mountain cabin in Lake Tahoe. He has three different contexts in which he might be accessing his site. One context is work-related. The other two are home-life related, but in different locations. The user can create a Persona for Work **1220**, a Persona for Home **1230**, and a Persona for his cabin home **1240**. Each Persona references a different General Profile **1250**, **1260** and **1270** which contains the address for that location. Hence, there are three General Profiles. Each Persona also references one of two Travel Profiles. The user maintains a Work Travel Profile **1280** that contains all of the business rules related to booking tickets and making reservations. This Profile may specify, for instance, that this person only travels in Business or First Class and his preferred airline is United Airlines. The Work Persona references this Work Travel Profile. The user may also maintain a Home Travel Profile **1290** that specifies that he prefers to travel in coach and wants to find non-refundable fairs, since they are generally cheaper. Both the Persona for Home and the Persona for the cabin home point to the Home Travel Profile.

FIG. 13 describes the data model that supports the Persona concept. The user table **1310** contains a record for each user who has an account in the system. This table contains a username and a password **1320** as well as a unique identifier. Each user can have multiple Personas **1330**, which act as containers for more specialized structures called Profiles **1340**. Profiles contain the detailed personal information in Profile Field **1350** records. Attached to each Profile are sets of Profile Restriction **1360** records. These each contain a Name **1370** and a Rule **1380**, which define the restriction. The Rule is in the form of a pattern like (if x then y), which allows the Rule to be restricted to certain uses. An example Profile Restriction would be the rule that dictates that the user cannot book a flight on a certain airline contained in the list. This Profile Restriction could be contained in the "Travel" Profile of the "Work" Persona set up by the user's employer, for instance. Each Profile Field also contains a set of Permissions **1390** that are contained in that record. These permissions dictate who has what access rights to that particular Profile Field's information.

Intention-Centric Interface

Satisfying Customer Intentions, such as Planning for Retirement or Relocating requires a specialized interface. Customer Intentions require extensive planning and coordination across many areas, ranging from financial security, housing and transportation to healthcare, personal and professional development, and entertainment, among others. Satisfying Intentions requires a network of complementary businesses, working across industries, to help meet consumers' needs.

An Intention-Centric Interface is a user interface designed to help the user manage personal Intentions. At any given point, the interface content is customized to show only content that relates to that particular Intention. The Intention-Centric Interface allows the user to manage the process of satisfying that particular Intention. This involves a series of discrete steps and a set of content areas the user can access. At any point, the user can also switch the interface to manage a different Intention, and this act will change the content of the interface to include only that content which is relevant to the satisfaction of the newly selected Intention.

FIG. 14 provides a detailed description of the data model needed to support an Intention-Centric Interface. Each User

Persona **1410** (see FIG. **13** for a more detailed description of the Persona data model.) has any number of active User Intentions **1420**. Each active User Intention is given a Nickname **1430**, which is the display name the user sees on the screen. Each active User Intention also contains a number of Data Fields **1440**, which contain any user data collected throughout the interaction with the user. For instance, if the user had filled out a form on the screen and one of the fields was Social Security Number, the corresponding Data Field would contain Name = "SSN" **1450**, Value = "999-99-9999" **1460**. Each User Intention also keeps track of Intention Step **1470** completion status. The Completion **1480** field indicates whether the user has completed the step. Every User Intention is a user-specific version of a Generic Intention **1490**, which is the default model for that Intention for all users. The Generic Intention is customized through Custom Rules **1411** and **1412** that are attached to the sub-steps in the Intention. These Custom Rules are patterns describing how the system will customize the Intention for each individual user using the individual user's profile information.

Statistical Agent

An agent keeps track of key statistics for each user. These statistics are used in a manner similar to the Tamagochi virtual reality pet toy to encourage certain behaviors from the user. The statistics that are recorded are frequency of login, frequency of rating of content such as news articles, and activity of agents, measured by the number of tasks which it performs in a certain period. This information is used by the system to emotionally appeal to the user to encourage certain behaviors.

FIG. **15** describes the process for generating the page that displays the agent's current statistics. When the user requests the agent statistics page **1510** with the client browser, the server retrieves the users' statistics **1520** from the users' profile database **1530**. The server then performs the mathematical calculations necessary to create a normalized set of statistics **1540**. The server then retrieves the formulas **1550** from the content database **1560** that will be used to calculate the user-centric statistics. Graphs are then generated **1570** using the generic formulas and that user's statistics. These graphs are inserted into a template to create the statistics page **1580**. This page is then returned to the user **1590**.

Personalized Product Report Service

The system provide Consumer Report-like service that is customized for each user based on a user profile. The system records and provides ratings from users about product quality and desirability on a number of dimensions. The difference between this system and traditional product quality measurement services is that the ratings that come back to the users are personalized. This service works by finding the people who have the closest match to the user's profile and have previously rated the product being asked for. Using this algorithm will help to ensure that the product reports sent back to the user only contain statistics from people who are similar to that user.

FIG. **16** describes the algorithm for determining the personalized product ratings for a user. When the user requests a product report **1610** for product X, the algorithm retrieves the profiles **1620** from the profile database **1630** (which includes product ratings) of those users who have previously rated that product. Then the system retrieves the default thresholds **1640** for the profile matching algorithm

from the content database **1650**. It then maps all of the short list of users along several dimensions specified in the profile matching algorithm **1660**. The top n (specified previously as a threshold variable) nearest neighbors are then determined and a test is performed to decide if they are within distance y (also specified previously as a threshold variable) of the user's profile in the set **1670** using the results from the profile matching algorithm. If they are not within the threshold, then the threshold variables are relaxed **1680**, and the test is run again. This processing is repeated until the test returns true. The product ratings from the smaller set of n nearest neighbors are then used to determine a number of product statistics **1690** along several dimensions. Those statistics are inserted into a product report template **1695** and returned to the user **1697** as a product report.

Personal Profile and Services Ubiquity

This system provides one central storage place for a person's profile. This storage place is a server available through the public Internet, accessible by any device that is connected to the Internet and has appropriate access. Because of the ubiquitous accessibility of the profile, numerous access devices can be used to customize services for the user based on his profile. For example, a merchant's web site can use this profile to provide personalized content to the user. A Personal Digital Assistant (PDA) with Internet access can synchronize the person's calendar, email, contact list, task list and notes on the PDA with the version stored in the Internet site. This enables the person to only have to maintain one version of this data in order to have it available whenever it is needed and in whatever formats it is needed.

FIG. **17** presents the detailed logic associated with the many different methods for accessing this centrally stored profile. The profile database **1710** is the central storage place for the users' profile information. The profile gateway server **1720** receives all requests for profile information, whether from the user himself or merchants trying to provide a service to the user. The profile gateway server is responsible for ensuring that information is only given out when the profile owner specifically grants permission. Any device that can access the public Internet **1730** over TCP/IP (a standard network communications protocol) is able to request information from the profile database via intelligent HTTP requests. Consumers will be able to gain access to services from devices such as their televisions **1740**, mobile phones, Smart Cards, gas meters, water meters, kitchen appliances, security systems, desktop computers, laptops, pocket organizers, PDAs, and their vehicles, among others. Likewise, merchants **1750** will be able to access those profiles (given permission from the consumer who owns each profile), and will be able to offer customized, personalized services to consumers because of this.

One possible use of the ubiquitous profile is for a hotel chain. A consumer can carry a Smart Card that holds a digital certificate uniquely identifying him. This Smart Card's digital certificate has been issued by the system and it recorded his profile information into the profile database. The consumer brings this card into a hotel chain and checks in. The hotel employee swipes the Smart Card and the consumer enters his Pin number, unlocking the digital certificate. The certificate is sent to the profile gateway server (using a secure transmission protocol) and is authenticated. The hotel is then given access to a certain part of the consumer's profile that he has previously specified. The hotel can then retrieve all of the consumer's billing information as well as preferences for hotel room, etc. The hotel can also access the consumer's movie and dining prefer-

ences and offer customized menus for both of them. The hotel can offer to send an email to the consumer's spouse letting him/her know the person checked into the hotel and is safe. All transaction information can be uploaded to the consumer's profile after the hotel checks him in. This will allow partners of the hotel to utilize the information about the consumer that the hotel has gathered (again, given the consumer's permission).

Intention Value Network

In an Intention Value Network, the overall integrator system coordinates the delivery of products and services for a user. The integrator manages a network of approved suppliers providing products and services, both physical and virtual, to a user based on the user's preferences as reflected in the user's profile. The integrator manages the relationship between suppliers and consumers and coordinates the suppliers' fulfillment of consumers' intentions. It does this by providing the consumer with information about products and suppliers and offering objective advice, among other things.

FIG. 18 discloses the detailed interaction between a consumer and the integrator involving one supplier. The user accesses a Web Browser **1810** and requests product and pricing information from the integrator. The request is sent from the user's browser to the integrator's Web/Application Server **1820**. The user's preferences and personal information is obtained from an integrator's customer profile database **1830** and returned to the Web/Application server. The requested product information is extracted from the supplier's product database **1840** and customized for the particular customer. The Web/Application server updates the supplier's customer information database **1850** with the inquiry information about the customer. The product and pricing information is then formatted into a Web Page **1860** and returned to the customer's Web Browser.

Summary Agent

A suite of software agents running on the application and web servers are programmed to take care of repetitive or mundane tasks for the user. The agents work according to rules set up by the user and are only allowed to perform tasks explicitly defined by the user. The agents can take care of paying bills for the user, filtering content and emails, and providing a summary view of tasks and agent activity. The user interface for the agent can be modified to suit the particular user.

FIG. 19 discloses the logic in accordance with a preferred embodiment processing by an agent to generate a verbal summary for the user. When the user requests the summary page **1900**, the server gets the user's agent preferences **1920**, such as agent type, rules and summary level from the user profile database **1930**. The server gets the content **1940**, such as emails, to do list items, news, and bills, from the content database **1950**. The agent parses all of this content, using the rules stored in the profile database, and summarizes the content **1960**. The content is formatted into a web page **1970** according to a template. The text for the agent's speech is generated **1980**, using the content from the content database **1990** and speech templates stored in the database. This speech text is inserted into the web page **1995** and the page is returned to the user **1997**.

Trusted Third Party

The above scenario requires the web site to maintain a guarantee of privacy of information according to a published policy. This system is the consumer's Trusted Third Party,

acting on his behalf in every case, erring on the side of privacy of information, rather than on the side of stimulation of commerce opportunities. The Trusted Third Party has a set of processes in place that guarantee certain complicity with the stated policy.

"meCommerce"

This word extends the word "eCommerce" to mean "personalized electronic commerce." FIG. 20 illustrates a display login in accordance with a preferred embodiment. The display is implemented as a Microsoft Internet Explorer application with an agent **2000** that guides a user through the process of interacting with the system to customize and personalize various system components to gather information and interact with the user's personal requirements. A user enters a username at **2010** and a password at **2020** and selects a button **2040** to initiate the login procedure. As the logo **2030** suggests, the system transforms electronic commerce into a personalized, so called "me" commerce.

FIG. 21 illustrates a managing daily logistics display in accordance with a preferred embodiment. A user is greeted by an animated agent **2100** with a personalized message **2190**. The user can select from various activities based on requirements, including travel **2110**, household chores **2120**, finances **2130** and marketplace activities **2140**. Icons **2142** for routine tasks such as e-mail, calendaring and document preparation are also provided to facilitate rapid navigation from one activity to another. Direct links **2146** are also provided to allow transfer of news and other items of interest. Various profiles can be selected based on where the user is located. For example, work, home or vacation. The profiles can be added **2170** as a user requires a new profile for another location. Various items **2180** of personal information are collected from the user to support various endeavors. Moreover, permissions **2150** are set for items **2180** to assure information is timely and current.

FIG. 22 illustrates a user main display in accordance with a preferred embodiment. World **2200** and local news **2210** is provided based on a user's preference. The user has also selected real estate **2230** as an item to provide direct information on the main display. Also, a different agent **2220** is provided based on the user's preference.

FIG. 23 illustrates an agent interaction in accordance with a preferred embodiment. The agent **2310** is communicating information **2300** to a user indicating that the user's life insurance needs have changed and pointing the user to the chart that best summarizes the information for the user. Particular tips **2395** are provided to facilitate more detailed information based on current user statistics. A chart **2370** of the user's life insurance needs is also highlighted at the center of the display to assist the user in determining appropriate action. A button **2380** is provided to facilitate changing the policy and a set of buttons **2390** are provided to assist a user in selecting various views of the user's insurance requirements.

Event Backgrounder

An Event Backgrounder is a short description of an upcoming event that is sent to the user just before an event. The Event Backgrounder is constantly updated with the latest information related to this event. Pertinent information such as itinerary and logistics are included, and other useful information, such as people the user knows who might be in the same location, are also included. The purpose of the Event Backgrounder is to provide the most up-to-date information about an event, drawing from a number of resources,

such as public web sites and the user's calendar and contact lists, to allow the user to react optimally in a given situation.

Vicinity Friend Finder

This software looks for opportunities to tell the user when a friend, family member or acquaintance is or is going to be in the same vicinity as the user. This software scans the user's calendar for upcoming events. It then uses a geographic map to compare those calendar events with the calendar events of people who are listed in his contact list. It then informs the user of any matches, thus telling the user that someone is scheduled to be near him at a particular time.

Information Overload

The term information overload is now relatively understood in both its definition as well as its implications and consequences. People have a finite amount of attention that is available at any one time, but there is more and more vying for that attention every day. In short, too much information and too little time are the primary factors complicating the lives of most knowledge workers today.

The first attempts to dynamically deal with information overload were primarily focused on the intelligent filtering of information such that the quantity of information would be lessened. Rather than simply removing random bits of information, however, most of these approaches tried to be intelligent about what information was ultimately presented to the user. This was accomplished by evaluating each document based on the user's interests and discarding the less relevant ones. It follows, therefore, that the quality was also increased.

Filtering the information is only a first step in dealing with information in this new age. Arguably, just as important as the quality of the document is having ready access to it. Once you have entered a meeting, a document containing critical information about the meeting subject delivered to your office is of little value. As the speed of business continues to increase fueled by the technologies of interconnectedness, the ability to receive quality information wherever and whenever you are becomes critical. This new approach is called intelligent information delivery and is heralding in a new information age.

A preferred embodiment demonstrates the intelligent information delivery theory described above in an attempt to not only reduce information overload, but to deliver high quality information where and when users' require it. In other words, the system delivers right information to the right person at the right time and the right place.

Active Knowledge Management System Description

FIG. 24 is a block diagram of an active knowledge management system in accordance with a preferred embodiment. The system consists of the following parts: back-end **2400** connection to one or more servers, personal mobile wireless clients (Awareness Machine)**2430**, **2436**, public clients (Magic Wall) **2410**, **2420**, web clients **2446**, **2448**, e-mail clients **2450**, **2460**.

Back-end Server (2400) Processes

FIG. 25 is a block diagram of a back end server in accordance with a preferred embodiment. The back-end (**2400** of FIG. 24) is a computer system that has the following software active: Intelligent Agents Coordinator

(Munin) **2580**, Information Prioritization Subsystem **2530**, a set of continuously and periodically running information gathering and processing Intelligent Agents **2500**, **2502** and **2504**, User Profiles Database **2542** and supporting software, Information Channels Database **2542** and supporting software, communications software **2550**, information transformation software **2560**, and auxiliary software.

The Awareness Machine (**2446** & **2448** of FIG. 24)

The Awareness Machine is a combination of hardware device and software application. The hardware consists of handheld personal computer and wireless communications device. The Awareness Machine reflects a constantly updated state-of-the-owner's-world by continually receiving a wireless trickle of information. This information, mined and processed by a suite of intelligent agents, consists of mail messages, news that meets each user's preferences, schedule updates, background information on upcoming meetings and events, as well as weather and traffic. The Awareness Machine is covered by another patent application.

FIG. 26 is a block diagram of a magic wall in accordance with a preferred embodiment.

The Magic Wall

The Magic Wall hardware includes:

Computer system **2640** connected to the back-end server
Sensor array **2634**, **2630** and **2632** detects presence, position, and identity of a person

Large touch-sensitive display **2620**

Sound input **2610**/output **2614** hardware

The Magic Wall software supports:

Multimedia output compatible with current Web standards

Speech recognition

Tactile input

Intelligent agents representations in the form of speech-enabled animated characters

The Magic Wall operates as follows:

1. If a user appears in the vicinity of Magic Wall, the sensor array triggers "user here" event that sends an environmental cue containing the person's id and the location to the Intelligent Agent Coordinator.
2. User is identified based on the information returned by the sensor array.
3. The Magic Wall switches to "locked on the user" mode. If another user approaches, the system will notify him or her that it cannot serve another user while the current user is being served.
4. Intelligent Agent Coordinator is notified about the user presence.
5. The Intelligent Agent Coordinator decides if there is pertinent to that user and Magic Wall location time-sensitive information to show (e.g. traffic report, meeting reminder). If such information exists, it is prepared for delivery. If not, control is transferred to the Information Prioritization Subsystem.
6. Information Prioritization Subsystem decides what information is most relevant to the user based on their personal profile, freshness of the information, and the Intelligent Agent Coordinator's prior suggestions.
7. The page of information identified as the most relevant to the user at this time and place is shown. The act of the information delivery can also include animation and speech output of the intelligent agent representation.

8. If user desires so, he or she can ask Magic Wall to show a particular page. The Magic Wall recognizes the speech fragment and then identifies and shows the requested page.

9. As the user departs from the Magic Wall area, the sensor array triggers "user left" event.

10. The Magic Wall switches back to the waiting state.

Other Clients

The Web client is a standard browser navigating to a set of Web pages which allow user to see the same information that is available via the Magic Wall.

The e-mail client is any standard e-mail program.

Intelligent Agent Coordinator Description

This piece of code is the coordinating agent (or meta-agent) for the Active Knowledge Management system. This means that all communications between the system and each user, as well as communication between the different minion agents are handled (coordinated) by the Intelligent Agent Coordinator. Examples of these minion agents are:

BackgroundFinder—an agent that parses meeting text determining important keywords and phrases and finds background information on the meeting for each user

TrafficFinder—an agent that finds traffic information for each user based on where they live

Several other agents that are responsible for doing statistical analysis of the data in each user's profile and updating fields pertinent to that data

The Intelligent Agent Coordinator 2580 of FIG. 25 is also the user's "interface" to the system, in that whenever the user interacts with the system, regardless of the GUI or other end-user interface, they are ultimately dealing with (asking questions of or sending commands to) the Intelligent Agent Coordinator. The Intelligent Agent Coordinator has four primary responsibilities: 1) monitoring user activities, 2) handling information requests, 3) maintaining each user's profile, and 4) routing information to and from users and to and from the other respective agents.

Monitoring User Activities

Anytime a user triggers a sensor the Intelligent Agent Coordinator receives an "environmental cue." These cues not only enable the Intelligent Agent Coordinator to gain an understanding where users' are for information delivery purposes, but also to learn the standard patterns (arrival time, departure time, etc.) of each persons' life. These patterns are constantly being updated and refined in an attempt to increase the system's intelligence when delivering information. For instance, today it is not uncommon for a person to have several email accounts (work-based, home-based, mobile-based, etc.) as well as several different computers involved in the retrieval process for all of these accounts. Thus, for the Intelligent Agent Coordinator to be successful in delivering information to the correct location it must take into account all of these accounts and the times that the user is likely to be accessing them in order to maximize the probability that the user will see the information. This will be discussed further in another section.

Handling Information Requests

The Intelligent Agent Coordinator handles information requests from other agents in order to personalize information intended for each user and to more accurately reflect each user's interests in the information they are given. These requests will commonly be related to the user's profile. For instance, if an agent was preparing a traffic report for a user

it may request the traffic region (search string) of that user from the Intelligent Agent Coordinator. All access to the user's profile data is accessed in this method.

Maintaining User Profiles

User profiles contain extensive information about the users. This information is a blend of user-specified data and information that the Intelligent Agent Coordinator has learned and extrapolated from each user's information and activities. In order to protect the data contained in the profiles, the Intelligent Agent Coordinator must handle all user information requests. The Intelligent Agent Coordinator is constantly modifying and updating these profiles by watching the user's activities and attempting to learn the patterns of their lives in order to assist in the more routine, mundane tasks. The Intelligent Agent Coordinator also employs other agents to glean meaning from each user's daily activities. These agents mine this data trying to discover indications of current interests, long-term interests, as well as time delivery preferences for each type of information. Another important aspect of the Intelligent Agent Coordinator's observations is that it also tries to determine where each user is physically located throughout the day for routing purposes.

Information Routing

Most people are mobile throughout their day. The Intelligent Agent Coordinator tries to be sensitive to this fact by attempting to determine, both by observation (unsupervised learning) and from cues from the environment, where users are or are likely to be located. This is certainly important for determining where to send the user's information, but also for determining in which format to send the information. For instance, if a user were at her desk and using the web client, the Intelligent Agent Coordinator would be receiving indications of activity from her PC and would know to send any necessary information there. In addition, because desktop PCs are generally quite powerful, a full-featured, graphically intense version could be sent. However, consider an alternative situation: the Intelligent Agent Coordinator has received an indication (via the keycard reader next to the exit) that you have just left the building. Minutes later the Intelligent Agent Coordinator also receives notification that you have received an urgent message. The Intelligent Agent Coordinator, knowing that you have left the building and having not received any other indications, assumes that you are reachable via your handheld device (for which it also knows the capabilities) and sends the text of the urgent message there, rather than a more graphically-oriented version.

Inherent Innovations

The Active Knowledge Management system represents some of the most advanced thinking in the world of knowledge management and human computer interaction. Some of the primary innovations include the following:

The Intelligent Agent Coordinator as illustrated above.

The development, demonstration, and realization of the theory of Intelligent Information Delivery

Support for several channels of information delivery, all of which utilize a common back-end. For instance, if a user is in front of a Magic Wall the information will be presented in a multimedia-rich form. If the system determines that the user is mobile, the information will be sent by to their Awareness Machine in standard text. It facilitates delivery of information whenever and wherever a user requires the information.

Personalization of information based not only on a static user profile, but also by taking into account history of the user interactions and current real-time situation including "who, where, and when" awareness.

Utilization of fast and scalable Information Prioritization Subsystem that takes into account Intelligent Agents Coordinator opinion, user preferences, and history of user interactions. It takes the load of mundane decisions off the Intelligent Agents part therefore allowing the agents to be much more sophisticated and precise without compromising the system scalability.

Speech recognition and speech synthesis in combination with intelligent agent animated representation and tactile input provides for efficient, intuitive, and emotionally rewarding interaction with the system.

Supporting Code in Accordance With A Preferred Embodiment

The following code is written and executed in the Microsoft Active Server Pages environment in accordance with a preferred embodiment. It consists primarily of Microsoft Jscript with some database calls embedded in the code to query and store information in the database.

Intention-Centric Interface

Create an Intention ASP Page ("intention_create.asp")

```

<%@ LANGUAGE = "JScript" %>
<%
Response.Buffer = true;
Response.Expires = 0;
%>
<html>
<head>
    <title>Create An Intention</title>
</head>
<body bgcolor="#FFE9D5" style="font-family: Arial" text="#000000">
<%
//Define some variables
upl = Server.CreateObject("SoftArtisans.FileUp")
intention_name = upl.Form("intention_name")
intention_desc = upl.Form("intention_desc")
//intention_name = Request.Form("intention_name")
//intention_desc = Request.Form("intention_desc")
//intention_icon = Request.Form("intention_icon")
submitted = upl.Form("submitted")
items = new Enumerator(upl.Form)
%>
<%
//Establish connection to the database
objConnection = Server.CreateObject("ADODB.Connection")
objConnection.Open("Maelstrom").
%>
<%
//Check to see if the person hit the button and do the appropriate thing
if (submitted == "Add/Delete")
{
    flag = "false"
    //loop through all the inputs
    while(!items.atEnd())
    {
        i = items.item()
        //if items are checked then delete them
        if(upl.Form(i) == "on")
        {
            objConnection.Execute("delete from user_intention where
intention_id = " + i);
            objConnection.Execute("delete from intentions where intention_id = "
+ i);
            objConnection.Execute("delete from tools_to_intention where
intention_id = " + i)
            flag = "true"
        }
        items.moveNext()
    }
    // if items were not deleted then insert whatever is in the text field in the
database
    if(flag == "false")
    {
        intention_name_short = intention_name.replace(/ /gi,"")
        objConnection.Execute("INSERT INTO intentions
(intention_name,intention_desc,intention_icon) values('"+ intention_name + "','"+
intention_desc + "','"+ intention_name_short + ".gif" + "')")
        Response.write("the intention short name is " + intention_name_short);
        upl.SaveAs("E:development/asp_examples/" + intention_name_short + ".gif")
    }
}

// Query the database to show the most recent items.
rsCustomersList = objConnection.Execute("SELECT * FROM intentions")

```

-continued

Create an Intention ASP Page ("intention_create.asp")

```

%>
☐

```

Retrieve Intentions List ASP Page ("intentions_list.asp")

```

<!-- #include file="include/check_authentication.inc" -->
<HTML>
<HEAD>
<TITLE>mySite! Intentions List</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function intentionsList () {
this.internalArray = new Array();

```

```
<%  
    // establish connection to the database  
    objConnection = Server.CreateObject("ADODB.Connection");  
    objConnection.Open("Maelstrom");  
    // create query  
    intentionsQuery = objConnection.Execute("SELECT * FROM intentions ORDER BY  
intention__name asc");  
%>  
  
// write out the options  
  
numOptions = 0  
while (!intentionsQuery.EOF) {  
    intentionName = intentionsQuery("intention__name");  
    intentionIcon = intentionsQuery("intention__icon");  
  
    this.internalArray[<%= numOptions %>] = new Array(2);  
    this.internalArray[<%= numOptions %>][0] = "<%= intentionName %>";  
    this.internalArray[<%= numOptions %>][1] = "images/<%= intentionIcon  
%>";  
    numOptions++; intentionsQuery.MoveNext();    %>  
}  
}  
numIntentions = <%= numOptions %>;  
intentionArray = new intentionsList().internalArray;  
function selectIntention () {  
    for (i=0;i<numIntentions;i++) {  
        if (IntentionsListSelect.options[i].selected) {  
            intentionNameTextField.value = intentionArray[i][0];  
            //intentionPicture.src = intentionArray[i][1];  
            break;  
        }  
    }  
}  
</SCRIPT>  
</HEAD>  
<BODY BGCOLOR="<%=Session("main__background")%" style="font-family: Arial">  
<CENTER>  
<!- --><FORM NAME="intention_list"> --->  
<TABLE FRAME="BOX" border=0 CELLPADDING="2" CELLSPACING="2">  
<TR><TD COLSPAN="3" STYLE="font: 20pt arial" ALIGN="CENTER"><B>Add a mySite!  
Intention</B></TD></TR>  
<TR><TD COLSPAN="3">&nbsp;</TD></TR>  
<TR>  
    <TD width="100"><font size="-1">Please Select An Intention You Would Like to Add  
to Your List</font></TD>  
    <TD colspan=2>  
        <SELECT ID="IntentionsListSelect" NAME="IntentionsListSelect" SIZE="10"  
style="font: 9pt Arial;" onClick="selectIntention()">  
            <%  
                intentionsQuery.moveFirst();  
                for(j=0;j<numOptions;j++) { %>  
                    <OPTION VALUE="<%= intentionsQuery("intention_id") %" <% if (j ==  
0) { %> SELECTED <% } %>>  
                        <%= intentionsQuery("intention__name") %>  
                        <% intentionsQuery.moveToNext()  
                            }  
                                intentionsQuery.moveFirst();  
                                    %>  
                                        </SELECT>  
                                            </TD>  
                                                </TR>  
                                                    <TR><TD COLSPAN="3">&nbsp;</TD></TR>  
                                                        <TR>  
                                                            <TD width="100"><font size="-1">Customize the Intention name</font></TD>  
                                                                <TD COLSPAN="2"><INPUT TYPE="text" NAME="intentionNameTextField"  
ID="intentionNameTextField" SIZE="30" VALUE="<%= intentionsQuery("intention__name")  
%>"></TD>  
                                                                    </TR>  
                                                                        <TR><TD COLSPAN="3">&nbsp;</TD></TR>  
                                                                            <TR>  
                                                                                <TD COLSPAN="3" ALIGN="CENTER">  
                                                                                    <INPUT TYPE="button" NAME="intentionOKButton" VALUE=" OK " SIZE="10"  
ID="intentionOKButton" onClick="javaScript:top.opener.top.navframe.addAnIntention();">  
                                                                                        &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~  
P;  
                                                                    <INPUT TYPE="button" NAME="intentionCancelButton" VALUE="Cancel" SIZE="10"  
ID="intentionCancelButton" onClick="self.close();">
```

-continued

```

Retrieve Intentions List ASP Page ("intentions_list.asp")
</TD>
</TR>
</TABLE>
<!-- </FORM> --->
</CENTER>
<% objConnection.Close(); %>
</BODY>
</HTML>

```

```

Display User Intention List ASP Page (excerpted from "navigation.asp")
<DIV ID="intentionlist" style="position: absolute; width:210; height:95; left: 365pt; top:
-5; visibility: hidden; font-family: Arial; font-color: #000000; font: 8pt Arial ;" >
<DIV style="position: absolute; top:7; left:7; height:78; width:210; z-index:2;
background: <%=Session("main_background")%>; border: solid 1pt #000000; padding: 3pt;
overflow: auto; alink: black; link: black;">
<body LINK="#000000" ALINK="#000000" vlink="black">
    <%
        // create query
        intentionsQuery = objConnection.Execute("SELECT user_intention.*
FROM user_intention, user_intention_to_persona WHERE
user_intention_to_persona.user_persona_id = " + Session("currentUser.Persona") + " AND
user_intention_to_persona.user_intention_id = user_intention.user_intention_id");
        numintentions = 0;
        Response.Write("<SCRIPT>numintentions=" +
intentionsQuery.RecordCount + "</SCRIPT><TABLE cellpadding='0' width='100%'
cellspacing='0'>");
        while (!intentionsQuery.EOF)
        {
            <%>
            <TR><TD><a href="javascript:changeIntention('<%=
intentionsQuery("user_intention_id") %>','<%=numintentions%>')">
onmouseover="mouseOverTab()" onmouseout="mouseOutOfTab()"><font color="Black" face="arial"
size="-2"><%= intentionsQuery("intention_custom_name") %></font></a></TD><TD><IMG
align="right" SRC="images/delete.gif" alt="Delete this intention"
onClick="confirmDelete(<%= intentionsQuery("user_intention_id") %>)"></TD></TR>
            <%numintentions++; intentionsQuery.moveToNext(); %>
            <%
                }
            Response.Write("<SCRIPT>numintentions="+numintentions
+ "</SCRIPT>");
            <%>
            <tr><td colspan="2"><hr></td></tr>
            <TR><td colspan="2"><a href="javascript:changeIntention('add
...',<%=numintentions%>');" onmouseover="mouseOverTab()"
onmouseout="mouseOutOfTab()"><font color="Black" face="arial" size="-2">add
...</font></a></td></TR>
            </table>
        </body>
    </DIV>
<DIV style="position: absolute; top:0; left:-5; width: 230; height:105; z-index:1;"
onmouseout="intentionlist.style.visibility='hidden'"
onmouseover="intentionlist.style.visibility='hidden'"
onmouseover="intentionlist.style.visibility='hidden'"></DIV>
</DIV>
</DIV>

```

While various embodiments have been described above, 55 it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should not be limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following 60 claims and their equivalents.

What is claimed is:

1. A method for creating a user network interface, comprising the steps of:

- (a) obtaining user profile information;
- (b) retrieving content of a particular application;

(c) filtering the content based on the user profile information;

(d) parsing the filtered content;

(e) retrieving user-centric content;

(f) matching the user-centric content to the parsed content;

(g) presenting the parsed content in a format based on information in the user's profile on a display; and

(h) adding the user-centric content to the parsed content such that the user-centric content demonstrates how the parsed content relates to the user.

49

2. A method for creating a user network interface as recited in claim 1, including the step of searching content for text that matches predefined patterns.

3. A method for creating a user network interface as recited in claim 1, including the step of storing user custom patterns in the user profile to subsidize the predefined patterns.

4. A method for creating a user network interface as recited in claim 1, including the step of matching the pattern utilizing a text comparison to obtain one or more matched content.

5. A method for creating a user network interface as recited in claim 1, including the step of matching the pattern utilizing a database query to obtain one or more matched content.

6. A method for creating a user network interface as recited in claim 1, including the step of comparing the matched content to user application content.

7. A method for creating a user network interface as recited in claim 1, wherein the user applications comprise: calendar, electronic mail, spreadsheet, contact list, word processing, task list, stocks and news.

8. A method for creating a user network interface as recited in claim 1, wherein the patterns comprise at least one comparison indicative of a match.

9. A method for creating a user network interface as recited in claim 1, including the step of reporting information with at least one match to a user.

10. A method for creating a user network interface as recited in claim 1, further comprising selecting one of a plurality of personas of the user, wherein each persona of the user has unique user profile information, wherein the user profile information is obtained based on the selected persona.

11. An apparatus that creates an information summary, comprising;

- (a) a processor;
- (b) a memory that stores information under the control of the processor;
- (c) logic that obtains user profile information;
- (d) logic that retrieves content of a particular application;
- (e) logic that filters the content based on the user profile information;
- (f) logic that parses the filtered content;
- (g) logic that retrieves user-centric content;
- (h) logic that matches the user-centric content to the parsed content;
- (i) logic that presents the parsed content in a format based on information in the user's profile on a display; and
- (j) logic that adds the user-centric content to the parsed content such that the user-centric content demonstrates how the parsed content relates to the user.

12. A computer program embodied on a computer-readable medium that creates an information summary, comprising:

50

(a) a code segment that obtains user profile information;

(b) a code segment that retrieves content of a particular application;

(c) a code segment that filters the content based on the user profile information;

(d) a code segment that parses the filtered content;

(e) a code segment that retrieves user-centric content;

(f) a code segment that matches the user-centric content to the parsed content;

(g) a code segment that presents the parsed content in a format based on information in the user's profile on a display; and

(h) a code segment that adds the user-centric content to the parsed content such that the user-centric content demonstrates how the parsed content relates to the user.

13. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 11, including logic that searches content for text that matches predefined patterns.

14. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, including logic that stores user custom patterns in the user profile to subsidize the predefined patterns.

15. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, including logic that matches a pattern utilizing a text comparison to obtain one or more matched content.

16. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, including logic that matches a pattern utilizing a database query to obtain one or more matched content.

17. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, including logic that compares the matched content to user application content.

18. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, wherein the user applications comprise: calendar, electronic mail, spreadsheet, contact list, word processing, task list, stocks and news.

19. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 12, wherein the patterns comprise at least one comparison indicative of a match.

20. A computer program embodied on a computer-readable medium that creates an information summary as recited in claim 11, including logic that reports information with at least one match to a user.

* * * * *