

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AKAMAI TECHNOLOGIES, INC.,

Petitioner,

v.

EQUIL IP HOLDINGS LLC,

Patent Owner.

Case IPR2023-00329

U.S. Patent No. 6,792,575

**DECLARATION OF VIJAY K. MADISETTI IN SUPPORT OF PETITION
FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 6,792,575**

TABLE OF CONTENTS

	Page
I. INTRODUCTION.....	1
II. QUALIFICATIONS.....	3
III. LEGAL UNDERSTANDING.....	11
A. My Understanding of Claim Construction.....	11
B. My Understanding of Obviousness.....	13
IV. BACKGROUND OF THE TECHNOLOGY	18
V. THE '575 PATENT.....	21
A. '575 Patent Overview	21
B. Prosecution History of the '575 Patent	26
VI. THE '575 PATENT PRIORITY DATE	27
VII. LEVEL OF ORDINARY SKILL IN THE ART	28
VIII. CLAIM CONSTRUCTION	29
A. Claim elements [1.f]-[1.i] and [9.d]-[9.f.4].....	30
B. Preambles	34
IX. GROUNDS OF UNPATENTABILITY	34
A. Ground 1: Lafer in View of Tso Renders Obvious Claim 1	39
1. U.S. 6,192,382 ("Lafer") Overview	39
2. Overview of Tso and Motivation to Modify Lafer with Tso's Teachings.....	45
3. Invalidity of Claim 1 Over Lafer in view of Tso	57
(a) Element [1.pre]	57
(b) Element [1.a].....	61
(c) Element [1.b]	70
(d) Element [1.c].....	75

	(e) Element [1.d]	82
	(f) Element [1.e].....	85
	(g) Element [1.f]	88
	(h) Element [1.g]	99
	(i) Element [1.h]	104
	(j) Element [1.i]	108
	(k) Element [1.j]	111
	(l) Element [1.k]	114
	(m) Element [1.l]	118
B.	Ground 2: Lafer in View of Tso and Cao (Claim 9).....	123
1.	Overview of Cao	123
2.	Motivation to Modify Lafer (in view of Tso) with Cao	125
3.	Invalidity of Claim 9 Over Lafer in View of Tso and Cao	132
	(a) Element [9.pre]	132
	(b) Element [9.a].....	141
	(c) Element [9.b]	146
	(d) Element [9.c].....	152
	(e) Element [9.d]	154
	(f) Element [9.d.1]	159
	(g) Element [9.d.2]	161
	(h) Element [9.e].....	164
	(i) Element [9.e.1].....	167
	(j) Element [9.e.2].....	168
	(k) Element [9.e.3].....	168
	(l) Element [9.e.4].....	169
	(m) Element [9.e.5].....	170
	(n) Element [9.f]	170
	(o) Element [9.f.1]	171
	(p) Element [9.f.2]	171
	(q) Element [9.f.3]	172
	(r) Element [9.f.4]	172
C.	Grounds 3-4: Lafer in view of Tso, (Cao, for Claim 9), and Moore	173

X.	SECONDARY CONSIDERATIONS.....	175
XI.	CONCLUSION	175

I. INTRODUCTION

1. I have been retained by Akamai Technologies, Inc. (I will refer to them as “Akamai”) to provide my opinions on certain issues related to U.S. Patent No. 6,792,575 (the “’575 patent,” which I understand has been designated as Exhibit 1001) in connection with the above-captioned *inter partes* review (IPR) proceeding. In particular, I have been asked to provide my insights, analysis, and opinions regarding whether claims 1 and 9 of the ’575 patent (“Challenged Claims”) are obvious over the prior art references identified below and as to whether persons interested and ordinarily skilled in the subject matter or art exercising reasonable diligence could have located these references.

2. I understand the ’575 patent is titled “Optimization of Media Content Using Generated Intermediate Media Content,” identifies as its named inventors Christopher Samaniego, Nelson Offner, Adrian Thewlis, and David Boyd, and is currently owned by Equil IP Holdings LLC. I have considered the ’575 patent.

3. I understand that the file history of the ’575 patent has been designated as Exhibit 1002. I have considered this file history, and I will refer to it as the “’575 File History” or by its exhibit number.

4. I understand that the ’575 patent was filed as U.S. Patent Application 09/425,326 on October 21, 1999.

5. I have considered the prior art cited in my declaration, including:

1. **U.S. Patent No. 6,192,382** to Lafer (“Lafer”), titled “Method and System for Web Site Construction Using HTML Fragment Caching,” filed September 24, 1997 and issued February 20, 2001. I understand that a copy of this patent has been designated as Exhibit 1004.
2. **U.S. Patent No. 6,421,733** to Tso (“Tso”), titled “System for Dynamically Transcoding Data Transmitted Between Computers,” filed on September 8, 1997, and issued July 16, 2002. I understand that a copy of this publication has been designated as Exhibit 1005.
3. Pei Cao & Chengjie Liu, *Maintaining strong cache consistency in the World Wide Web*, 47 IEEE TRANSACTIONS ON COMPUTS. 445 - 457 (1998) (“**Cao**”), which was published in April 1998 and available through IEEE no later than April 1998, and available from the British Library no later than May 1998. I understand that a copy of this publication has been designated as Exhibit 1006.

6. I am being compensated by Akamai at my standard hourly consulting rate of \$650 for my time on this matter. My compensation is not dependent on the outcome of this proceeding.

7. In forming my opinions, I relied on the documents cited in this declaration and the documents identified in the attached Appendix B. These documents comprise patents, file histories, printed publications, and other related documents. As discussed below, each document is a type that experts in my field would have reasonably relied upon when forming their opinions. Further, experts in my field would have had access to each document either through the applicable patent offices and/or well-known libraries, conferences, or publications in the field. My opinions are also based upon my personal and professional experience.

II. QUALIFICATIONS

8. I received my Bachelor of Technology (Honors) in Electronics and Electrical Communication Engineering at the Indian Institute of Technology (IIT) in Kharagpur, India in 1984. I obtained my Ph.D. in Electrical Engineering and Computer Science at the University of California, Berkeley, in 1989. I received the Demetri Angelakos Outstanding Graduate Student Award from the University of California, Berkeley and the IEEE/ACM Ira M. Kay Memorial Paper Prize in 1989.

9. I am a tenured Full Professor in the Colleges of Computing and Engineering at Georgia Tech. I am knowledgeable and familiar with software

engineering, web applications, cloud computing, data analytics, wireless communications, microprocessor architecture, hardware, RF, cellular networks, ASIC design, computer engineering, embedded systems, digital signal processing, and associated software and firmware design. I have created and taught undergraduate and graduate courses in hardware and software design for computer science and engineering applications at Georgia Tech for the past thirty years. Additionally, I have been active in the areas of wireless communications, digital signal processing, integrated circuit design (analog & digital), software engineering, system-level design methodologies and tools, and software systems. I have been the principal investigator (“PI”) or co-PI in several active research programs in these areas, including DARPA’s Rapid Prototyping of Application Specific Signal Processors, the State of Georgia’s Yamacraw Initiative, the United States Army’s Federated Sensors Laboratory Program, and the United States Air Force Electronics Parts Obsolescence Initiative. I have received an IBM Faculty Award and NSF’s Research Initiation Award.

10. I have designed several specialized computer and communication systems over the past two decades at Georgia Tech for tasks such as wireless audio and video processing and protocol processing for portable platforms, such as cell phones and PDAs. I have worked on designing systems that are efficient from

performance, size, weight, area, and thermal considerations. I have developed courses and classes for the industry on these topics, and many of my lectures in advanced computer system design, developed under the sponsorship of the United States Department of Defense in the late 1990s, are available for educational use at “<http://www.eda.org/rassp>” and have been used by several U.S. and international universities as part of their course work. Some of my recent publications in the area of design of wireless communications systems and associated protocols are listed in Appendix A. I graduated more than 20 Ph.D. students that now work as professors or in technical positions around the world.

11. I have been active in research in the area of the World Wide Web and some of my peer-reviewed publications in this area include (i) Arshdeep Bahga and Vijay K. Madiseti, Performance Evaluation Approach for Multi-Tier Cloud Applications, Journal of Software Engineering and Applications 74-83 (2013); (ii) Arshdeep Bahga and Vijay K. Madiseti, Synthetic Workload Generation for Cloud Computing Applications, Journal of Software Engineering and Applications 396-410 (2011); (iii) Venu Dasigi and Vijay Madiseti, A Web-Based Interface for a Digital Broadband Home, Yamacraw IAB Workshop (2000); and (iv) Arshdeep Bahga and Vijay K. Madiseti, Rapid Prototyping of Multitier Cloud-Based Services and Systems, Computer, vol. 46, no. 11, pp. 76-83 (Nov. 2013).

12. I have been an active consultant to industry and various research laboratories (including Massachusetts Institute of Technology Lincoln Labs and Johns Hopkins University Applied Physics Laboratory). My consulting work for MIT Lincoln Labs involved high resolution imaging for defense applications, where I worked in the area of prototyping complex and specialized computing systems. My consulting work for the Johns Hopkins Applied Physics Lab (“APL”) mainly involved localization of objects in image fields, where I worked on identifying targets in video and other sensor fields and identifying computer architectures and circuits for power and space-efficient designs.

13. I have founded three companies in the areas of embedded software, military chipsets involving imaging technology, and wireless communications. I have supervised the Ph.D. dissertations of over twenty engineers in the areas of computer engineering, signal processing, communications, rapid prototyping, and system-level design methodology, of which five have resulted in thesis prizes or paper awards. The first of the companies I founded, VP Technologies, offers products in the area of semiconductor integrated circuits, including building computing systems for imaging systems for avionics electronics for the United States Air Force and the US Navy, since 1995. I remain a director of VP Technologies. The second of these companies, Soft Networks, LLC, offers software

for multimedia and wireless computing platforms, including the development of a set-top box for Intel that decodes MPEG-2 video streams, wireless protocol stacks, and imaging codecs for multimedia phones. The technology involved with the design, development, and implementation of the Intel set-top box included parsing the bit streams, decoding communications protocols, extracting image and video data, and then processing for subsequent display or storage. The third of these companies, Elastic Video, uses region of interest based video encoding or decoding for capturing high quality video at very low bit rates, with primary application for wireless video systems.

14. I have authored more than sixty refereed journal publications and around forty peer reviewed conference publications. I have been active in research in the area of wireless and mobile communications and some of my recent peer-reviewed publications in this area include: (i) Mustafa Turkboylari & Vijay K. Madisetti, Effect of Handoff Delay on the System Performance of TDMA Cellular Systems, Proceedings of the Fourth IEEE Conference on Mobile and Wireless Communications Network 411-15 (Sept. 9-11, 2002); (ii) Loran A. Jatunov & Vijay K. Madisetti, Computationally-Efficient SNR Estimation for Bandlimited Wideband CDMA Systems, 5 IEEE Transactions on Wireless Communications, no. 12 (2006) at 3480-91; and (iii) Nimish Radio, Ying Zhang, Mallik Tatipamula & Vijay K.

Madisetti, Next Generation Applications on Cellular Networks: Trends, Challenges, and Solutions, 100 Proceedings of the IEEE, no. 4 (April 2012) at 841-54. I have extensive experience analyzing, designing, and testing systems based on 3GPP Technical Specifications, including specifications describing WCDMA and HSDPA technologies. I have been active in the area of location-based services and wireless localization techniques since the mid-1990s, and have authored several papers on location-based services, including, Vijay K. Madisetti et al., Mobile Fleet Application Using SOAP and System on Devices (SyD) Middleware Technologies, Communications, Internet, and Information Technology (2002) at 426-31. I have served as associate editor or on the editorial board for technical journals, including IEEE Transactions on Circuits & Systems II, International Journal in Computer Simulation, and International Journal in VLSI Signal Processing.

15. I have authored or co-authored several books, including VLSI Digital Signal Processors (IEEE Press 1995) and the Digital Signal Processing Handbook (CRC Press, 1998, 2010). I co-authored Quick-Turnaround ASIC Design in VHDL (Kluwer Academic Press 1996) and Platform-Centric Approach to System-on-Chip (SoC) Design (Springer 2004). I am also the editor of several books, including the three-volume DSP Handbook set: Volume 1: Digital Signal Processing Fundamentals, Volume 2: Video, Speech, and Audio Signal Processing and

Associated Standards, and Volume 3: Wireless, Networking, Radar, Sensory Array Processing, and Nonlinear Signal Processing, published in 2010 by CRC Press, Boca Raton, Florida. More recently I have authored Cloud Computing (2014, CreateSpace Press), and Internet of Things (2014, CreateSpace), and the book, Cloud Computing, was nominated as a Notable Book of 2014 by the Association of Computing Machinery (ACM) in July 2015.

16. My experience is relevant to this case. In addition to authoring textbooks and research papers in the area of cloud computing, virtualization, web-services, and web-based applications, I have developed web-based document management systems and also web-based fintech applications. I am listed as an inventor on over a dozen patents on web-based applications. I also invented a new protocol that replaces the HTTP protocol for token-based networks, called VTTP, that is a subject of a few issued US patents.

17. I have been extensively involved in the activities of one of the premier standard setting organizations in the world, the IEEE, since the 1980s, and I have participated in the development of standards for hardware design and description languages, such as VHDL, used in design of computer chips – IEEE 1076.6. I was also a member of the IEEE Press Board from 1995-1997. I was the Technical Program Chair of IEEE ICASSP in 1996. I was also the Associate Editor of IEEE

Transactions on Circuits & Systems II from 1994 to 1997. I am also the director of the IEEE Atlanta Chapter, and the author of several drafts of IETF proposals.

18. I have served as associate editor or on the editorial board for technical journals, including IEEE Transactions on Circuits & Systems II, International Journal in Computer Simulation, and International Journal in VLSI Signal Processing. I have been elected a Fellow of the IEEE, for contributions to embedded computing systems. The Fellow is the highest grade of membership of the IEEE, a world professional body consisting of over 300,000 electrical and electronics engineers, with only one tenth of one percent (0.1%) of the IEEE membership being elected to the Fellow grade each year. Election to Fellow is based upon votes cast by existing Fellows in IEEE.

19. I have also been awarded the 2006 Frederick Emmons Terman Medal by the American Society of Engineering Education for contributions to Electrical Engineering, including authoring a widely used textbook in the design of VLSI digital signal processors. I was awarded VHDL International Best PhD Dissertation Advisor Award in 1997 and the NSF RI Award in 1990. I was Technical Program Chair for both the IEEE MASCOTS in 1994 and the IEEE Workshop on Parallel and Distributed Simulation in 1990. In 1989 I was recognized with the Ira Kay

IEEE/ACM Best Paper Award for Best Paper presented at the IEEE Annual Simulation Symposium.

20. I have submitted approximately 100 invention disclosures and provisional patents over the past ten years. I am listed as a first inventor on over thirty issued US patents.

21. I have testified as an expert witness before. Over the past two years, I've testified as an expert in more than 20 proceedings, and several of these were in the area of enterprise, cloud and web-based software applications.

22. I have attached a more detailed list of my qualifications as **Appendix A**.

III. LEGAL UNDERSTANDING

A. My Understanding of Claim Construction

23. I have been informed that patent claims are construed from the viewpoint of a person having ordinary skill in the art of the patent at the time of the invention. I have been informed that patent claims generally should be interpreted consistent with their plain and ordinary meaning as understood by a person having ordinary skill in the art in the relevant time period (*i.e.*, at the time of the purported invention, or the so called “effective filing date” of the patent application), after reviewing the patent claim language, the specification, and the prosecution history (*i.e.*, the intrinsic record).

24. I have further been informed that a person having ordinary skill in the art must read the claim terms in the context of the claim itself, as well as in the context of the entire patent specification. I understand that in the specification and prosecution history, the patentee may specifically define a claim term in a way that differs from the plain and ordinary meaning. I understand that the prosecution history of the patent is a record of the proceedings before the U.S. Patent and Trademark Office, and may contain explicit representations or definitions made during prosecution that affect the scope of the patent claims. I understand that an applicant may, during the course of prosecuting the patent application, limit the scope of the claims to overcome prior art or to overcome an examiner's rejection, by clearly and unambiguously arguing to overcome or distinguish a prior art reference, or clearly and unambiguously disavowing claim coverage.

25. In interpreting the meaning of the claim language, I understand that a person having ordinary skill in the art may also consider "extrinsic" evidence, including expert testimony, inventor testimony, dictionaries, technical treatises, other patents, and scholarly publications. I understand this evidence is considered to ensure that a claim is construed in a way that is consistent with the understanding of those of ordinary skill in the art at the time of the claimed invention. For example, this can be useful for a technical term whose meaning may differ from its ordinary

English meaning. I understand that extrinsic evidence may not be relied on if it contradicts or varies the meaning of claim language provided by the intrinsic evidence, particularly if the applicant has explicitly defined a term in the intrinsic record.

B. My Understanding of Obviousness

26. I understand that a claim may be invalid if the subject matter described by the claim as a whole would have been obvious to a hypothetical person of ordinary skill in the art in view of a prior art reference or in view of a combination of references at the time the claimed invention was made. Therefore, I understand that obviousness is determined from the perspective of a hypothetical person of ordinary skill in the art and that the asserted claims of the patent should be read from the point of view of such a person at the time the claimed invention was made. I further understand that a hypothetical person of ordinary skill in the art is assumed to know and to have all relevant prior art in the field of endeavor covered by the patent in suit.

27. I have been informed that there are two criteria for determining whether prior art is analogous and thus can be considered prior art: (1) whether the art is from the same field of endeavor, regardless of the problem addressed, and (2) if the reference is not within the field of the patentee's endeavor, whether the reference still is reasonably pertinent to the particular problem with which the patentee is

involved. I have also been informed that the field of endeavor of a patent is not limited to the specific point of novelty, the narrowest possible conception of the field, or the particular focus within a given field. I have also been informed that a reference is reasonably pertinent if, even though it may be in a different field from that of the patentee's endeavor, it is one which, because of the matter with which it deals, logically would have commended itself to a patentee's attention in considering their problem.

28. I have also been advised that an analysis of whether a claimed invention would have been obvious should be considered in light of the scope and content of the prior art, the differences (if any) between the prior art and the claimed invention, and the level of ordinary skill in the pertinent art involved. I understand as well that a prior art reference should be viewed as a whole.

29. I have also been advised that in considering whether a claimed invention could be obvious over a combination of prior art references, I may assess whether there are apparent reasons to combine known elements in the prior art in the manner claimed in view of interrelated teachings of multiple prior art references, the effects of demands known to the design community or present in the marketplace, and/or the background knowledge possessed by a person having ordinary skill in the art. I understand that other principles may be relied on in evaluating whether a

claimed invention would have been obvious, and that these principles include the following:

- A combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results;
- When a device or technology is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or in a different one, so that if a person having ordinary skill can implement a predictable variation, the variation is likely obvious;
- If a technique has been used to improve one device, and a person having ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond their skill;
- An explicit or implicit teaching, suggestion, or motivation to combine two prior art references to form the claimed combination may demonstrate obviousness, but proof of

obviousness does not depend on or require showing a teaching, suggestion, or motivation to combine;

- Market demand, rather than scientific literature, can drive design trends and may show obviousness;
- In determining whether the subject matter of a patent claim would have been obvious, neither the particular motivation nor the avowed purpose of the named inventor controls whether the claim is obvious;
- One of the ways in which a patent's subject can be proved obvious is by noting that there existed at the time of invention a known problem for which there was an obvious solution encompassed by the patent's claims;
- Any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed;
- "Common sense" teaches that familiar items may have obvious uses beyond their primary purposes, and in many cases a person having ordinary skill will be able to fit the teachings of multiple patents together like pieces of a puzzle;

- A person having ordinary skill in the art is also a person having ordinary creativity, and is not an automaton;
- A patent claim can be proved obvious by showing that the claimed combination of elements was “obvious to try,” particularly when there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions such that a person having ordinary skill in the art would have had good reason to pursue the known options within his or her technical grasp; and
- One should be cautious of using hindsight in evaluating whether a claimed invention would have been obvious.

30. I further understand that, in making a determination as to whether the claimed invention would have been obvious to a person having ordinary skill, the Board may consider certain objective factors if they are present, such as: commercial success of products practicing the claimed invention; long-felt but unsolved need; teaching away; unexpected results; copying; and praise by others in the field. These factors are generally referred to as “secondary considerations” or “objective indicia” of nonobviousness. I understand, however, that for such objective evidence to be relevant to the obviousness of a claim, there must be a causal relationship (called a

“nexus”) between the claim and the evidence and that this nexus must be based on a novel element of the claim rather than something in the prior art. I also understand that even when they are present, secondary considerations may be unable to overcome primary evidence of obviousness (such as motivation to combine with predictable results) that is sufficiently strong.

31. I have been asked to consider the validity of claims 1 and 9 of the '575 patent, which I refer to as the “Challenged Claims.” I understand that for *inter partes* reviews, invalidity must be shown under a preponderance of the evidence standard. I have concluded that each of Challenged Claims is rendered obvious by the prior art based on the references described below and as explained herein.

IV. BACKGROUND OF THE TECHNOLOGY

32. As discussed in Section V, the purported invention of the '575 patent is a process for delivering media using proprietary media tags. Specifically, the Challenged Claims recite “actuating a user request” for content by parsing an HTML document for embedded “proprietary media tags.” If such a proprietary media tag is identified in the HTML document, a media tags database is searched for a corresponding standard HTML tag with which the tag is replaced in the HTML document. Once one or more such proprietary media tags have been replaced with a standard HTML tag in the HTML document and the end of the file has been reached, the modified HTML document with standard HTML tags in place of the proprietary

media tags is delivered to a customer's browser. If a proprietary media tag is not found in the media tag database, the system generates and caches Web media using the proprietary media tag that is embedded in the HTML document. The proprietary media tag and a standard HTML tag referring to the cached version of the newly generated media are then stored in the media tag database.

33. Based on my experience in the field, however, the functionalities that are claimed in the '575 patent were well-known in the art prior to the '575 patent's priority date. One example of such prior art is U.S. Patent No. 6,192,382, which was originally assigned to MediaOne Group, Inc., and identifies Lafer as its first named inventor. I understand that a copy of Lafer has been designated as Exhibit 1004. Lafer discloses that a system responds to a request for a page by parsing a "tag-extended HTML file" with embedded tags that "do not constitute a part of the standard HTML tag library." Lafer, 2:65-3:7, 3:52-55, 3:65-67, 4:8-15. The tag includes "substitution tokens" specifying the "content data" associated with that tag, as well as "user session information" that is used for "dynamic substitution" of content in the HTML page. Lafer, 4:24-31, 3:15-21. This is done by the system, which looks up the tags in a "tag cache," and replaces the tags in the file with "HTML fragments," which are linked to underlying data stored in a Content Database ("Content DB"). Lafer, 4:32-35, 3:32-43, Figure 1. As was understood in the art,

such HTML fragments are chunks of HTML code—for example, an HTML tag to embed content, such as an image, in a webpage. If the tag does not exist in the cache, then the system uses the tag as embedded in the tag-extended HTML file to create a tag-content pair, which links the specialized tag to an HTML fragment and the corresponding content stored in the Content DB. Lafer, 4:46-54, 3:22-36. This tag-content pair is then cached. *Id.*

34. Another relevant prior art reference is U.S. Patent No. 6,421,733, which was originally assigned to Intel, and which identifies Tso as the first named inventor. I understand that a copy of Tso has been designated as Exhibit 1005. Tso further discloses that, when requested media content is not already available in a cache, the system retrieves the original content, transcodes it based on “selection criteria” contained within the request. Tso, 6:64-7:62. The selection criteria information specifies the operations to further transform the content, including “the degree of alteration” of the content that is required. *Id.* After transcoding, the system caches the transcoded content and then returns the transcoded content to the user. Tso, 4:1-5, 6:28-50, 6:64-8:13. As explained in greater detail below in Section IX.A.2, a person having ordinary skill in the art would have been motivated to modify Lafer with Tso’s teachings to advantageously use the substitution tokens to carry selection criteria used to transcode content. Applying Tso’s teachings to Lafer yields a system

in which the substitution tokens include all necessary information used to transcode, generate, and cache the content. Lafer, 4:24-31; Tso, 4:1-5, 6:28-50, 6:64-8:13. None of the features recited in the Challenged Claims of the '575 patent was novel at the time of the alleged invention, and there was nothing novel as to the manner in which the '575 patent combined those features. As I discuss below, the Challenged Claims are obvious over the prior art. Further, there were no technological barriers to combining these elements to form the claimed invention and a person having ordinary skill in the art would have been motivated to combine these elements and would have understood that the combination would have yielded predictable results.

V. THE '575 PATENT

A. '575 Patent Overview

35. The Challenged Claims of the '575 patent are method claims that generally recite methods in which a system responds to a user request by reading a proprietary tag in an HTML file, processing media content according to the tag, and replacing the tag with a standard HTML tag referencing the processed version of content to be returned to the user. '575 patent, 1:9-11. Aspects of an embodiment of a system in which the claimed methods in the Challenged Claims are implemented are shown in the annotated version of Figure 6 below:

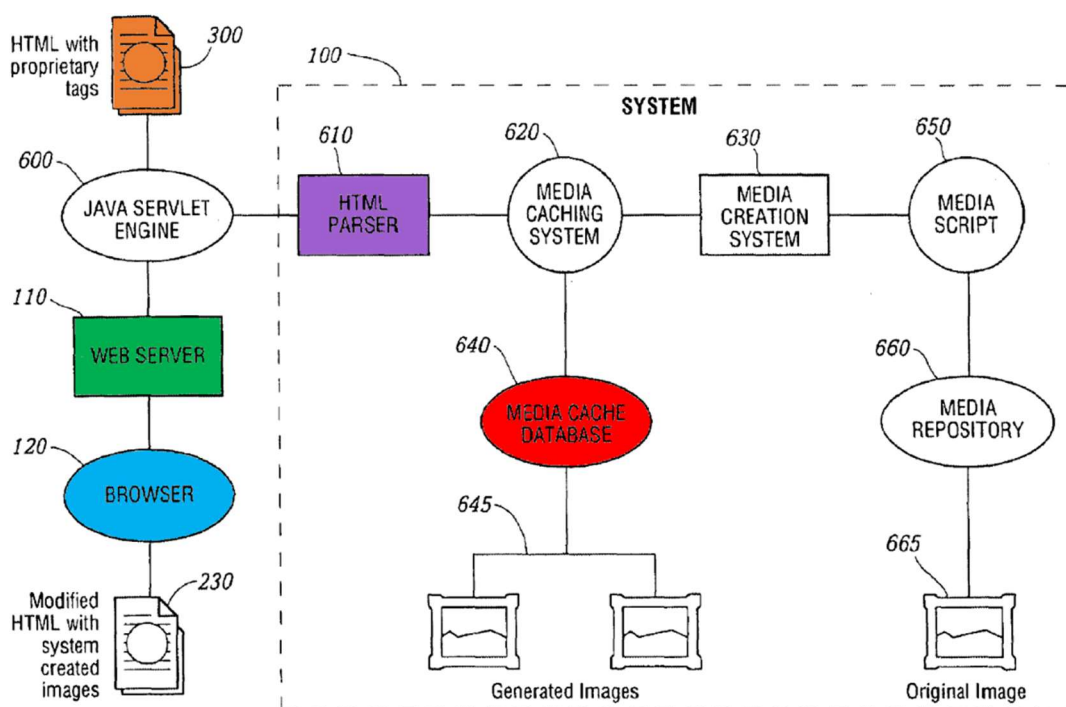


FIG. 6

'575, Figure 6. As shown, the system includes a **web server 110** that communicates with a user's **browser 120**. '575 patent, Figure 6, 7:28-29. The server stores **HTML with proprietary tags 300** for content requested by a user, and is connected to a system that includes an **HTML parser 610**, and a **media cache database 640**. '575 patent, Figure 6, 8:18-20, 7:21-26, 8:21-24. The database stores both proprietary tags and generated web media. '575 patent, Figure 6, 7:47-57.

36. Figure 9, annotated below, summarizes how the system handles user requests for cached content:

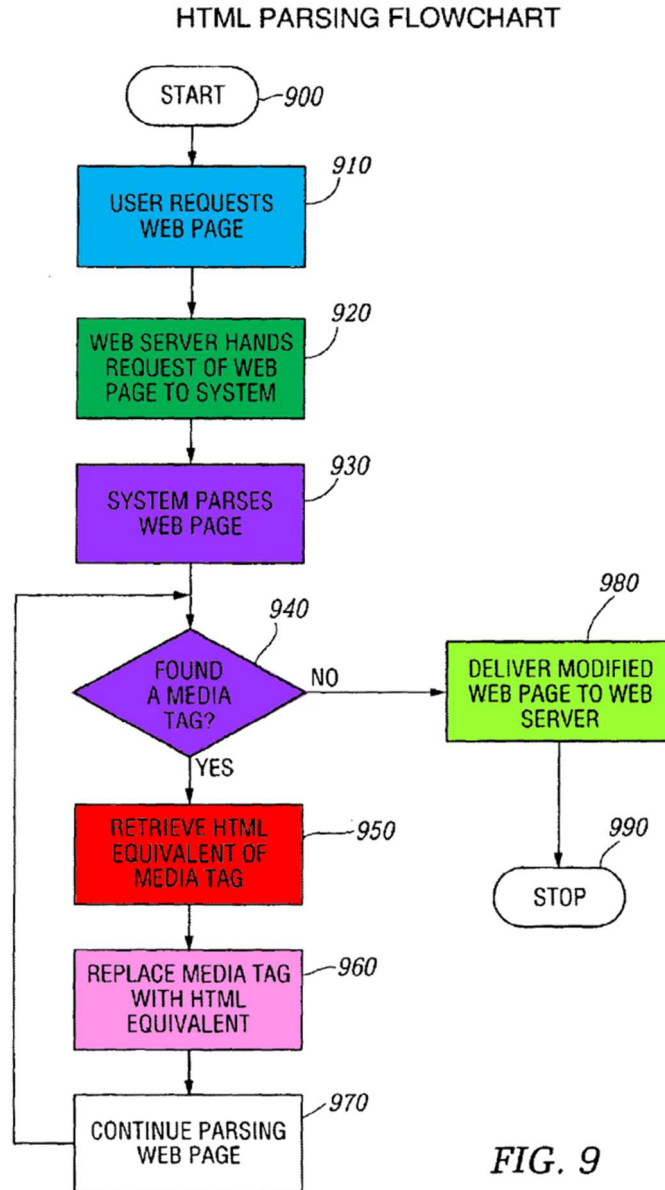
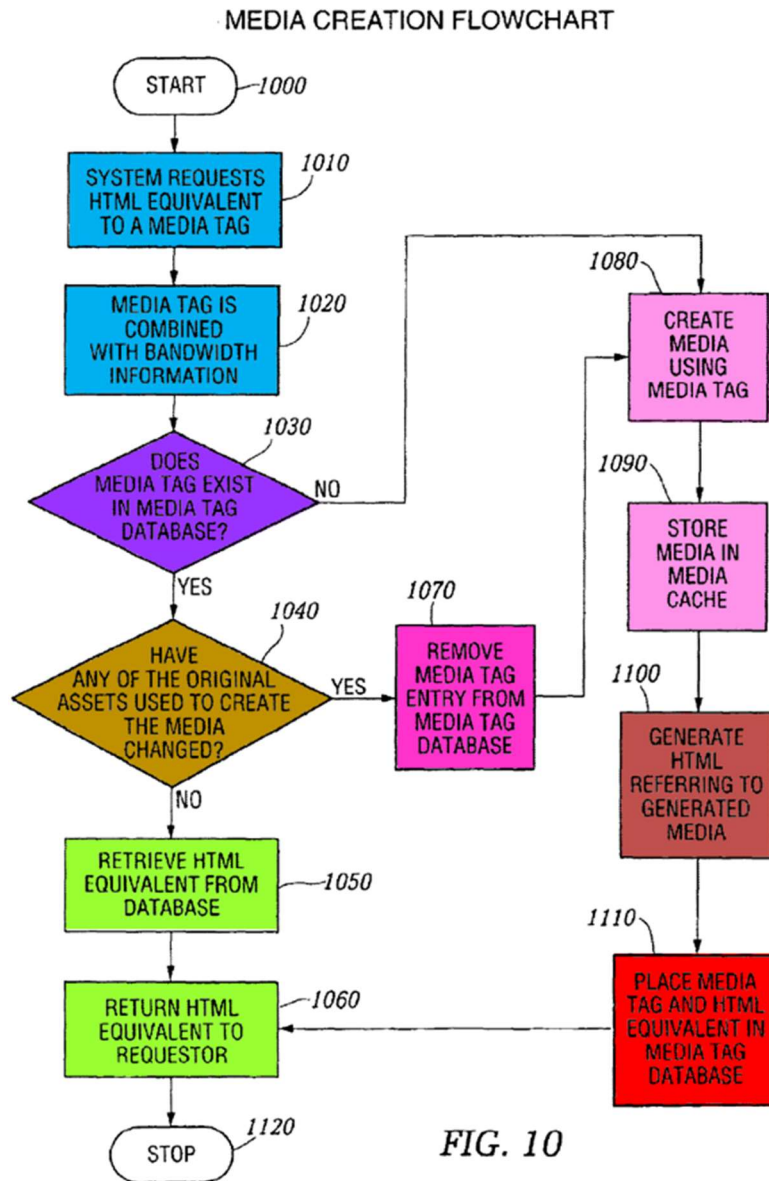


FIG. 9

'575, Figure 9; When a **user requests a web page (step 910)**, the **server passes the request to the system (step 920)**. '575 patent, 8:48-49. The system then **parses the requested web page for media tags (steps 930 and 940)**. '575 patent, 8:49-50. When a media tag is found, the system **retrieves an HTML equivalent of the tag from a cache database (steps 950 and 960)**, and **replaces the tag with an HTML**

equivalent tag (step 960). '575 patent, 8:49-54. The process repeats until the system finishes parsing the page, at which point the system **delivers the page back to the server (step 980)**, which then returns the page to the user. '575 patent, 8:54-56.

37. The steps involved in determining whether a media tag is cached, and how to handle the situation in which a media tag is not found in cache are set forth in annotated Figure 10, below:



'575, Figure 10. First, the system **requests a media tag** (step 1010), which is **combined with bandwidth information**, from the media tag database (step 1020). '575 patent, 8:57-61. The database is checked to **determine whether the requested media tag exists** (step 1030). '575 patent, 8:61-63. If yes, the system **checks if the original asset is unchanged** (step 1040) and, if so, **retrieves and**

returns the HTML equivalent tag (steps 1050, 1060). '575 patent, 8:63-67. If the original asset changed, the system updates the database by removing the old media tag entry (step 1070), using the media tag to create and store a new media in cache (steps 1080, 1090), generating HTML referring to the generated media (step 1100), and storing the media tag and HTML equivalent in a media tag database (step 1110) before returning the generated HTML back to the system (step 1060). See '575 patent, 8:61-9:9, Figure 10.

38. When checking if a media tag exists in a database at step 1030, if the tag does not exist, the system uses the same steps, 1080-1110, to create and store the generated media, the media tag, and the HTML equivalent tag. *Id.*

B. Prosecution History of the '575 Patent

39. During prosecution, the claim that issued as Challenged Claim 1 was numbered as draft claim 35, while draft claim 45 issued as Challenged Claim 9. During prosecution, the Examiner issued a restriction requirement that required the applicant to make an election of claims, in response to which the Applicant elected a set of claims including draft claims 35 and 45. Ex. 1002, 80-83, 86-101. The Examiner then allowed draft claim 35, and indicated that draft claim 45 “would be allowable if rewritten or amended to overcome the rejection(s) under 35 U.S.C. 112, second paragraph” by adding the word “media” in the claim elements reciting “to create said media has changed,” and “to create said media has not changed.” Ex.

1002, 102-15. The Challenged Claims were then allowed without further rejections. Ex. 1002, 139-41.

40. Prior to allowance, the Examiner did reject other claims based on prior art. I compared the claims that were rejected with the allowed claims. Based on my comparison, the rejected claims did not recite certain claim elements that appear in claims 1 and 9 including, for example, elements relating to “actuating a user request” or “receiving a request” (which are recited in elements [1.c] and [9.a], respectively), or “removing said media tag entry from said database” (which is recited in element [9.g]). *See* Ex. 1002, 102-15. The applicant responded to those prior art rejections by canceling the rejected claims. Ex. 1002, 124-38.

VI. THE '575 PATENT PRIORITY DATE

41. I understand the '575 patent was filed on October 21, 1999. I have not formulated any opinion about the correct priority date for the '575 patent. For purposes of this declaration, I was asked to assume, with respect to each of the prior art Grounds discussed in this declaration, that the '575 patent is entitled to the priority date of October 21, 1999. I have been instructed to consider the patent and the prior art from the perspective of a person having ordinary skill in the art as of October 21, 1999.

VII. LEVEL OF ORDINARY SKILL IN THE ART

42. In rendering the opinions set forth in this declaration, I was asked to consider the patent claims and the prior art through the eyes of a person having ordinary skill in the art at the time of the alleged invention. I understand that the factors considered in determining the ordinary level of skill in a field of art include the level of education and experience of persons working in the field; the types of problems encountered in the field, the teachings of the prior art, and the sophistication of the technology at the time of the alleged invention. I understand that a person having ordinary skill in the art is not a specific real individual, but rather is a hypothetical individual having the qualities reflected by the factors above. I understand that a person having ordinary skill in the art would also have knowledge from the teachings of the prior art, including the art cited below.

43. I have been asked to assume that the '575 patent is entitled to the priority date of October 21, 1999. Upon consideration of these factors, in my opinion, on or before this priority date, a person having ordinary skill in the art relating to the technology of the '575 patent would have had a minimum of either a bachelor's degree in computer systems, computer science, or the equivalent thereof, and at least two years of experience with networked media delivery or related technologies. Additional graduate education could substitute for professional

experience, or significant experience in the field could substitute for formal education.

44. Well before October 21, 1999, my level of skill in the art was at least that of a person having ordinary skill. I am qualified to opine concerning what a person having ordinary skill in the art would have known and understood at the time of the '575 patent's purported invention. My analysis and conclusions herein are from the perspective of a person having ordinary skill in the art as of October 21, 1999. Even if that priority date were earlier, however, my opinions would not change as long as the identified prior art were still prior art.

VIII. CLAIM CONSTRUCTION

45. I have been instructed that the Challenged Claims 1 and 9 of the '575 patent are to be given their ordinary and customary meaning as understood by one of ordinary skill in the art at the time of the alleged invention in light of the specification and the prosecution history pertaining to the patent. I have followed these principles in my analysis set forth in this document.

46. I understand that there is a pending district court litigation involving the '575 patent, and the parties have not engaged in claim construction in that litigation. I reserve the right to supplement my opinions in this declaration to the extent that the parties later propose claim constructions for any claim terms in the '575 patent.

A. Claim elements [1.f]-[1.i] and [9.d]-[9.f.4]

47. Claim elements [1.f]-[1.i] read as follows, with key language emphasized and indentation reflecting the indentation in the claims as set forth in the '575 patent:

[1.f] **if** at least one media tag of said proprietary media tags is **not found, generating** a Web media by using said at least one media tag;

[1.g] **placing** said generated Web media in a media cache on said network system;

[1.h] **converting** said at least one media tag to at least one standard HTML equivalent tag that refers to said media in said cache; **and**

[1.i] **storing** said at least one media tag and said at least one standard HTML equivalent tag in said database;

48. Claim elements [9.d] to [9.f.4] read as follows, with key language emphasized and indentation reflecting the indentation in the claims as set forth in the '575 patent:

[9.d] **if said media tag entry exists** in said media database,

determining if at least one original asset used to create said media **has changed**;

[9.d.1] **if said at least one original asset** used to create said media **has not changed**,

retrieving said HTML equivalent tag from said database; **and**

[9.d.2] **return** said HTML equivalent tag to said system;

[9.e] **if said at least one original asset** used to create said media **has changed**,

removing said media tag entry from said database;

[9.e.1] **create** said media using said media tag;

[9.e.2] **storing** said media in a media cache;

[9.e.3] **generating** an HTML document referring to said generated media;

[9.e.4] **placing** said media tag and said HTML equivalent tag in said media database; **and**

[9.e.5] **return** said HTML equivalent tag to said system;

[9.f] **if said media tag entry does not exist** in said media database,

create said media using said media tag;

[9.f.1] **storing** said media in a media cache;

[9.f.2] **generating** HTML or browser language referring to said generated media;

[9.f.3] **placing** said media tag and said HTML equivalent tag in said media database; **and**

[9.f.4] **return** said HTML equivalent tag to said system.

49. I have been informed that a method claim that includes a conditional step can be thought of as covering two methods—one method where the condition for the step is met and the step is performed, and the second method where the condition is not met and the step is not performed. In other words, such a claim encompasses the method where only the non-conditional steps are performed (i.e., this occurs when the condition is not met). I have further been informed that conditional steps may be mutually exclusive and cover the possibilities for the condition. I understand that when a limitation recites a mutually exclusive condition, the language encompasses a method where one of the conditional steps is performed and the other conditional step is not performed.

50. I have been informed that where a claim sets forth a plurality of elements or steps, each element or step of the claim should be separated by a line indentation. Within a claim, there may be plural indentations to further segregate subcombinations or related steps.

51. For purposes of this declaration, I have been instructed to assume that elements [1.f]-[1.i] are conditional limitations. Specifically, claim 1 recites that “if at least one media tag ... is not found,” then the “placing,” “converting ... and” “storing” steps of [1.g]-[1.i] are performed. But if the media tag is found, those steps need not be performed. Accordingly, claim 1 may be met without showing that

elements [1.f]-[1.i] are met because in the circumstance in which a media tag *is* found, those elements need not be performed. I have considered this construction in formulating my opinions below, and Claim 1 is rendered obvious under this construction. In addition, Claim 1 is rendered obvious even if every step of the claimed method is required as shown in Sections IX.A.3 and IX.C.

52. For purposes of this declaration, I have also been instructed to assume that elements [9.d]-[9.f.4] are mutually exclusive conditional limitations. Specifically, I have been instructed to assume that claim 9 covers methods in which (1) said “media tag entry exists” and the “original asset” “has not changed” (*see* elements [9.d]-[9.d.2]); (2) the entry exists and the “original asset” “has changed” (*see* elements [9.d], [9.e]-[9.e.5]); and (3) the entry “does not exist” (*see* elements [9.f]-[9.f.4]). I have been instructed to assume that these are mutually exclusive conditional steps, and meeting the claim requires only one of these sets of steps to be performed. I have considered this construction in formulating my opinions below, and Claim 9 is rendered obvious under this construction. In addition, Claim 9 is rendered obvious even if every step of the claimed method is required as shown in Sections IX.B.3 and IX.C.

53. Although I have been instructed to assume that the above-listed claim elements as conditional limitations, I have also been instructed to consider whether

the claims would be rendered obvious to the extent that these claims are not construed to recite conditional limitations. As set forth below in Section IX, under such an interpretation, the claims are still rendered obvious because the prior art not only renders obvious all of the recited claim elements, including each of the conditional limitations, but affirmatively discloses them.

B. Preambles

54. I understand the preamble of a claim may or may not be limiting. For purposes of this declaration, I have not been asked to formulate any opinion as to whether any claim preamble is limiting, and have not formulated any such opinion. Instead, I have been instructed to consider whether the prior art discloses the preambles, to the extent they are limiting. To the extent they are limiting, they are disclosed by the prior art as discussed in Section IX.

IX. GROUNDS OF UNPATENTABILITY

55. As I explained above in Section IV, the methods for delivering media using proprietary media tags claimed in the Challenged Claims were well-known in the art.

56. In this section, I will explain why combinations of teachings from the following references render obvious each of the Challenged Claims:

Name	US Patent/ Title	Exhibit	Filed	Published/ Issued
Lafer	U.S. Patent 6,192,382	1004	9/24/1997	2/20/2001
Tso	U.S. Patent 6,421,733	1005	9/8/1997	7/16/2002
Cao	Maintaining Strong Cache Consistency in the World Wide Web	1006		4/1998
Moore	U.S. Patent 6,310,601	1007	5/12/1998	10/30/2001

I have been instructed to assume that each of these references qualifies as prior art.

57. I have been asked to consider whether Ex. 1006, Pei Cao & Chengjie Liu, *Maintaining strong cache consistency in the World Wide Web*, 47 IEEE TRANSACTIONS ON COMPUTS. 445 - 457 (1998) (“Cao”) would have been publicly accessible to persons having ordinary skill in the art and interested researchers. As set forth above in Section II, I am familiar with publications that are generated by IEEE, including the IEEE Transactions on Computers. Before October 1999, IEEE Transactions on Computers was a well-known, monthly publication that, as stated by the IEEE, has a “wide distribution to researchers, developers, technical managers, and educators in the computer field.” *See* Exs. 1022-1024,

<https://ieeexplore.ieee.org/xpl/aboutJournal.jsp?punumber=12> (separate exhibits in which the “Aims & Scope,” “Publication Details,” and “Subjects” tabs on the page are expanded). Well before October 1999 and April 1998, the journal was known to involve topics relevant to the field of the ’575 patent, including for example “computer organizations and architectures,” “software systems, and communication protocols,” “real-time systems,” “digital devices,” “interconnection networks,” and “new and important applications and trends” in the field. *See id.*; *see also* Ex. 1025,

<https://ieeexplore.ieee.org/xpl/issues?punumber=12&isnumber=9984045> (index of all volumes and years for the journal, showing the journal has been published since the 1960s); Ex. 1026,

<https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=14873&punumber=12> (index of articles in the issue in which Cao appears). IEEE publications, including IEEE Transactions on Computers, were indexed and easily searchable at that time such that a person having ordinary skill in the art and/or an interested researcher would have been readily capable of locating Cao either through IEEE directly, personal subscriptions, or through various libraries. Indeed, Cao has been cited by numerous patents and publications with priority dates from 1999 or earlier. *See* Ex. 1027,

<https://ieeexplore.ieee.org/document/675713/citations?tabFilter=papers#citations;>

Ex. 1028,

<https://ieeexplore.ieee.org/document/675713/citations?tabFilter=patents#citations;>

(listing patents and publications that cited to Cao, including, *e.g.*, Ex. 1029, Maarten van Steen, Andrew S Tanenbaum, Ihor Kuz, Henk J Sips, “A scalable middleware solution for advanced wide-area Web services”, Distributed Systems Engineering, vol.6, no.1, pp.34, 1999; Ex. 1030, J.B. Lim, A.R. Hurson, “Data duplication and consistency in a mobile, multidatabase environment”, Proceedings 1998 International Conference on Parallel and Distributed Systems (Cat. No.98TB100250), pp.50-58, 1998; Ex. 1031, Xiangping Chen, P. Mohapatra, “Lifetime behavior and its impact on Web caching”, Proceedings 1999 IEEE Workshop on Internet Applications (Cat. No.PR00197), pp.54-61, 1999; Ex. 1032, K.Y. Wong, K.H. Yeung, “Site-based approach in HTTP proxy design”, Proceedings of the 1999 ICPP Workshops on Collaboration and Mobile Computing (CMC’99). Group Communications (IWGC). Internet ’99 (IWI’99). Industrial Applications on Network Computing (INDAP). Multime, pp.228-233, 1999; Ex. 1033, U.S. Patent No. 6,088,721 (filed 10/20/1998); Ex. 1034, Patent No. WO2000062185 (with US priority date of 9/4/1999). My opinion is further confirmed by the letter from Mr. Corsini stating that a copy of Cao was publicly

available as of May 1, 1998 out of the British Library and the accompanying hard copy of Cao (Exs. 1020, 1021), as well as the declaration of Mr. MacPherson, attesting that IEEE made a copy of Cao publicly available no later than the end of April 1998 (Ex. 1008).

58. As I explain below, the Challenged Claims are invalid based upon the following prior art grounds:

- a. **Ground 1:** Challenged Claim 1 of the '575 patent is rendered obvious by Lafer in view of Tso;
- b. **Ground 2:** Challenged Claim 9 is rendered obvious by Lafer in view of Tso and Cao (Ex. 1006);
- c. **Ground 3:** Challenged Claim 1 is rendered obvious by Lafer in view of Tso and Moore (Ex. 1007).
- d. **Ground 4:** Challenged Claim 9 is rendered obvious by Lafer in view of Tso, Cao and Moore.

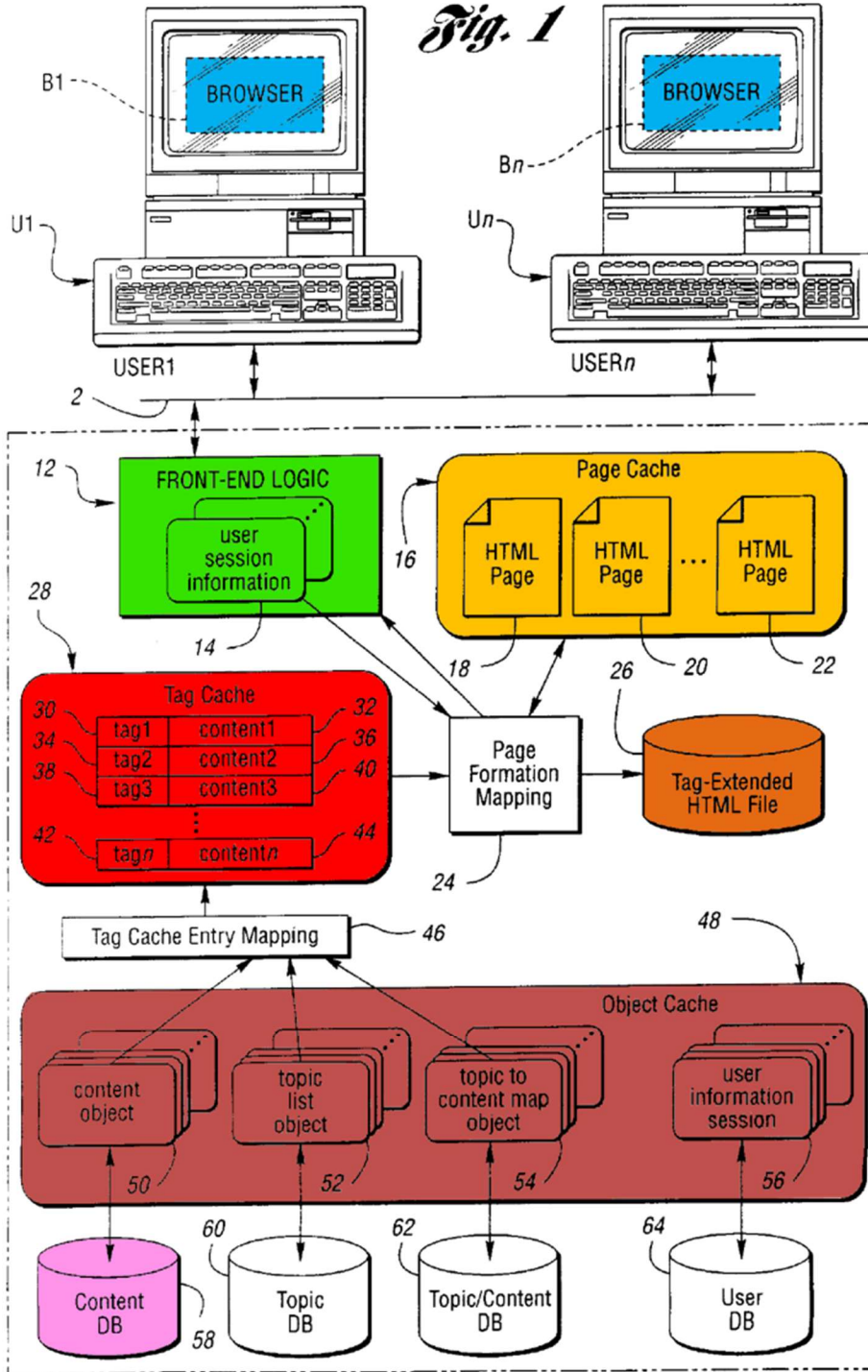
59. It is my opinion that Challenged Claims 1 and 9 of the '575 patent are unpatentable as explained below. I note that in this discussion, I may focus on the particular language in specific claims, but it should be understood that my analysis applies to the corresponding elements of all Challenged Claims containing identical or substantially identical language.

A. Ground 1: Lafer in View of Tso Renders Obvious Claim 1

60. Claim 1 of the '575 patent is obvious over Lafer in view of Tso. Lafer discloses a system that uses a tag cache to substitute special tags embedded in an HTML page with HTML fragments associated with the underlying data for content, which includes media data that is cached in a content database. Tso discloses caching both original and transcoded versions of content, including transcoding content based on “selection criteria” specifying how that original content data is manipulated according to user, network, and proxy server parameters, and that both the original and manipulated data are stored in cache for future use. Thus, as further set forth below, Lafer in view of Tso renders obvious claim 1 of the '575 patent.

1. U.S. 6,192,382 (“Lafer”) Overview

61. Lafer teaches a method and system for “rapidly rendering personalized HTML pages” by replacing “particular tags” that are embedded in “a stored HTML page” with “content information in the form of HTML fragments previously created based on user preference information.” Lafer, Abstract, 1:45-48, 1:54-58. An embodiment of this system is shown in Figure 1, which is annotated below:



62. As shown and described, users communicate with Lafer's system by contacting the system's **front-end logic using a browser**. Lafer, 2:65-3:7. The

system stores “tag-extended HTML files,” which correspond to Web pages that a user may request, in **“storage medium or device” 26**. Lafer, 3:8-22. The tag-extended HTML files are embedded with “specialized tags” that are not “part of the standard HTML library.” Lafer, 1:54-58, 3:15-16, 4:10-13, 4:55-60. The contents of each specialized tag “includes a set of tag attributes,” which “can contain substitution tokens.” Lafer, 4:24-31. These substitution tokens are “replaced by values” such as “content data” as well as “user session information” during the rendering of personalized Web pages in response to a user’s request. *Id.* The content data with which the system replaces particular specialized tags is stored as “HTML fragments” corresponding to “content objects” that are “fetched from **an Object Cache” 48**. Lafer, 3:37-43. The HTML fragments are associated by the system with content data stored in, for example, **content database 58 (“Content DB”)**. *Id.* The system stores “tag-content pairs” that associate the “specialized tags” with their corresponding HTML fragments in a **“tag cache.”** Lafer, 3:22-43. Once the page is constructed “in accordance with the present invention,” the system stores the generated personalized page in **“page cache” 16** if it is “likely to be used again in the future” to “reduce[] latency.” Lafer, 3:8-14. A **“tag cache”** stores tag-content pairs associating the special tags with the HTML fragments (including fragments

that are standard HTML tags). Lafer, 3:22-43. A “page cache” stores previously-generated personalized pages for reuse. Lafer, 3:8-14.

63. Lafer uses the term “HTML fragments” consistent with the understanding of that term in the art to refer to HTML code, such as HTML tags. Another patent, U.S. Patent No. 6,310,601, which names Moore as its first inventor, a copy of which I understand has been designated as Exhibit 1007, provides general teachings of such tags consistent with the understanding of a person having ordinary skill in the art. *See* Moore, 3:41-53, 4:5-13, 5:29-30, 6:59-7:13. Moore provides an example of an “HTML code sequence,” which Moore also refers to as an “HTML fragment,” consistent with how that term is used in the ’745 patent. Moore, 6:59-7:13. As Moore explains, webpages display images, for example, using “commands” implemented using “img” tags—a standard HTML tag. Ex. 1007, 3:41-4:35, 5:29-30. These tags provide “options” for basic adjustments to the image, including changing its “height” or “width.” *Id.* An example of such a tag is the following: “.” Moore, 5:19-33. This tag specifies that the image “image.gif” should be embedded within the webpage, and that the image should be scaled to be 300 pixels tall, and 400 pixels wide. *Id.* Image tags can include other parameters as well, including “align,” “border,” and “alt,” each of which sets parameters for how the image (or alternative

text) should be displayed. Moore, 4:4-35. Thus, as described in Lafer, when using such a tag as a “fragment,” the “fragment” itself is the “img” tag, and the underlying “data” for that tag would be the image itself. *See id.*; Lafer, 3:37-43. This is because Lafer’s “fragments” are substituted for tag values, and are associated with “content” data that is stored in the Content DB. Lafer, 3:22-35. As was understood by a POSITA, a fragment used to refer to content is an HTML tag, such as an “IMG” tag used to reference an image file. Moore, 3:41-4:35, 5:19-33. Thus, a person having ordinary skill in the art would have found Lafer’s teachings to not only render it obvious to use an HTML “image” tag as a “fragment” referencing an image to be inserted in a webpage, but actually disclose this. *Id.*

64. Lafer further discloses that the logic blocks shown in Figure 1 “could be combined into larger logical units or further subdivided to conveniently implement the system.” Lafer, 4:65-5:2. For example, a person having ordinary skill in the art would have understood that the **object cache** accessed by the **“tag cache,”** to obtain content objects in the form of HTML fragments, along with the content items stored in, for example, the **“Content DB,”** could be grouped together and characterized as a “back-end system.” *Id.* Similarly, a person having ordinary skill in the art would have understood that well-known front-end server components, including the **front-end logic**, the **page cache** of previously-generated pages, and

the **storage for tag-extended HTML files**, could be logically integrated and cumulatively characterized as a “front-end system.” *Id.*

65. Below is annotated Figure 2, summarizing certain portions of the process disclosed by Lafer:

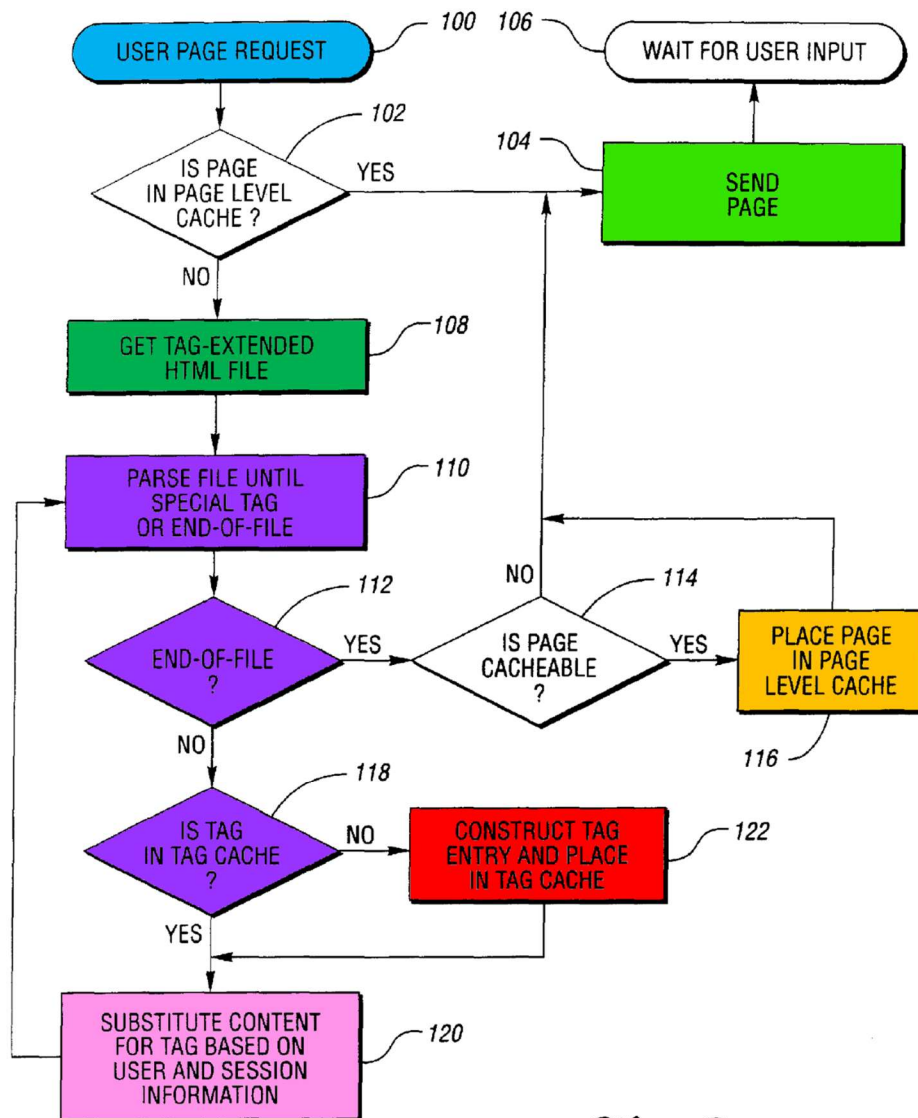


Fig. 2

66. After receiving a **user request for a Web page**, the system checks if the requested page was previously generated and stored in the page-level cache. *Id.* If not, the system **retrieves a tag-extended HTML file corresponding to the requested Web page (step 108)**. *Id.*; Lafer, 3:52-65. The system then **parses the tag-extended HTML file “until either a particular tag or the end-of-file is encountered” (steps 110-118)**. Lafer, 3:8-10, 3:65-67. If and when a special tag is identified in the HTML file during parsing, **the tag is replaced with a corresponding HTML fragment based on user and session information**. Lafer, 4:8-10, 3:22-25. “If the tag does not exist in the tag cache” at the time it is identified during parsing, **“a tag-content pair is constructed from information in the object cache and databases” and placed in the tag cache for future reference (step 122)**. Lafer, 4:46-48, Figure 2. Once the end of the file is reached, page generation is complete and the page is **“placed in the page cache” (step 116)** for future use and then **“sent to the user” to fulfill the request (step 104)**. Lafer, 3:67-4:7.

2. Overview of Tso and Motivation to Modify Lafer with Tso’s Teachings

67. Tso discloses a “system for dynamically transcoding data transmitted between two computers over a communications link,” where the two computers are “a network server computer and a network client computer.” Tso, 1:13-14, 2:9-12. Tso provides that transcoding refers to “virtually any manipulation of data including,

but not limited to, adding, modifying or deleting data.” Tso, 2:47-49. When a client sends a request for “image, video, or HTML” content, the “request packet” carries “information” in both its “header” and “data portion[s]” that includes one or more “selection criterion,” which are used to specify what transcoding should be applied to the content. Tso, 3:46-54, 4:6-10, 6:64-8:9.¹ Each “selection criterion” may specify, for example, user-specific parameters, including “user,” “network,” or “proxy” server parameters such as “bandwidth” to be accommodated in transcoding the requested media content to fulfill the request. Tso, 7:15-8:9.

68. After the request is received, the server checks the “server-side cache memory 30” for a transcoded version of the requested content that can satisfy the request. Tso, 9:49-56, 10:24-27, 5:62-6:28. If such a suitable version is not available in the server-side cache memory 30, the server retrieves an “original” version of content from the network (for example, the Internet). Tso, 10:27-32, 6:28-30, 5:8-

¹ Tso sometimes inconsistently uses the terms “selection criterion” and “selection criteria” to collectively refer to either singular selection criterion and combinations of different selection criteria. *See, e.g.*, Tso, 7:20-21 (using the singular “criterion” to refer to a list of criteria: “The predetermined selection criterion may comprise: [examples of selection criteria]”), 8:10-13 (using the plural “criteria” to refer to the same list of criteria: “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.”), 15:36-38 (“location-specific characteristics are to be used as predetermined selection criteria”). I have used “criteria” and “criterion” consistent with Tso’s teachings of these terms, and generally use “criterion” to refer to the singular and “criteria” to refer to the plural.

10. The server then transcodes the retrieved content by “selectively invok[ing] one of transcode server providers 24 based upon satisfaction of a predetermined selection criterion” to generate a transcoded version of content that will satisfy the user’s request. Tso, 10:32-47, 6:37-44, 6:64-7:14. Both the original and transcoded versions of the content are cached “for later transmission,” and the transcoded version of the content is returned to the user. Tso, 10:47-49, 4:1-5, 5:8-26, 6:28-31, 6:48-50.

69. A person having ordinary skill in the art would have been motivated to modify Lafer with relevant teachings from Tso. Lafer and Tso are analogous prior art and in the same field as the ’575 patent which, as I discussed above, with respect to the level of ordinary skill in the art, is networked media caching and delivery. *See* ’575 patent, 1:9-11 (“[T]he invention relates to an Internet server-based software system that provides delivery of automated graphics and other media to Web sites for access by an end user or consumer”). Both Lafer and Tso are also reasonably pertinent to the alleged problems identified in the ’575 patent, namely, automating media delivery and reducing the required overhead and time to generate Web pages.’575 patent, 4:47-67 (“provide[] a less expensive and less time-consuming process.”), 4:64-67, 5:33-37, 6:36-40, 13:1-24; Lafer, 4:46-54 (“[t]he use of the object cache shortens database access latency”), 1:15-18, 1:24-35, 4:55-61; Tso,

1:10-31 (“a system for dynamically transcoding data transmitted between two computers over a communications link”), 1:59-2:5, 4:1-5.

70. While Lafer generally discloses caching media objects and generating personalized content in accordance with, for example, user preferences and session information stored in “substitution tokens” to generate personalized content (*e.g.*, Lafer, 3:2-7, 3:15-43, Fig. 1), Lafer leaves it to the person having ordinary skill in the art to determine how to implement this caching and content generation. Tso provides detailed teachings about the caching and personalization of content and web pages that a person having ordinary skill in the art would have understood to be compatible with Lafer’s disclosures.

71. As one example, Tso explicitly teaches generating and storing original and transcoded versions of media objects in server-side cache memory. Tso, 4:1-5. And with respect to Lafer’s teaching of substituting tags with content “based on stored user preferences and session information,” Tso specifically teaches that “selective[] transcod[ing]” of content is achieved using “a predetermined selection criterion” or “combinations thereof” that indicate user and session information. Tso, 3:48-54, 7:20-8:21. Tso teaches that such criteria may provide information about, for example, “network client 12,” “network characteristics, including ... bandwidth,” and “user preferences” that can inform the system on how to

appropriately transcode the requested content. Tso, 7:20-8:21. A person having ordinary skill in the art would have been motivated to modify Lafer's broad teachings with these specific teachings from Tso to achieve a system in which requested media objects that are not already in cache are retrieved from the network, transcoded according to user preferences and network conditions, and stored in the "Content DB," with a corresponding "HTML fragment" such as a standard HTML tag—*e.g.*, an "img" tag for an image file—is stored in the "object cache" as an "HTML fragment" to be substituted for the specialized tags embedded in "tag-extended HTML files." A person having ordinary skill in the art would have further understood that this resulting system transcodes and generates the requested media content, which would be cached and then served to the client using session-specific selection criterion information, such as bandwidth, as part of the "user session information" contained in the "substitution tokens" within Lafer's "specialized tags." A person having ordinary skill in the art would have been motivated to modify Lafer with Tso's teachings for at least the following independent reasons.

72. **First, Lafer** leaves it to a person having ordinary skill in the art to implement its teachings of substituting tags with HTML fragments referencing media content "based on stored user preferences and session information" carried in a special tag's "substitution tokens." Lafer, 3:2-35, 4:7-31. **Tso** teaches that the

request packet received by the system includes “selection criteria” that are used to specify how to transcode content, and provides detailed teachings of examples of types of selection criteria including parameters based on the “network client,” “network,” and “user preferences,” including “bandwidth”; Tso also provides flexible teachings of how such information is provided to the server as part of the request (*e.g.*, by sending in different parts of a request packet). Tso, 3:48-54, 7:20-8:21, 9:26-33. A person having ordinary skill in the art would have looked to modify **Lafer** with **Tso’s** teachings to implement Lafer’s “substitution tokens” because Tso teaches a known technique for providing session information to the system, and provides a broad disclosure of including such information as selection criteria in a request, including, *e.g.*, describing the use of session information such as “bandwidth” to adapt to the “condition of a communications link” during a session, improving reliability of the communication. Tso, 7:4-41, 16:63-17:9. Modified with **Tso’s** teachings, **Lafer’s** substitution tokens carry user preference and session information, including **Tso’s** selection criteria such as “bandwidth.” Indeed, it was well known in the art to use such parameters about a client’s system when determining how to transcode content. For example, U.S. Patent No. 6,144,996, which names Starnes as its first named inventor, a copy of which I understand has been designated as Exhibit 1011, teaches that users will perform a “content

negotiation” to ask a server to send, for example, a “smaller (*e.g.*, lest robust, lower quality, or smaller image area)” content item in order to advantageously “facilitate[] improved bandwidth utilization and responsiveness of the Internet.” Starnes, 2:24-41. As another example, Harini Bharadvaj et al., *An Active Transcoding Proxy to Support Mobile Web Access*, PROCS. 17TH IEEE SYMP. ON RELIABLE DISTRIBUTED SYS. 118 (1998), dated October 1998, a copy of which I understand has been designated as Exhibit 1012,² teaches a “proxy based system” for supporting “web browsing from mobile clients over wireless networks.” Bharadvaj, 118. Bharadvaj teaches transcoding image files by performing transformation operations such that “[t]he image files are scaled down in size, or the number of colors is reduced, or both,” and caching content, in order to address bandwidth issues for such mobile users. *Id.*, 120; *see also id.* 118-21. It was thus already well-understood in the art to apply transcoding operations based on user parameters, such as bandwidth, to better allow for the efficient generation and provision of web pages to users over the Internet.

73. **Second**, both **Lafer** and **Tso** address similar needs to improve the efficiency of a system. Lafer, 1:15-18, 1:27-35, 4:46-61; Tso, 4:45-47. For example,

² I understand that Bharadvaj et al. was available on or around October 1998. *See* <https://ieeexplore.ieee.org/document/740482>.

Lafer discloses that its system will “improve distribution efficiency” by “caching ... the information stored in a database accessed by a server” to “reduc[e] the amount of time required by a server to construct or generate” a requested Web page and “improve distribution efficiency.” *Lafer*, 1:15-18, 1:27-35, 4:45-47. **Tso** similarly teaches that caching original and transcoded versions of content advantageously eliminates “the need to re-retrieve” and “re-transcode” content, further improving efficiency. *Tso*, 4:1-5. For example, *Tso* teaches that server transcoding capacity may be impacted by the “current processing load” and the “condition of a communications link”; a person having ordinary skill in the art would have recognized that minimizing unnecessary re-retrieval and re-processing of data would reduce this processing load. *See Tso*, 7:7-12. Indeed, *Tso* recognizes that high processing loads will force servers to prioritize work in response to some requests at the expense of other requests. *Tso*, 16:39-54 (transcoding servers “may boost thread scheduling priorities ... for the VIP service” and “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”). A person having ordinary skill in the art would have recognized that applying **Tso’s** teaching of caching original media content in a server-side cache would further improve the efficiency of **Lafer’s** caching teachings, allowing content to be delivered to users

more quickly. A person having ordinary skill in the art therefore would have been motivated to modify **Lafer** with **Tso's** teachings by caching both original and transcoded versions of media in **Lafer's** Content DB, as taught by **Tso**, to advantageously minimize re-retrieval and re-transcoding of content. Indeed, the use caches to store both originally-retrieved and processed/generated content was well-understood in the art. For example, U.S. Patent No. 6,745,368, which names Boucher as its first named inventor, a copy of which I understand has been designated as Exhibit 1013, teaches that using a cache for storing both originally-retrieved and “fully rendered” content, advantageously resulting in “decrease[d] retrieval latencies” and “reduce[d] access times,” “significantly increas[ing] content viewing speeds ... while decreasing processing requirements.” Boucher, 1:41-50, 5:1-5. Another example teaching in the art comes from, Rakesh Mohan et al., *Adapting Multimedia Internet Content for Universal Access*, 1 IEEE TRANSACTIONS ON MULTIMEDIA 104 (1999), dated from March 1999, a copy of which I understand has been designated Exhibit 1014. Mohan et al. teaches that caching the different formats of content will “improve response times” for content. Mohan et al., 106. A third example is from Armando Fox et al., *Adapting to Network and Client Variation Using Infrastructural Proxies: Lessons and Perspectives*, 5 IEEE PERS. COMMC'NS. 10 (1998), dated from August 1998, a copy of which I understand has been

designated as Exhibit 1015. Fox et al. similarly teaches using caching of “both original and transformed content” as part of providing Internet services. Fox, 13.

74. **Third, Tso** provides additional detail about how to implement an efficient caching process that is applicable to **Lafer**. **Lafer** discloses using a “Content DB” to store the cached underlying “data” for HTML fragments, which describes for example the underlying media itself that is referred to in an HTML fragment with a tag such as an “img” tag pointing to image media, but leaves it to a person having ordinary skill in the art to implement populating media cached in the “Content DB.” Lafer, 3:32-43. **Tso** teaches retrieving and caching both “original” and “transcoded” versions of content when the content is transcoded in order to eliminate the need for re-transcoding and re-retrieval of content in response to later requests. Tso, 14:37-46, 3:48-4:5. **Tso** thus teaches how to implement **Lafer’s** “Content DB” by storing both the transcoded version of content data to be accessed by a user as taught in **Lafer**, and the original content data, which allows the cache to efficiently deliver requested content to the user quickly by, for example, eliminating the need to re-retrieve new content in response to further requests for the content. Tso, 4:1-5. Indeed, such implementations in which transcoded versions of content are saved were well-understood in the art. For example, U.S Patent 5,420,967, which names Delp as its first inventor, a copy of which I understand has

been designated as Exhibit 1010, teaches that “intermediate images” produced at each step as an “existing compressed image” are processed by being “decompressed, scaled to the correct size, corrected to the original color, matched to the color of the display, converted to the resolution of the display, and rotated,” “may even be optionally saved” as a “representation” in an “image table.” Delp, 3:64-4:10, 6:20-32. Boucher teaches that “rendered content can be stored in a traditional cache, a database, a file system or other storage media.” Boucher, 9:34-37.

75. In light of the above teachings, a person having ordinary skill in the art would have had a reasonable expectation of success in modifying **Lafer** with **Tso’s** teachings by (1) using **Lafer’s** “substitution tokens” in its “specialized tags” to carry **Tso’s** selection criteria, and (2) transcoding and caching content as taught by **Tso** in order to provide such content to clients. **Tso**, like **Lafer**, already teaches using its disclosures for “customizing content.” Tso, 7:62-8:4, 9:26-29; Lafer, 3:44-47, Abstract, 1:27-30. Indeed, it was well understood for a user request to specify parameters for customizing content, and for a cache to store original and customized versions of content, using, *e.g.*, standard HTML. For example, Moore teaches commands that “cause an image to display” and provide “options such as HEIGHT=nn (where nn is the number of pixels), WIDTH=nn (where nn is the number of pixels), WIDTH=nn % (where nn is the percentage of screen width 1-

100%), [that] will greatly alter the dimensions of the image 201 being displayed.” Moore, 4:5-60. Mohan teaches “a system that adapts multimedia Web documents to optimally match the capabilities of the client device requesting it” that employs “[a] representation scheme called the InfoPyramid that provides a multimodal, multiresolution representation hierarchy for multimedia” and “[a] customizer that selects the best content representation.” Mohan, 104. Mohan also teaches that “client specific versions of content” are stored in cache “to improve response times.” Mohan, 106. As a third example, Delp teaches that “[h]igh quality images 250, 260, and 270 are to be generated from this compressed image. In order to generate high quality image 250, image 200 must first be decompressed, corrected to the original color (also known as dynamic range adjust, input gamma correct, and/or sigma correct), scaled to the correct size, matched to the color of the display (also known as linear transformation), converted to the resolution of the display (also known as halftoning, dithering, and/or diffusing), and rotated.” Delp, 3:42-51. Delp also teaches that the intermediate image resulting from each of the aforementioned transformations is stored in as a representation in an image map. Delp, 6:20-22. As a fourth example, Fox teaches “a particular subset of Internet services, based on transformation (distillation, filtering, format conversion, etc.), aggregation (collecting and collating data from various sources, as search engines do), caching

(both original and transformed content), and customization (maintenance of a per-user preferences database that allows workers to tailor their output to the user's needs or device characteristics)." Fox, 13. Boucher teaches "Semantic Representation," which is "[a] description of the characteristics, attributes, logical structure, and features of multimedia elements (or objects) that form a rendered representation of multimedia content, or a portion thereof The semantic representation ... is structured such that the semantic representation can be saved as formatted and indexed semantic content in a file or database, and rapidly restored from the semantic content." Boucher, 6:1-12. Modifying **Lafer** with **Tso's** teachings would have therefore been a straightforward application of those teachings in a manner that advantageously improves the efficiency of a system that endeavors to quickly provide customized content to fulfill user requests.

3. Invalidity of Claim 1 Over Lafer in view of Tso

(a) Element [1.pre]

76. **Element [1.pre] recites "A process for delivering an original media, comprising the steps of."** This element is met by Lafer alone and the combined teachings of Lafer and Tso.

77. **Lafer discloses a process for delivering an original media** (*e.g.*, "distributing" "content" in an "HTML page[]" in response to "user requests"). *See, e.g.*, Lafer:

- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.*”)
- **3:23-42** (“*A tag cache* as shown in reference 28 *contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs* as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, *the tag cache is implemented*

using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, *other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on *the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.*

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The data for these objects are maintained in respective databases 58, 60, 62 and 64.”)*

- **4:55-61** (“Thus, the present invention provides a system and *method for* selectively constructing and *distributing personalized HTML pages* by *storing selected fragments of HTML text in a tag cache*, and rapidly rendering the personalized HTML pages via *special tags embedded in extended HTML files which allow identification and retrieval of the desired fragments.*”)
- *See also* Abstract, 1:54-58, 3:2-3, 3:16-20, 4:48-54.

78. Tso further explicitly discloses delivering original media (e.g., “transmission” of “content” with an “original” version). See, e.g., Tso:

- **Figure 3**

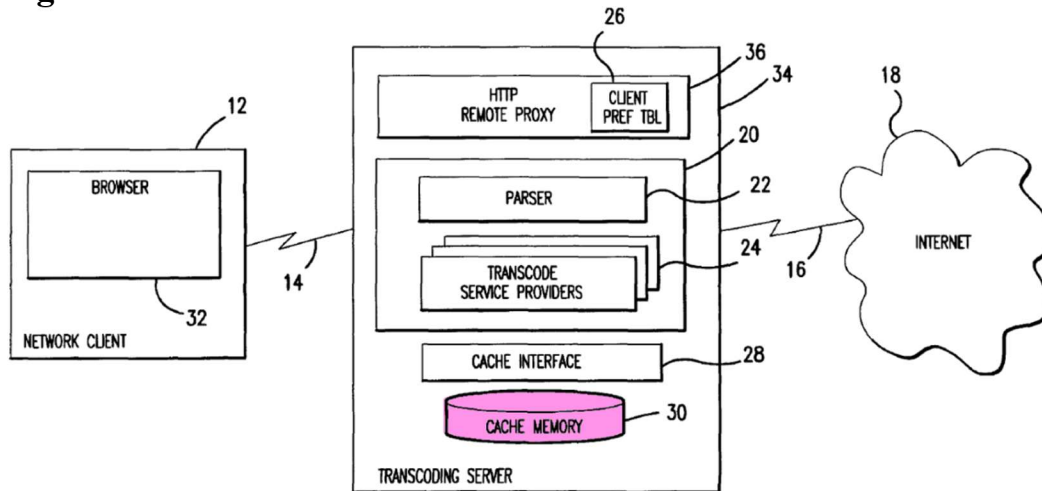


FIG.3

- **3:51-54** (“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).”)
- **4:1-5** (“*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.*”)
- **See also 2:44-53, 3:31-34, 5:8-10, 5:62-64, 6:9-13, 6:19-23, 10:37-44.**

79. Lafer discloses a “method for selectively constructing and distributing personalized HTML pages” containing “content” by “dynamic substitution of content for tag values” in HTML files that “are selectively augmented with a

plurality of particular tags each associated with one or more of the predetermined HTML fragments.” Lafer, 4:55-61, 3:15-21, 2:15-33.

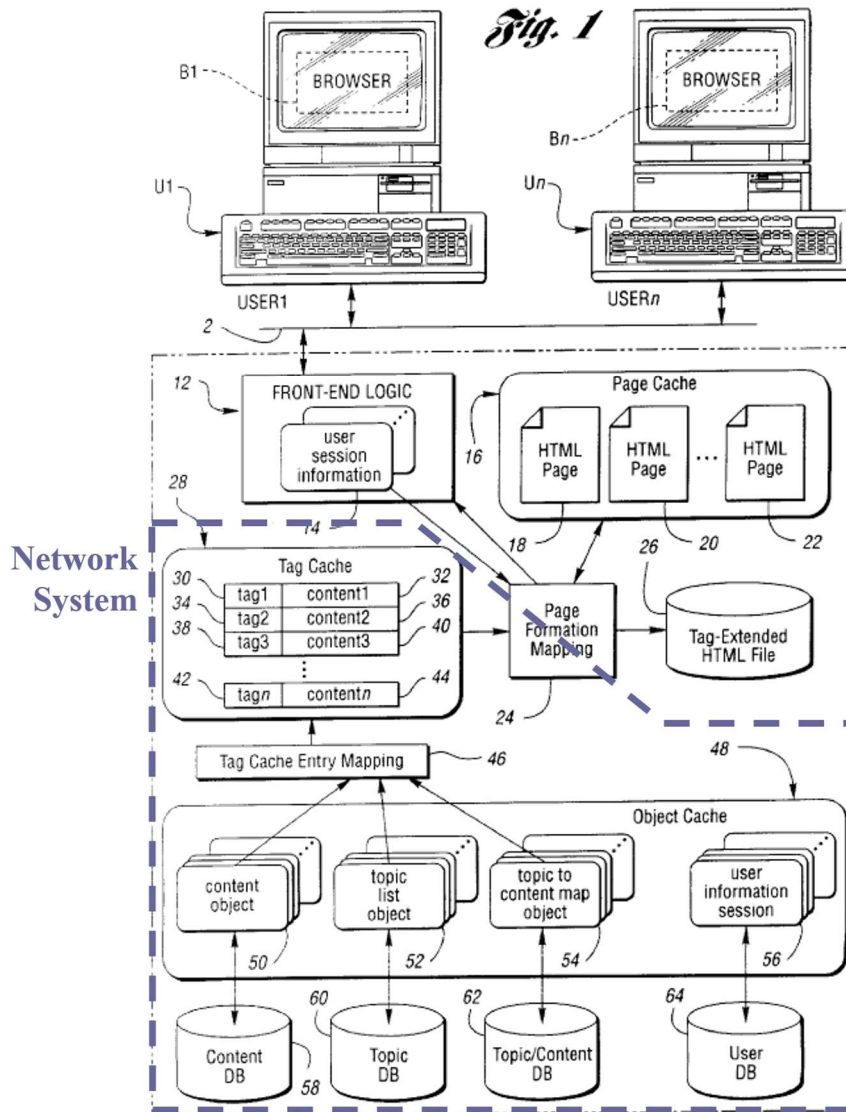
80. **Lafer** also already discloses that the “Content DB” caches the underlying “data” for content, which a person having ordinary skill in the art would have understood includes original media content. Lafer, 3:32-43. Furthering this teaching, **Tso** provides additional explicit disclosure of caching “original” versions of content, including “image” and “video” content for “transmission” to users. Tso, 3:41-54, 4:1-5. As discussed in Section IX.A.2, a person having ordinary skill in the art would have been motivated to modify **Lafer’s** general disclosure with **Tso’s** more detailed teachings of caching and delivering media content to advantageously and efficiently deliver personalized Web pages that include media content.

(b) Element [1.a]

81. **Element [1.a] recites “placing said original media in a network system.”** This element is met by Lafer alone and the combined teachings of Lafer and Tso.

82. **Lafer discloses placing said original media** (*e.g.*, placing “data for” “content”) **in a network system** (*e.g.*, in the “Content DB” of a network system that also includes “logical units” for an “Object Cache” and “Tag Cache,” and communicates over the “Internet”). *See, e.g.*, Lafer:

• **Figure 1:**



- **1:59-2:1** (“In accordance with one aspect of the present invention, *a method for selectively distributing information including personalized Web pages on a network such as the Internet* is characterized by the steps of storing in a tag cache at least one HTML fragment derived from information stored in a database as determined from a user's input preference information, and

associating the at least one HTML fragment stored in the tag cache with a particular tag to be selectively embedded in an HTML document or file stored in a server.”)

- **3:15-21** (“*A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic substitution of content for tag values based on stored user preferences and session information. The storage device may be implemented as a directory system or as a database.*”);
- **3:23-42** (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache.*
A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list

data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50*, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The *data for these objects are maintained in respective databases 58, 60, 62 and 64.*")

- **4:32-35** ("If a particular tag is encountered prior to the end-of-file the tag cache is checked as is shown in block 118. *If the tag is already associated with contents in the tag cache, the contents are substituted for the tag* as in block 120.")
- **4:58-5:2** ("Further, while a preferred embodiment of the present invention has been described, it will be appreciated that actual implementation of the system architecture is susceptible to modification. For example, *the logic blocks as described in conjunction with FIG. 1 can be implemented in hardware, software or a combination thereof, and that the blocks could be combined into larger logical units or further subdivided to conveniently implement the system.*")
- *See also Abstract, 2:50-57.*

83. To the extent additional disclosure of “original media” is required, **Tso** further explicitly discloses placing said original media in a network system (e.g., original media (see [1.pre]) is cached in “[s]erver-side cache memory”). See, e.g., Tso:

- **Figure 3**

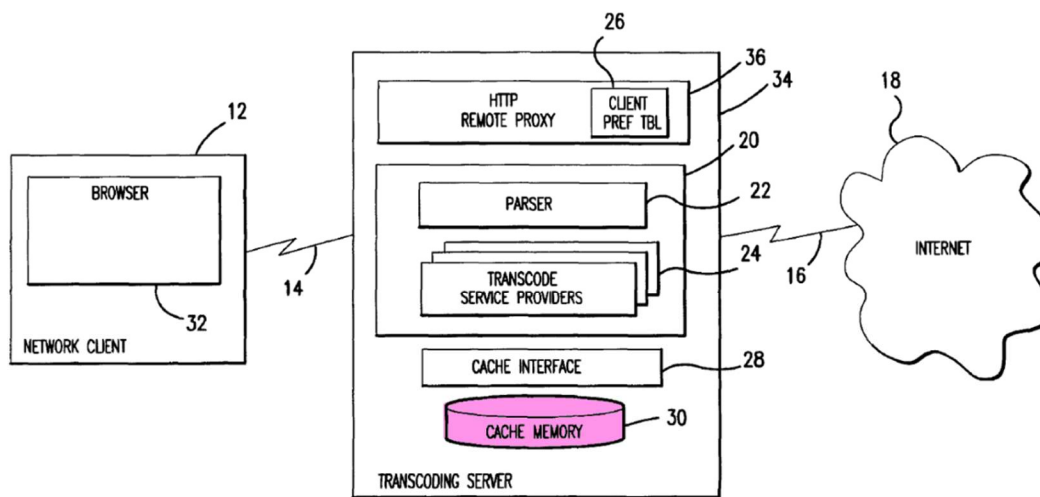


FIG.3

- **3:51-54** (“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).”)
- **4:1-5** (“*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.”)

- **10:37-44** (“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, video/mpeg)* by interrogating a MIME type in the content-type header record that appears at the beginning of the incoming HTTP data stream.”)
- **See also 2:44-53, 3:31-34, 5:8-10, 5:62-64, 6:9-13, 6:19-23.**

84. Lafer discloses that “data” for “content objects” that are ultimately displayed on webpages is stored in a “Content DB” database. Lafer, 3:23-43, Figure 1. As I discussed above in Section IX.A.1, the term “content” refers to the HTML fragments stored in the media tag database, while the underlying “data” stored in the “Content DB” includes the actual media content. Because Lafer characterizes the “HTML fragments that are substituted for tag values” as the “content” portion of a “tag-content pair,” a person having ordinary skill in the art would have understood the “data” underlying what is displayed on a webpage and

associated with the HTML fragments to refer to content such as media content like images and video. *See id.*

85. Lafer discloses that the “Content DB” that is part of a server-side system includes several other logical components, including a “Tag Cache” and “Object Cache” that are used to manage tags and media content objects, respectively. Lafer, Figure 1. Though these components are shown as “logical units,” Lafer discloses that these blocks “could be combined into larger logical units or further subdivided to conveniently implement the system.” Lafer, 4:65-5:2. Moreover, Lafer discloses that within the “server 10,” the “front-end logic” logical block can be separately “implemented as a standard web server.” Lafer, 2:65-3:3. The fact that this logical component of the system can itself be implemented as its own standard web server further confirms that a person having ordinary skill in the art would have understood that Lafer discloses that “server 10” may itself be broken up into multiple server-side devices. *Id.*

86. Lafer further discloses that, to enable substitution of “HTML fragments” for “tag values, its method employs a “Tag Cache,” “Object Cache,” and “Content DB,” all shown in Figure 1. Lafer, 3:23-43. The “Object Cache” stores these “HTML fragments,” which reference the “data” for the “objects” including, for example, the media “content objects” stored in the Content DB. *Id.* The “Tag

Cache” holds a “plurality of tag-content pairs” that map a “particular tag” to its “associated ... contents” that are substituted for the tag. *Id.*, Lafer, 4:32-35. Based on Lafer’s disclosures that these components work together to replace tags with values and that the “logical units” shown in Figure 1 may be “combined” or “subdivided to conveniently implement the system,” a person having ordinary skill in the art would have understood that Lafer not only renders obvious implementing these components together as part of a network system, but actually discloses it. Indeed, the logical components as shown in Lafer are amenable to such an organization: the front-end logic, page cache, and tag-extended HTML file storage all communicate, through the “Page Formation Mapping” process, with the back-end tag and object caches. *See* Lafer, Fig. 1. As depicted, the page formation mapping process is also used to communicate between the front-end logic, page cache, and tag-extended HTML files to handle front-end operations as well, meaning that some of the “Page Formation Mapping” process also occurs in the front end. In the circumstances in which the page was previously-generated, the front end components alone are able to handle a user’s request. But when further parsing is required to generate a page, the back-end components must interface with these front-end components to generate the page. *See id.*

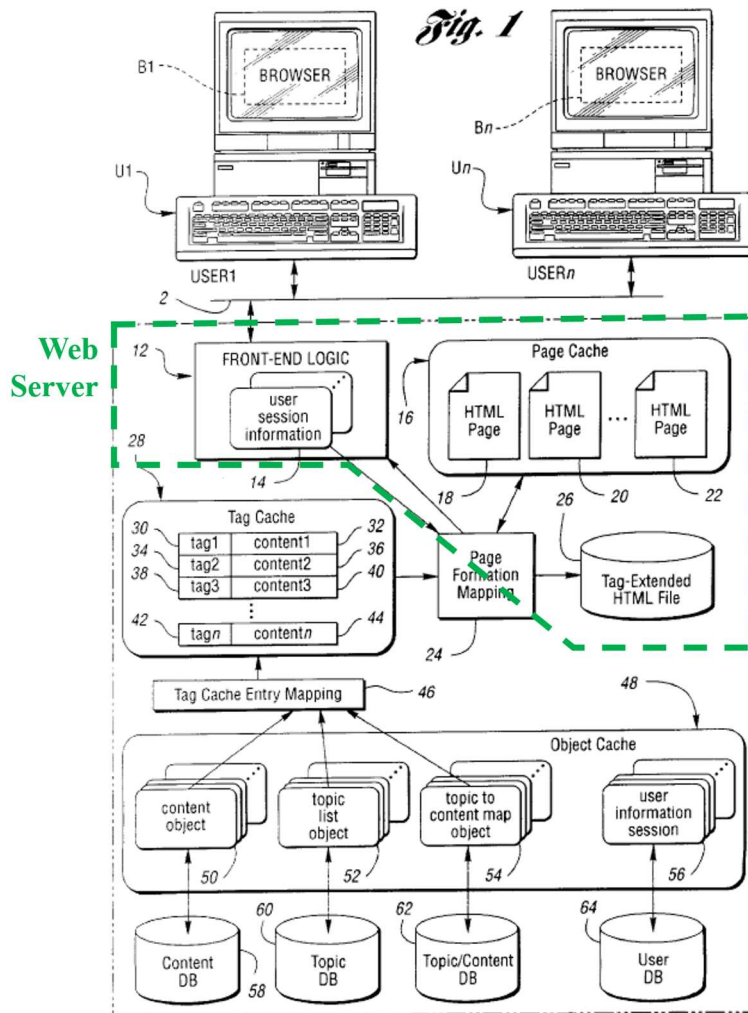
87. To the extent additional disclosure of placing the “original media” in the network system is required, **Tso** discloses caching “original” versions of content, including “image” and “video” content, in “[s]erver-side cache memory 30” on a server connected to a network such as the “Internet.” Tso, 3:41-54, 4:1-5. As I discussed for element [1.pre], **Lafer** already discloses caching the underlying “data” for content in the “Content DB”, which a person having ordinary skill in the art would have understood includes original media. Lafer, 3:32-43. **Tso** provides additional, explicit disclosure of original media, and this original media is cached in “[s]erver-side cache memory 30” on a server that is connected to the “Internet.” Tso, 3:41-54, 4:1-5. As I discussed with respect to element [1.pre] and §IX.A.2, a person having ordinary skill in the art would have been motivated to modify Lafer’s teachings with Tso’s disclosure of caching original versions of media since caching was generally understood to advantageously reduce the need to re-retrieve and transcode data, advantageously and efficiently delivering personalized pages without the unnecessary re-retrieval of media data. Lafer itself states “a need ... exists for improving distribution efficiency and server response time” in responding to user requests. Lafer, 1:39-41.

(c) Element [1.b]

88. **Element [1.b] recites “creating an HTML document or browser language having proprietary media tags and placing said HTML document or browser language onto a Web server.”** This element is met by Lafer.

89. **Lafer discloses creating an HTML document or browser language having proprietary media tags** (*e.g.*, using a “page formation mapping process” to create “tag-extended HTML files” having “specialized tags” that are not “part of the standard HTML tag library”) **and placing said HTML document or browser language onto a Web server** (*e.g.*, storing “tag-extended HTML files” in the “storage medium” of a server that also comprises “front-end logic” and “Page Cache”). *See, e.g.*, Lafer:

• **Figure 1**



- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a*

plurality of particular tags each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.”)*

- **2:65-3:22** (“Considering the server 10 in detail, *front-end logic is responsible for interfacing with users U1 to Un through the communication medium 2*. As represented by block 12, *the front-end logic can be implemented as a standard web server*. The front-end logic services the user by returning requested information in the form of HTML pages. *The front-end logic generally operates to collect session information as shown in block 14. This information can be obtained, for example, through the use of HTML forms as is commonly done with web pages.* *A page formation mapping process, as shown in reference 24, provides construction of the HTML pages* in accordance with the present invention.

A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.

A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic substitution of content for tag values based on stored user preferences and session information. The storage device may be implemented as a directory system or as a database.”)

- **4:10-22** (“More specifically, *a tag in accordance with the present invention does not constitute a part of the standard HTML tag library.* The tag syntax chosen for the preferred embodiment uses a special begin symbol and end symbol to mark the particular tag as follows:

<! - - CW>	begin symbol
type=typename	tag type attribute
. . .	additional tag attributes
<CW - - >	end symbol

)

- **4:58-5:2** (“Further, while a preferred embodiment of the present invention has been described, it will be appreciated that actual implementation of the

system architecture is susceptible to modification. For example, *the logic blocks as described in conjunction with FIGURE 1 can be implemented in hardware, software or a combination thereof, and that the blocks could be combined into larger logical units or further subdivided to conveniently implement the system.*")

- **See also 1:54-58, 2:50-51, 2:67-3:2, 4:55-61.**

90. See my discussion of elements [1.pre]-[1.a]. As explained for element [1.a] above, Lafer not only renders obvious that the “Tag Cache,” “Object Cache,” and “Content DB” together form a network system, but actually discloses it. See [1.a]. Similarly, Lafer discloses the “front-end logic,” “page cache,” and “storage medium” work together as part of the “page formation mapping process” to receive user requests, return pre-cached copies of pages, and retrieve tag-extended HTML files that will be further processed by the network system components discussed for element [1.a]. Lafer, 2:65-3:22. Because these logical units are “combin[able]” as discussed above in element [1a], Lafer thus not only renders obvious the claimed “Web server,” but actually discloses it because a person having ordinary skill in the art would have understood it to teach combining the “front-end logic” (which “can be implemented as a standard web server”), “page cache,” and “storage medium” together to form the claimed Web server which, through part of the “Page Formation

Process,” interact with one another to handle front-and aspects of requests for pages as I discussed with respect to element [1.a]. *See id.*, Lafer, 4:65-5:2. And as I explained with respect to [1.a], Lafer’s teachings show that the back-end “Web server” components work together in a manner that not only renders obvious organizing these components together, but actually discloses it. *See* Lafer, Fig. 1.

91. In addition, Lafer discloses that the “page formation mapping process” is used to create, and then store in the server’s “storage medium,” “tag-extended HTML files” on this Web server. 2:65-3:22. These tag-extended HTML files, which are HTML documents, are “selectively augmented with a plurality of particular tags.” Lafer, 2:20-25. Lafer describes these tags as “specialized tags” that are not “a part of the standard HTML tag library.” Lafer, 1:49-58, 4:10-22.

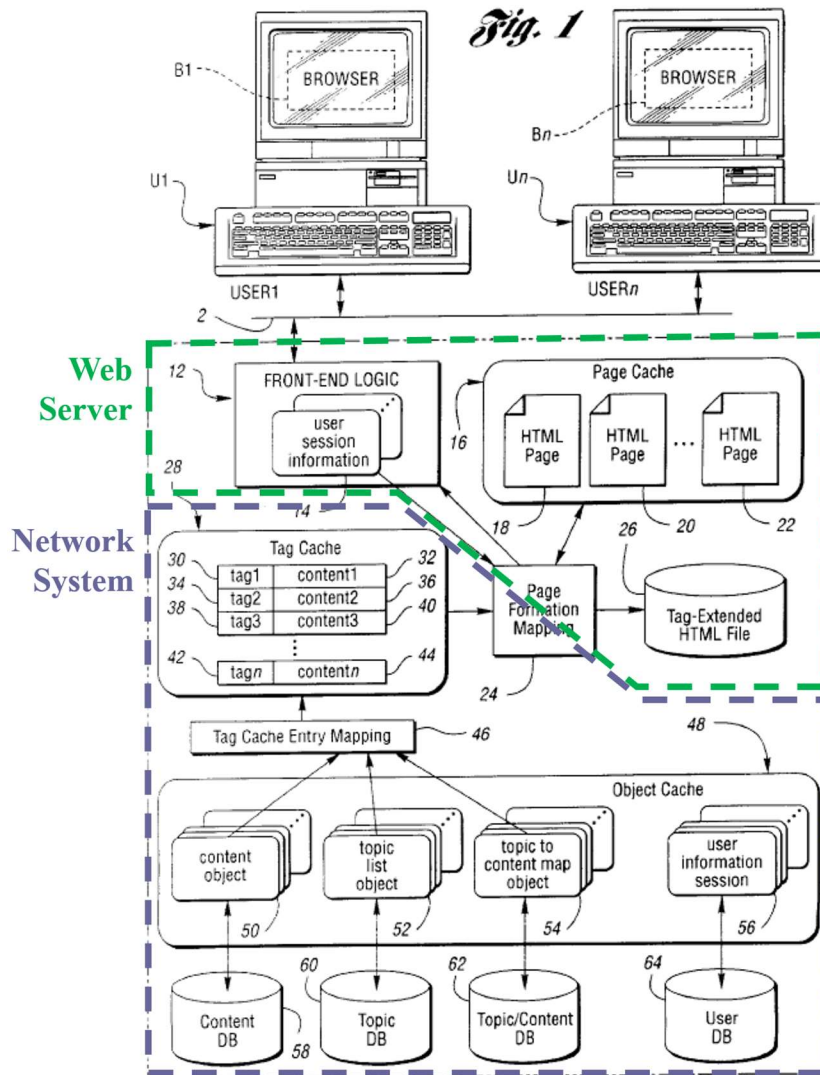
(d) Element [1.c]

92. **Element [1.c] recites “actuating a user request from a customer browser for a Web page by having said Web server pass said requested Web page to an HTML parser on said network system.”** This element is met by Lafer.

93. **Lafer discloses actuating a user request from a customer browser for a Web page** (*e.g.*, a “user” views pages on a “browser,” and the system “services the user by returning requested information”) **by having said Web server pass said requested Web page to an HTML parser on said network system** (*e.g.*, web server (*see* [1.b]) “obtain[s]” the “tag-extended HTML file” from cache and provides

it to “appropriate software, hardware, or a combination of both” on a network system (see [1.a]) used to “parse[]” the page). See, e.g., Lafer:

- **Figure 1**



• **Figure 2**

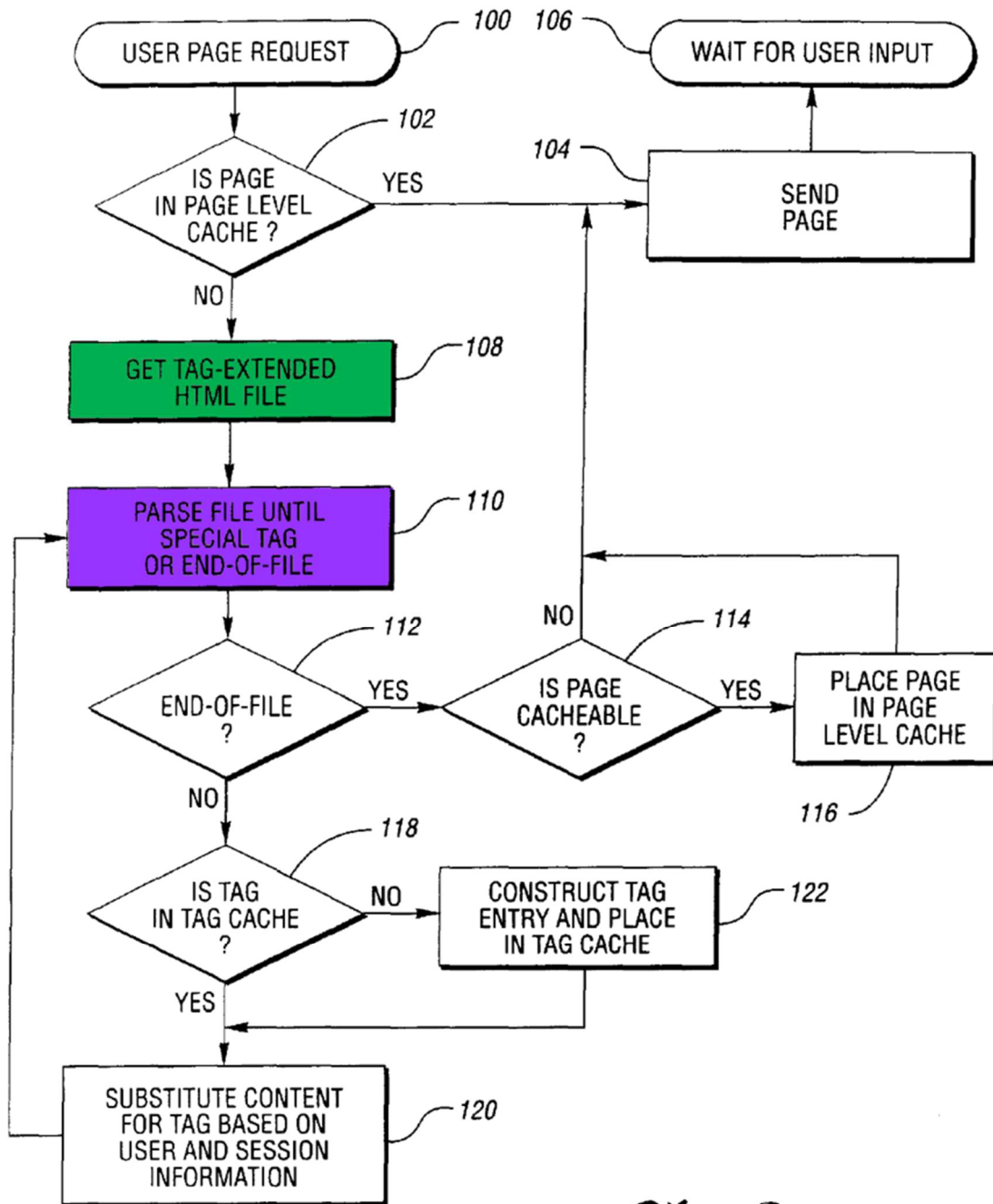


Fig. 2

- **1:15-18** (“As such, a key factor in optimizing distribution efficiency is reducing the amount of time required by a server to *construct or generate a particular Web page requested by a user.*”)
- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.*”)

- 2:62-64 (“*Each user may view the HTML pages using a conventional Internet browser as represented by B1 through Bn.*”)
- 2:65-3:22 (Considering the server 10 in detail, *front-end logic is responsible for interfacing with users U1 to Un through the communication medium*

2. As represented by block 12, *the front-end logic can be implemented as a standard web server*. The front-end logic services the user by returning requested information in the form of HTML pages. *The front-end logic generally operates to collect session information as shown in block 14. This information can be obtained, for example, through the use of HTML forms as is commonly done with web pages.*

A page formation mapping process, as shown in reference 24, provides construction of the HTML pages in accordance with the present invention.

A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.

A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic

substitution of content for tag values based on stored user preferences and session information. The storage device may be implemented as a directory system or as a database.”)

- **3:8-10** (“*A page formation mapping process, as shown in reference 24, provides construction of the HTML pages* in accordance with the present invention.”)
- **3:44-55** (“Referring now to FIG. 2, a flow chart illustrates overall operation for providing selective distribution of information and construction of customizable pages in accordance with the present invention. *The illustrated steps are not necessarily sequential operations, and may be implemented using appropriate software, hardware, or a combination of both.* The sequential flow of the steps has been shown for ease of illustration. Block 100 of the flow diagram represents *a user request for information in the form of an HTML page. As is well known, this can be a Uniform Resource Locator (URL) address for Internet servers.*”)
- **3:62-4:7** (“*If the page is not in the cache, the page must be constructed or generated. The generation process begins by obtaining the tag-extended HTML file* as shown in block 108. Block 110 shows that *the tag-extended file is then parsed until either a particular tag or the end-of-file is*

encountered. If the end-of-file is encountered, as is shown in block 112, a check is made to determine if the page should be cached as in block 114. If so, *block 116 indicates that the page is placed in the page cache*. This process can be implemented using any known algorithm or technique available for managing a page cache. Flow then passes to block 104 where *the page is sent to the user* as previously described.”)

- **4:58-5:2** (“Further, while a preferred embodiment of the present invention has been described, it will be appreciated that actual implementation of the system architecture is susceptible to modification. For example, *the logic blocks as described in conjunction with FIGURE 1 can be implemented in hardware, software or a combination thereof, and that the blocks could be combined into larger logical units or further subdivided to conveniently implement the system.*”)

94. **Lafer** also discloses that when a user uses a browser (as shown in Figure 1) to issue a request to the Web server “front-end logic” (*see* [1.b]), the Web server obtains “requested information” by interfacing with the network system (which I discussed with respect to element [1.a]) to pass a tag-extended HTML file to that network system when additional back-end processing is required to form the page that will be returned to the user. Lafer, 2:65-3:22, 3:62-67, 1:15-18. In

response to this request, the network system implements a portion of the “page formation mapping” process involving the network system in which “the tag-extended HTML file” is “parsed” until the system finds a “particular tag”—referring to Lafer’s “special tags”—is “encountered.” Lafer, 3:8-10, 2:65-3:22, 3:62-67. Within this network system, the components of the system that are used as part of the process to parse the HTML file together comprise an HTML parser. *See id.*

(e) Element [1.d]

95. **Element [1.d] recites “parsing said HTML document or browser language on said parser by looking for said proprietary media tags.”** This element is met by Lafer.

96. **Lafer discloses parsing said HTML document or browser language on said parser by looking for said proprietary media tags** (*e.g.*, “the tag extended file is ... parsed” by the parser (*see* [1.c]) to look for a “specialized tag[]” before the “end-of-file is encountered” (*see* [1.b])). *See, e.g.*, Lafer:

- **Figure 2**

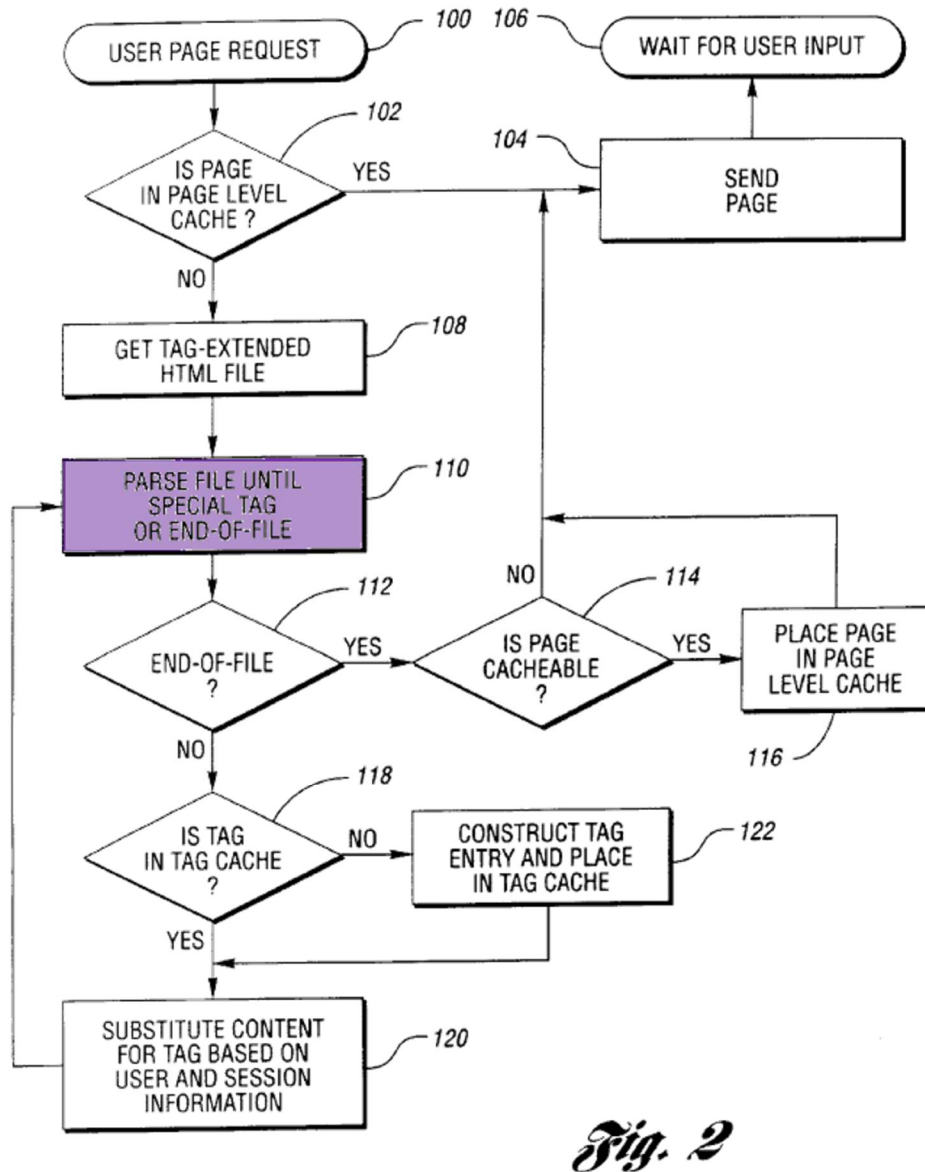


Fig. 2

- **1:54-58** (“In accordance with these and other objects, the present invention provides a method and system for *specifying fragments of HTML text using specialized tags*, holding these fragments in a tag cache, and rapidly rendering personalized HTML pages using the cached fragments.”)

- **3:8-10** (“*A page formation mapping process, as shown in reference 24, provides construction of the HTML pages in accordance with the present invention.*”)
- **3:62-4:7** (“*If the page is not in the cache, the page must be constructed or generated. The generation process begins by obtaining the tag-extended HTML file as shown in block 108. Block 110 shows that the tag-extended file is then parsed until either a particular tag or the end-of-file is encountered. If the end-of-file is encountered, as is shown in block 112, a check is made to determine if the page should be cached as in block 114. If so, block 116 indicates that the page is placed in the page cache. This process can be implemented using any known algorithm or technique available for managing a page cache. Flow then passes to block 104 where the page is sent to the user as previously described.*”)
- **4:10-22** (“More specifically, *a tag in accordance with the present invention does not constitute a part of the standard HTML tag library.* The tag syntax chosen for the preferred embodiment uses a special begin symbol and

end symbol to mark the particular tag as follows:

<! - - CW>	begin symbol
type=typename	tag type attribute
. . .	additional tag attributes
<CW - - >	end symbol
	)

- *See also 4:8-22.*

97. *See* my discussion of elements [1.b]-[1.c]. As I discussed above with respect to element [1.c], Lafer discloses that “the tag-extended [HTML] file is ... parsed until either a particular tag or the end-of-file is encountered.” Lafer, 3:62-67, 1:54-58. The “specialized tags” embedded within the tag-enhanced HTML file do not “constitute a part of the standard HTML tag library.” Lafer, 1:54-58, 4:11-13.

(f) Element [1.e]

98. **Element [1.e] recites “looking up said proprietary media tags in a media tags database on said network system.”** This element is met by Lafer.

99. **Lafer discloses looking up said proprietary media tags in a media tags database on said network system** (*e.g.*, “check[]” for the “specialized tag[]” (*see* [1.b]) in a “tag cache” on the network system (*see* [1.a])). *See, e.g.*, Lafer:

- **Figure 2**

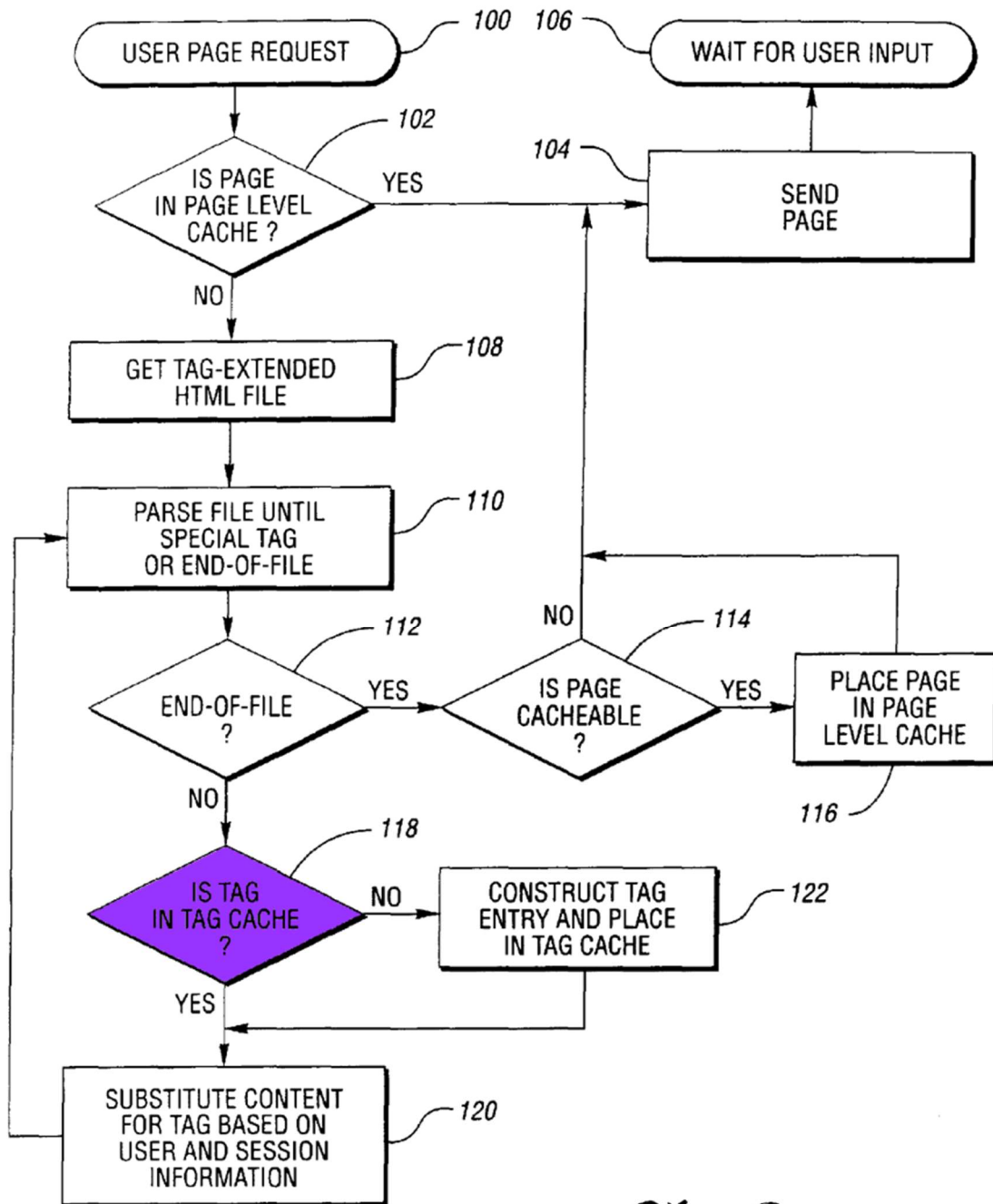


Fig. 2

- **1:54-58** (“In accordance with these and other objects, the present invention provides a method and system for *specifying fragments of HTML text*”

using specialized tags, holding these fragments in a tag cache, and rapidly rendering personalized HTML pages using the cached fragments.”)

- **3:23-42** (“*A tag cache* as shown in reference 28 *contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs* as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, *the tag cache is implemented using a persistent, e.g., disk-based, hash table*. However, as will be appreciated by one of ordinary skill in the art, *other software or hardware methods may be used to implement the tag cache*.”)

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on *the type of tag into which the HTML data will be substituted, the mapping may require content data*, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50*, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The *data for these objects are maintained in respective databases 58, 60, 62 and 64.*”);

- **4:32-35** (“*If a particular tag is encountered prior to the end-of-file, the tag cache is checked as is shown in block 118. If the tag is already associated with contents in the tag cache, the contents are substituted for the tag as in block 120.*”)
- **See also 3:37-43, 4:55-61.**

100. See my discussion of elements [1.a]-[1.b], [1.d]. As discussed for [1.a], Lafer discloses that the network system includes a “tag cache. In addition, Lafer discloses that while a tag-extended HTML file is being parsed, “[i]f a particular tag is encountered prior to the end-of-file, the tag cache is checked.” Lafer, 4:32-35, 1:54-58. “A tag cache ... contains HTML fragments that are substituted for tag values.” Lafer, 3:22-26. Lafer provides that the “tag cache” is implemented using a “hash table” or other “software or hardware methods.” Lafer, 3:22-36. Based on these disclosures, a person having ordinary skill in the art would have understood that Lafer not only renders obvious implementing the “tag cache” as a database, but actually discloses it. For example, a person having ordinary skill in the art would have understood Lafer’s reference to a hash table to describe using a hash table for database indexing, as was commonly implemented in the art.

(g) Element [1.f]

101. **Element [1.f] recites “if at least one media tag of said proprietary media tags is not found, generating a Web media by using said at least one**

media tag.” As I previously discussed in Section VIII.A, elements [1.f]-[1.i] are conditional limitations and [1.f]-[1.i] do not need to be met to render obvious claim 1. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Tso.

102. **Lafer discloses if at least one media tag of said proprietary media tags is not found** (*e.g.*, “[i]f the tag does not exist in the tag cache”), **using said at least one media tag** (*e.g.*, “a tag-content pair is constructed from information in the object cache or databases” based on “tag attributes” from the proprietary media tag (*see* [1.b]) in the tag-extended HTML file indicating “content data from a database” and “user session information”). *See, e.g.*, Lafer:

• **Figure 2**

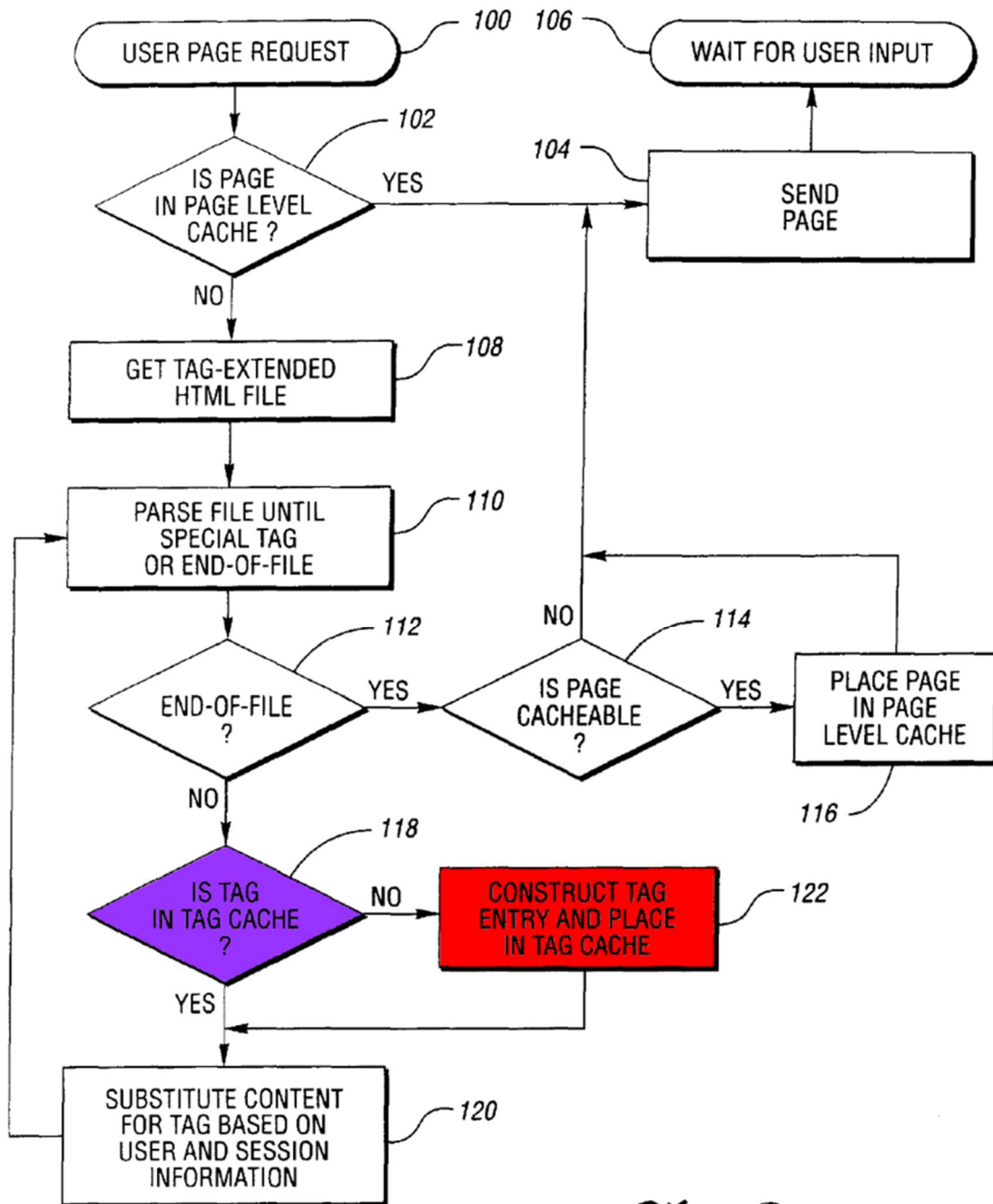


Fig. 2

- 3:23-42 (“A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality

of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, *the tag cache is implemented using a persistent, e.g., disk-based, hash table*. However, as will be appreciated by one of ordinary skill in the art, *other software or hardware methods may be used to implement the tag cache*.

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on *the type of tag into which the HTML data will be substituted, the mapping may require content data*, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50*, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. *The data for these objects are maintained in respective databases 58, 60, 62 and 64.*")

- 4:24-31 ("*The content of a particular tag includes a set of tag attributes*."

One tag attribute determines the tag type. *Other tag attributes have text values that can contain substitution tokens. These tokens are replaced by values depending on the tag type. Possible token replacements include* but

are not limited to *content data from a database*, URLs for predefined forms, *user session information*, and URL attributes and form fields.”)

- **4:46-54** (“*If the tag does not exist in the tag cache, a tag-content pair is constructed from information in the object cache or databases* as shown in block 122. *The use of the object cache shortens database access latency.*”)

The content associated with the current topic is matched with a key formed from information including relevant topic information, content type, and a reference to the HTML page. This page reference may be the base file name or may include all or part of the URL address.”)

- **5:36-39** (“The method of claim 1 further comprising the steps of determining whether *an HTML fragment associated with a particular tag* is currently stored in the tag cache, and if not, *regenerating and restoring the fragment in the tag cache.*”)
- *See also 4:13-31.*

103. **Tso discloses if said media is not found** (*e.g.*, if “no entry exists in server-side cache memory 30 for the requested hypertext object” media), **generating a Web media** (*e.g.*, “retriev[ing]” the media and “transcod[ing] the data stream appropriately”) **using said at least one media tag** (*e.g.*, using reference to “hypertext object[]” embedded in “web page” and “selection criteria” relating to

“client,” “network,” and “proxy” characteristics such that the system can “transmit[] the transcoded data stream to network client 12” appropriately). *See, e.g.,* Tso:

- **6:66-7:3** (“*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.”)
- **7:15-8:9** (“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*. ... 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) ... (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”)

- 8:10-13 (“*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.*”)
- 9:60-65 (“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.*”)
- 10:24-49 (“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. ...

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.”)

- **14:1-7** (“[P]rocessing 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.”)
- **16:39-44** (“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.”)
- **See also 3:43-60, 4:1-5, 4:45-50, 4:62-66, 5:36-42, 6:9-50, 6:64-66, 10:24-32, 14:21-63, 14:37-63.**

104. See my discussion of elements [1.b], [1.d]-[1.e]. **Lafer** also discloses that when a special tag in the tag-extended HTML file “does not exist in the tag cache,” the tag is used to “construct[]” a “tag-content pair” linking a “tag” to an

“HTML fragment.” Lafer, 3:32-39, 4:13-31, 4:46-54. As I previously discussed in Section IX.A.1, it was known by a person having ordinary skill in the art that an “HTML fragment” is a piece of standard HTML, such as an image tag when an image is referenced; this fragment references a piece of “content data” stored in the “Content DB.” Lafer, 3:22-30, Fig. 1. Within each specialized tag, there are “tag attributes” including “substitution tokens” that are used to carry tokens specifying “user session information” and “content data from a database.” Lafer, 4:24-31. For example, as also discussed in §IX.A.1, a person having ordinary skill in the art would have understood and found it obvious that a “fragment” for a special tag associated with image content uses a standard “img” HTML tag to link to an image media file because such “img” tags were standard HTML tags used to insert an image into an HTML page.

105. Lafer leaves it to a person having ordinary skill in the art to choose the “content data from a database” or “user session information” to be specified by a tag. A person having ordinary skill in the art would have identified Tso as providing more detail on this matter. Tso further discloses that a user issues a request for a webpage with embedded hypertext objects. Tso, 9:60-65, 14:1-17. A person having ordinary skill in the art would have understood that hypertext objects are embedded using HTML tags, such as, an “img” tag that is used to embed an image, along with

certain parameters used to specify how the image is presented in the page. For example, as previously discussed, Moore provides an explanation of how “img” tags are used to include image files in a webpage. Moore, 3:41-53, 4:5-13, 5:29-30, 6:59-7:13. The tags within the page cause the processing browser to issue separate requests for hypertext objects, such as embedded image files embedded using “img” tags. Tso, 9:60-65.

106. Tso further teaches that these requests are processed, along with corresponding selection criteria, to identify and transcode content that is responsive to the request. Tso, 10:24-49, 6:64-7:3, 8:10-13. The “combinations” of one or more “selection criteria” included in a request, specify how content is to be transcoded before being returned to the user. Tso, 10:24-49, 6:64-7:3, 8:10-13. Tso provides examples of possible selection criteria including “network characteristics” like “bandwidth,” information about the “network client” and “proxy” server, as well as “user preferences” like “quality/speed tradeoff.” Tso, 7:15-8:9. Based on the predetermined selection criterion or criteria, the parser 22 “dictate[s] which of transcode providers 24 are invoked” to “transcode[] the data stream appropriately.” *Id.*, Tso, 10:24-49. However, as I discussed in Section IX.A.2, **Tso’s** broad teachings do not require a specific technique for specifying or including such criteria in a request.

107. As I further discussed in Section IX.A.2, a person having ordinary skill in the art would have been motivated to modify Lafer's teaching of "user session information" in a "tag" with Tso's more specific selection criteria teachings as the "session information" substitution tokens within **Lafer's** tags within the tag-enhanced HTML page that **Lafer** uses to generate requests for personalized content. **Tso's** selection criteria specifying properties including, for example, "network characteristics," "network client," and "proxy" characteristics describe variables that are a type of "user session information" as taught by **Lafer**. Adding **Tso's** transcoding and caching teachings to **Lafer** by using these substitution tokens to specify selection criteria for transcoding and caching content allows the system to implement **Lafer's** teachings of using cached data to personalize webpages in a manner that has the advantageous effect of allowing personalized transcoding of content for individual users while simultaneously eliminating unnecessary re-transcoding and re-caching of content. *E.g.*, Lafer, 4:24-31; Tso, 7:15-8:9. Modifying Lafer with Tso's teachings therefore yields a system in which the substitution token information of **Lafer's** tags is used to specify how to generate, transcode, and cache Web media using transcoding algorithms as taught by **Tso**. Lafer, 4:24-31.

(h) Element [1.g]

108. Element [1.g] recites “**placing said generated Web media in a media cache on said network system.**” As I previously discussed in Section VIII.A, elements [1.f]-[1.i] are conditional limitations and [1.f]-[1.i] do not need to be met to render obvious claim 1. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer alone and the combined teachings of Lafer and Tso.

109. Lafer discloses placing said generated Web media in a media cache on said network system (*e.g.*, “Content DB” holds content “data” for “content objects”). *See, e.g.*, Lafer:

- **3:23-42** (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache. A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list*”).

data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50*, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. *The data for these objects are maintained in respective databases 58, 60, 62 and 64.*”)

- 4:46-54 (“*If the tag does not exist in the tag cache, a tag-content pair is constructed from information in the object cache or databases* as shown in block 122. *The use of the object cache shortens database access latency.*”)

The content associated with the current topic is matched with a key formed from information including relevant topic information, content type, and a reference to the HTML page. This page reference may be the base file name or may include all or part of the URL address.”)

110. **To the extent additional disclosure of this element is required, Tso further discloses placing said generated Web media in a media cache on said network system** (*e.g.*, “caching of an original version of the retrieved hypertext object” and “transcoded versions of content” in “cache” on the network system (*see* [1.a])). *See, e.g.*, Tso:

• **Figure 8**

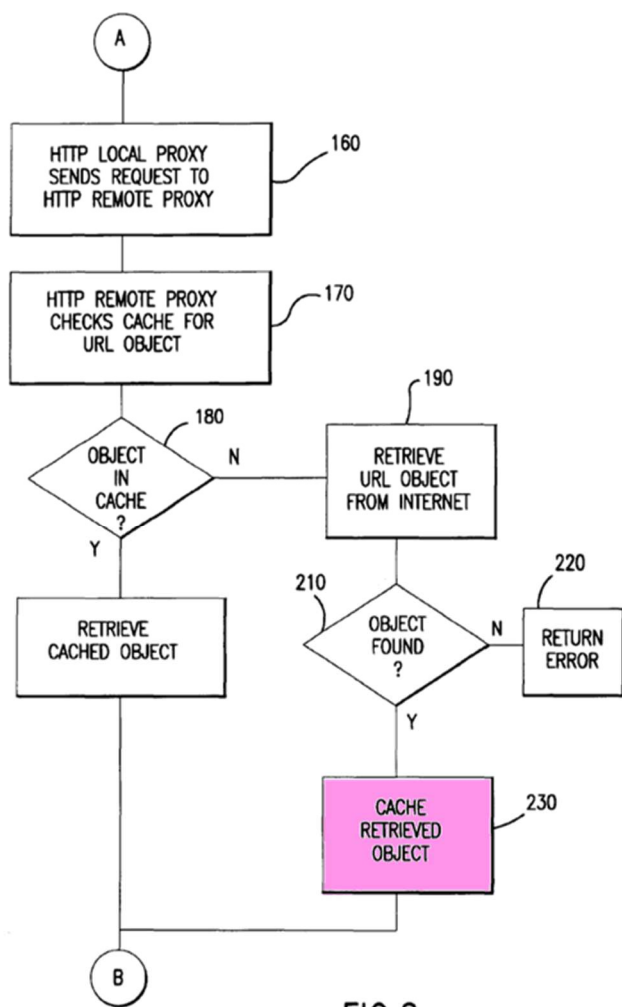


FIG.8

- **Figure 9**

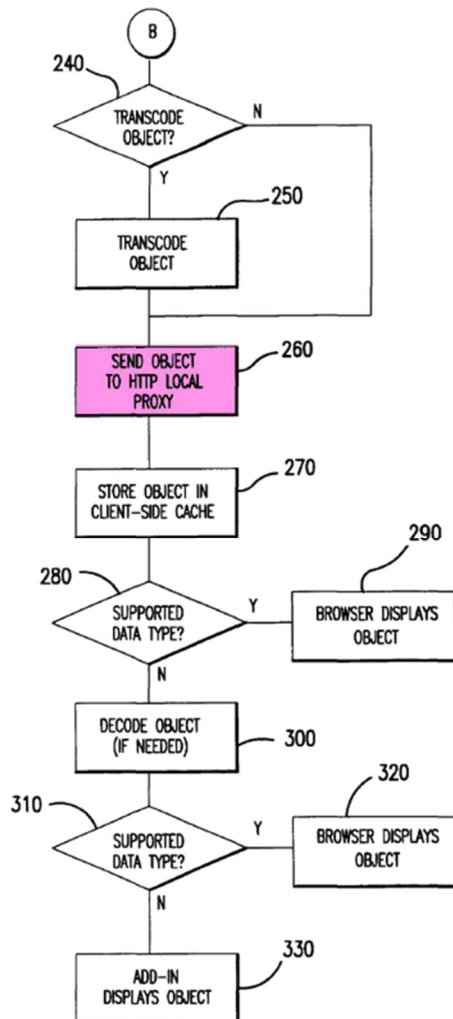


FIG.9

- **4:1-5** (“*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.*”)

- **4:62-5:7** (“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.”)
- **14:1-7** (“[P]rocessing 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.”)
- **See also 4:38-48, 6:9-61, 7:15-65, 14:37-46.**

111. Lafer teaches that “content objects” are stored in an “object cache.” Lafer, 3:37-43. “The data for these objects are maintained in respective databases including a “Content DB.” Lafer, 3:37-43. Lafer also teaches that this content “is constructed from information in ... databases” that “maintain[]” “data for these [content] objects.” *Id.*, Lafer, 4:46-54. As I previously discussed for element [1.a], a person having ordinary skill in the art would have understood that the underlying “data” for content stored in the “content DB” is “media,” including media generated as discussed in [1.f]. The generated content is cached in the “content DB.” Lafer, 3:23-42, 4:46-54.

112. Tso discloses using “[s]erver-side cache memory 30 ... to store original and transcoded versions of content for later transmission to network client 12.” Tso, 4:1-5. Among these stored content versions are “original version[s] ... retrieved” in response to a “determin[ation] that the requested hypertext object does not exist in the server-side cache memory 30.” Tso, 14:1-7, 14:37-46. Such hypertext objects include displayed images. *Id.* As discussed above for element [1.f] and Section IX.A.2, a person having ordinary skill in the art would have been motivated to modify Lafer with this teaching to reduce the re-transcoding of data such that both “original” and “transcoded” versions of content are stored in the “Content DB.” Tso, 4:1-5.

(i) Element [1.h]

113. **Element [1.h] recites “converting said at least one media tag to at least one standard HTML equivalent tag that refers to said media in said cache.”** As I previously discussed in Section VIII.A, elements [1.f]-[1.i] are conditional limitations and [1.f]-[1.i] do not need to be met to render obvious claim 1. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

114. **Lafer discloses converting said at least one media tag to at least one standard HTML equivalent tag that refers to said media in said cache** (*e.g.*, “a tag-content pair is constructed from information in the object cache” that “map[s]

HTML fragments [including standard HTML tags] into the tag” to allow “HTML data” to be “substituted” for the tag). *See, e.g., Lafer:*

- **Figure 2**

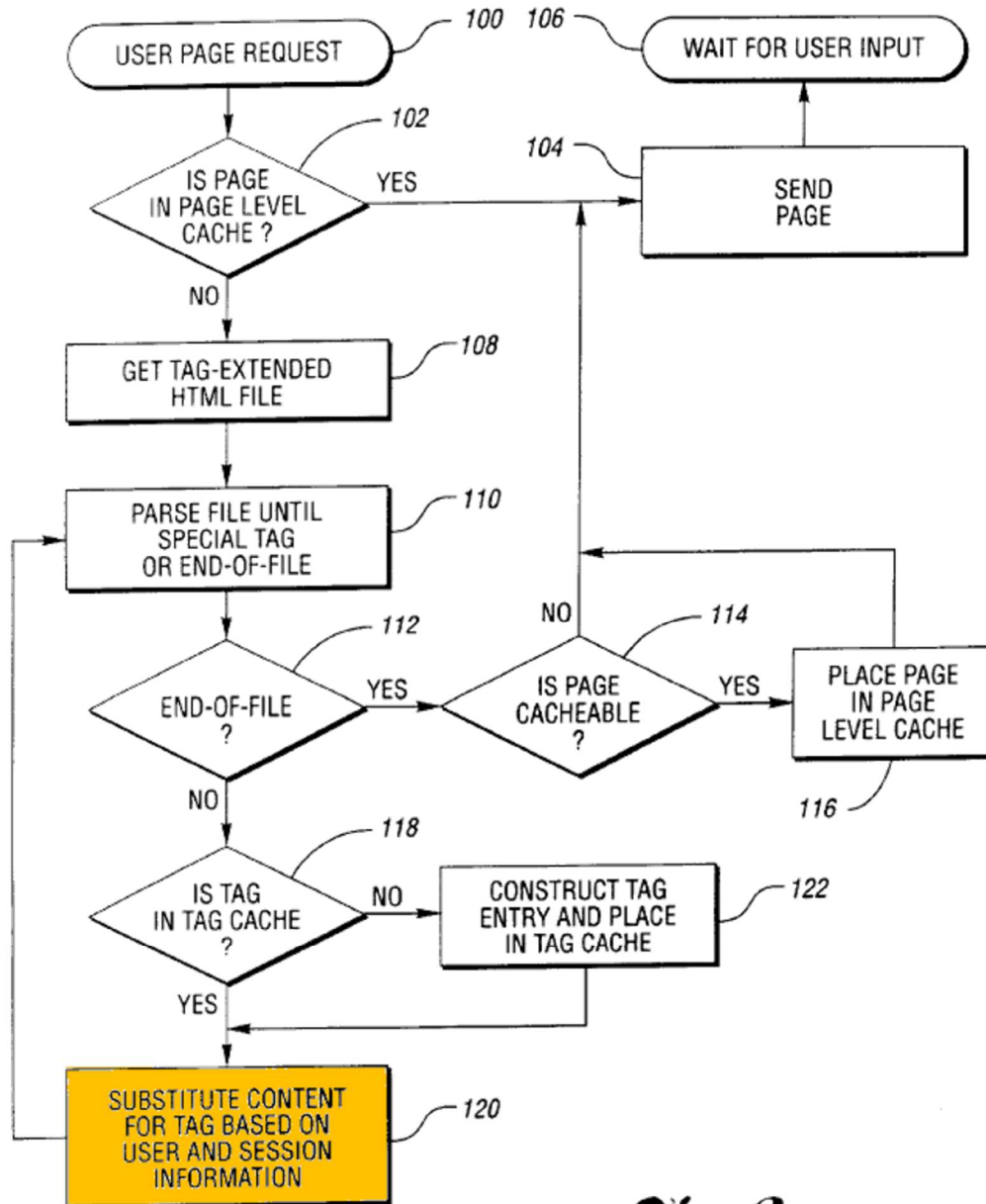


Fig. 2

- **Abstract** (“A method and system for web site construction using HTML fragment caching includes embedding particular tags in a stored HTML page, and when a request for distribution of the HTML page is received from a user, *replacing each tag with content information in the form of HTML fragments previously created based on user preference information.*”)
- **3:23-43** (“*A tag cache* as shown in reference 28 *contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs* as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, *the tag cache is implemented using a persistent, e.g., disk-based, hash table.* However, as will be appreciated by one of ordinary skill in the art, *other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds*

content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The *data for these objects are maintained in respective databases 58, 60, 62 and 64.*”)

- **4:46-54** (“*If the tag does not exist in the tag cache, a tag-content pair is constructed from information in the object cache* or databases as shown in block 122. The use of the object cache shortens database access latency. The content associated with the current topic is matched with a key formed from information including relevant topic information, content type, and a reference to the HTML page. This page reference may be the base file name or may include all or part of the URL address.”)
- **See also 1:49-2:3, 3:22-25, 4:32-35.**

115. See my discussions of elements [1.d]-[1.e] and [1.g]. As discussed for [1.e], Lafer discloses that each particular disclosed “specialized tag,” which is not “part of the standard HTML tag library,” is associated in a “tag-content pair” with a standard “HTML fragment” that refers to corresponding “content data” in the “Content DB.” Lafer, 1:54-58, 4:11-13, 3:32-43, Figure 1.

116. As a result, the “specialized tags” in the file are replaced with “content information in the form of HTML fragments,” which include standard HTML tags particularly when referencing an image (as I further discussed in Section IX.A.1),

by “substitut[ing]” the special tag with the equivalent HTML fragment/tag when a tag-extended HTML file is parsed in response to a user request for the Web page to which the HTML file corresponds. Lafer, 1:54-58, Abstract, Figure 2. This process of replacing the specialized tags with HTML fragments describes, and at a minimum renders obvious, converting such tags in the page into standard HTML equivalent tags.

(j) Element [1.i]

117. **Element [1.i] recites “storing said at least one media tag and said at least one standard HTML equivalent tag in said database.”** As I previously discussed in Section VIII.A, elements [1.f]-[1.i] are conditional limitations and [1.f]-[1.i] do not need to be met to render obvious claim 1. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

118. **Lafer discloses storing said at least one media tag and said at least one standard HTML equivalent tag in said database** (*e.g.*, “tag-content pair,” linking the “specialized tag” and “HTML fragment,” is stored in “tag cache”). *See, e.g.*, Lafer:

- **Figure 2**

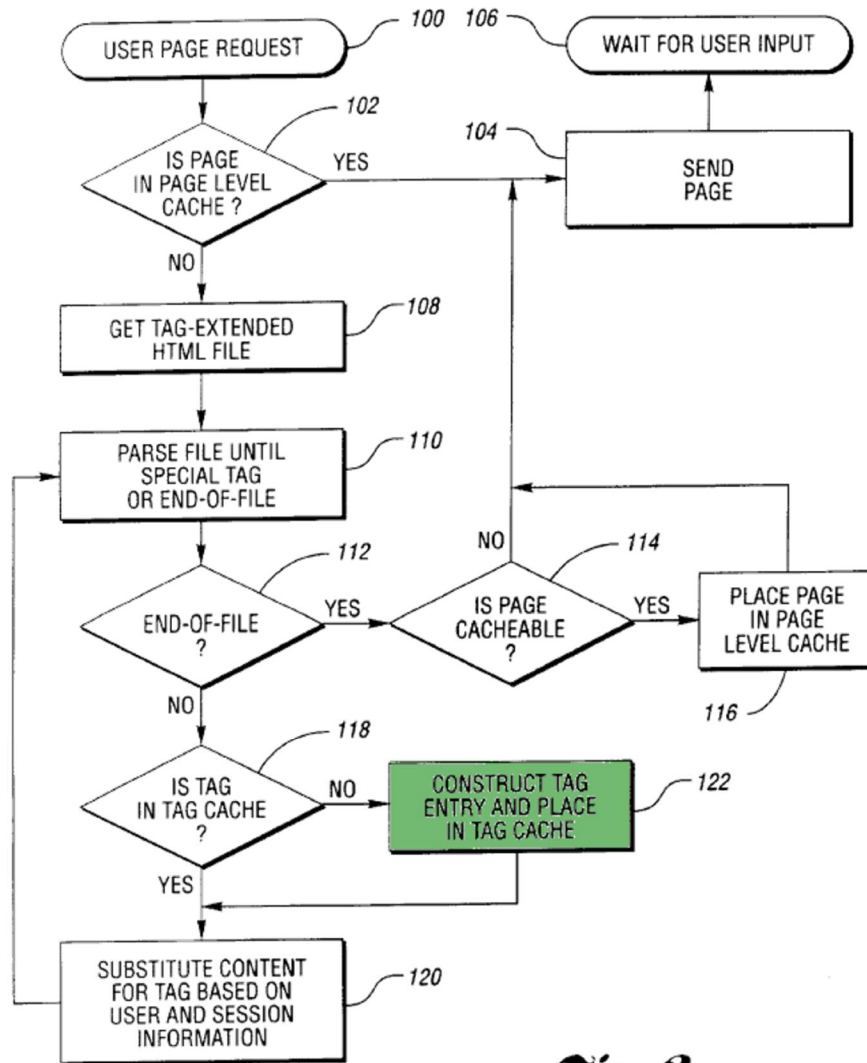


Fig. 2

- 4:46-54 (“If the tag does not exist in the tag cache, a tag-content pair is constructed from information in the object cache or databases as shown in block 122. The use of the object cache shortens database access latency. The content associated with the current topic is matched with a key formed from information including relevant topic information, content type, and a

reference to the HTML page. This page reference may be the base file name or may include all or part of the URL address.”)

- **3:23-42** (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache. A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data. More specifically, HTML fragments mapped into the tag cache are fetched from an object cache referenced as block 48. The object cache holds content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The data for these objects are maintained in respective databases 58, 60, 62 and 64.*”);”)
- **See also 5:36-39.**

119. *See* my discussion of elements [1.b] and [1.e]-[1.h]. As discussed for element [1.h], Lafer also discloses a “tag cache” that stores “a plurality of tag-content pairs,” which are “constructed from information in the object cache” and help the system “to map HTML fragments into the tag.” Lafer, 3:23-42, 4:46-54, 3:32-39. As discussed for element [1.f], the “tag” is a “specialized tag” that “does not constitute a part of the standard HTML tag library,” while an “HTML fragment” not only renders obvious a standard HTML equivalent tag that refers to content in the “Content DB,” but actually discloses it. Lafer, 1:54-59, 4:11-13, 3:22-30. A person having ordinary skill in the art would have understood that the constructed “tag-content pair” is stored in the tag cache to enable the system to “rapidly render[]” HTML pages with the same tag in response to future requests. Lafer, 4:46-54, 1:54-59.

(k) Element [1.j]

120. **Element [1.j] recites “modifying said HTML document or browser language by replacing said at least one media tag by said at least one standard HTML equivalent tag.”** This element is met by Lafer.

121. **Lafer discloses modifying said HTML document or browser language by replacing said at least one media tag by said at least one standard HTML equivalent tag** (*e.g.*, “substituting ... predetermined HTML fragments for

each particular tag embedded in a requested HTML file based on the stored user preference information”). *See, e.g., Lafer:*

- **Figure 2**

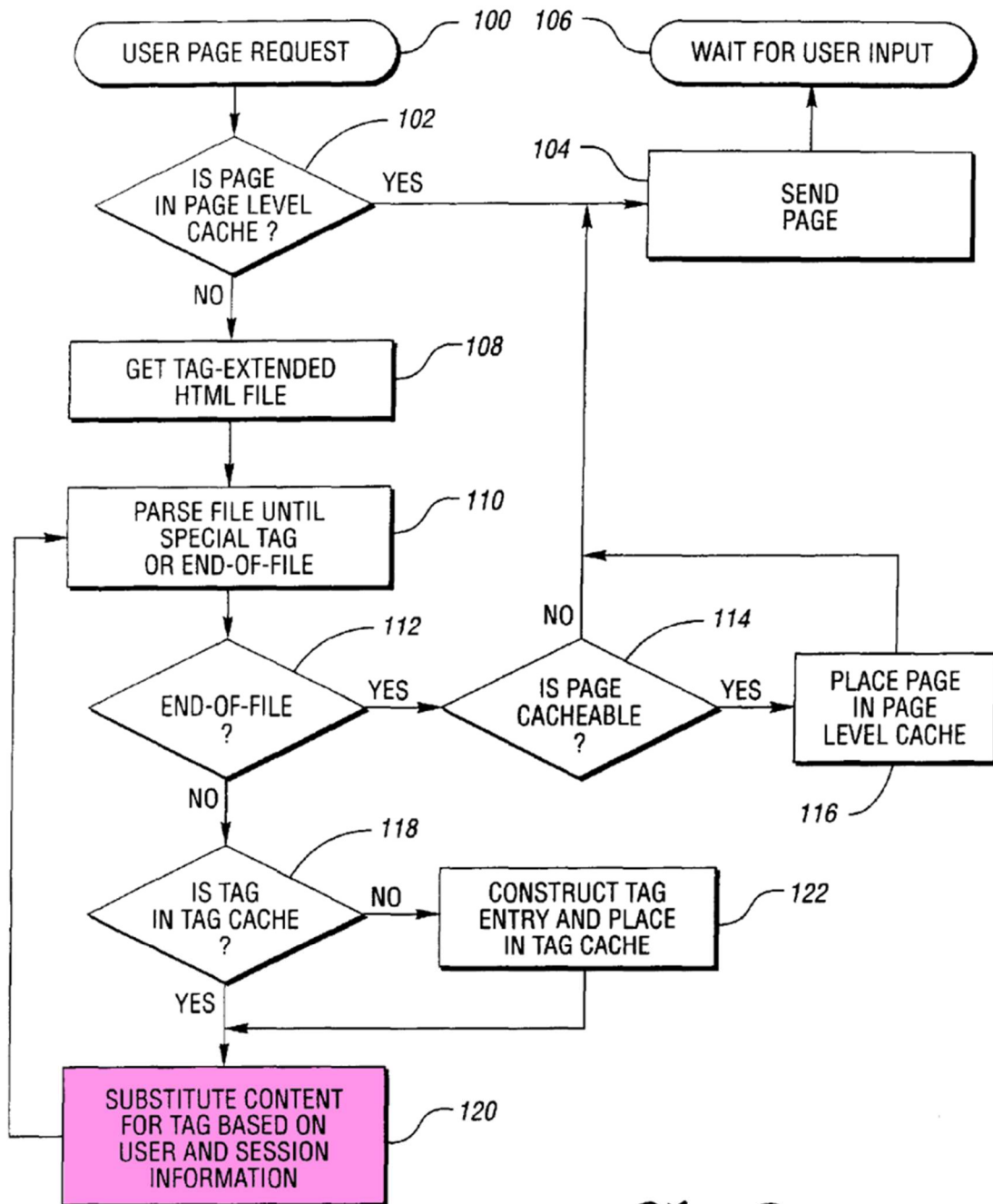


Fig. 2

- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.*”)
- **4:8-13** (“*If, while parsing the file and before the end-of-file is reached, a particular tag is encountered, the tag is replaced with one or more HTML fragments associated with the tag.* More specifically, a tag in accordance

with the present invention does not constitute a part of the standard HTML tag library.”)

- *See also* 4:32-45, 4:55-61.

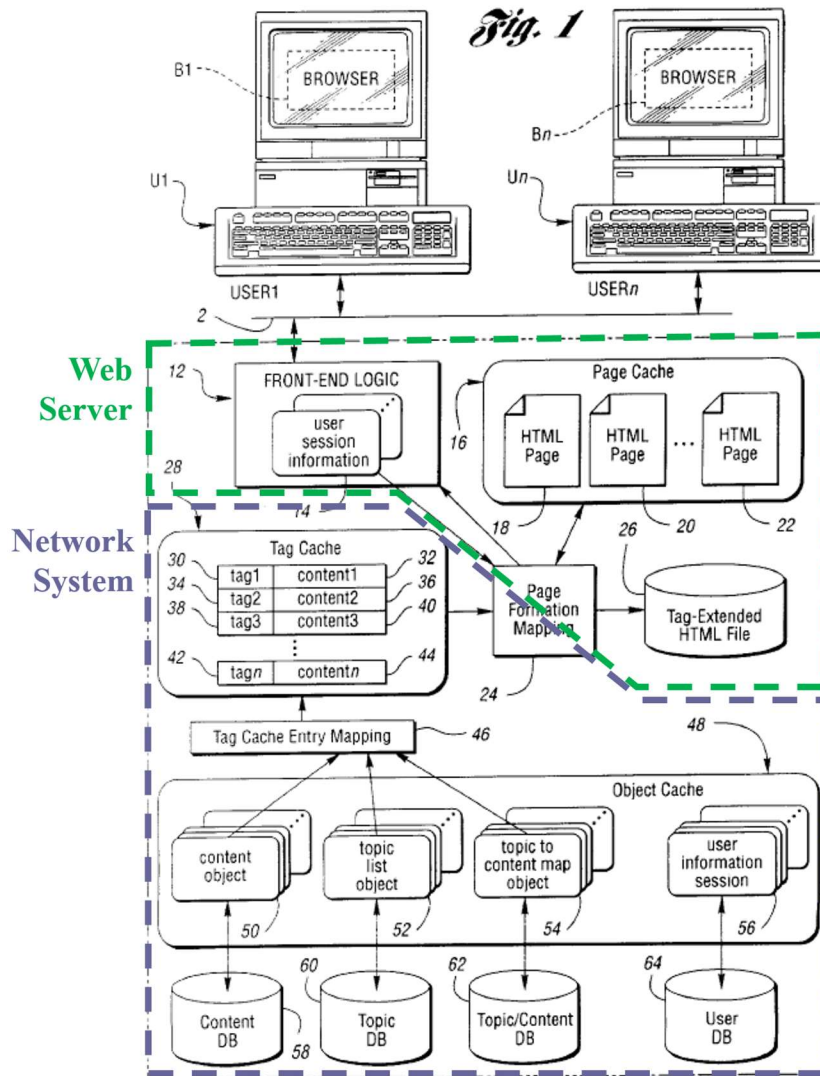
122. *See* my discussion of elements [1.d]-[1.e] and [1.h]-[1.i]. As discussed for elements [1.e] and [1.h]-[1.i], Lafer also discloses “substituting one or more predetermined HTML fragments” (standard HTML equivalent tags) “for each particular tag” “embedded in a requested HTML file” to “construct[]” personalized HTML pages.

(l) Element [1.k]

123. **Element [1.k] recites “delivering said modified HTML document or browser language from said network system to said Web server.”** This element is met by Lafer.

124. **Lafer discloses delivering said modified HTML document or browser language from said network system to said Web server** (*e.g.*, the network system (*see* [1.a]) returns the processed “page” to the web server (*see* [1.b]) after “end-of-file is encountered” and the “page is placed in the page cache”). *See, e.g.*, Lafer:

• **Figure 1**



- **Figure 2**

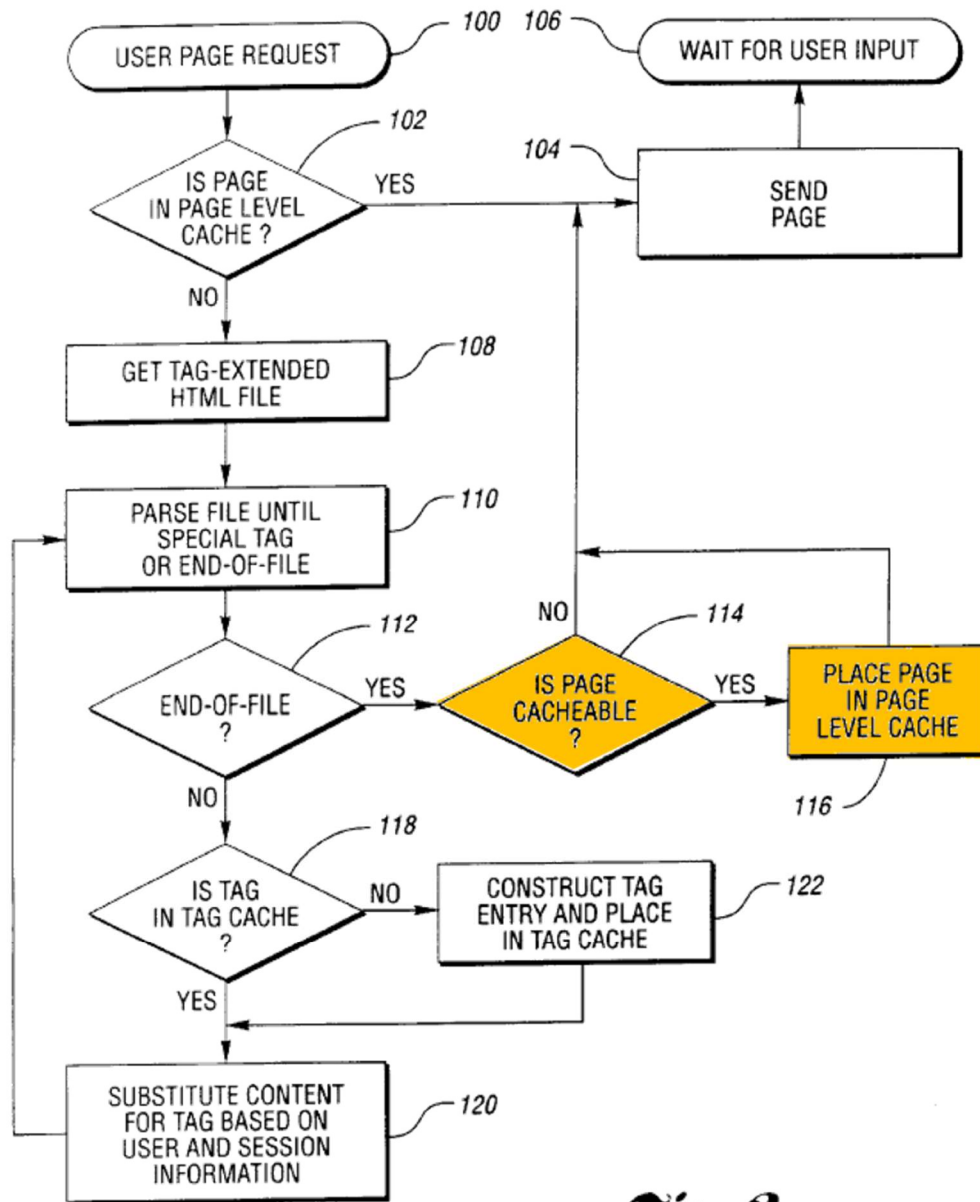


Fig. 2

- **3:62-4:7** (“*If the page is not in the cache, the page must be constructed or generated. The generation process begins by obtaining the tag-extended HTML file* as shown in block 108. Block 110 shows that *the tag-extended*

file is then parsed until either a particular tag or the end-of-file is encountered. If the end-of-file is encountered, as is shown in block 112, a check is made to determine if the page should be cached as in block 114. If so, *block 116 indicates that the page is placed in the page cache.* This process can be implemented using any known algorithm or technique available for managing a page cache. Flow then passes to block 104 where *the page is sent to the user* as previously described.”)

- *See also 2:50-55, 2:65-67, 3:8-10, 4:3-7.*

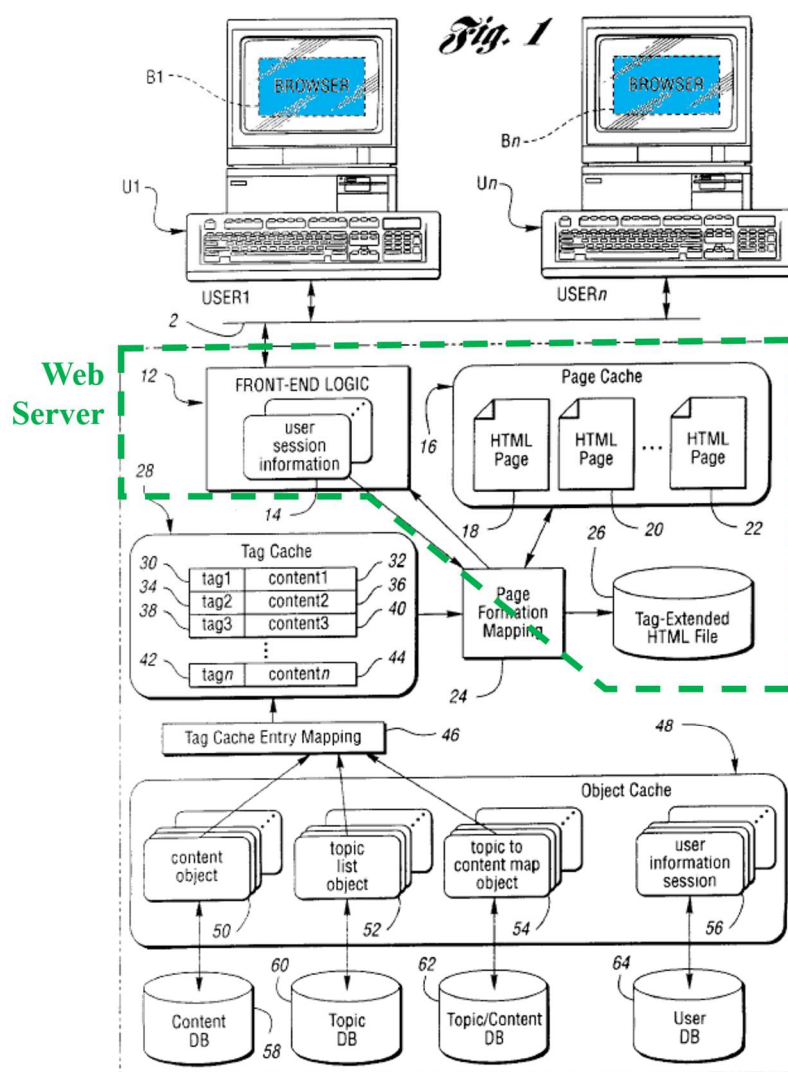
125. See my discussion of elements [1.a]-[1.b] and [1.j]. Lafer also discloses that “a check is made to determine if the page should be cached” once the modified HTML document is generated as discussed for [1.j] as part of the “page formation mapping process” 24. Lafer, 3:62-4:5. If the system decides that the HTML page is “likely to be used again in the future,” “the page is placed in the page cache” to “reduce[] latency” in fulfilling future requests. *Id.*, 3:10-14 (“A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.”). As I discussed for element [1.b], the “page cache” is part of the Web server.

(m) Element [1.l]

126. **Element [1.l] recites “delivering said modified HTML document or browser language from said Web server to said customer browser for said customer to view.”** This element is met by Lafer.

127. **Lafer discloses delivering said modified HTML document or browser language from said Web server to said customer browser for said customer to view** (*e.g.*, the Web server (*see* [1.b]) sends the “page” to a customer’s “browser” (*see* [1.c]) used to “view the HTML pages”). *See, e.g.*, Lafer:

• Figure 1



• **Figure 2**

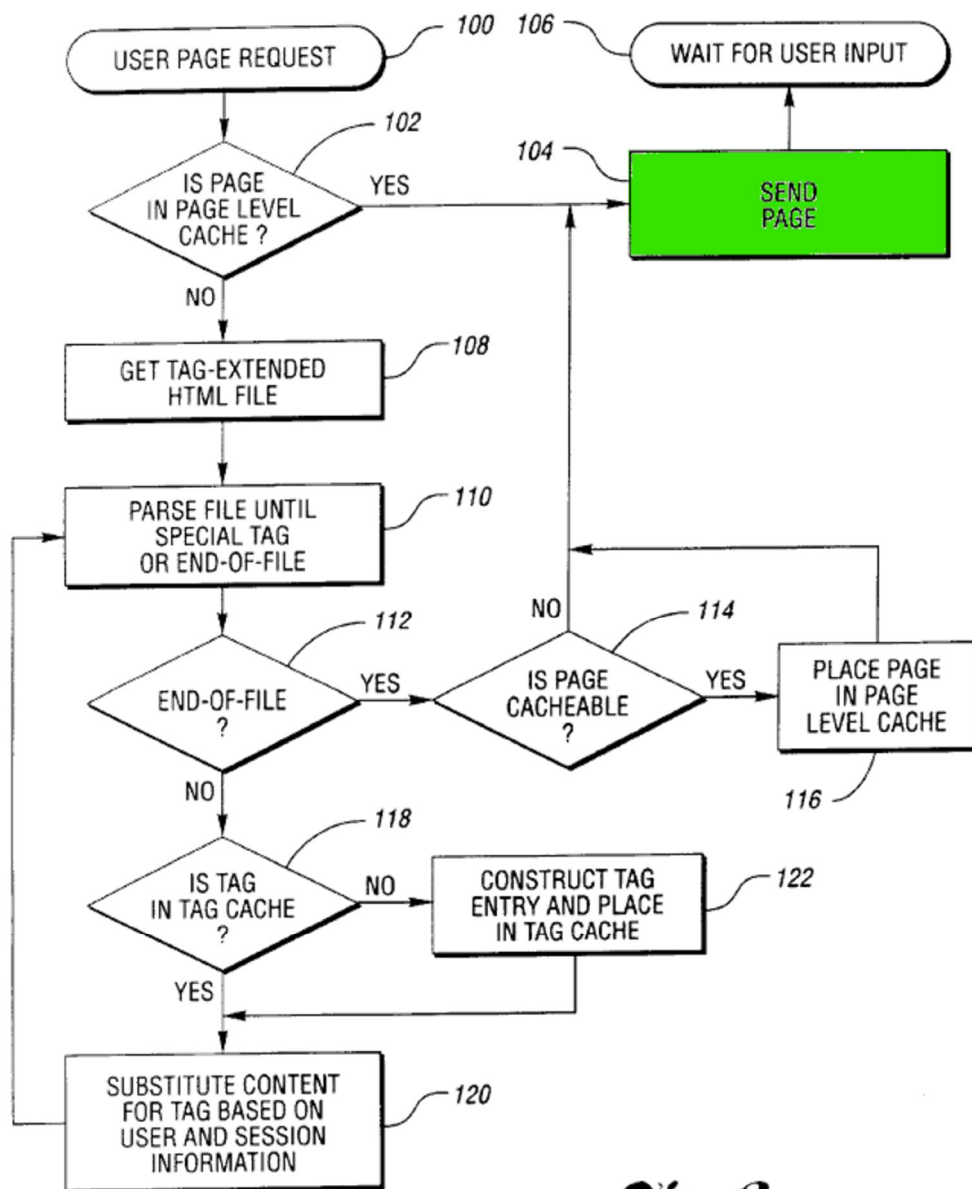


Fig. 2

- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests,*

and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.”)*

- 2:62-64 (“*Each user may view the HTML pages using a conventional Internet browser* as represented by B1 through Bn.”)
- 2:65-3:22 (“Considering the server 10 in detail, *front-end logic is responsible for interfacing with users U1 to Un through the communication medium 2*. As represented by block 12, *the front-end logic can be implemented as a standard web server*. The front-end logic services the user by returning requested information in the form of HTML pages. *The*

front-end logic generally operates to collect session information as shown in block 14. This information can be obtained, for example, through the use of HTML forms as is commonly done with web pages.

A page formation mapping process, as shown in reference 24, provides construction of the HTML pages in accordance with the present invention.

A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.

A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic substitution of content for tag values based on stored user preferences and session information. The storage device may be implemented as a directory system or as a database.”)

- **3:67-4:7** (“If the end-of-file is encountered, as is shown in block 112, a check is made to determine if the page should be cached as in block 114. If so, *block 116 indicates that the page is placed in the page cache.* This process can be implemented using any known algorithm or technique

available for managing a page cache. This process can be implemented using any known algorithm or technique available for managing a page cache.

Flow then passes to block 104 where *the page is sent to the user* as previously described.”)

128. See my discussion of elements [1.j]-[1.k]. Lafer also discloses that “the server processor ... distribute[s] the constructed HTML files to the user” via “front-end logic” that “interfac[es] with users U1 to Un through the communication medium 2” after the page has been cached. Lafer, 3:62-4:5, 2:32-33, 2:65-67. As discussed for [1.b], the “front-end logic” is part of the Web server. The user views the HTML page using a standard “browser.” Lafer, 2:62-64, Figure 1.

B. Ground 2: Lafer in View of Tso and Cao (Claim 9)

1. Overview of Cao

129. The Cao reference is an IEEE publication titled “Maintaining Strong Cache Consistency in the World Wide Web” written by Pie Cao and Chengjie Liu. Cao teaches several approaches of maintaining “cache consistency” such that “cached copies” are “updated when the originals change.” Cao, 445. Cao focuses on approaches that provide “strong consistency” in the cache, meaning that such approaches will ensure that “no stale copy of a modified document will ever be returned to the user.” Cao, 445.

130. Cao highlights two approaches to maintaining “strong consistency” of a cache: a “polling-every-time” approach and an “invalidation” approach. Cao, 448-49, 455. In both of these approaches, the system uses “if-modified-since” requests to verify cache consistency. Namely, in the “polling-every-time” consistency method, the system sends the “if-modified-since” request each time a user requests cached content, while the “invalidation” method sends these same “if-modified-since” requests when handling user requests, for example, when the system needs to handle cache server failures, or to avoid scalability issues that arise out of using “if-modified-since” requests for every request, as is implemented in “polling-every-time” consistency methods. Cao, 448-49, 455. Each of these cache consistency methods is used frequently in networked systems. *See, e.g.*, Cao, 446-447, 454-455. Indeed, this “if-modified-since” request is a well-understood, standard HTTP request that was specified as part of the HTTP protocol. *See* Ex. 1017, Fielding et al., *Hypertext Transfer Protocol – HTTP/1.1*, RFC 2068, Jan. 1997, at 118-19.

131. Using these disclosed consistency methods, the system “sends an ‘if-modified-since’ request” when “a request for a document hits in the cache.” Cao, 445, 447. This request prompts the server that maintains the most current version of the asset to “check[] whether the document has been modified since the timestamp.” Cao, 447. The timestamp indicates “when the document was last seen modified;”

that is, it indicates the last time the cached entry was updated. Cao, 449. If, in response to this ‘if-modified-since’ request, “the server returns the code ‘304,’ which stands for ‘document unmodified,’” “the cached copy is returned to the user” without further processing. Cao, 447, 449. Conversely, if the document has been updated since the timestamp, “the server returns the status code ‘200’ and the new data” as an updated form of “the document.” Cao, 447, 449. In this case, “the cached copy is deleted, [and] the new copy is put in the cache and returned to the user.” Cao, 449.

2. Motivation to Modify Lafer (in view of Tso) with Cao

132. Like Lafer and Tso, Cao is analogous prior art and in the same field as the ’575 patent, networked media caching and delivery, and is reasonably pertinent to the alleged problems identified in the ’575 patent of updating original assets that are stored in cache. ’575 patent, 4:57-60 (“original assets need to be updated”), 3:59-60, 13:49-58, 14:9-13; Lafer, 3:37-43 (“objects are maintained in respective databases”); Tso, 7:20-8:9 (“predetermined selection criterion may comprise ... cache restriction of prioritization parameters such as update frequency or replacement preferences”); Cao, 445 (“cache consistency must be maintained, that is, cached copies should be updated when the originals change”), 446-49.

133. While both Lafer and Tso disclose caching media objects, Cao provides details about how to maintain cache consistency such that the cached objects remain up-to-date—a requirement “[f]or Web caches to be useful.” Cao, 445. For example,

while Lafer broadly teaches “maintain[ing]” a database” and Tso similarly teaches subjecting the cache to “prioritization parameters such as update frequency or replacement preferences,” Cao provides specific details about implementing processes to maintain “strong” cache consistency to ensure a request is never fulfilled with a “stale” document. Lafer, 3:37-43; Tso, 7:20-8:9; Cao, 445, 449. For multiple independent reasons, a person having ordinary skill in the art would have been motivated to modify both Lafer’s and Tso’s teachings of content requests using specialized tags, with Cao’s teachings of using an “if-modified-since” request to verify that the original content requested by a user has not changed at the server. Cao, 447. Applying Cao’s teachings, when receiving a request for a cached item based on a specialized tag as taught by Lafer (and modified by Tso), upon receiving a cache hit for a tag-content pair, the system issues an if-modified-since request to verify that the original version of the cached content is unmodified. If the original version of the content has changed, the system deletes the outdated cached information, which includes the tag-content pair because the content is being deleted, and replaces it with an updated version of the information, which is a new tag-content pair. *Id.* A person having ordinary skill in the art would have been motivated to modify Lafer and Tso with Cao’s teachings for at least the following independent reasons.

134. **First**, Cao discloses the important advantages that applying its teachings to maintain cache consistency would have for caching systems such as those disclosed in Lafer and Tso to address networked content distribution in response to challenges that arose from “[the] exploding popularity of the World Wide Web,” including ensuring that only up-to-date content is served. Cao, 445. Lafer discloses that its system was developed to accommodate the “dramatic growth in the amount of information which can be distributed from a Web server to a user.” Lafer, 1:13-18. Tso similarly discloses the utility of caches in eliminating the “need to re-retrieve content from Internet 18,” and considering “network characteristics,” such as “latency, bandwidth and/or error rates,” when determining how to transcode content. Tso, 4:1-5, 7:33-43. Cao teaches its disclosed cache consistency methods “were developed to accommodate challenges that arise from the “Web continu[ing] to explode in size”: namely that the “increase of network load has led to increased latency in accessing Web documents” and that “[f]or Web caches to be useful, cache consistency must be maintained.” Cao, 445. Cao teaches that its disclosed strong cache consistency methods would advantageously address both challenges by improving “both network traffic and document access latency,” and also ensuring that “no stale copy of the modified document will ever be returned to the user.” *Id.*

135. **Second**, Lafer and Tso both recognize the importance of maintaining cache consistency, but leave it to a person having ordinary skill in the art to determine how to maintain cache consistency. Lafer broadly states that its system databases are “maintained.” Lafer, 3:37-43. Tso similarly discloses that selection criterion that dictate how a request for content is processed might specify “cache restriction or prioritization parameters such as update frequency or replacement preferences” without providing any specific cache consistency protocol that uses these parameters to ensure cache consistency. Tso, 7:60-65. A person having ordinary skill in the art would have understood that Tso’s and Lafer’s goals of maintaining the cache in a manner that avoids providing users with out-of-date content and reliably fulfilling content requests by utilizing caches would be frustrated if cache consistency is not maintained because “the lack of strong consistency reduces the effectiveness of client caches as a way to improve the scalability of the Web.” Cao, 445. A person having ordinary skill in the art therefore would have been motivated to consider Cao’s teachings of methods to maintain strong cache consistency and ensure content stored in cache has not been modified prior to returning the cached content in response to a user request.

136. **Third**, a person having ordinary skill in the art would have understood and found it obvious that when content is modified, the tag in the cache referencing

that content would itself also be removed because the content associated with the tag-content pair has changed. Lafer discloses that each tag stores “attributes” such as “content data” for the associated content, which are used to generate the tag-content pair. Lafer, 3:22-30, 4:8-31. Cao discloses that when content is modified, the information about that content, including the “content data” attributes, is no longer valid and therefore is deleted. Cao, 449. Indeed, Cao equates deleting the “cached copy” of content with deleting an “entry” for the content, and also teaches deleting “all information” when a document is invalidated. Cao, 449, 456. Indeed, when maintaining a database, it is common to delete entries that need to be updated and replace them with new entries to avoid the risk of artifacts from old entries causing instability within the database. A person having ordinary skill in the art therefore would have been motivated to modify **Lafer** with **Cao’s** teachings by “deleting” both the content, and the tag-content pair, when content changes, ensuring that tag values in the “tag cache,” including content data associated the tag-content pairs, are up-to-date. Lafer, 3:22-30; Cao, 449. This would have been easy to implement since, as I discussed above in for element [1.e] (section IX.A.3(f)), Lafer provides that the “tag cache” is implemented using a “hash table” or other “software or hardware methods.” Lafer, 3:22-36. Based on these disclosures, a person having ordinary skill in the art would have understood that Lafer not only renders obvious

implementing the “tag cache” as a database, but actually discloses it. Based on this disclosure, a person having ordinary skill in the art would have known to employ a hash table in conjunction with known methods to keep the contents of a hash table up-to-date. For example, Chaman L. Sabharwal and Sanjiv K. Bhatia, *Image databases and near-perfect hash table*, 30 PATTERN RECOGNITION 1867 (November 19, 1997) teaches methods of inserting and deleting information from a hash table used to implement an image database. Ex. 1019. It was well understood that old entries in a hash table would need to be removed when the content stored in the hash table is updated because the hash value of a file is generated based upon the content of the file itself such that a file that is modified will have a different hash value. Indeed, Sabharwal and Bhatia provided methods to improve the efficiency of such updates. *Id.* at 1867-68, 1875.

137. In light of the above teachings, a person having ordinary skill in the art would have had a reasonable expectation of success in applying Cao’s teachings of using “if-modified-since” requests to Lafer’s object cache by issuing such requests when a user requests content to maintain strong cache consistency. This ensures that, if the cached version of original content requested by a user is out-of-date, that the user is not served with that out-of-date content. Indeed, Lafer and Tso both teach the advantages of maintaining objects in databases, and Cao discloses that its

methods were already successfully implemented in different file systems and web browsers. Lafer, 3:37-43 (“objects are maintained in respective databases”); Tso, 7:60-65 (“update frequency or replacement preferences”); Cao, 445-47 (similar teachings implemented in, for example, “Netscape browser,” “Sprite” file systems, “callbacks in AFS”). Indeed, persons of ordinary skill in the art continued to use similar techniques for managing cache consistency, as reflected in U.S. Patent No. 5,581,704, which names Barbará as the first named inventor, a copy of which I understand has been designated as Exhibit 1016. Barbará teaches that when cache entries are out of date, they are marked as “invalid” such that the system is aware that the cache entries must be replaced. Barbará, 1:15-50, 4:7-37. And as discussed earlier, the HTTP protocol itself specifies Cao’s “if-modified-since” calls. Ex. 1017, 118-119. The resulting combination yields a system in which, when the system verifies that the original asset has *not* changed, the cached version of the tag is used to retrieve the associated HTML fragment, which a person having ordinary skill in the art would have understood and found obvious is an HTML equivalent tag, and place this equivalent tag in the web page. *Id.* Applying Cao’s teachings to implement Lafer would therefore have been nothing more than a straightforward application of these teachings in a manner that beneficially improves the overall efficacy of the system.

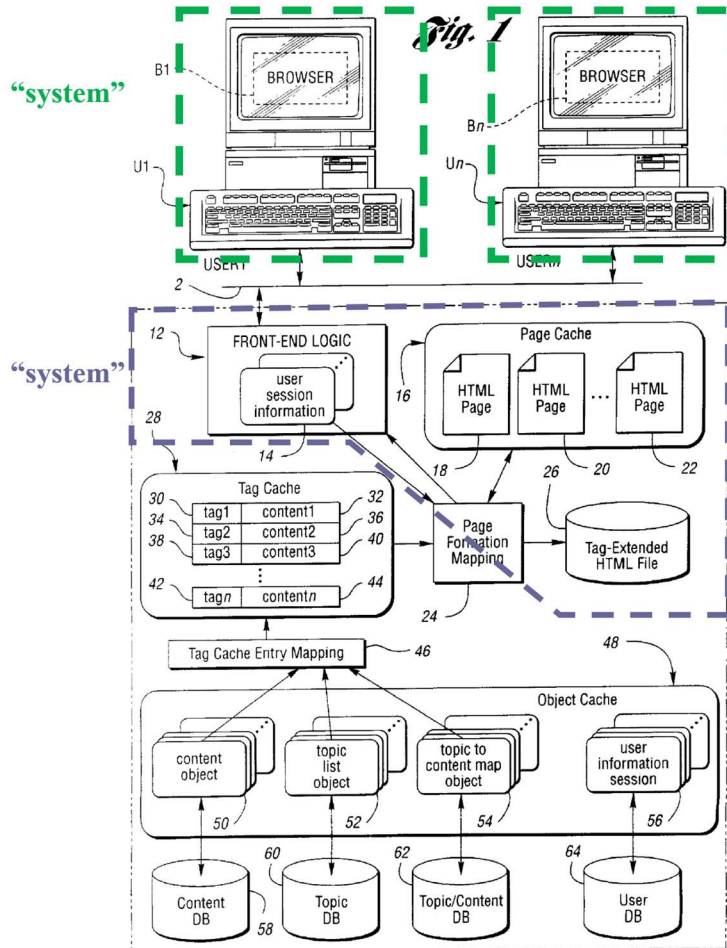
3. Invalidity of Claim 9 Over Lafer in View of Tso and Cao

(a) Element [9.pre]

138. Element [9.pre] recites “**A process for creating a media associated with an HTML equivalent tag and associated with a media tag for a system, comprising the steps of.**” This element is met by the combined teachings of Lafer and Tso.

139. Lafer discloses a process for creating a media associated with an **HTML equivalent tag** (*e.g.*, a process for creating “content data” associated with an “HTML fragment” including standard HTML tags (*see* [1.h])) **and associated with a media tag** (*e.g.*, associated, through a “tag-content pair,” with a media tag (*see* [1.b])) **for a system** (*e.g.*, for either a “user” computer or a system including “page formation mapping” components). *See, e.g.*, Lafer:

• Figure 1



- **Figure 2**

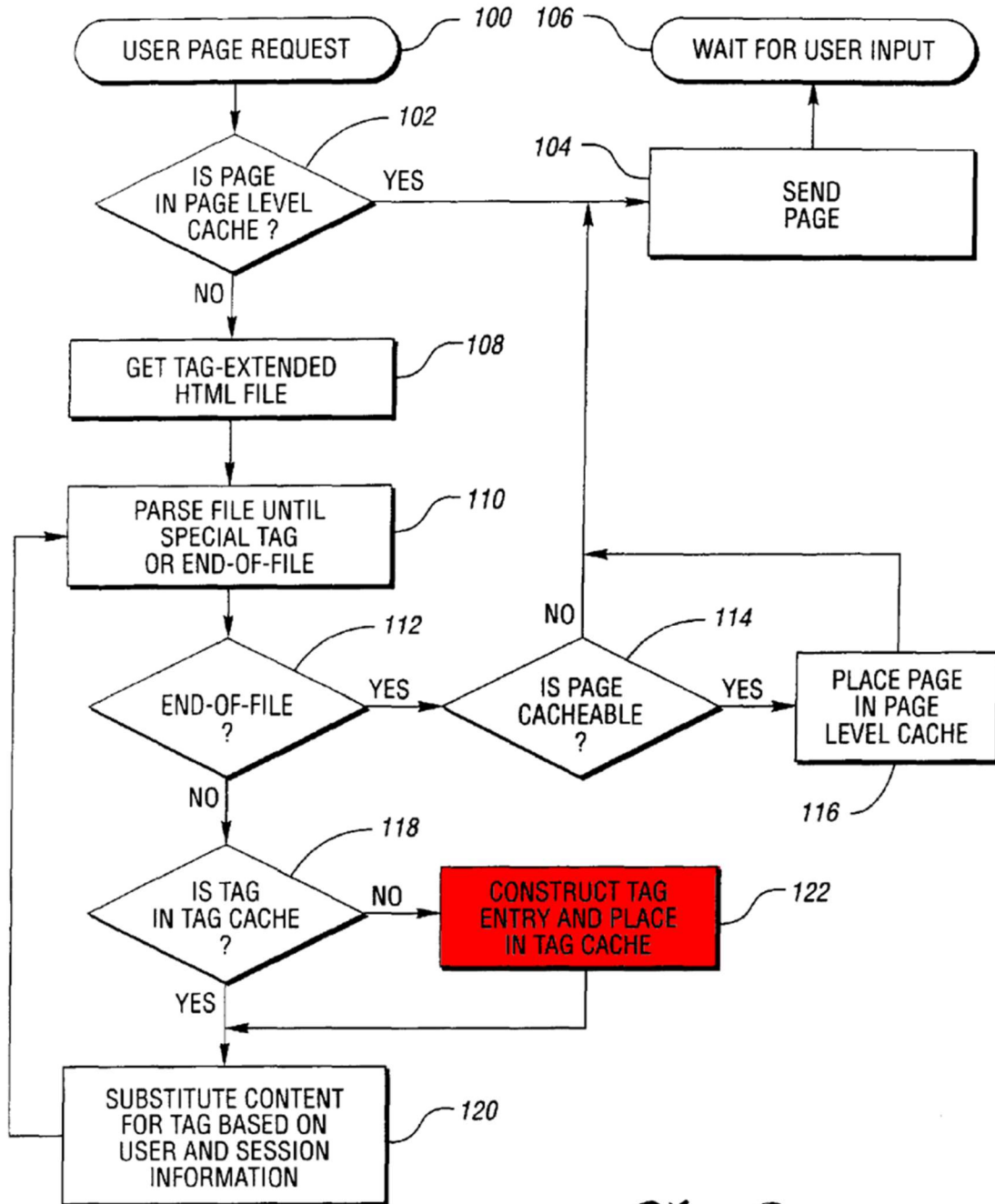


Fig. 2

- 2:62-64 (“Each user may view the HTML pages using a conventional Internet browser as represented by B1 through Bn.”)

- **2:65-3:22** (“Considering the server 10 in detail, *front-end logic is responsible for interfacing with users U1 to Un through the communication medium 2*. As represented by block 12, *the front-end logic can be implemented as a standard web server*. The front-end logic services the user by returning requested information in the form of HTML pages. *The front-end logic generally operates to collect session information as shown in block 14. This information can be obtained, for example, through the use of HTML forms as is commonly done with web pages.*
A page formation mapping process, as shown in reference 24, provides construction of the HTML pages in accordance with the present invention. A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.
A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic substitution of content for tag values based on stored user preferences and

session information. The storage device may be implemented as a directory system or as a database.”)

- **3:23-42** (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The data for these objects are maintained in respective databases 58, 60, 62 and 64.”)*

- 4:8-13 (“*If, while parsing the file and before the end-of-file is reached, a particular tag is encountered, the tag is replaced with one or more HTML fragments associated with the tag.* More specifically, a tag in accordance with the present invention does not constitute a part of the standard HTML tag library.”)
- 4:46-54 (“*If the tag does not exist in the tag cache, a tag-content pair is constructed from information in the object cache or databases* as shown in block 122. *The use of the object cache shortens database access latency.* The content associated with the current topic is matched with a key formed from information including relevant topic information, content type, and a reference to the HTML page. This page reference may be the base file name or may include all or part of the URL address.”)
- 4:58-5:2 (“[A]ctual implementation of the system architecture is susceptible to modification. For example, *the logic blocks as described in conjunction with FIG. 1 can be implemented in hardware, software or a combination thereof, and that the blocks could be combined into larger logical units or further subdivided to conveniently implement the system.*”)
- See also Abstract, 3:22-30, 3:65-67.

140. **To the extent additional disclosure is required, Tso discloses a process for creating a media** (*e.g.*, “retriev[ing] the hypertext object” and “transcod[ing] the data stream appropriately,” *see* [1.f]) **for a system** (*e.g.*, for the content to be “transmit[ted] the transcoded data stream to network client 12” (*see* [1.f])).

141. *See* my discussion of elements [1.pre]-[1.b] and [1.e]-[1.j] (§ IX.A.3(a)-(c), (f)-(k)). **Lafer** also discloses that “Content DB” stores the “content data,” which a person having ordinary skill in the art would have understood to include media data, associated with “content” items in the media tag cache. Lafer, 3:37-43, Figure 1. As discussed for [1.f], Lafer’s system provides for the creation and storage of a “tag-content pair” that associates a “specialized tag” and an “HTML fragment.” Lafer, 3:37-43. This HTML fragment in turn would have been understood to be associated with the “content data” stored in “Content DB” because the “HTML fragment” is disclosed in Lafer as referring to the “content data.” *Id.* And as I previously discussed in section IX.A.1, Lafer not only renders obvious HTML fragments that are HTML tags such as “img” tags for images, but actually discloses this.

142. Lafer discloses that this process is performed for at least two different systems that a person having ordinary skill in the art would have understood to meet

the claimed “system.” First, the user computers would have been understood to meet the claimed “system” (“User System”). Media, including media content objects, is created for the “User System’s” computers to view in requested HTML pages on a browser. Lafer, 2:62-64, Figure 1.

143. Second, as I discuss in more detail below for elements [9.b] and [9.d.2], a system that comprises at least the “front-end logic” and “page cache” (together, the “Front-End System”) would have been understood to meet the claimed “system.” The “Front-End System” is used to implement the “Page Formation Mapping” process 24 as shown in Figure 1. Lafer, Figure 1, 3:8-14. During the page formation process disclosed in Lafer, the system “parses” a “tag-extended” HTML file for “specialized” tags (which are not standard HTML tags) and requests “HTML fragments,” which are standard HTML equivalent tags that the system uses to substitute for the specialized tags in the HTML file prior to delivery to the requesting user. Lafer, 4:8-35. This “system” is made up of “hardware” and “software” components used to form one or more “logical blocks” and rearranging them from how they are set forth in Figure 1. For example, a person having ordinary skill in the art would have understood and found it obvious to implement a system comprising one or more of the “front-end logic,” “page cache,” and “storage medium” for “tag-extended HTML files.” Lafer, 3:8-21, 4:58-5:2. *See id.* And as I previously

discussed with respect to element [1.a] in Section IX.A.3(b), these elements communicate with one another through part of the “Page Formation Mapping” process. The components shown in Fig. 1 used to implement the features of the “page formation mapping” function implemented by the claimed system can be “combined” or “further subdivided,” and comprise the claimed “system” that includes front-end logic (“[Front-End System](#)”). *See id.*

144. In addition to Lafer’s teaching of a process in which a “tag-content pair” that is associated with a standard HTML fragment (and standard HTML equivalent tag) for content is constructed when the tag is not found in Lafer’s tag cache, **Tso** further discloses a process in which, when content requested by a user is not available in cache, an original version of the content is retrieved in the form of a hypertext object and then transcoded to generate content that is cached for future use and then sent to the requesting user. Tso, 6:28-44, 4:1-5, 10:24-49. A person having ordinary skill in the art would have been motivated to modify Lafer with this teaching, to achieve a system in which, in response to finding that a specialized tag does not yet exist in the tag cache, the system creates the media content object that corresponds to the specialized tag, associates this created media content object with an HTML fragment (tag) used by the HTML page to reference the content, and associates the content with the identified specialized “tag” so that future appearances

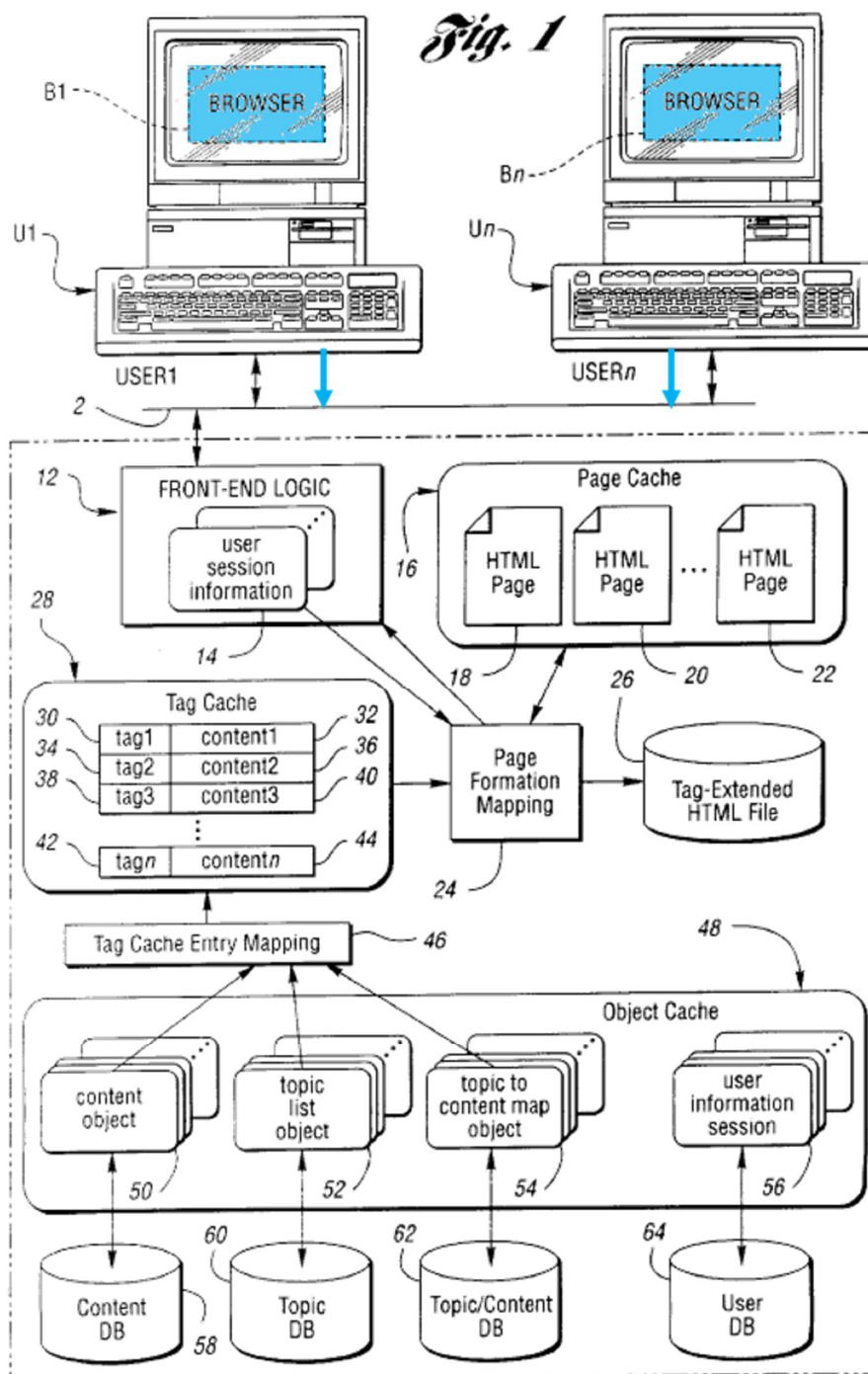
of this same tag in a tag-extended HTML file are addressed more efficiently. Lafer, 1:54-58, 4:46-54; Tso, 6:28-44, 4:1-5, 10:24-49.

(b) Element [9.a]

145. **Element [9.a]** recites “receiving a request from said system for said **HTML equivalent tag**.” This element is met by Lafer.

146. **Lafer discloses receiving a request from said system for said HTML equivalent tag** (*e.g.*, “request” from the system (*see* [9.pre]) for a “page” causes “tag cache” to be “checked” for associated “HTML fragments” (*see* [1.c]), which include HTML equivalent tags). *See, e.g.*, Lafer:

• **Figure 1**



- **Figure 2**

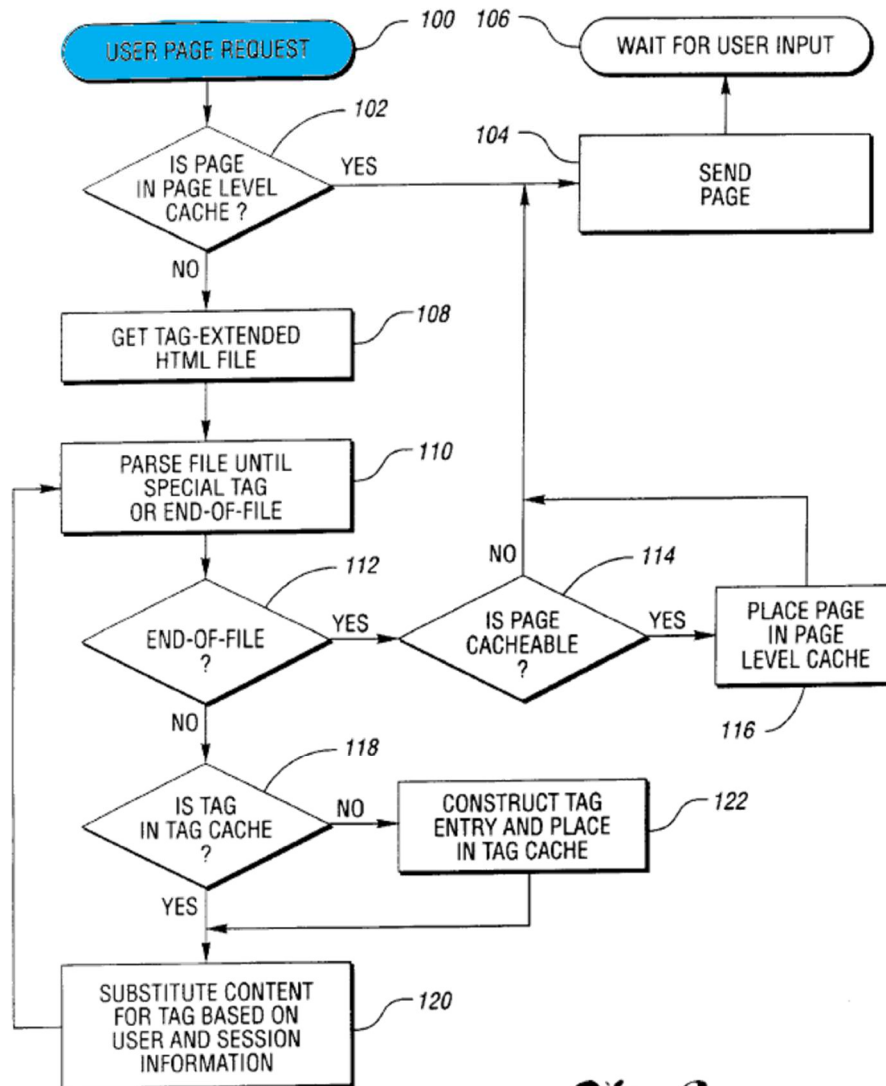


Fig. 2

- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments

stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information. *The server processor is further operative to distribute the constructed HTML files to the user.”)*

- 3:44-55 (“*The illustrated steps are not necessarily sequential operations, and may be implemented using appropriate software, hardware, or a combination of both. ...*

Block 100 of the flow diagram represents *a user request for information in the form of an HTML page. As is well known, this can be a Uniform Resource Locator (URL) address for Internet servers.”)*

- 3:62-4:7 (“*If the page is not in the cache, the page must be constructed or generated. The generation process begins by obtaining the tag-extended*

HTML file as shown in block 108. Block 110 shows that *the tag-extended file is then parsed until either a particular tag or the end-of-file is encountered*. If the end-of-file is encountered, as is shown in block 112, a check is made to determine if the page should be cached as in block 114. If so, *block 116 indicates that the page is placed in the page cache*. This process can be implemented using any known algorithm or technique available for managing a page cache. Flow then passes to block 104 where *the page is sent to the user* as previously described.”);4:32-35 (“*If a particular tag is encountered prior to the end-of-file the tag cache is checked as is shown in block 118*. If the tag is already associated with contents in the tag cache, the contents are substituted for the tag as in block 120.”)

- 4:32-35 (“*If a particular tag is encountered prior to the end-of-file, the tag cache is checked* as is shown in block 118. If the tag is already associated with contents in the tag cache, the contents are substituted for the tag as in block 120.”)
- *See also 3:2-3, 3:22-27, 3:32-36, 4:8-10.*

147. See my discussion of elements [1.c] and [1.h]-[1.j]. As discussed for [1.c] (§IX.A.3(d)), Lafer discloses that a user issues a request for HTML content in

the form of a Web page embedded with content objects. Lafer, 1:15-18. For the User System, the request for HTML is a request for a page including the HTML equivalent tag that references the content object, and is serviced by providing the user with an HTML page that includes that HTML tag as I discussed for elements [1.j] to [1.l] by substituting it for the specialized tag that in the tag-enhanced HTML file. Lafer, 3:62-67.

148. For the Front-End System, Lafer discloses that during parsing of a page to process it for delivery to the requesting user, “the tag-extended file is ... parsed until either a particular tag or the end-of-file is encountered.” Lafer, 3:62-67. Lafer further provides that “[i]f a particular tag is encountered prior to the end-of-file, the tag cache is checked” for “HTML fragments that are substituted for tag values.” Lafer, 4:32-35, 3:22-27. This “check” is a request from the Front-End System for an HTML equivalent tag (and renders it obvious as I discussed in [1.h] to [1.j]). *See id.*

(c) Element [9.b]

149. **Element [9.b] recites “combining said media tag with bandwidth information.”** This element is met by Lafer.

150. **Lafer discloses combining said media tag with information (e.g., incorporating “session information” during “dynamic substitution of content for tag values” using a “substitution token” in the “tag”).** *See, e.g.,* Lafer:

- **2:65-3:22** (“Considering the server 10 in detail, *front-end logic is responsible for interfacing with users U1 to Un through the communication medium 2*. As represented by block 12, *the front-end logic can be implemented as a standard web server*. The front-end logic services the user by returning requested information in the form of HTML pages. *The front-end logic generally operates to collect session information as shown in block 14. This information can be obtained, for example, through the use of HTML forms as is commonly done with web pages.*
A page formation mapping process, as shown in reference 24, provides construction of the HTML pages in accordance with the present invention.
A page cache, represented by block 16, reduces latency by holding HTML pages that are likely to be used again in the future, as represented by references 18, 20 and 22. Techniques for the caching of pages is well known.
A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for dynamic substitution of content for tag values based on stored user preferences and session information. The storage device may be implemented as a directory

system or as a database.”);3:15-21 (“A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for *dynamic substitution of content for tag values based on stored user preferences and session information*. The storage device may be implemented as a directory system or as a database.”)

- **3:15-21** (“*A storage medium or device, shown as block 26, is used to hold tag-extended HTML files. In accordance with the present invention, the tags form an extension to the HTML language that allow for *dynamic substitution of content for tag values based on stored user preferences and session information*. The storage device may be implemented as a directory system or as a database.*”)
- **4:24-31** (“*The content of a particular tag includes a set of tag attributes. One tag attribute determines the tag type. Other tag attributes have text values that can contain substitution tokens. These tokens are replaced by values depending on the tag type. Possible token replacements include* but are not limited to content data from a database, URLs for predefined forms, *user session information*, and URL attributes and form fields.”)
- **See also 3:37-50.**

151. **Tso further discloses bandwidth information** (e.g., “predetermined selection criterion” specifying “network client,” “network” and “proxy” characteristics including “bandwidth,” and “user preferences” for “content quality/speed tradeoff”). *See, e.g., Tso:*

- **7:15-8:9** (“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*. ... 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) ... (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*’)

- **8:10-13** (“*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.*”)
- **See also 3:8-17, 6:64-7:3, 10:37-49, 17:9-15.**

152. As I discussed for element [1.f], Lafer discloses that contents of each of the “specialized tags” include “a set of tag attributes.” Lafer, 4:24-31. These attributes include “text values that can contain substitution tokens” that are “replaced by values depending on the tag type.” *Id.* One of the values that Lafer provides that may replace a substitution token is “user session information.” *Id.* This “user session information” is collected by the “front-end logic” that is responsible for “servic[ing] the user” by facilitating the conveyance of a request and a Web page that satisfies the request. Lafer, 2:65-3:5. Lafer’s disclosure of replacing the “substitution token” with “user session information” would have been understood to disclose and render obvious combining the media tag with bandwidth information, since bandwidth was commonly understood as a relevant piece of “user session information” in the Web page delivery context in which Lafer’s disclosures are implemented.

153. Indeed, **Lafer** leaves it to a person having ordinary skill in the art to select the “user session information” to be included in a substitution token of a specialized tag. **Tso** further discloses that when a request for content is made, “selection criteria” transmitted as part of the request packet are used to determine how to transcode content in response to the request—but **Tso** does not require such selection criteria to be transmitted in any specific way. **Tso**, 6:64-7:3. A person having ordinary skill in the art would have recognized that the selection criteria in **Tso** describing “network characteristics,” including “bandwidth,” as well as information about the “network client,” “proxy” server, and “user preferences” including “quality/speed tradeoff” describe information about a user session communicating with a server. **Tso**, 7:15-8:9. As a result, as I further discussed in Section IX.A.2, a person having ordinary skill in the art would have been motivated to modify **Lafer**’s teaching of combining “user session information” in a “tag” with **Tso**’s selection criteria teachings describing types of session information, by using the “session information” substitution token in **Lafer**’s tag to carry **Tso**’s selection criteria describing such a session. **Lafer**, 4:24-31; **Tso**, 7:15-8:9. Modifying **Lafer** with **Tso**’s teachings yields a system in which **Lafer**’s “special tags” carry “attributes with “substitution tokens” that specify information about the user session including the bandwidth for that session, disclosing this element. **Lafer**, 4:24-31.

(d) Element [9.c]

154. Element [9.c] recites “determining if a media tag entry associated with said media tag exists in a media tag database.” This element is met by Lafer.

155. Lafer discloses determining if a media tag entry associated with said media exists in a media tag database (e.g., “check[ing]” a “tag cache” implemented in a “hash table” for a “particular tag,” associated with “content data,” see [1.e]-[1.f]). See, e.g., Lafer:

- 3:23-42 (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds*

content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The *data for these objects are maintained in respective databases 58, 60, 62 and 64.*”)

- **4:32-35** (“*If a particular tag is encountered prior to the end-of-file, the tag cache is checked* as is shown in block 118. If the tag is already associated with contents in the tag cache, the contents are substituted for the tag as in block 120.”)
- **4:55-61** (“Thus, the present invention provides a system and method for selectively constructing and distributing personalized HTML pages by *storing selected fragments of HTML text in a tag cache*, and rapidly rendering the personalized HTML pages via *special tags embedded in extended HTML files which allow identification and retrieval of the desired fragments.*”)

156. See my discussion of elements [1.e]-[1.f]. As discussed above in § IX.A.3(f)-(g) ([1.e]-[1.f]), Lafer discloses that “[i]f a particular tag is encountered prior to the end-of-file, the tag cache is checked” to determine “[i]f the tag is already associated with contents in the tag cache.” Lafer, 4:32-35. In the case that the tag is already associated with “content” stored in a “Content DB,” in a tag-content pair, the content portion of this pair comprises an “HTML fragment” (that includes or

renders obvious a standard HTML equivalent tag) that identifies, and is associated with, content data in the “Content DB.” Lafer, 3:22-30, Figure 1; *see* [1.f].

(e) Element [9.d]

157. Claim 9 recites process steps for how to handle three cases as part of the process: (1) when a “media tag entry exists” and the “original asset” “has not changed” (*see* [9.d]-[9.d.2]); (2) when a “media tag entry exists” and the “original asset” “has changed” (*see* [9.e]-[9.e.5]); and (3) when the “media tag entry does not exist” (*see* [9.f]-[9.f.4]). Each case ends with “return[ing] said HTML equivalent tag to said system.” *See* [9.d.2], [9.e.5], [9.f.4].

158. **Element [9.d] recites “if said media tag entry exists in said media database, determining if at least one original asset used to create said media has changed.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Cao.

159. **Lafer discloses [determining] if said media tag entry exists in said media database** (*e.g.*, “determining” that a “tag is already associated with contents in the tag cache,” (*see* [1.e]-[1.f]). *See, e.g.*, Lafer:

- 3:23-42 (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.

More specifically, HTML fragments mapped into the tag cache are fetched from an object cache referenced as block 48. The object cache holds content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The data for these objects are maintained in respective databases 58, 60, 62 and 64.”)

- 4:32-35 (“If a particular tag is encountered prior to the end-of-file, *the tag cache is checked as is shown in block 118. If the tag is already associated*

with contents in the tag cache, the contents are substituted for the tag as in block 120.”)

- *See also 5:36-39.*

160. **Cao discloses if said media entry exists in said media database** (*e.g.*, if “there is a cached copy”), **determining if at least one original asset used to create said media has changed** (*e.g.*, “check[ing] whether the document has been modified”). *See, e.g.*, Cao:

- **445** (“In the polling-every-time approach, *every time the user requests a document and there is a cached copy, the cache first contacts the Web server to validate the cached copy*, then returns the copy to the user. In this approach, a write is complete when the modification is registered in the server’s file system.”)
- **447** (“In addition, each client can send an ‘if-modified-since’ request, containing the URL of the document and a timestamp, to the Web server. *Upon receiving the request, the server checks whether the document has been modified since the timestamp. If so, the server returns the status code ‘200’ and the new data; otherwise, the server returns the code ‘304,’ which stands for ‘document unmodified.’*”)

- **447** (“The polling-every-time approach simply *sends an ‘if-modified-since’ request every time a request for a document hits in the cache*. The approach has the advantage that it can be implemented easily in the existing HTTP protocol, and, indeed, the Netscape browser allows users to select this approach for cache consistency.”)
- **448** (“Polling-every-time guarantees that, *when the user requests a document, the returned document is up-to-date at the time when the server processes the ‘if-modified-since’ message*. If a server or network failure happens when the user requests a document, polling-every-time can inform the user of the failure. The user can then choose to view the cached copy while being aware that the copy maybe stale and server or networks failures are hindering the propagation of updates.”)
- **449** (“For polling-every-time, we simply *send an ‘if-modified-since’ request to the server every time a cache hit happens*. If the server replies with status code 304, the cached copy is returned to the user. If the server replies with the document, the cached copy is deleted, the new copy is put in the cache and returned to the user.”);
- *See also 447, 449.*

161. See my discussion for elements [1.e]-[1.f], [9.c]. For this element, as discussed above in Sections IX.A.3(f)-(g) (elements [1.e]-[1.f]), Lafer also discloses “determining whether an HTML fragment associated with a particular tag is currently stored in the tag cache.” Lafer, 5:36-39.

162. And with respect to Cao, as discussed above in Sections IX.A.3(f)-(g) ([1.e]-[1.f]), Lafer teaches determining whether a tag appears in cache; if it does, then the content associated with the tag is then retrieved. Lafer, 4:32-35, 3:23-42. Lafer further teaches that if content is no longer stored in the cache, that content must be “regenerat[ed] and restor[ed]”—disclosing that content in a cache changes and must be updated—but Lafer leaves it to a person having ordinary skill in the art to determine how to confirm that only up-to-date content is provided to clients. Lafer, 5:36-39. Cao provides additional and complementary teachings of how to perform this confirmation. Specifically, when a request for content “hits in the cache,” which means that a cache entry associated with that content exists, Cao teaches using “an ‘if-modified-since’ request” to confirm that the content item being requested is up to date. Cao, 445, 447.

163. As I discussed above in Section IX.B.2, a person having ordinary skill in the art would have been motivated to modify Lafer with this teaching to, for example, ensure that clients are not provided with out-of-date content, yielding a

system in which the system checks whether the associated underlying original asset used to generate the media associated with the media tag (*see* [9.pre]) has changed when a request is made using a “media tag.”

(f) Element [9.d.1]

164. **Element [9.d.1] recites “if said at least one original asset used to create said media has not changed, retrieving said HTML equivalent tag from said database.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer alone and the combined teachings of Lafer and Cao.

165. **Lafer discloses that if said at least one original asset used to create said media has not changed** (*e.g.*, “the tag is already associated with contents in the tag cache”), **retrieving said HTML equivalent tag from said database** (*e.g.*, “the contents are substituted for the tag”). *See, e.g.*, Lafer:

- **4:32-35** (“If a particular tag is encountered prior to the end-of-file, *the tag cache is checked as is shown in block 118. If the tag is already associated with contents in the tag cache*, the contents are substituted for the tag as in block 120.”)
- ***See also* 5:36-39.**

166. To the extent further disclosure of this element is required, Cao further discloses that if said at least one original asset used to create said media has not changed (e.g., “the server returns the code ‘304,’ which stands for ‘document unmodified’”), retrieving said cached copy from said database (e.g., “the cached copy is returned to the user”). See, e.g., Cao:

- 447 (“In addition, each client can send an ‘if-modified-since’ request, containing the URL of the document and a timestamp, to the Web server. Upon receiving the request, the server checks whether the document has been modified since the timestamp. If so, the server returns the status code ‘200’ and the new data; otherwise, *the server returns the code ‘304,’ which stands for ‘document unmodified.’*”)
- 449 (“For polling-every-time, we simply *send an ‘if-modified-since’ request to the server every time a cache hit happens. If the server replies with status code 304, the cached copy is returned to the user.* If the server replies with the document, the cached copy is deleted, the new copy is put in the cache and returned to the user.”).

167. As I discussed for elements [1.h] and [1.j], cached entries of content allow “HTML data” to be substituted for a special tag. Lafer discloses that “[i]f the tag is already associated with contents in the tag cache, the contents are substituted

for the tag as in block 120.” Lafer, 4:32-35; *see* my further discussion in elements [1.e], [1.h], and [1.j].

168. With respect to Cao, as I discussed further in element [9.d], **Lafer** discloses that when an entry exists in cache for a tag (in other words, there is a successful cache hit for the tag), the tag is substituted for the HTML fragment in the file. Cao discloses that in response to “an ‘if-modified-since’ request,” the server “return[s] the code ‘304,’” if the “document [is] unmodified.” Cao, 447. After the cache receives a status code 304 that confirms the cached copy is up-to-date, “the cached copy is returned to the user.” Cao, 449.

169. As I further discussed with respect to element [9.d] and in Section IX.B.2, a person having ordinary skill in the art would have been motivated to modify **Lafer** with **Cao’s** teachings to avoid serving out-of-date content by verifying that the content is unmodified. This combination yields a system in which, when the system verifies that the original asset has not changed, the cached version of the tag is used to retrieve the associated HTML fragment, which not only renders obvious an HTML equivalent tag and placing this equivalent tag in the web page, but actually discloses this. *Id.*

(g) Element [9.d.2]

170. **Element [9.d.2] recites “return said HTML equivalent tag to said system.”** As I previously discussed in Section VIII.A, only one of the three sets of

elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

171. **Lafer discloses returning said HTML equivalent tag to said system** (e.g., returning from “[a] tag cache ... HTML fragments that are substituted for tag values,” see [1.k]-[1.l]). See, e.g., Lafer:

- **2:15-33** (“In accordance with another aspect of the present invention, a system for constructing and selectively distributing Web pages comprises *a server processor for translating user requests into HTML page requests*, and a tag cache containing a plurality of predetermined HTML fragments stored therein. A first storage device is provided for *storing a plurality of HTML files, one or more of which are selectively augmented with a plurality of particular tags* each associated with one or more of the predetermined HTML fragments. A second storage device is provided for storing user preference information represented as topics. A mapping logic processor communicates with the first storage device, second storage device and the tag cache so as to construct an HTML file by *substituting one or more predetermined HTML fragments for each particular tag embedded in a requested HTML file* based on the stored user preference information.

The server processor is further operative to distribute the constructed HTML files to the user.”)

- **3:23-42** (“*A tag cache as shown in reference 28 contains HTML fragments that are substituted for tag values. The tag cache is composed of a plurality of tag-content pairs as shown in blocks 30 and 32, 34 and 36, 38 and 40, and 42 and 44. In the preferred embodiment, the tag cache is implemented using a persistent, e.g., disk-based, hash table. However, as will be appreciated by one of ordinary skill in the art, other software or hardware methods may be used to implement the tag cache.*

A tag cache entry mapping, as shown in block 46, is used to map HTML fragments into the tag. Depending on the type of tag into which the HTML data will be substituted, the mapping may require content data, topic list data, or topic-to-content mapping data.

More specifically, *HTML fragments mapped into the tag cache are fetched from an object cache* referenced as block 48. *The object cache holds content objects 50, topic list objects 52, and topic-to-content map objects 54 as well as user information objects shown as block 56. The data for these objects are maintained in respective databases 58, 60, 62 and 64.”)*

- **4:32-35** (“*If a particular tag is encountered prior to the end-of-file, the tag cache is checked*” as is shown in block 118. If the tag is already associated with contents in the tag cache, the contents are substituted for the tag as in block 120.”).

172. See my discussion of elements [1.h], [1.k]-[1.l]. Lafer also discloses that “[i]f a particular tag is encountered prior to the end-of-file, the tag cache is checked.” Lafer, 4:32-35. “If the tag is already associated with contents in the tag cache, the contents are substituted for the tag.” *Id.*

173. With respect to the Front-End System, as I discussed for elements [1.k]-[1.l] in Sections IX.A.3(l)-(m), the modified HTML document, which includes the HTML equivalent tag substituted for each particular specialized tag, is cached in the “page cache” and sent to the “front-end logic” to be returned to the user computer in fulfillment of the user’s request. This therefore discloses that the HTML equivalent tag is returned to the system. Lafer, 3:22-31, 4:32-35.

174. Similarly, for the User System, the HTML equivalent tag is embedded in the HTML page that then is returned to the user computer after it was “constructed” as I discussed in [1.l]. Lafer, 2:15-33. See § IX.A.3(m).

(h) Element [9.e]

175. **Element [9.e]** recites “**if said at least one original asset used to create said media has changed, removing said media tag entry from said**

database.” As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer, Tso, and Cao and/or Lafer and Cao.

176. Lafer and alternatively Lafer in view of Tso not only renders obvious, but affirmatively discloses removing said media tag entry from said database (*e.g.*, removing a “tag” such that it must be “regenerat[ed] and restor[ed]”).

177. Cao discloses if said at least one original asset used to create said media has changed (*e.g.*, if “the document has been modified since the timestamp”), **removing said entry from said database** (*e.g.*, “the cached copy is deleted”). *See, e.g.*, Cao:

- **447** (“In addition, each client can send an ‘if-modified-since’ request, containing the URL of the document and a timestamp, to the Web server.

Upon receiving the request, *the server checks whether the document has been modified since the timestamp. If so, the server returns the status code ‘200’ and the new data*; otherwise, the server returns the code ‘304,’ which stands for ‘document unmodified.’”)

- **449** (“We improved the original adaptive TTL implementation in Harvest. Every time a cache hit happens, the cache entry’s time-to-live is checked. If it expires, *the original implementation deletes the entry* and sends a regular GET request for the document. Our implementation keeps the cached file and sends an “if-modified-since” request instead.

...

For polling-every-time, we simply *send an ‘if-modified-since’ request to the server every time a cache hit happens*. If the server replies with status code 304, the cached copy is returned to the user. *If the server replies with the document, the cached copy is deleted, the new copy is put in the cache and returned to the user.*”).

178. As I discussed with respect to element [1.f], Lafer discloses that tags must be “regenerat[ed]” and “restor[ed],” which discloses that previously-existing tags have been removed from cache. Lafer, 5:36-39 (*see* [1.f]). And Tso further discloses content “data” that is media, such as image data (*see* [1.pre]).

179. Cao discloses that if “the document has been modified since the timestamp,” “the server returns the ... new data.” Cao, 447. Cao further teaches that documents in the cache are managed using cache “entries.” *See* Cao, 449, 455. In response to an if-modified-since request, “[i]f the server replies with the

document” that contains the new data, “the cached copy is deleted.” Cao, 449. When the server responds to an update request with an updated version of content, the cached copy is removed by deleting the “entry” in cache, which teaches that both the content itself and a cached copy of the content, both of which form part of the entry, are deleted when out of date. Cao, 449.

180. As I further discussed in Section IX.B.2, a person having ordinary skill in the art would have been motivated to modify Lafer with Cao’s teachings of verifying that cache entries are up-to-date to ensure that clients are served with current versions of data. This yields a system in which the system “regenerat[es] and restor[es]” out-of-date data as taught by Lafer by deleting all cached information, including the media tag, and replacing it with new data when the cached copy is out-of-date, rendering obvious this element. Cao, 447, 449.

(i) Element [9.e.1]

181. **Element [9.e.1] recites “create said media using said media tag.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Tso.

182. **Lafer in view of Tso renders obvious creating said media using said media tag.**

183. *See* my discussion of element [1.f] in Section IX.A.3(g).

(j) Element [9.e.2]

184. **Element [9.e.2] recites “storing said media in a media cache.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer alone and the combined teachings of Lafer and Tso.

185. **Lafer and, alternatively, Lafer in view of Tso discloses storing said media in a media cache.**

186. *See* my discussion of element [1.g] in Section IX.A.3(h).

(k) Element [9.e.3]

187. **Element [9.e.3] recites “generating an HTML document referring to said generated media.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Tso.

188. **Lafer discloses generating an HTML document referring to said media** (*see* [1.j]).

189. **Tso discloses said generated media** (*see* [1.j]). *See, e.g.,* Tso:

- **1:54-58** (“In accordance with these and other objects, the present invention provides a method and system for *specifying fragments of HTML text using specialized tags*, holding these fragments in a tag cache, and rapidly rendering personalized HTML pages using the cached fragments.”)
- *See also* **1:59-2:1, 4:55-61.**

190. *See* my discussion of element [1.j] in §IX.A.3(k). Lafer not only renders obvious, but actually discloses, modifying an HTML document by replacing at least one media tag with at least one standard HTML equivalent tag that references a content object. Modifying Lafer with Tso’s teachings as I discussed for elements [1.f]-[1.g], [9.e.1]-[9.e.2], those HTML equivalent tags refer to said generated media in order to “personalize[] HTML pages.” Lafer, 1:54-58.

(I) Element [9.e.4]

191. **Element [9.e.4] recites “placing said media tag and said HTML equivalent tag in said media database.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

192. **Lafer discloses placing said media tag and said HTML equivalent tag in said media database** (*see* [1.i]).

193. Lafer discloses this element for the reasons that I explained for [1.i] in Section IX.A.3(j).

(m) Element [9.e.5]

194. **Element [9.e.5] recites “return said HTML equivalent tag to said system.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

195. **Lafer discloses returning said HTML equivalent tag to said system.** *See* my discussion of element [9.d.2].

(n) Element [9.f]

196. **Element [9.f] recites “if said media tag entry does not exist in said media database, create said media using said media tag.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Tso.

197. **Lafer in view of Tso renders obvious if said media tag entry does not exist in said media database, create said media** (*see* [1.f], [9.e.1]).

198. *See* my discussion of element [1.f] in Section IX.A.3(g) and my discussion of element [9.e.1].

(o) Element [9.f.1]

199. **Element [9.f.1] recites “storing said media in a media cache.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Tso.

200. **Tso discloses storing said media in a media cache.**

201. *See* my discussion of element [9.e.2].

(p) Element [9.f.2]

202. **Element [9.f.2] recites “generating HTML or browser language referring to said generated media.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by the combined teachings of Lafer and Tso.

203. **Lafer discloses generating HTML or browser language referring to said media.**

204. **Tso discloses said generated media.**

205. *See* my discussion of element [9.e.3]. The difference between the terms “HTML or browser language” and “an HTML document” is immaterial to my opinion.

(q) Element [9.f.3]

206. **Element [9.f.3] recites “placing said media tag and said HTML equivalent tag in said media database.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

207. **Lafer discloses placing said media tag and said HTML equivalent tag in said media database.**

208. *See* my discussion of element [9.e.4].

(r) Element [9.f.4]

209. **Element [9.f.4] recites “return said HTML equivalent tag to said system.”** As I previously discussed in Section VIII.A, only one of the three sets of elements [9.d]-[9.d.2], [9.e]-[9.e.5], and [9.f]-[9.f.4] must be met in order to render

obvious claim 9. Nonetheless, as set forth for these elements below, the prior art discloses this element. This element is met by Lafer.

210. **Lafer discloses returning said HTML equivalent tag to said system.**

211. *See* my discussion of elements [9.e.5] and [9.d.2].

C. Grounds 3-4: Lafer in view of Tso, (Cao, for Claim 9), and Moore

212. Lafer in view of Tso renders obvious Challenged Claim 1 and in further view of Cao renders obvious Challenged Claim 9, including Lafer's teachings of HTML fragments that not only render obvious, but actually disclose, the elements reciting "HTML equivalent tag[s]." *See* §IX.A.1-2. Nonetheless, to the extent that further disclosure is required for the recitation of an "HTML equivalent tag" in elements [1.h]-[1.j], [9.pre]-[9.a], [9.d.1]-[9.d.2], [9.e.3]-[9.e.5], or [9.f.2]-[9.f.4] (which I will refer collectively to as the "HTML Limitations"), one example of prior art further explicitly disclosing this is U.S. Patent No. 6,310,601, which was originally assigned to IBM, and identifies Moore as its first named inventor. I understand that a copy of Moore has been designated as Exhibit 1007. Moore provides explicit teachings of an HTML fragment that is a standard HTML "IMG" tag.

213. Specifically, Moore teaches that webpages are generated using HTML source code for rendering, as disclosed, "graphics" and "pictures." Moore, 3:40-53. For example, including an "IMG command," which takes the form of an "HTML

tag,” causes “an image to display” on a Web page. Moore, 4:5-35; 5:29-30. Moore also uses the term “fragment” consistent with its generally understood meaning in the art at the time to refer to a set of lines of code, and therefore not only renders obvious, but actually discloses, a “fragment” that is a “tag.” Thus, Moore discloses a standard “HTML equivalent tag” (*e.g.*, “IMG command” “HTML tag”) as recited in the HTML Limitations.

214. **Moore** is analogous prior art and in the same field as the ’575 patent and **Lafer** (and **Cao** and **Tso**), networked media caching and delivery, because Moore is directed to “methods and apparatus for enhancing the performance of displaying images on Internet browsers.” Moore, 1:11-14. While **Lafer** provides broad teachings of using “HTML fragments” with underlying data maintained in a database, as has been discussed, **Lafer** leaves it to a person having ordinary skill in the art to implement the mapping between such fragments and the underlying data. Lafer, 3:32-43. **Moore** teaches the specific standard HTML tag—an “IMG” tag—used to cause an image to display within a Web page. Moore, 4:5-35, 5:26-29. **Moore** uses the term “fragment” (consistent with the meaning to a person having ordinary skill in the art) to refer to an “HTML code sequence,” and a person having ordinary skill in the art further would have understood these “sequences” can include one image tag. Moore, 4:5-35, 6:59-7:12. A person having ordinary skill in the art

therefore would have been motivated to modify **Lafer** (in view of **Tso** (and **Cao**)) with **Moore**'s teachings of including image "content" in a page because **Moore** teaches a well known, standard HTML tag used to include such images in a page. *Id.* And because such known tags were standard, a person having ordinary skill in the art would have had a reasonable expectation of success using them in implementing **Lafer**'s teachings (in view of **Tso** and **Cao**), which renders obvious the elements reciting a "[standard] HTML equivalent tag."

X. SECONDARY CONSIDERATIONS

215. I have not seen any evidence in the prosecution history of the '575 patent or any related application that I considered of any arguments regarding secondary considerations exist. Nor have I seen any evidence that any such evidence would have a nexus to any of the Challenged Claims 1 and 9. Moreover, as shown above with respect to the prior art referenced in Section IX, any alleged solutions to problems or unexpected results in the '575 patent were already well known.

XI. CONCLUSION

216. In summary, I have concluded that the Challenged Claims 1 and 9 are rendered obvious over the prior art references as discussed above.

217. Specifically, in my opinion, for at least the reasons above, the Challenged Claim 1 is rendered obvious over **Lafer** in view of **Tso** and in view of

Tso and Moore; and Claim 9 is rendered obvious over Lafer in view of Tso and Cao and in view of Tso, Cao, and Moore.

218. To the extent it is argued that any further disclosure is required for an element in the Challenged Claims 1 and 9 that I have identified as being disclosed, a person having ordinary skill in the art would certainly have found that element obvious to include based on the same explicit disclosure or inherent disclosures and analysis I have identified above.

219. I reserve the right to supplement my opinions in the future to respond to any arguments that Patent Owner or its expert(s) may raise and to take into account new information as it becomes available to me.

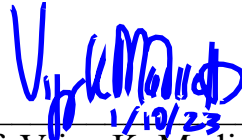
220. In signing this declaration, I recognize that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross-examination within the United States during the time allotted for cross-examination.

221. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful

false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed this 10th day of January 2023, in Atlanta, Georgia

Respectfully submitted,



Prof. Vijay K. Madiseti, Ph.D.

Appendix A

Dr. Vijay K. Madiseti

Fellow, IEEE

vkm@madiseti.com

Cell: 770-527-0177

Address:

**56 Creekside Park Drive
Johns Creek, GA 30022**

Employment:

- 1989-present: Full Professor of Cybersecurity & Privacy, and Electrical & Computer Engineering (**Georgia Tech, Atlanta, GA 30332**).

Areas of Technical Interest – Software Engineering, Cloud Computing Signal Processing, Embedded Systems, Wireless & Mobile Communications, Computer Engineering, Circuit Design (Analog/Digital), Biomedical Instrumentation Hardware/Software, Software Engineering, Digital Signal Processing, Sensors, Wireline & Wireless Computer Networking, Internet of Things (IoT).

Education: PhD (EECS, University of California at Berkeley, 1989), **B.Tech** (Hons) in Electronics and Electrical Communications Engineering (Indian Institute of Technology, Kharagpur, 1984).

Startup Companies:

Director, **VP Technologies, Inc.** (1995-): A startup commercialized through Georgia Tech's Advanced Technology Development Corporation (ATDC) focusing on digital software and hardware design services for military market. <http://www.vptinc.net>

Director, **Soft Networks, LLC** (2001-2007): A startup commercialized through Georgia Tech support focusing on software development tools and compilers for Cellular/WiFi/VOIP/telecommunication products. <http://www.soft-networks.com>

Director, **Elastic Video Inc.** (2007- 2009): A startup commercialized through Georgia Tech's VentureLab (<http://venturelab.gatech.edu>) development image and video processing software for wireless & IP networking.

Litigation Experience (2016-2022) With Testimony

(Note: There may be multiple cases between the parties, e.g., District Court v. ITC, US versus Foreign Cases)

Chrimar v. Adtran, Alcatel, et al.

Technology: Power over Ethernet (2015-2017)

Submitted reports & Deposition & Trial

Chamberlain v. Ryobi/TTI

Case No: 1:16-cv-06097 (ND Illinois)

Expert for Ryobi

Technology: Wireless/IoT/Barrier Movement (2016 – 2019)

Submitted reports & deposition & trial testimony

IOEngine v. IMC/Imation

Case No: cv-14-1572-GMS (US Delaware)

Expert for IMC/Imation

Technology: Networked Storage Device (2016-2017)

Submitted reports & deposition & trial testimony

Huawei v. Samsung

Case No: 3:16-cv-2787-WHO (ND Cal)

Expert for Samsung

Technology: 4G/LTE Random Access Protocols (2016-2019)

Submitted reports & deposition

Hitachi Maxell v. ZTE/Huawei

Case No: 5:16-cv-00178-RWS (ED Texas)

Expert for Hitachi Maxell

Technology: Digital Cameras

Submitted reports & deposition (2017 – 2018)

Hitachi Maxell v. Apple

Case No: 5:19-cv-00036-RWS

Expert for Hitachi Maxell

Technology: Digital Cameras

Submitted Expert Reports (2020-)

Qualcomm v. Apple

Case No: 17-ccv-0108-GPC-MDD (SD Cal) Also, Related ITC/FTC Matters

Expert for Qualcomm

Technology: 4G/Wireless Communications/Smartphones (2017-2019)

Submitted Reports and Deposition

Qualcomm v. Apple

Case No: 3:17-cv-01375-DMS-MDD (SD Cal)

Expert for Qualcomm

Technology: 4G/Wireless Communications/Smartphones (2017-2019)

Submitted Reports and Deposition

Optis v. Huawei

Case No: 2:17-cv-123 (E.D. Texas)

Expert for Optis Wireless

Technology: 4G/Video (2017-2019)

Submitted reports & deposition

Beckman Coulter v. Sysmex

Case No: 1:17-cv-24049-DPG (ND Illinois)

Expert for Sysmex

Technology: Medical Instrumentation Automation (2017-present)

Testifying Expert

TQ Delta v. Xyvel/Adtran

Expert for TQ Delta

Technology: DSL Technologies (2018-present)

Testifying Expert

3GL/KPN v. LG, Blackberry, HTC

Expert for 3GL/KPN

Technology: 4G/LTE Protocols (2018-present)

Reports and Deposition / Testifying Expert

Rovi v. Comcast

ITC No. 337-TA-1103 (ITC)

Expert for Rovi

Technology: Digital Video & interactive GUI (Jan 2018-May 2018)

Reports and Deposition

Cirba/Densify v. VMWare

Case No: 1:19-cv-00742-LPS

Expert for Cirba/Densify

Technology: Virtualization (2019-present)

Reports Deposition/PI Hearing / Testifying Expert

Power Integrations v. On Semiconductor

Case No: 17-cv-03189-BLF

Expert for On Semiconductor

Technology: Power Electronics (2018-2019)

Reports & Deposition/Testifying Expert.

Pathway v. Lumens, IPEVO, Aver JDA

Case No: ITC 337-TA-1045

Expert for Lumens JDA

Technology: Document Cameras (2016-2019)

Reports & Deposition (for IPR/ITC, no trial)

Wilan v. Apple

Case No: 3:14-cv-2235

Expert for WiLan

Technology: Voice over LTE/LTE Protocols (2017-2020)

Reports and Testified through Deposition/Trials

St. Lawrence Comm. v. Amazon

Case No. 2:19-CV-00027-JRG

Expert for Amazon

Technology: Multimedia Codecs

Submitted Expert Reports/Deposition (2020-2020)

PanOptis v. Apple

Case No. 2:19-CV-00066-JRG

Expert for PanOptis

Technology: LTE

Submitted Expert Reports/Deposition/Trial (2019-2020)

GAS v. Sprint Communications

Case No. 2:20-cv-00007-RWS

Expert for General Access (GAS)

Technology: LTE

Submitted Expert Reports/Deposition (9/21-)

Causam v. Ecobee, Itron, et al

Case No. ITC 337-TA-1277

Expert for Causum

Technology: Load Management

Submitted Expert Reports and Deposition (6/22)

Masimo v. Apple

Case No. ITC 337-TA-1276 and related cases

Expert for Masimo

Technology: Pulse Oximetry

Submitted Expert Reports, Deposition & Trial

TQ Delta v. Nokia, Adtran, CommScope, 2Wire, et al

Cases in Delaware & Texas

Expert for TQ Delta

Technology: DSL and Cable Modems

Submitted Expert Reports, Depositions & Hearings.

Paltalk Holdings v. Cisco Systems Inc.

Western District of Texas (6:21-CV-00757-ADA)

Expert for Paltalk Holdings

Technology: VOIP Technologies

Submitted Expert Reports and Deposition

Ollnova Technologies v. Carrier Group

Southern District of Florida (Case 9:22-cv-80388-DMM)

Expert for Ollnova

Technology: IoT

Submitted Report and Deposition.

Additional matters include declarations supporting IPRs at the PTAB for Google (US Patent 8,601,154), On Semiconductor (US Patent 6,212,079), Ubisoft (US Patent 5,490,216), Lumens JDA (2017-2019), Broadcom (WiFi), Sony (US Patent 6,101,534), Kia, (US Patent 5,530,431), Qualcomm, Ericsson, Daimler Benz, VW, Estech, Amazon, Ring, Digital Ally, On Semiconductor, LoganTree, Unified Patents, BMC Software, Masimo, Sentius, BMW, Facebook, PAWS, Google., Daimler and Dolby.

Earned Degrees

1. B. Tech (Hons), Electronics & Electrical Comm. Engineering

Indian Institute of Technology (IIT), Kharagpur, India

1984.

2. Ph.D., Electrical Engineering & Computer Sciences (EECS)

University of California (UC), Berkeley. CA

1989.

Books

1. VLSI Digital Signal Processors

Madiseti, V.K.;

Boston: MA, IEEE Press: Butterworth Heinemann, 1995, 525 pp.

2. Quick-Turnaround ASIC Design in VHDL

Romdhane, M., Madiseti, V.K., Hines, J.

Boston: MA, Kluwer Academic Press, 1996, 190 pp.

3. The Digital Signal Processing Handbook (First Edition)

Madiseti, V. K., Williams, D. (Editors)

CRC Press, Boca Raton, Fla, 1998, 2500 pp.

4. VHDL: Electronics Systems Design Methodologies.

Madiseti, V. K. (Editor)

Boston: MA, IEEE Standards Press, 2000, ISBN 0-7381-1878-8.

5. Platform-Centric Approach to System-on-Chip (SoC) Design.

Madiseti, V. K., Arpikanondt, A.

Springer, Boston: MA, Springer, 2004, 280 pp.

6. The Digital Signal Processing Handbook – Second Edition.

Madiseti, V. K. (2009)

CRC Press, Boca Raton, Fla.

7. Cloud Computing: A Hands-On Approach

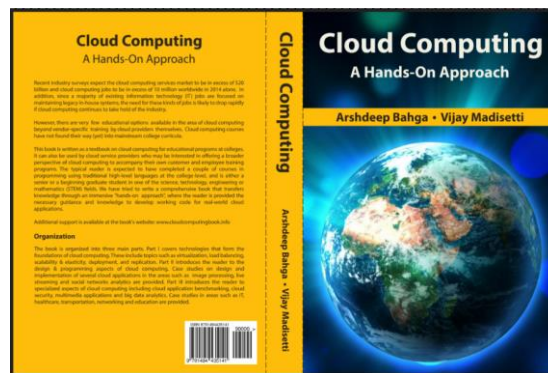
A Bahga, V. Madiseti (2013)

Amazon CreateSpace Publishing, 2013, 454 pp.

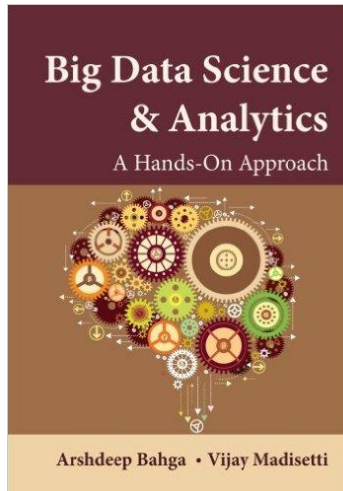
8. Internet of Things: A Hands-On Approach

A Bahga, V. Madiseti (2014)

Amazon CreateSpace Publishing, 2014, 450 pp.



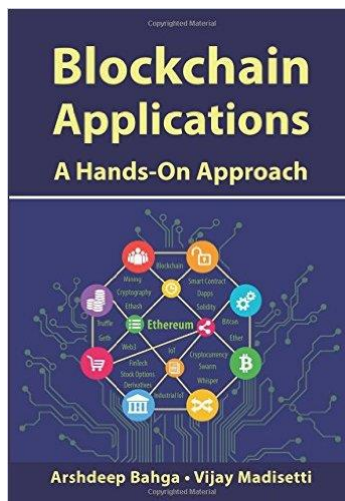
9. Big Data Science & Analytics: A Hands-On Approach
A Bahga, V. Madiseti (2016)
 Amazon CreateSpace Publishing, 2016, 542 pp.



10. Blockchain Applications: A Hands-On Approach

A Bahga, V. Madiseti (2017)

Amazon CreateSpace Publishing, 2017, 380 pp.



Edited Books & Collection of Papers

1. Advances in Parallel & Distributed Simulation

Madiseti, V.K.; Nicol, D., Fujimoto, R. (Editors)

San Diego, CA: SCS Press, 1991, 200 pp.

2. Modeling, Analysis, Simulation of Computer & Telecommunications Systems

Madiseti, V., Gelenbe, E., Walrand, J. W. (Editors)

Los Alamitos: CA, IEEE Computer Society Press, 1994, 425 pp.

3. Modeling & Simulations on Microcomputers

Madiseti, V.K. (Editor)

San Diego, CA: SCS Press, 138 pp. 1990.

Editorship of Journals & Transactions

1. IEEE Design & Test of Computers

Special Issue: Reengineering Digital Systems

April – June 1999 (Vol 16, No 2)

Madiseti, V.K (Editor)

Los Alamitos: CA, IEEE Computer Society Press, 1999.

2. IEEE Design & Test of Computers

Special Issue: Rapid Prototyping of Digital Systems

Fall 1996 (Vol 13, No 3)

Madiseti, V., Richards, M. (Editors)

Los Alamitos: CA, IEEE Computer Society Press, 1994, 425 pp.

3. IEEE Transactions on Circuits & Systems II

Associate Editor: 1993-1995.

4. International Journal in Computer Simulation

Associate Editor: 1990-1993

5. International Journal in VLSI Signal Processing

Editorial Board: 1995 - Present

Recent Issued US Patents

- [1] [10,503,927](#) [Method and system for securing cloud storage and databases from insider threats and optimizing performance](#), Issued Dec 10, 2019
- [2]. [10,460,283](#) [Smart contract optimization for multiparty service or product ordering system](#), Issued Oct 29, 2019
- [3]. [10,459,946](#) [Method and system for tuning blockchain scalability, decentralization, and security for fast and low-cost payment and transaction processing](#), Issued Oct 29, 2019

- [4]. [10,402,589](#) [Method and system for securing cloud storage and databases from insider threats and optimizing performance](#), Issued Sep 3, 2019
- [5]. [10,394,845](#) [Method and system for tuning blockchain scalability for fast and low-cost payment and transaction processing](#), Issued Aug 27, 2019
- [6]. [10,289,631](#) [Method and system for tuning blockchain scalability for fast and low-cost payment and transaction processing](#), Issued May 24, 2019
- [7]. [10,255,342](#) [Method and system for tuning blockchain scalability, decentralization, and security for fast and low-cost payment and transaction processing](#), Issued April 9, 2019
- [8]. [10,243,743](#) [Tokens or crypto currency using smart contracts and blockchains](#), Issued March 26, 2019
- [9]. [10,204,339](#) [Method and system for blockchain-based combined identity, ownership, integrity and custody management](#), Issued February 12, 2019
- [10]. [10,204,148](#) [Method and system for tuning blockchain scalability, decentralization, and security for fast and low-cost payment and transaction processing](#), Issued Feb 12, 2019
- [11]. [10,121,143](#) [Method and system for blockchain-based combined identity, ownership, integrity and custody management](#), Issued Nov 6, 2018
- [12]. [10,102,526](#) [Method and system for blockchain-based combined identity, ownership, integrity and custody management](#), October 16, 2018
- [13]. [10,102,265](#) [Method and system for tuning blockchain scalability for fast and low-cost payment and transaction processing](#), October 16, 2018
- [14]. [9,935,772](#) [Methods and systems for operating secure digital management aware applications](#), April 3, 2018
- [15]. [9,769,213](#) [Method and system for secure digital object management](#), Issued Sep 19, 2017

Refereed Journal Publications

1. Trends in the Electronic Control of Mine Hoists

*Madiseti, V. and Ramlu, M.,
IEEE Transactions on Industry Applications, Vol IA-22, No. 6,
November/December 1986. Pages 1105-1112*

2. Multilevel range/NEXT performance in digital subscriber loops

Brand, G.; Madiseti, V.; Messerschmitt, D.G.;
Communications, Speech and Vision, IEE Proceedings I [see also IEE

Proceedings-Communications] ,Volume: 136 , Issue: 2 , April 1989
Pages:169 – 174

3. Seismic migration algorithms on parallel computers

Madiseti, V.K.; Messerschmitt, D.G.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and
Signal Processing, IEEE Transactions on] ,Volume: 39 , Issue: 7 , July 1991
Pages:1642 – 1654

4. Asynchronous algorithms for the parallel simulation of event-driven dynamical systems

Madiseti, V.K.; Walrand, J.C.; Messerschmitt, D.G.:
ACM Transactions on Modeling and Computer Simulation, v 1, n 3, July 1991,
Pages: 244-74

5. Synchronization mechanisms for distributed event-driven computation

Madiseti, V.K.; Hardaker, D.:
ACM Transactions on Modeling and Computer Simulation, v 2, n 1, Jan. 1992,
Pages: 12-51

6. Efficient VLSI Architectures for the Arithmetic Fourier Transform (AFT)

Kelley, B.T.; Madiseti, V.K.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and
Signal Processing, IEEE Transactions on] ,Volume: 41 , Issue: 1 , January
1993
Pages:365-378

7. The fast discrete Radon transform. I. Theory

Kelley, B.T.; Madiseti, V.K.;
Image Processing, IEEE Transactions on ,Volume: 2 , Issue: 3 , July 1993
Pages:382 – 400

8. The Georgia Tech digital signal multiprocessor

Barnwell, T.P., III; Madiseti, V.K.; McGrath, S.J.A.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and
Signal Processing, IEEE Transactions on] ,Volume: 41 , Issue: 7 , July 1993
Pages:2471 – 2487

9. The MIMDIX Environment for Parallel Simulation

Madiseti, V.K.; Hardaker, D.; Fujimoto, R.M.:
Journal of Parallel and Distributed Computing, v18, no. 4, August 1993,
Pages: 473-83.

- 10. LMSGEN: a prototyping environment for programmable adaptive digital filters in VLSI**
Romdhane, M.S.B.; Madiseti, V.K.;
Chapter in VLSI Signal Processing, VII, 1994.,
Pages:33 – 42
- 11. Fixed-point co-design in DSP**
Egolf, T.W.; Famorzadeh, S.; Madiseti, V.K.;
Chapter in VLSI Signal Processing, VII, 1994.,
Pages:113 - 126
- 12. A fast spotlight-mode synthetic aperture radar imaging system**
Madiseti, V.K.;
Communications, IEEE Transactions on ,Volume: 42 , Issue: 234 , February-April 1994
Pages:873 – 876
- 13. Rapid prototyping on the Georgia Tech digital signal multiprocessor**
Curtis, B.A.; Madiseti, V.K.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on] ,Volume: 42 , Issue: 3 , March 1994
Pages:649 – 662
- 14. Low-power signaling in asymmetric noisy channels via spectral shaping**
Sipitca, M.; Madiseti, V.K.;
Signal Processing Letters, IEEE, Volume: 1 , Issue: 8 , Aug 1994
Pages:117 – 118
- 15. A quantitative methodology for rapid prototyping and high-level synthesis of signal processing algorithms**
Madiseti, V.K.; Curtis, B.A.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on] ,Volume: 42 , Issue: 11 , Nov. 1994
Pages:3188 – 3208
- 16. Computer Simulation of Application-Specific Signal Processing Systems**
Casinovi, G.; Madiseti, V.K.;
International Journal in Computer Simulation, Vol. 4, No. 4, Nov 1994.
- 17. System partitioning of MCMs for low power**
Khan, S.A.; Madiseti, V.K.;
Design & Test of Computers, IEEE ,Volume: 12 , Issue: 1 , Spring 1995
Pages:41 – 52

18. Error correcting run-length limited codes for magnetic recording

Jaejin Lee; Madiseti, V.K.;

Magnetics, IEEE Transactions on ,Volume: 31 , Issue: 6 , Nov. 1995

Pages:3084 – 3086

19. Virtual prototyping of embedded microcontroller-based DSP systems

Madiseti, V.K.; Egolf, T.W.;

Micro, IEEE ,Volume: 15 , Issue: 5 , Oct. 1995

Pages:9 – 21

20. Constrained multitrack RLL codes for the storage channel

Lee, J.; Madiseti, V.K.;

Magnetics, IEEE Transactions on ,Volume: 31 , Issue: 3 , May 1995

Pages:2355 – 2364

21. Rapid digital system prototyping: current practice, future challenges

Madiseti, V.K.;

Design & Test of Computers, IEEE ,Volume: 13 , Issue: 3 , Fall 1996

Pages:12 – 22

22. Conceptual prototyping of scalable embedded DSP systems

Dung, L.-R.; Madiseti, V.K.;

Design & Test of Computers, IEEE ,Volume: 13 , Issue: 3 , Fall 1996

Pages:54 – 65

23. Advances in rapid prototyping of digital systems

Madiseti, V.K.; Richards, M.A.;

Design & Test of Computers, IEEE ,Volume: 13 , Issue: 3 , Fall 1996

Pages:9

24. Combined modulation and error correction codes for storage channels

Jaejin Lee; Madiseti, V.K.;

Magnetics, IEEE Transactions on ,Volume: 32 , Issue: 2 , March 1996

Pages:509 – 514

25. Model-based architectural design and verification of scalable embedded DSP systems-a RASSP approach

Dung, L.-R.; Madiseti, V.K.; Hines, J.W.;

Chapter in VLSI Signal Processing, IX, 1996.

Pages:147 – 156

26. Low-power digital filter implementations using ternary coefficients

Hezar, R.; Madiseti, V.K.;

Chapter in VLSI Signal Processing, IX, 1996.,

Pages:179 – 188

27. All-digital oversampled front-end sensors

Romdhane, M.S.B.; Madiseti, V.K.;

Signal Processing Letters, IEEE, Volume: 3 , Issue: 2 , Feb. 1996

Pages:38 – 39

28. Modeling COTS components in VHDL

Calhoun, S., Reese, R; Egolf, T., Madiseti, V.K.;

Journal of VLSI Signal Processing, Volume: 14 , Issue: 2 , Nov 1996

Pages: 24 – 31

29. VHDL-Based Rapid Systems Prototyping

Egolf, T.; Madiseti, V.K.;

Journal of VLSI Signal Processing, Volume: 14 , Issue: 2 , Nov 1996

Pages: 40-52

30. Interface design for core-based systems

Madiseti, V.K.; Lan Shen;

Design & Test of Computers, IEEE ,Volume: 14 , Issue: 4 , Oct.-Dec. 1997

Pages:42 - 51

31. Incorporating cost modeling in embedded-system design

Debardelaben, J.A.; Madiseti, V.K.; Gadiant, A.J.;

Design & Test of Computers, IEEE ,Volume: 14 , Issue: 3 , July-Sept. 1997

Pages:24 – 35

32. On homomorphic deconvolution of bandpass signals

Marenco, A.L.; Madiseti, V.K.;

Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and

Signal Processing, IEEE Transactions on] ,Volume: 45 , Issue: 10 , Oct. 1997

Pages:2499 – 2514

33. A case study in the development of multi-media educational material: the VHDL interactive tutorial

Gadiant, A.J.; Stinson, J.A., Jr.; Taylor, T.C.; Aylor, J.H.; Klenke, R.H.;

Salinas, M.H.; Madiseti, V.K.; Egolf, T.; Famorzadeh, S.; Karns, L.N.; Carter,

H.W.;

Education, IEEE Transactions on ,Volume: 40 , Issue: 4 , Nov. 1997

Pages:17 pp.

34. Adaptive mobility management in wireless networks

Jeongwook Kim; Madiseti, V.K.;

Electronics Letters ,Volume: 34 , Issue: 15 , 23 July 1998

Pages:1453 – 1455

35. Efficient implementation of two-band PR-QMF filterbanks

Hezar, R.; Madiseti, V.K.;

Signal Processing Letters, IEEE ,Volume: 5 , Issue: 4 , April 1998

Pages:92 – 94

36. On fast algorithms for computing the inverse modified discrete cosine transform

Yun-Hui Fan; Madiseti, V.K.; Mersereau, R.M.;

Signal Processing Letters, IEEE ,Volume: 6 , Issue: 3 , March 1999

Pages:61 – 64

37. System on chip or system on package?

Tummala, R.R.; Madiseti, V.K.;

Design & Test of Computers, IEEE ,Volume: 16 , Issue: 2 , April-June 1999

Pages:48 – 56

38. Reengineering legacy embedded systems

Madiseti, V.K.; Jung, Y.-K.; Khan, M.H.; Kim, J.; Finnessy, T.;

Design & Test of Computers, IEEE ,Volume: 16 , Issue: 2 , April-June 1999

Pages:38 – 47

39. Reengineering digital systems

Madiseti, V.K.;

Design & Test of Computers, IEEE ,Volume: 16 , Issue: 2 , April-June 1999

Pages:15 – 16

40. Parameter optimization of robust low-bit-rate video coders

Sangyoun Lee; Madiseti, V.K.;

Circuits and Systems for Video Technology, IEEE Transactions on, Volume: 9

Issue: 6 , Sept. 1999

Pages:849 – 855

41. Closed-form for infinite sum in bandlimited CDMA

Jatunov, L.A.; Madiseti, V.K.;

Communications Letters, IEEE ,Volume: 8 , Issue: 3 , March 2004

Pages:138 – 140

42. A new protocol to enhance path reliability and load balancing in mobile ad hoc networks

Argyriou, A.; Madiseti, V.K.;

Journal of Ad Hoc Networks, Elsevier Press, 2004

43. Closed-form analysis of CDMA systems using Nyquist pulse

Jatunov, L.A.; Madiseti, V. K.;

Communications Letters, IEEE (Under Revision), 2005.

44. Systematic Design of End-to-End Wireless Mobility Management Protocols,

Argyriou, A.; Madiseti, V. K.;

ACM/Springer Wireless Networks (WINET), Accepted 2005.

45. A Novel End-to-End Approach for Video Streaming Over the Internet,

Argyriou, A.; Madiseti, V. K.;

Kluwer Telecommunications Systems, Vol. 28, No. 2, Pages 133-150, Jan 2005. *Special Issue on Multimedia Streaming.*

46. An Analytical Framework of RD Optimized Video Streaming with TCP,

Argyriou, A.; Madiseti, V. K.;

IEEE Transactions on Multimedia, Submitted for review in March 2005.

47. Modeling the Effect of Handoffs on Transport Protocol Performance,

Argyriou, A.; Madiseti, V. K.;

IEEE Transactions on Mobile Computing, Submitted for review in March 2005

48. Throughput Models for Transport Protocols with CBR and VBR Traffic Workloads",

Argyriou, A.; Madiseti, V. K.;

ACM Transactions on Multimedia Computing, Communications & Applications, Submitted for review in April 2005.

49. "Electronic System, Platform & Package Codesign",

Madiseti, V. K.

IEEE Design & Test of Computers, Volume 23, Issue 3, June 2006. pages 220-233.

50. "The Design of an End-to-End Handoff Management Protocol",

A. Argyriou, Madiseti, V. K.

Wireless Networks, Springer, May 2006.

51. "A Soft-Handoff Transport Protocol for Media Flows in Heterogeneous Mobile Networks ",

A. Argyriou, Madiseti, V. K.

Computer Networks, Vol 50, Issue 11, Pages 1860-1871, August 2006.

- 52. "Computationally Efficient SNR Estimation for Bandlimited WCDMA Systems"**
L. Jatunov, Madiseti, V. K.
IEEE Transactions on Wireless Communications, Volume 5, Issue 13,
December 2006, Pages 3480-3491.
- 53. "Space-Time Codes for Wireless & Mobile Applications",**
M. Sinnokrot, Madiseti, V.K.
DSP Handbook, Second Edition, 2009 (to be published)
- 54.A. Bahga, V. Madiseti, "Rapid Prototyping of Advanced Cloud-Based Systems",** *IEEE Computer*, vol. 46, no. 11, Nov 2013, pp 76-83, 2013
- 55.A. Bahga, V. Madiseti, "Cloud-Based Information Integration & Informatics Framework for Healthcare Applications",** *IEEE Computer*, February 2015.
- 56.A. Bahga, V. Madiseti, "A Cloud-based Approach for Interoperable EHRs",** *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 5, Sep 2013, pp. 894-906, 2013
- 57.A. Bahga, V. Madiseti, "Cloud-Based Information Technology Framework for Data Driven Intelligent Transportation Systems",** *Journal of Transportation Technologies*, vol.3 no.2, April 2013
- 58.A. Bahga, V. Madiseti, "Performance Evaluation Approach for Multi-tier Cloud Applications",** *Journal of Software Engineering and Applications*, vol. 6, no. 2, pp. 74-83, Mar 2013.
- 59.Yusuf, A., V. Madiseti, "Configuration for Predicting Travel Time Using Wavelet Packets and Support Vector Regression",** *Journal of Transportation Technologies*, vol 3, no. 3, June 2013.
- 60.A. Bahga, V. Madiseti, "Analyzing Massive Machine Maintenance Data in a Computing Cloud",** *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 10, pp. 1831 - 1843, 2012.
- 61.N. Radia. Y. Zhang, M. Tatimapula, V. Madiseti, "Next Generation Applications on Cellular Networks: Trends, Challenges, and Solutions,"** *Proceedings of the IEEE*, Vol 100, Issue 4, pp. 841-854, 2012.
- 62.A. Bahga, V. Madiseti, "Rapid Prototyping of Advanced Cloud-Based Systems",** *IEEE Computer*, vol. 46, no. 11, Nov 2013, pp 76-83, 2013

- 63.**A. Bahga, V. Madiseti, "**A Cloud-based Approach for Interoperable EHRs**", *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 5, Sep 2013, pp. 894-906, 2013
- 64.**A. Bahga, V. Madiseti, "**Cloud-Based Information Integration & Informatics Framework for Healthcare Applications**", *IEEE Computer*, February 2015.

Peer Reviewed Conference Publications

- 1. Dynamically-reduced complexity implementation of echo cancelers**
Madiseti, V.; Messerschmitt, D.; Nordstrom, N.;
Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '86. ,Volume: 11 , Apr 1986
- 2. Seismic migration algorithms using the FFT approach on the NCUBE multiprocessor**
Madiseti, V.K.; Messerschmitt, D.G.;
Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988
International Conference on , 11-14 April 1988
- 3. Seismic migration algorithms on multiprocessors**
Madiseti, V.K.; Messerschmitt, D.G.;
Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988
International Conference on , 11-14 April 1988
Pages:2124 - 2127 vol.4
- 4. WOLF: A rollback algorithm for optimistic distributed simulation systems**
Madiseti, V.; Walrand, J.; Messerschmitt, D.;
Simulation Conference Proceedings, 1988 Winter , December 12-14, 1988
Pages:296 – 305
- 5. Efficient distributed simulation**
Madiseti, V.; Walrand, J.; Messerschmitt, D.;
Simulation Symposium, 1989. The 22nd Annual , March 28-31, 1989
Pages:5 - 6
- 6. High speed migration of multidimensional seismic data**
Kelley, B.; Madiseti, V.;

Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991
International Conference on , 14-17 April 1991
Pages:1117 - 1120 vol.2

7. Performance of a fast analog VLSI implementation of the DFT

Buchanan, B.; Madiseti, V.; Brooke, M.;
Circuits and Systems, 1992., Proceedings of the 35th Midwest Symposium on
, 9-12 Aug. 1992
Pages:1353 - 1356 vol.2

8. Task scheduling in the Georgia Tech digital signal multiprocessor

Curtis, B.A.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE
International Conference on ,Volume: 5 , 23-26 March 1992
Pages:589 - 592 vol.5

9. The fast discrete Radon transform

Kelley, B.T.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE
International Conference on ,Volume: 3 , 23-26 March 1992
Pages:409 - 412 vol.3

10.Yield-based system partitioning strategies for MCM and ASEM design

Khan, S.; Madiseti, V.;
Multi-Chip Module Conference, 1994. MCMC-94, Proceedings., 1994 IEEE,15-
17 March 1994
Pages:144 - 149

11. Multitrack RLL codes for the storage channel with immunity to intertrack interference

Lee, J.; Madiseti, V.K.;
Global Telecommunications Conference, 1994. GLOBECOM '94.
'Communications: The Global Bridge', IEEE,Volume: 3 , 28 Nov.-2 Dec. 1994
Pages:1477 - 1481 vol.3

12. A parallel mapping of backpropagation algorithm for mesh signal processor

Khan, S.A.; Madiseti, V.K.;
Neural Networks for Signal Processing [1995] V. Proceedings of the 1995
IEEE Workshop , 31 Aug.-2 Sept. 1995
Pages:561 - 570

13. Virtual prototyping of embedded DSP systems

Madiseti, V.K.; Egolf, T.; Famorzadeh, S.; Dung, L.-R.;
Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995

International Conference on ,Volume: 4 , 9-12 May 1995
Pages:2711 - 2714 vol.4

14. Assessing and improving current practice in the design of application-specific signal processors

Shaw, G.A.; Anderson, J.C.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995
International Conference on ,Volume: 4 , 9-12 May 1995
Pages:2707 - 2710 vol.4

15. Introduction to ARPA's RASSP initiative and education/facilitation program

Corley, J.H.; Madiseti, V.K.; Richards, M.A.;
Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995
International Conference on ,Volume: 4 , 9-12 May 1995
Pages:2695 - 2698 vol.4

16. DSP design education at Georgia Tech

Madiseti, V.K.; McClellan, J.H.; Barnwell, T.P., III;
Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995
International Conference on ,Volume: 5 , 9-12 May 1995
Pages:2869 - 2872 vol.5

17. Rapid prototyping of DSP systems via system interface module generation

Famorzadeh, S.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on ,Volume: 2 , 7-10 May 1996
Pages:1256 - 1259 vol. 2

18. Rapid prototyping of DSP chip-sets via functional reuse

Romdhane, M.S.B.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on ,Volume: 2 , 7-10 May 1996
Pages:1236 - 1239 vol. 2

19. A constructive deconvolution procedure of bandpass signals by homomorphic analysis

Marenco, A.L.; Madiseti, V.K.;
Geoscience and Remote Sensing Symposium, 1996. IGARSS '96. 'Remote Sensing for a Sustainable Future.', International ,Volume: 3 , 27-31 May 1996
Pages:1592 - 1596 vol.3

20. BEEHIVE: an adaptive, distributed, embedded signal processing environment

Famorzadeh, S.; Madiseti, V.; Egolf, T.; Nguyen, T.;
Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on ,Volume: 1 , 21-24 April 1997
Pages:663 - 666 vol.1

21. Target detection from coregistered visual-thermal-range images

Perez-Jacome, J.E.; Madiseti, V.K.;
Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on ,Volume: 4 , 21-24 April 1997
Pages:2741 - 2744 vol.4

22. Variable block size adaptive lapped transform-based image coding

Klausutis, T.J.; Madiseti, V.K.;
Image Processing, 1997. Proceedings., International Conference on ,Volume: 3 , 26-29 Oct. 1997
Pages:686 - 689 vol.3

23. A Rate 8/10 (0, 6) MTR Code And Its Encoder/decoder

Jaejin Lee; Madiseti, V.K.;
Magnetics Conference, 1997. Digests of INTERMAG '97., 1997 IEEE International , 1-4 April 1997
Pages:BS-15 - BS-15

24. VHDL models supporting a system-level design process: a RASSP approach

DeBardelaben, J.A.; Madiseti, V.K.; Gadiant, A.J.;
VHDL International Users' Forum, 1997. Proceedings , 19-22 Oct. 1997
Pages:183 - 188

25. A performance modeling framework applied to real time infrared search and track processing

Pauer, E.K.; Pettigrew, M.N.; Myers, C.S.; Madiseti, V.K.;
VHDL International Users' Forum, 1997. Proceedings , 19-22 Oct. 1997
Pages:33 - 42

26. System design and re-engineering through virtual prototyping: a temporal model-based approach

Khan, M.H.; Madiseti, V.K.;
Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on ,Volume: 2 , 1-4 Nov. 1998
Pages:1720 - 1724 vol.2

27. A debugger RTOS for embedded systems

Akgul, T.; Kuacharoen, P.; Mooney, V.J.; Madiseti, V.K.;
Euromicro Conference, 2001. Proceedings. 27th , 4-6 Sept. 2001
Pages:264 - 269

28. Adaptability, extensibility and flexibility in real-time operating systems

Kuacharoen, P.; Akgul, T.; Mooney, V.J.; Madiseti, V.K.;
Digital Systems, Design, 2001. Proceedings. Euromicro Symposium on , 4-6 Sept. 2001
Pages:400 - 405

29. Effect of handoff delay on the system performance of TDMA cellular systems

Turkboydari, M.; Madiseti, V.K.;
Mobile and Wireless Communications Network, 2002. 4th International Workshop on , 9-11 Sept. 2002
Pages:411 - 415

30. Enforcing interdependencies and executing transactions atomically over autonomous mobile data stores using SyD link technology

Prasad, S.K.; Bourgeois, A.G.; Dogdu, E.; Sunderraman, R.; Yi Pan; Navathe, S.; Madiseti, V.;
Distributed Computing Systems Workshops, 2003. Proceedings. 23rd International Conference on , 19-22 May 2003
Pages:803 - 809

31. Performance evaluation and optimization of SCTP in wireless ad-hoc networks

Argyriou, A.; Madiseti, V.;
Local Computer Networks, 2003. LCN '03. Proceedings. 28th Annual IEEE International Conference on , 20-24 Oct. 2003
Pages:317 - 318

32. Implementation of a calendar application based on SyD coordination links

Prasad, S.K.; Bourgeois, A.G.; Dogdu, E.; Sunderraman, R.; Yi Pan; Navathe, S.; Madiseti, V.;
Parallel and Distributed Processing Symposium, 2003. Proceedings. International , 22-26 April 2003
Pages:8 pp.

33. Bandwidth aggregation with SCTP

Argyriou, A.; Madiseti, V.;
Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE Volume: 7 , 1-5 Dec. 2003
Pages:3716 - 3721 vol.7

34. Software streaming via block streaming

Kuacharoen, P.; Mooney, V.J.; Madisetti, V.K.;
Design, Automation and Test in Europe Conference and Exhibition, 2003
, 2003
Pages:912 – 917

35. Frequency-dependent space-interleaving for MIMO OFDM systems

Mohajerani, P.; Madisetti, V.K.;
Radio and Wireless Conference, 2003. RAWCON '03. Proceedings , Aug. 10-13, 2003
Pages:79 - 82

36. A media streaming protocol for heterogeneous wireless networks

Argyriou, A.; Madisetti, V.;
Computer Communications, 2003. CCW 2003. Proceedings. 2003 IEEE 18th Annual Workshop on , 20-21 Oct. 2003
Pages:30 – 33

37. Realizing load-balancing in ad-hoc networks with a transport layer protocol

Argyriou, A.; Madisetti, V.;
Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE ,Volume: 3 , 21-25 March 2004
Pages:1897 - 1902 Vol.3

38.Streaming H.264/AVC video over the Internet

Argyriou, A.; Madisetti, V.;
Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE , 5-8 Jan. 2004
Pages:169 – 174

Other Publications

1. **A Transport Layer Technology for Improving the QoS of Networked Multimedia Applications <draft-madisetti-arguriou-qos-sctp-00.txt>.**
Madisetti, V., Argyriou, A.
IETF Internet-Draft, Jul 25, 2002.
2. **Voice & Video over Mobile IP Networks <draft-madisetti-argyriou-voice-video-mip-00.txt>**
Madisetti, V., Argyriou, A.
IETF Internet-Draft, Nov 20, 2002.
3. **Enhancements to ECRTTP with Applications to Robust Header Compression for Wireless Applications. <draft-madisetti-rao-suresh-rohc-00.txt>**
Madisetti, V.; Rao, S., Suresh, N.
IETF Internet-Draft, June 30, 2003.

Ph.D. Students Graduated

1. **Brian T. Kelley, 1992**
VLSI Computing Architectures for High Speed Signal Processing
Member of Technical Staff, Motorola.

Winner of Dr. Thurgood Marshall Dissertation Fellowship Award
2. **Bryce A. Curtis, 1992**
Special Instruction Set Multiple Chip Computer for DSP
Member of Technical Staff, IBM
3. **Jaejin Lee, 1994**
Robust Multitrack Codes for the Magnetic Channel
Professor, Yonsei University, Korea
4. **Mohamed S. Ben Romdhane, 1995**
Design Synthesis of Application-Specific IC for DSP
Director of IP, Rockwell.
5. **Shoab A. Khan, 1995**
Logic and Algorithm Partitioning on MCMs
Professor, National University of Science & Technology, Pakistan
6. **Lan-Rong Dung, 1997**
VHDL-based Conceptual Prototyping of Embedded DSP Architectures
Professor, National Chaio Tung University, Taiwan.

Winner of VHDL International Best PhD Thesis Award, 1997

7. Thomas W. Egolf, 1997

Virtual Prototyping of Embedded DSP Systems

Distinguished Member of Technical Staff, Agere

8. Alvaro Marenco, 1997

On Homomorphic Deconvolution of Bandpass Signals

Professor, Texas A&M University.

Winner of GIT ECE Outstanding Teaching Assistant Award

9. Shahram Famorzadeh, 1997

BEEHIVE: A Distributed Environment for Adaptive Signal Processing

Member of Technical Staff, Rockwell.

10. Timothy J. Klausutis, 1997

Adaptive Lapped Transforms with Applications to Image Coding.

US Air Force/Univ. of Florida.

11. Lan Shen, 1998

Temporal Design of Core-Based Systems

Member of Technical Staff, IBM

12. James DeBardelaben, 1998

Optimization Based Approach to Cost Effective DSP Design

Research Scientist, Johns Hopkins University

Georgia Tech ECE Faculty Award

13. Sangyoun Lee, 1999

Design of Robust Video Signal Processors

Professor, Yonsei University

US Army Sensors Lab Research Excellence Award, 1999

14. Rahmi Hezar, 2000

Oversampled Digital Filters

Member of Technical Staff, Texas Instruments

15. Yong-kyu Jung, 2001

Model-Based Processor Synthesis

Professor, Texas A&M University

16. Mustafa Turkboylari, 2002

Handoff Algorithms for Wireless Applications

Member of Technical Staff, Texas Instruments

17. Yun-Hui Fan, 2002

A Stereo Audio Coder with Nearly Constant Signal to Noise Ratio

Post-Doctoral Research Associate, Northeastern University

18. Subrato K. De, 2002

Design of a Retargetable Compiler for DSP
Member of Technical Staff, Qualcomm

US Army Sensors Lab Research Excellence Award, 1999

19. Chonlameth Aripnikanondt, 2004

System-on-Chip Design with UML
Professor, King Mongkut's University, Thailand.

US Army Sensors Lab Research Excellence Award, 1999

20. Loran Jatunov, 2004

Performance Analysis of 3G CDMA Systems
Senior Research Scientist, Soft Networks, LLC.

21. Antonios Argyriou, 2005, Serving in Hellenic Army.

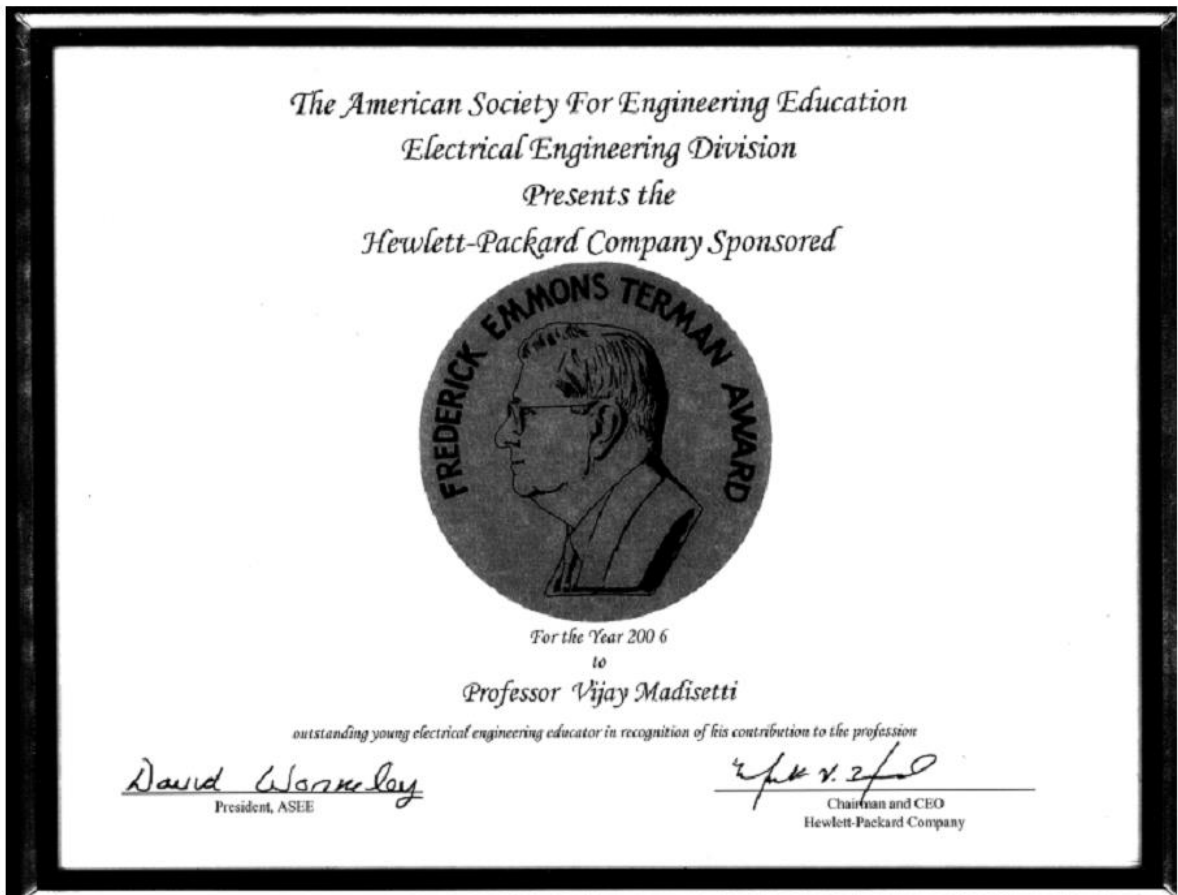
22. Pilho Kim, 2009, Scientist, VP Technologies, Inc.

23. M. Sinnokrot, 2009, Staff Engineer, Qualcomm.

Awards & Honors

1. **Jagadis Bose National Science Talent Fellowship**, Indian Institute of Technology, Kharagpur, 1980-1984.
2. **General Proficiency Prize**, Indian Institute of Technology, Kharagpur, 1984.
3. **Demetri Angelakos Outstanding Graduate Student Award**, Univ. of California, Berkeley, 1989
4. **Ira Kay IEEE/ACM Best Paper Award** for Best Paper presented at IEEE Annual Simulation Symposium, 1989
5. **IBM Faculty Development Award** 1990
6. **Technical Program Chair**, IEEE Workshop on Parallel and Distributed Simulation. 1990.
7. **Technical Program Chair**, IEEE MASCOTS'94
8. **NSF RI Award**, 1990
9. **VHDL International Best PhD Dissertation Advisor Award**, 1997

10. Georgia Tech Outstanding Doctoral Dissertation Advisor Award, 2001.
11. ASEE 2006 Frederick Emmons Terman Medal, 2006.
12. Fellow of IEEE



Intellectual Property Disclosures (Georgia Tech)

<u>Patent</u>	<u>Date</u>	<u>Description</u>
2843	2004	Method and Apparatus for Improving the Performance of Wireless LANs
2825	2003	Method and Apparatus for Optimal Partitioning and Ordering of Antennas for Layered Space-Time Block Codes in MIMO Communications Systems
2815	2003	How to Rapidly Develop a SyD Application
GSU-023	2003	Rapid Development of SyD Applications

2810	2003	System on Mobile Devices Middleware Design
2718	2003	A Transport Layer Algorithm for Improved Anycast Communication
2717	2003	A Novel Transport Layer Load-Balancing Algorithm
2716	2003	A Transport Layer QoS Algorithm
2715	2003	A Novel Transport Layer Algorithm for MPLS Performance
2659	2002	A New Algorithm and Technology for Implementing Mobile IP with Applications to Voice and Video over Mobile IP
2656	2002	Debugging with Instruction-Level Reverse Execution
2655	2002	Embedded Software Streaming
2539		System of Databases: An Enabling Technology for Programming
2517	2002	A Dynamic Instantiated Real-Time Operating System Debugger
2516	2002	A Dynamic Real-Time Operating System
GSU-009	2001	System of Databases: Architecture,, Global Queries, Triggers and Constraints
2480	2001	Mobile Fleet Application based on SyD Technology
2479	2001	System of Databases: A model with coordination links and a calendar application
1893	1999	Beehive
1726	1995	Very High Scale Integrated Circuit Hardware Description Language Models (VHDL Models)
1401	1995	Self-Compensation Receiver (SCR)

Issued Patents: 9,935,772, 9,769,213

Appendix B

APPENDIX B: LIST OF MATERIALS CONSIDERED

Ex. 1001	U.S. Patent No. 6,792,575 (“’575”)
Ex. 1002	File History of U.S. Application No. 09/425,326 (“’575 FH)
Ex. 1004	U.S. Patent No. 6,192,382 (“Lafer”)
Ex. 1005	U.S. Patent No. 6,421,733 (“Tso”)
Ex. 1006	Pei Cao & Chengjie Liu, <i>Maintaining strong cache consistency in the World Wide Web</i> , 47 IEEE TRANSACTIONS ON COMPUTS. 445 - 457 (1998)
Ex. 1007	U.S. Patent No. 6,310,601 (“Moore”)
Ex. 1008	Declaration of Gordon MacPherson
Ex. 1010	U.S. Patent No. 5,420,967 (“Delp”)
Ex. 1011	U.S. Patent No. 6,144,996 (“Starnes”)
Ex. 1012	Harini Bharadvaj et al., <i>An Active Transcoding Proxy to Support Mobile Web Access</i> , PROCS. 17TH IEEE SYMP. ON RELIABLE DISTRIBUTED SYS. 118 (1998), https://ieeexplore.ieee.org/document/740482
Ex. 1013	U.S. Patent No. 6,745,368 (“Boucher”)
Ex. 1014	Rakesh Mohan et al., <i>Adapting Multimedia Internet Content for Universal Access</i> , 1 IEEE TRANSACTIONS ON MULTIMEDIA 104 (1999), https://ieeexplore.ieee.org/document/748175
Ex. 1015	Armando Fox et al., <i>Adapting to Network and Client Variation Using Infrastructural Proxies: Lessons and Perspectives</i> , 5 IEEE PERS. COMMC’NS. 10 (1998), https://ieeexplore.ieee.org/abstract/document/709365
Ex. 1016	U.S. Patent No. 5,581,704 (“Barbará”)
Ex. 1017	Fielding et al., <i>Hypertext Transfer Protocol – HTTP/1.1</i> , RFC 2068, Jan. 1997, https://www.rfc-editor.org/rfc/rfc2068

Ex. 1019	Chaman L. Sabharwal and Sanjiv K. Bhatia, <i>Image databases and near-perfect hash table</i> , 30 PATTERN RECOGNITION 1867 (November 19, 1997), https://www.sciencedirect.com/science/article/pii/S0031320396001872
Ex. 1020	Letter from Tony Corsini of the British Library
Ex. 1021	British Library copy of Pei Cao & Chengjie Liu, <i>Maintaining strong cache consistency in the World Wide Web</i> , 47 IEEE TRANSACTIONS ON COMPUTS. 445 - 457 (1998)
Ex. 1022	IEEE Transactions on Computers About Journal IEEE Xplore (part 1), https://ieeexplore.ieee.org/xpl/aboutJournal.jsp?punumber=12
Ex. 1023	IEEE Transactions on Computers About Journal IEEE Xplore (part 2), https://ieeexplore.ieee.org/xpl/aboutJournal.jsp?punumber=12
Ex. 1024	IEEE Transactions on Computers About Journal IEEE Xplore (part 3), https://ieeexplore.ieee.org/xpl/aboutJournal.jsp?punumber=12
Ex. 1025	IEEE Transactions on Computers All Issues IEEE Xplore, https://ieeexplore.ieee.org/xpl/issues?punumber=12&isnumber=9984045
Ex. 1026	Table of Contents, 47 IEEE TRANSACTIONS ON COMPUTS., https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=14873&punumber=12
Ex. 1027	Maintaining strong cache consistency in the World Wide Web IEEE Journals & Magazine IEEE Xplore, Citations by Papers, https://ieeexplore.ieee.org/document/675713/citations?tabFilter=papers#citations
Ex. 1028	Maintaining strong cache consistency in the World Wide Web IEEE Journals & Magazine IEEE Xplore, Citations by Patents, https://ieeexplore.ieee.org/document/675713/citations?tabFilter=patents#citations

Ex. 1029	Maarten van Steen et al., <i>A scalable middleware solution for advanced wide-area Web services</i> , 6 DISTRIBUTED SYS. ENG'G 34 (1999), https://iowpscience.iop.org/article/10.1088/0967-1846/6/1/304
Ex. 1030	James B. Lim & Ali R. Hurson, <i>Data duplication and consistency in a mobile, multidatabase environment</i> , PROCS. 1998 INT'L CONF. ON PARALLEL AND DISTRIBUTED SYS. 50 (1998) (Cat. No.98TB100250), https://ieeexplore.ieee.org/document/800066
Ex. 1031	Xiangping Chen & Prasant Mohapatra, <i>Lifetime behavior and its impact on Web caching</i> , PROCS. 1999 IEEE WORKSHOP ON INTERNET APPLICATIONS 54 (1999) (Cat. No.PR00197), https://ieeexplore.ieee.org/document/788017
Ex. 1032	K.Y. Wong, K.H. Yeung, <i>Site-based approach in HTTP proxy design</i> , PROCS. 1999 ICPP WORKSHOPS ON COLLABORATION AND MOBILE COMPUTING (1999), https://ieeexplore.ieee.org/document/800066
Ex. 1033	U.S. Patent No. 6,088,721 ("Lin")
Ex. 1034	PCT Patent App. No. WO2000062185 ("Lea")