IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

ORACLE CORPORATION and ORACLE U.S.A. INC.,	)	
Plaintiffs/Counterclaim Defendants,	)	C.A. No. 06-414-SLR
v.	)	**JURY TRIAL DEMANDED**
EPICREALM LICENSING, LP,	)	**PUBLIC VERSION**
Defendant/Counterclaim Plaintiff.	)	

APPENDIX IN SUPPORT OF DEFENDANTS' CLAIM CONSTRUCTION BRIEF

VOLUME II of II

EXHIBITS 7-9

Richard L. Horwitz (#2246)
David E. Moore (#3983)
POTTER ANDERSON & CORROON LLP
Hercules Plaza, 6th Floor
1313 N. Market Street
Wilmington, DE 19899
Tel: (302) 984-6000
rhorwitz@potteranderson.com
dmoore@potteranderson.com

Attorneys for Defendant/Counterclaim Plaintiff epicRealm Licensing, LP

OF COUNSEL:

Harry J. Roper
George S. Bosy
Aaron A. Barlow
Patrick L. Patras
David R. Bennett
Paul D. Margolis
Benjamin J. Bradford
Emily C. Johnson
JENNER & BLOCK
330 N. Wabash Avenue
Chicago, IL 60611-7603
Tel: (312) 923-8305

Dated: July 31, 2008
876823 / 31393 / Oracle

Public Version Dated: August 7, 2008

TABLE OF CONTENTS

Exh.	Document
1.	U.S. Patent 5,894,554, Lowery et al.
2.	U.S. Patent 6,415,335, Lowery et al.
3.	epicRealm's Proposed Claim Construction Chart, December 14, 2007
4.	Oracle's Proposed Claim Construction Chart, December 14, 2007
5.	Parties' Proposed Element-by-Element Claim Construction Chart, June 30, 2008
6 .	File History of U.S. Patent 5,894,554, Lowery et al., EPIC000174-324
7.	File History of U.S. Patent 6,415,335, Lowery et al., EPIC000325-565
8.	Report and Recommendation Regarding Claim Construction, August 15, 2006, *EpicRealm Licensing, LLC v. Autoflex Leasing, Inc. et al.,* (E.D. Tex.)
9.	Order Adopting Report and Recommendation Regarding Claim Construction, October 30, 2006, *EpicRealm Licensing, LLC v. Autoflex Leasing, Inc. et al.,* (E.D. Tex.)

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

CERTIFICATE OF SERVICE

I, David E. Moore, hereby certify that on August 7, 2008, the attached document was electronically filed with the Clerk of the Court using CM/ECF which will send notification to the registered attorney(s) of record that the document has been filed and is available for viewing and downloading.

I further certify that on August 7, 2008, the attached document was Electronically Mailed to the following person(s):

Mary B. Graham
Morris, Nichols, Arsht & Tunnell LLP
1201 North Market Street
P.O. Box 1347
Wilmington, DE 19899
MGraham@MNAT.com

James G. Gilliland
Igor Shoiket
Townsend and Townsend and Crew LLP
Two Embarcadero Center
Eighth Floor
San Francisco, CA 94111-3834
OracleEpicrealm@townsend.com

Theodore T. Herhold
Robert J. Artuz
Eric M. Hutchins
Eric A. Mercer
Joseph A. Greco
Nitin Gupta
Townsend and Townsend and Crew LLP
379 Lytton Avenue
Palo Alto, CA 94301
OracleEpicrealm@townsend.com

Chad E. King
Townsend and Townsend and Crew LLP
1200 Seventeenth Street
Suite 2700
Denver, CO 80202
OracleEpicrealm@townsend.com

/s/ David E. Moore
Richard L. Horwitz
David E. Moore
Potter Anderson & Corroon LLP
Hercules Plaza – Sixth Floor
1313 North Market Street
P.O. Box 951
Wilmington, DE 19899-0951
(302) 984-6000
rhorwitz@potteranderson.com
dmoore@potteranderson.com

Exhibit 7

jc518 U.S. PTO
09/234048
01/19/99

ISSUE CLASSIFICATION	
710	Class
5	Subclass

PATENT NUMBER
6415335
6415335

U.S. UTILITY PATENT APPLICATION

O.I.P.E.	PATENT DATE
SCANNED Q.A.	JAN 02 2002

SECTOR	CLASS	SUBCLASS	ART UNIT	EXAMINER
	710	5	2135	Perveen

FILED WITH: ☐ DISK (CRF) ☐ FICHE
(Attached in pocket on right inside flap)

PREPARED AND APPROVED FOR ISSUE

ISSUING CLASSIFICATION

ORIGINAL CLASS	ORIGINAL SUBCLASS	CROSS REFERENCE(S) CLASS	CROSS REFERENCE(S) SUBCLASS (ONE SUBCLASS PER BLOCK)					
710	5	710	7					
INTERNATIONAL CLASSIFICATION		709	219	223	238			
G06F 13/14								
G06F 13/20								

☐ Continued on Issue Slip Inside File Jacket

☒ TERMINAL DISCLAIMER	DRAWINGS			CLAIMS ALLOWED	
09/234,048	Sheets Drwg.	Figs. Drwg.	Print Fig.	Total Claims	Print Claim for O.G.
	5	5	1	29	1

☐ a) The term of this patent subsequent to _____ (date) has been disclaimed.

Rehana Perveen (Assistant Examiner) 2/7/02 (Date) ~~8/10/05~~

NOTICE OF ALLOWANCE MAILED

2-8-02

☒ b) The term of this patent shall not extend beyond the expiration date of U.S Patent. No. 5,894,554

(Primary Examiner) 2/7/02 (Date)

ISSUE FEE

Amount Due	Date Paid
$1280.00	5-20-02

☐ c) The terminal ____ months of this patent have been disclaimed.

(Legal Instruments Examiner) 2-13-02 (Date)

ISSUE BATCH NUMBER

WARNING:
The information disclosed herein may be restricted. Unauthorized disclosure may be prohibited by the United States Code Title 35, Sections 122, 181 and 368. *Possession outside the U.S. Patent & Trademark Office is restricted to authorized employees and contractors only.*

Form PTO-436A
(Rev. 6/98)

(LABEL AREA)

(FACE)

jc518 U.S. PTO
09/234048
01/19/99

PATENT APPLICATION

09234048

INITIALS _____

CONTENTS

JAN 20 99 45

		Date received (Incl. C. of M.) or Date Mailed			Date received (Incl. C. of M.) or Date Mailed
1.	Application 5 pty papers.		42.		
2.	IDS w/ Refs	1-19-99	43.		
3.	IDS w/ Refs	6-18-99	44.		
4.	Corr. / PR	6/18/99	45.		
5.	[illegible]	[illegible]	46.		
6.	[illegible]	[illegible]	47.		
2/22.	Rej (3 months)	2/23/01	48.		
8.	IDS	5-23-01	49.		
9.	Amdt B	5-23-01	50.		
10.	Terminal Disclaimer	5-23-01	51.		
11.	Rev. 99/A	6-11-01	52.		
12.	Notice of Acceptance	6-18-01	53.		
13.	Req for CPA	3/5/01	54.		
8/24.	Rej (3 months)	8/27/01	55.		
15.	Reconsideration	11-27-01	56.		
16.	Notice Allowability	2-8-02	57.		
17.	Small Entity	5-20-02	58.		
18.	[illegible] (5 sheets)	4-1-02	59.		
19.			60.		
20.			61.		
21.			62.		
22.			63.		
23.			64.		
24.			65.		
25.			66.		
26.			67.		
27.			68.		
28.			69.		
29.			70.		
30.			71.		
31.			72.		
32.			73.		
33.			74.		
34.			75.		
35.			76.		
36.			77.		
37.			78.		
38.			79.		
39.			80.		
40.			81.		
41.			82.		

(LEFT OUTSIDE)

ISSUE SLIP STAPLE AREA (for additional cross references)

POSITION	INITIALS	ID NO.	DATE
FEE DETERMINATION	D.B.	2[illegible]5	1-[illegible]-99
O.I.P.E. CLASSIFIER			
FORMALITY REVIEW	DM	92223	2-4-99

INDEX OF CLAIMS

✔ ..... Rejected
= ..... Allowed
— (Through numeral)... Canceled
÷ ..... Restricted
N ..... Non-elected
I ..... Interference
A ..... Appeal
O ..... Objected

Claim Final	Claim Original	Date									
		3/18/01	8/30/01	2/27/02							
	1										
	2										
	3										
	4										
	5										
	6										
	7										
	8										
	9										
	10										
	11										
	12										
	13										
	14										
	15										
	16										
1	17	✓	✔	=							
2	18	✓	O	=							
3	19	✓	✔	=							
4	20	✓	✔	=							
5	21	✓	✔	=							
6	22	✓	✔	=							
7	23	✓	✔	=							
8	24	✓	✔	=							
9	25	✓	✔	=							
10	26	✓	✔	=							
11	27	✓	✔	=							
12	28	✓	✔	=							
13	29	✓	✔	=							
14	30	✓	✔	=							
15	31	✓	✔	=							
16	32	✓	O	=							
17	33	✓	✔	=							
18	34	✓	✔	=							
19	35	✓	✔	=							
20	36	✓	✔	=							
21	37	✓	✔	=							
22	38	✓	✔	=							
23	39	✓	✔	=							
24	40	✓	✔	=							
25	41	✓	✔	=							
26	42	✓	✔	=							
27	43	✓	✔	=							
28	44	✓	✔	=							
29	45	✓	✔	=							
	46										
	47										
	48										
	49										
	50										

Claim Final	Claim Original	Date
	51	
	52	
	53	
	54	
	55	
	56	
	57	
	58	
	59	
	60	
	61	
	62	
	63	
	64	
	65	
	66	
	67	
	68	
	69	
	70	
	71	
	72	
	73	
	74	
	75	
	76	
	77	
	78	
	79	
	80	
	81	
	82	
	83	
	84	
	85	
	86	
	87	
	88	
	89	
	90	
	91	
	92	
	93	
	94	
	95	
	96	
	97	
	98	
	99	
	100	

Claim Final	Claim Original	Date
	101	
	102	
	103	
	104	
	105	
	106	
	107	
	108	
	109	
	110	
	111	
	112	
	113	
	114	
	115	
	116	
	117	
	118	
	119	
	120	
	121	
	122	
	123	
	124	
	125	
	126	
	127	
	128	
	129	
	130	
	131	
	132	
	133	
	134	
	135	
	136	
	137	
	138	
	139	
	140	
	141	
	142	
	143	
	144	
	145	
	146	
	147	
	148	
	149	
	150	

If more than 150 claims or 10 actions staple additional sheet here

(LEFT INSIDE)

[redacted]

SEARCHED

Class	Sub.	Date	Exmr.
709	238	2/18/01	[illegible]
709	223		
709	219		
710	5		
710	7		
710	20-21	2/18/01	[illegible]
UPDATE	SEARCH	8/10/01	[illegible]
UPDATE	SEARCH	2/7/02	[illegible]

INTERFERENCE SEARCHED

Class	Sub.	Date	Exmr.
709	238	8/10/01 2/7/02	[illegible]
709	223		
709	219		
710	5		
710	7	8/10/01	[illegible]
710	20-21	2/7/02	

SEARCH NOTES
(INCLUDING SEARCH STRATEGY)

	Date	Exmr.
USPT, JPAB, EPAB, DWPI, & TDBD search, See results attached.	2/18/01	[illegible]
USPT, JPAB, EPAB, DWPI, & TDBD search, See results attached.	2/7/02	[illegible]

(RIGHT OUTSIDE)

US006415335B1

(12) **United States Patent**
Lowery et al.

(10) **Patent No.: US 6,415,335 B1**
(45) **Date of Patent: *Jul. 2, 2002**

(54) **SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS**

(75) Inventors: **Keith Lowery**, Richardson; **Andrew B. Levine**, Plano; **Ronald L. Howell**, Rowlett, all of TX (US)

(73) Assignee: **epicRealm Operating Inc.**, Richardson, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/234,048**

(22) Filed: **Jan. 19, 1999**

Related U.S. Application Data

(62) Division of application No. 08/636,477, filed on Apr. 23, 1996, now Pat. No. 5,894,554.

(51) **Int. Cl.**[7] **G06F 13/14**; G06F 13/20

(52) **U.S. Cl.** **710/5**; 710/7; 709/219; 709/223; 709/238

(58) **Field of Search** 709/238, 223, 709/219; 710/5, 7, 20–21

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,866,706	A		9/1989	Christophersen et al.	370/85.7
5,341,499	A		8/1994	Doragh	395/700
5,392,400	A		2/1995	Berkowitz et al.	395/200
5,404,522	A		4/1995	Carmon et al.	395/650
5,404,523	A		4/1995	DellaFera et al.	395/650
5,404,527	A	*	4/1995	Irwin et al.	395/700
5,452,460	A		9/1995	Distelberg et al.	395/700
5,532,838	A		7/1996	Barbari	358/400
5,701,463	A	*	12/1997	Malcolm	395/610
5,751,956	A		5/1998	Kirsch	395/200.33
5,752,246	A	*	5/1998	Rogers et al.	707/10
5,754,772	A	*	5/1998	Leaf	395/200.33
5,761,673	A	*	6/1998	Bookman et al.	707/104
5,774,660	A		6/1998	Brendel et al.	395/200.31
5,774,668	A		6/1998	Choquier et al.	395/200.53

OTHER PUBLICATIONS

Hoffner, 'Inter–operability and distributed application platform design', Web URL:http://www.ansa.co.uk/, 1995, pp. 342–356.*

Mourad et al, 'Scalable Web Server Architectures', IEEE, Jun. 1997, pp. 12–16.*

Dias et al, 'A Scalable and Highly Available Web Server', IEEE, 1996, pp. 85–92.*

'Single System Image and Load Balancing for Network Access to a Loosely Coupled Complex', IBM TDB, vol. 34, Feb. 1992, pp. 464–467.*

Dias, Daniel M., et al.; A Scalable and Highly Available Web Server; IBM Research Division; T.J. Watson Research Center; 7 pages.

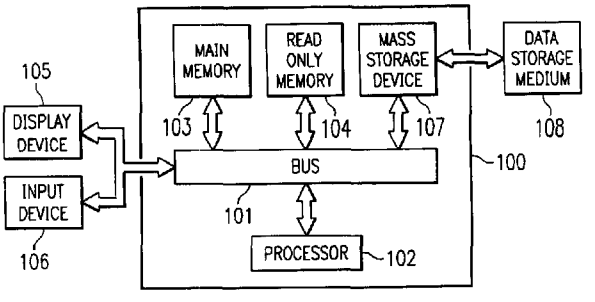
(List continued on next page.)

Primary Examiner—Jeffrey Gaffin
Assistant Examiner—Rehana Perveen
(74) *Attorney, Agent, or Firm*—Baker Botts L.L.P.

(57) **ABSTRACT**

The present invention teaches a method and apparatus for creating and managing custom Web sites. Specifically, one embodiment of the present invention claims a computer-implemented method for managing a dynamic Web page generation request to a Web server, the computer-implemented method comprising the steps of routing the request from the Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, processing the request, the processing being performed by the page server concurrently with the Web server, as the Web server processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources.

29 Claims, 4 Drawing Sheets



OTHER PUBLICATIONS

Andresen, Daniel, Et Al.; Scalability Issues for High Performance Digital Libraries on the World Wide Web; Department of Computer Science; University of California at Santa Barbara; 10 pages.

Andresen, Daniel, Et Al.; SWEB: Towards a Scalable World Wide Web Server on Multicomputers; Department of Computer Science; University of California at Santa Barbara; 7 pages.

Holmedahl, Vegard; Et Al.; Cooperative Caching of Dynamic Content on a Distributed Web Server; Department of Computer Science; University of California at Santa Barbara; 8 pages.

Overson, Nicole; NeXT Ships WebObjects—On Time—As Promised; Deja.com: NeXT Ships WebObjects—On Time—As Promishttp://X28..deja.com/=dnc/ST m=ps...EXT=927585438. 1744765032&hitnum=33.

International Search Report; 7 pages; dated Aug. 21, 1997.

Birman, Kenneth P. and van Renesse, Robbert; Software for Reliable Networks; Scientific American; May 1996; pp. 64–69.

"Beyond the Web: Excavating the Real World Via Mosaic"; Goldberg et al.; Second International WWW Conference; Oct. 17, 1994.

* cited by examiner

FIG. 1

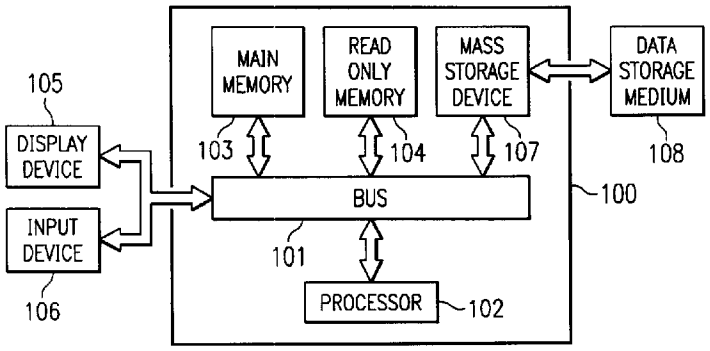


FIG. 2
(PRIOR ART)

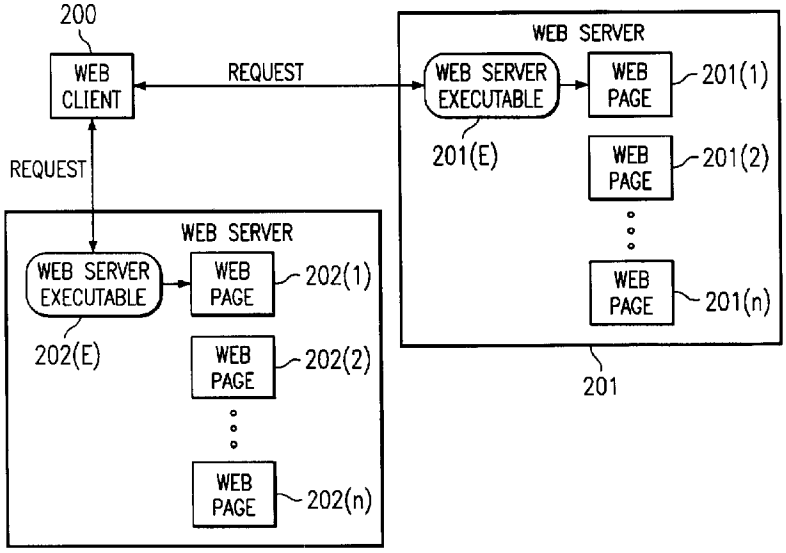
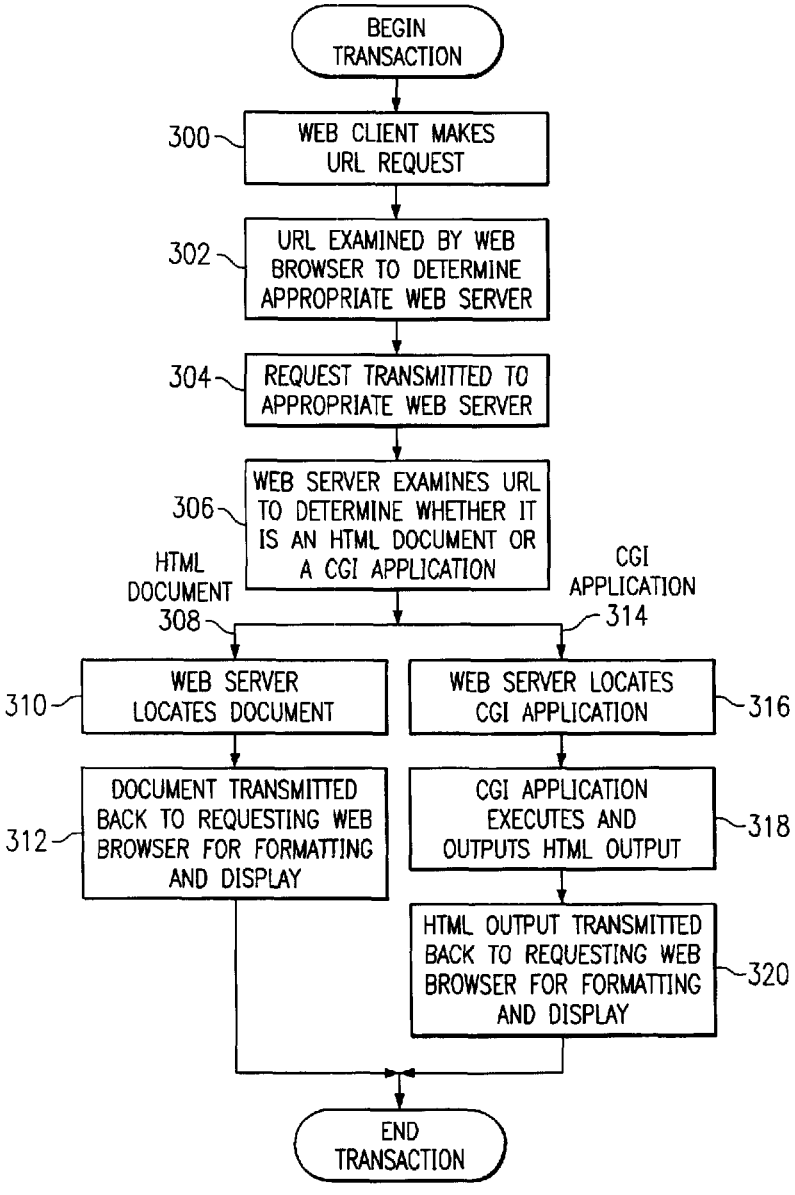


FIG. 3
(PRIOR ART)



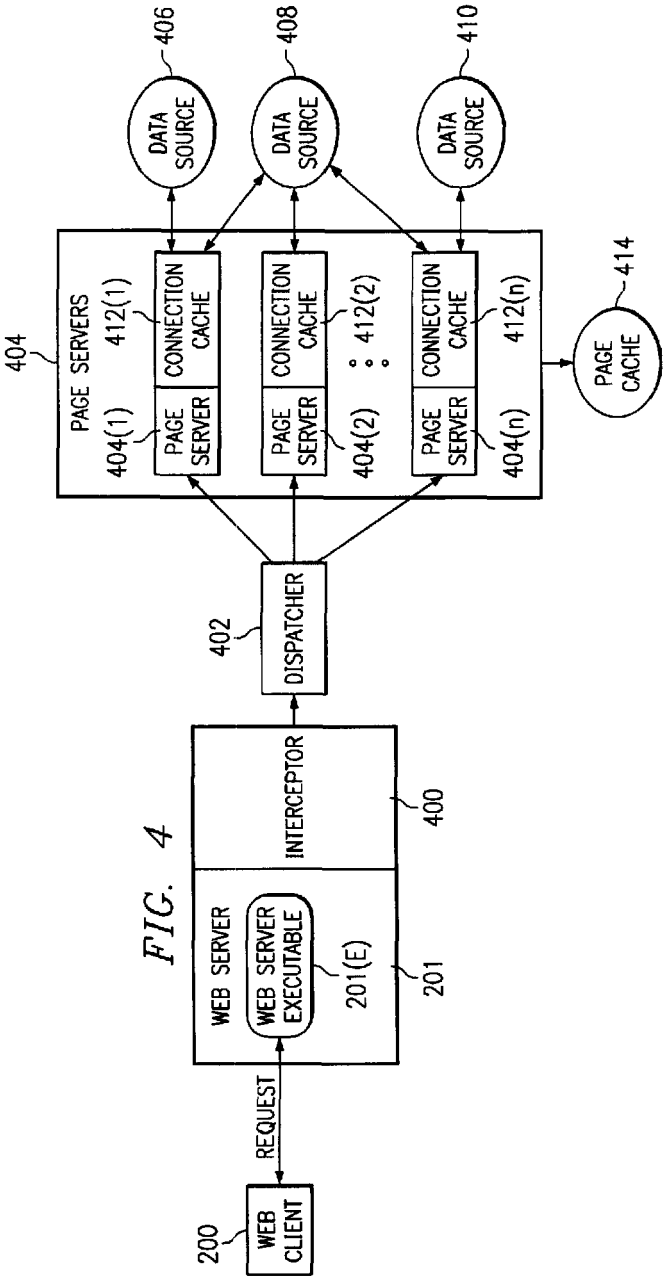
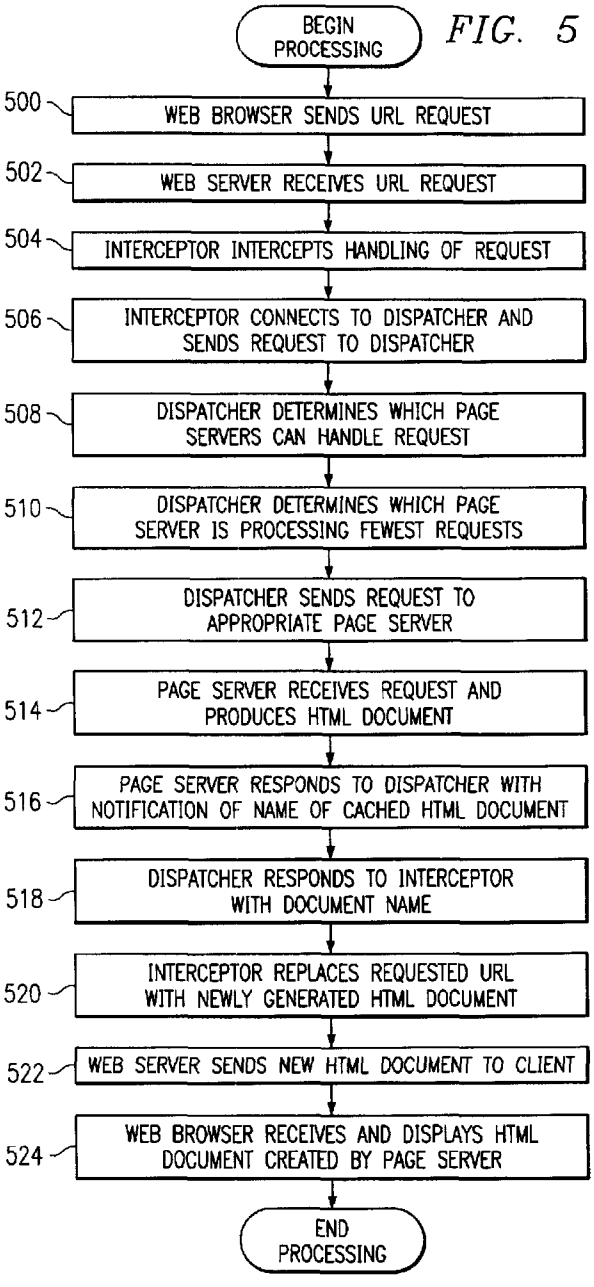

406
DATA SOURCE
408
DATA SOURCE
410
DATA SOURCE
404
PAGE SERVERS
412(1)
CONNECTION CACHE
PAGE SERVER
404(1)
412(2)
404(2)
412(n)
404(n)
414
PAGE CACHE
402
DISPATCHER
INTERCEPTOR
400
WEB SERVER
WEB SERVER EXECUTABLE
201(E)
201
REQUEST
200
WEB CLIENT

FIG. 4

FIG. 5


BEGIN PROCESSING
500
WEB BROWSER SENDS URL REQUEST
502
WEB SERVER RECEIVES URL REQUEST
504
INTERCEPTOR INTERCEPTS HANDLING OF REQUEST
506
INTERCEPTOR CONNECTS TO DISPATCHER AND SENDS REQUEST TO DISPATCHER
508
DISPATCHER DETERMINES WHICH PAGE SERVERS CAN HANDLE REQUEST
510
DISPATCHER DETERMINES WHICH PAGE SERVER IS PROCESSING FEWEST REQUESTS
512
DISPATCHER SENDS REQUEST TO APPROPRIATE PAGE SERVER
514
PAGE SERVER RECEIVES REQUEST AND PRODUCES HTML DOCUMENT
516
PAGE SERVER RESPONDS TO DISPATCHER WITH NOTIFICATION OF NAME OF CACHED HTML DOCUMENT
518
DISPATCHER RESPONDS TO INTERCEPTOR WITH DOCUMENT NAME
520
INTERCEPTOR REPLACES REQUESTED URL WITH NEWLY GENERATED HTML DOCUMENT
522
WEB SERVER SENDS NEW HTML DOCUMENT TO CLIENT
524
WEB BROWSER RECEIVES AND DISPLAYS HTML DOCUMENT CREATED BY PAGE SERVER
END PROCESSING

SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

This application is a division of Ser. No. 08/636,477, filed Apr. 23, 1996, now U.S. Pat. No. 5,894,554.

FIELD OF THE INVENTION

The present invention relates to the field of Internet technology. Specifically, the present invention relates to the creation and management of custom World Wide Web sites.

DESCRIPTION OF RELATED ART

The World Wide Web (the Web) represents all of the computers on the Internet that offer users access to information on the Internet via interactive documents or Web pages. These Web pages contain hypertext links that are used to connect any combination of graphics, audio, video and text, in a non-linear, non-sequential manner. Hypertext links are created using a special software language known as HyperText Mark-Up Language (HTML).

Once created, Web pages reside on the Web, on Web servers or Web sites. A Web site can contain numerous Web pages. Web client machines running Web browsers can access these Web pages at Web sites via a communications protocol known as HyperText Transport Protocol (HTTP). Web browsers are software interfaces that run on World Wide Web clients to allow access to Web sites via a simple user interface. A Web browser allows a Web client to request a particular Web page from a Web site by specifying a Uniform Resource Locator (URL). A URL is a Web address that identifies the Web page and its location on the Web. When the appropriate Web site receives the URL, the Web page corresponding to the requested URL is located, and if required, HTML output is generated. The HTML output is then sent via HTTP to the client for formatting on the client's screen.

Although Web pages and Web sites are extremely simple to create, the proliferation of Web sites on the Internet highlighted a number of problems. The scope and ability of a Web page designer to change the content of the Web page was limited by the static nature of Web pages. Once created, a Web page remained static until it was manually modified. This in turn limited the ability of Web site managers to effectively manage their Web sites.

The Common Gateway Interface (CGI) standard was developed to resolve the problem of allowing dynamic content to be included in Web pages. CGI "calls" or procedures enable applications to generate dynamically created HTML output, thus creating Web pages with dynamic content. Once created, these CGI applications do not have to be modified in order to retrieve "new" or dynamic data. Instead, when the Web page is invoked, CGI "calls" or procedures are used to dynamically retrieve the necessary data and to generate a Web page.

CGI applications also enhanced the ability of Web site administrators to manage Web sites. Administrators no longer have to constantly update static Web pages. A number of vendors have developed tools for CGI based development, to address the issue of dynamic Web page generation. Companies like Spider™ and Bluestone™, for example, have each created development tools for CGI-based Web page development. Another company, Haht Software™, has developed a Web page generation tool that uses a BASIC-like scripting language, instead of a CGI scripting language.

Tools that generate CGI applications do not, however, resolve the problem of managing numerous Web pages and requests at a Web site. For example, a single company may maintain hundreds of Web pages at their Web site. Current Web server architecture also does not allow the Web server to efficiently manage the Web page and process Web client requests. Managing these hundreds of Web pages in a coherent manner and processing all requests for access to the Web pages is thus a difficult task. Existing development tools are limited in their capabilities to facilitate dynamic Web page generation, and do not address the issue of managing Web requests or Web sites.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a method and apparatus for creating and managing custom Web sites. Specifically, the present invention claims a method and apparatus for managing dynamic web page generation requests.

In one embodiment, the present invention claims a computer-implemented method for managing a dynamic Web page generation request to a Web server, the computer-implemented method comprising the steps of routing the request from the Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, processing the request, the processing being performed by the page server concurrently with the Web server, as the Web server processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources. Other embodiments also include connection caches to the one or more data sources, page caches for each page server, and custom HTML extension templates for configuring the Web page.

Other objects, features and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a typical computer system in which the present invention operates.

FIG. **2** illustrates a typical prior art Web server environment.

FIG. **3** illustrates a typical prior art Web server environment in the form of a flow diagram.

FIG. **4** illustrates one embodiment of the presently claimed invention.

FIG. **5** illustrates the processing of a Web browser request in the form of a flow diagram, according to one embodiment of the presently claimed invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention relates to a method and apparatus for creating and managing custom Web sites. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent to one of ordinary skill in the art, however, that these specific details need not be used to practice the present invention. In other instances, well-known structures, interfaces and processes have not been shown in detail in order not to unnecessarily obscure the present invention.

FIG. **1** illustrates a typical computer system **100** in which the present invention operates. The preferred embodiment of

the present invention is implemented on an IBM™ Personal Computer manufactured by IBM Corporation of Armonk, New York. An alternate embodiment may be implemented on an RS/6000™ Workstation manufactured by IBM Corporation of Armonk, New York. It will be apparent to those of ordinary skill in the art that other computer system architectures may also be employed.

In general, such computer systems as illustrated by FIG. **1** comprise a bus **101** for communicating information, a processor **102** coupled with the bus **101** for processing information, main memory **103** coupled with the bus **101** for storing information and instructions for the processor **102**, a read-only memory **104** coupled with the bus **101** for storing static information and instructions for the processor **102**, a display device **105** coupled with the bus **101** for displaying information for a computer user, an input device **106** coupled with the bus **101** for communicating information and command selections to the processor **102**, and a mass storage device **107**, such as a magnetic disk and associated disk drive, coupled with the bus **101** for storing information and instructions. A data storage medium **108** containing digital information is configured to operate with mass storage device **107** to allow processor **102** access to the digital information on data storage medium **108** via bus **101**.

Processor **102** may be any of a wide variety of general purpose processors or microprocessors such as the Pentium™ microprocessor manufactured by Intel™ Corporation or the RS/6000™ processor manufactured by IBM Corporation. It will be apparent to those of ordinary skill in the art, however, that other varieties of processors may also be used in a particular computer system. Display device **105** may be a liquid crystal device, cathode ray tube (CRT), or other suitable display device. Mass storage device **107** may be a conventional hard disk drive, floppy disk drive, CD-ROM drive, or other magnetic or optical data storage device for reading and writing information stored on a hard disk, a floppy disk, a CD-ROM a magnetic tape, or other magnetic or optical data storage medium. Data storage medium **108** may be a hard disk, a floppy disk, a CD-ROM, a magnetic tape, or other magnetic or optical data storage medium.

In general, processor **102** retrieves processing instructions and data from a data storage medium **108** using mass storage device **107** and downloads this information into random access memory **103** for execution. Processor **102**, then executes an instruction stream from random access memory **103** or read-only memory **104**. Command selections and information input at input device **106** are used to direct the flow of instructions executed by processor **102**. Equivalent input device **106** may also be a pointing device such as a conventional mouse or trackball device. The results of this processing execution are then displayed on display device **105**.

The preferred embodiment of the present invention is implemented as a software module, which may be executed on a computer system such as computer system **100** in a conventional manner. Using well known techniques, the application software of the preferred embodiment is stored on data storage medium **108** and subsequently loaded into and executed within computer system **100**. Once initiated, the software of the preferred embodiment operates in the manner described below.

FIG. **2** illustrates a typical prior art Web server environment. Web client **200** can make URL requests to Web server **201** or Web server **202**. Web servers **201** and **202** include Web server executables, **201**(E) and **202**(E) respectively, that perform the processing of Web client requests. Each Web server may have a number of Web pages **201**(1)–(n) and **202**(1)–(n). Depending on the URL specified by the Web client **200**, the request may be routed by either Web server executable **201**(E) to Web page **201** (1), for example, or from Web server executable **202**(E) to Web page **202** (**1**). Web client **200** can continue making URL requests to retrieve other Web pages. Web client **200** can also use hyperlinks within each Web page to "jump" to other Web pages or to other locations within the same Web page.

FIG. **3** illustrates this prior art Web server environment in the form of a flow diagram. In processing block **300**, the Web client makes a URL request. This URL request is examined by the Web browser to determine the appropriate Web server to route the request to in processing block **302**. In processing block **304** the request is then transmitted from the Web browser to the appropriate Web server, and in processing block **306** the Web server executable examines the URL to determine whether it is a HTML document or a CGI application. If the request is for an HTML document **308**, then the Web server executable locates the document in processing block **310**. The document is then transmitted back through the requesting Web browser for formatting and display in processing block **312**.

If the URL request is for a CGI application **314**, however, the Web server executable locates the CGI application in processing block **316**. The CGI application then executes and outputs HTML output in processing block **318** and finally, the HTML output is transmitted back to requesting Web browser for formatting and display in processing block **320**.

This prior art Web server environment does not, however, provide any mechanism for managing the Web requests or the Web sites. As Web sites grow, and as the number of Web clients and requests increase, Web site management becomes a crucial need.

For example, a large Web site may receive thousands of requests or "hits" in a single day. Current Web servers process each of these requests on a single machine, namely the Web server machine. Although these machines may be running "multi-threaded" operating systems that allow transactions to be processed by independent "threads," all the threads are nevertheless on a single machine, sharing a processor. As such, the Web executable thread may hand off a request to a processing thread, but both threads will still have to be handled by the processor on the Web server machine. When numerous requests are being simultaneously processed by multiple threads on a single machine, the Web server can slow down significantly and become highly inefficient. The claimed invention addresses this need by utilizing a partitioned architecture to facilitate the creation and management of custom Web sites and servers.

FIG. **4** illustrates one embodiment of the presently claimed invention. Web client **200** issues a URL request that is processed to determined proper routing. In this embodiment, the request is routed to Web server **201**. Instead of Web server executable **201**(E) processing the URL request, however, Interceptor **400** intercepts the request and routes it to Dispatcher **402**. In one embodiment, Interceptor **400** resides on the Web server machine as an extension to Web server **201**. This embodiment is appropriate for Web servers such as Netsite™ from Netscape, that support such extensions. A number of public domain Web servers, such as NCSA™ from the National Center for Supercomputing Applications at the University of Illinois, Urbana-Champaign, however, do not provide support for

this type of extension. Thus, in an alternate embodiment, Interceptor **400** is an independent module, connected via an "intermediate program" to Web server **201**. This intermediate program can be a simple CGI application program that connects Interceptor **400** to Web server **201**. Alternate intermediate programs the perform the same functionality can also be implemented.

In one embodiment of the invention, Dispatcher **402** resides on a different machine than Web server **201**. This embodiment overcomes the limitation described above, in prior art Web servers, wherein all processing is performed by the processor on a single machine. By routing the request to Dispatcher **402** residing on a different machine than the Web server executable **201**(E), the request can then be processed by a different processor than the Web server executable **201**(E). Web server executable **201**(E) is thus free to continue servicing client requests on Web server **201** while the request is processed "off-line," at the machine on which Dispatcher **402** resides.

Dispatcher **402** can, however, also reside on the same machine as the Web server. The Web site administrator has the option of configuring Dispatcher **402** on the same machine as Web server **201**, taking into account a variety of factors pertinent to a particular Web site, such as the size of the Web site, the number of Web pages and the number of hits at the Web site. Although this embodiment will not enjoy the advantage described above, namely off-loading the processing of Web requests from the Web server machine, the embodiment does allow flexibility for a small Web site to grow. For example, a small Web site administrator can use a single machine for both Dispatcher **402** and Web server **201** initially, then off-load Dispatcher **402** onto a separate machine as the Web site grows. The Web site can thus take advantage of other features of the present invention regardless of whether the site has separate machines configured as Web servers and dispatchers.

Dispatcher **402** receives the intercepted request and then dispatches the request to one of a number of Page servers **404** (1)–(n). For example, if Page server **404** (1) receives the dispatched request, it processes the request and retrieves the data from an appropriate data source, such as data source **406**, data source **408**, or data source **410**. Data sources, as used in the present application, include databases, spreadsheets, files and any other type of data repository. Page server **404** (1) can retrieve data from more than one data source and incorporate the data from these multiple data sources in a single Web page.

In one embodiment, each Page server **404**(1)–(n) resides on a separate machine on the network to distribute the processing of the request. Dispatcher **402** maintains a variety of information regarding each Page server on the network, and dispatches requests based on this information. For example, Dispatcher **402** retains dynamic information regarding the data sources that any given Page server can access. Dispatcher **402** thus examines a particular request and determines which Page servers can service the URL request. Dispatcher **402** then hands off the request to the appropriate Page server.

For example, if the URL request requires financial data from data source **408**, dispatcher **402** will first examine an information list. Dispatcher **402** may determine that Page server **404**(3), for example, has access to the requisite data in data source **408**. Dispatcher **402** will thus route the URL request to Page server **404**(3). This "connection caching" functionality is described in more detail below, under the heading "Performance." Alternately, Dispatcher **402** also has the ability to determine whether a particular Page server already has the necessary data cached in the Page server's page cache (described in more detail below, under the heading "Performance"). Dispatcher **402** may thus determine that Page server **404**(1) and **404**(2) are both logged into Data source **408**, but that Page server **404**(2) has the financial information already cached in Page server **404**(2)'s page cache. In this case, Dispatcher **402** will route the URL request to Page server **404**(2) to more efficiently process the request.

Finally, Dispatcher **402** may determine that a number or all Page servers **404**(1)–(n) are logged into Data source **408**. In this scenario, Dispatcher **402** can examine the number of requests that each Page server is servicing and route the request to the least busy page server. This "load balancing" capability can significantly increase performance at a busy Web site and is discussed in more detail below, under the heading "Scalability".

If, for example, Page server **404**(2), receives the request, Page server **404**(2) will process the request. While Page server **404**(2) is processing the request, Web server executable **201**(E) can concurrently process other Web client requests. This partitioned architecture thus allows both Page server **404**(2) and Web server executable **201**(E) to. simultaneously process different requests, thus increasing the efficiency of the Web site. Page server **404**(2) dynamically generates a Web page in response to the Web client request, and the dynamic Web page is then either transmitted back to requesting Web client **200** or stored on a machine that is accessible to Web server **201**, for later retrieval.

One embodiment of the claimed invention also provides a Web page designer with HTML extensions, or "dyna" tags. These dyna tags provide customized HTML functionality to a Web page designer, to allow the designer to build customized HTML templates that specify the source and placement of retrieved data. For example, in one embodiment, a "dynatext" HTML extension tag specifies a data source and a column name to allow the HTML template to identify the data source to log into and the column name from which to retrieve data. Alternatively, "dyna-anchor" tags allow the designer to build hyperlink queries while "dynablock" tags provide the designer with the ability to iterate through blocks of data. Page servers use these HTML templates to create dynamic Web pages. Then, as described above, these dynamic Web pages are either transmitted back to requesting Web client **200** or stored on a machine that is accessible to Web server **201**, for later retrieval.

The presently claimed invention provides numerous advantages over prior art Web servers, including advantages in the areas of performance, security, extensibility and scalability

Performance

One embodiment of the claimed invention utilizes connection caching and page caching to improve performance. Each Page server can be configured to maintain a cache of connections to numerous data sources. For example, as illustrated in FIG. **4**, Page server **404**(1) can retrieve data from data source **406**, data source **408** or data source **410**. Page server **404**(1) can maintain connection cache **412**(1), containing connections to each of data source **406**, data source **408** and data source **410**, thus eliminating connect times from the Page servers to those data sources.

Additionally, another embodiment of the present invention supports the caching of finished Web pages, to optimize the performance of the data source being utilized. This "page

caching" feature, illustrated in FIG. **4** as Page cache **414**, allows the Web site administrator to optimize the performance of data sources by caching Web pages that are repeatedly accessed. Once the Web page is cached, subsequent requests or "hits" will utilize the cached Web page rather than re-accessing the data source. This can radically improve the performance of the data source.

Security

The present invention allows the Web site administrator to utilize multiple levels of security to manage the Web site. In one embodiment, the Page server can utilize all standard encryption and site security features provided by the Web server. In another embodiment, the Page server can be configured to bypass connection caches **412(1)**–(n), described above, for a particular data source and to require entry of a user-supplied identification and password for the particular data source the user is trying to access.

Additionally, another embodiment of the presently claimed invention requires no real-time access of data sources. The Web page caching ability, described above, enables additional security for those sites that want to publish non-interactive content from internal information systems, but do not want real-time Internet accessibility to those internal information systems. In this instance, the Page server can act as a "replication and staging agent" and create Web pages in batches, rather than in real-time. These "replicated" Web pages are then "staged" for access at a later time, and access to the Web pages in this scenario is possible even if the Page server and dispatcher are not present later.

In yet another embodiment, the Page server can make a single pass through a Web library, and compile a Web site that exists in the traditional form of separately available files. A Web library is a collection of related Web books and Web pages. More specifically, the Web library is a hierarchical organization of Web document templates, together with all the associated data source information. Information about an entire Web site is thus contained in a single physical file, thus simplifying the problem of deploying Web sites across multiple Page servers. The process of deploying the Web site in this embodiment is essentially a simple copy of a single file.

Extensibility

One embodiment of the present invention provides the Web site administrator with Object Linking and Embedding (OLE) 2.0 extensions to extend the page creation process. These OLE 2.0 extensions also allow information submitted over the Web to be processed with user-supplied functionality. Utilizing development tools such as Visual Basic, Visual C++ or PowerBuilder that support the creation of OLE 2.0 automation, the Web site administrator can add features and modify the behavior of the Page servers described above. This extensibility allows one embodiment of the claimed invention to be incorporated with existing technology to develop an infinite number of custom web servers.

For example, OLE 2.0 extensions allow a Web site administrator to encapsulate existing business rules in an OLE 2.0 automation interface, to be accessed over the Web. One example of a business rule is the steps involved in the payoff on an installment or mortgage loan. The payoff may involve, for example, taking into account the current balance, the date and the interest accrued since the last payment. Most organizations already have this type of business rule implemented using various applications, such as Visual Basic for client-server environments, or CICS programs on mainframes. If these applications are OLE 2.0 compliant, the Page server "dynaobject" HTML extension tag can be used to encapsulated the application in an OLE 2.0 automation interface. The Page server is thus extensible, and can incorporate the existing application with the new Page server functionality.

Scalability

One embodiment of the claimed invention allows "plug and play" scalability. As described above, referring to FIG. **4**, Dispatcher **402** maintains information about all the Page servers configured to be serviced by Dispatcher **402**. Any number of Page servers can thus be "plugged" into the configuration illustrated in FIG. **4**, and the Page servers will be instantly activated as the information is dynamically updated in Dispatcher **402**. The Web site administrator can thus manage the overhead of each Page server and modify each Page server's load, as necessary, to improve performance. In this manner, each Page server will cooperate with other Page servers within a multi-server environment. Dispatcher **402** can examine the load on each Page server and route new requests according to each Page server's available resources. This "load-balancing" across multiple Page servers can significantly increase a Web site's performance.

FIG. **5** illustrates the processing of a Web browser request in the form of a flow diagram, according to one embodiment of the presently claimed invention. A Web browser sends a URL request to a Web server in processing block **500**. In processing block **502**, the Web server receives the URL request, and an interceptor then intercepts the handling of the request in processing block **504**. The interceptor connects to a dispatcher and sends the URL request to the dispatcher in processing block **506**. In processing block **508**, the dispatcher determines which Page servers can handle the request. The dispatcher also determines which Page server is processing the fewest requests in processing block **510**, and in processing block **512**, the dispatcher sends the URL request to an appropriate Page server. The Page server receives the request and produces an HTML document in processing block **514**. The Page server then responds to the dispatcher with notification of the name of the cached HTML document in processing block **516**. In processing block **518**, the dispatcher responds to the interceptor with the document name, and the interceptor then replaces the requested URL with the newly generated HTML document in processing block **520**. The Web server then sends the new HTML document to the requesting client in processing block **522**. Finally, the Web browser receives and displays the HTML document created by the Page server at processing block **524**.

Thus, a method and apparatus for creating and managing custom Web sites is disclosed. These specific arrangements and methods described herein are merely illustrative of the principles of the present invention. Numerous modifications in form and detail may be made by those of ordinary skill in the art without departing from the scope of the present invention. Although this invention has been shown in relation to a particular preferred embodiment, it should not be considered so limited. Rather, the present invention is limited only by the scope of the appended claims.

We claim:

1. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing a request from a Web server to a page server, said page server receiving said request and releasing said

Web server to process other requests wherein said routing step further includes the steps of:

intercepting said request at said Web server and routing said request to said page server;

processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

2. The computer-implemented method in claim **1** wherein said step of routing said request includes the steps of:

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

3. The computer-implemented method in claim **1** wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

4. The computer-implemented method in claim **1** wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

5. The computer-implemented method in claim **1** wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

6. The computer-implemented method in claim **1** wherein said step of processing said request includes the step of logging into said one or more data sources.

7. The computer-implemented method in claim **1** wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

8. The computer-implemented method in claim **1** wherein said page server includes tag-based text templates for configuring said Web page.

9. The computer-implemented method in claim **8** wherein said step of processing said request further includes the step of inserting said-dynamically retrieved data from said one or more data sources into said tag-based text templates.

10. The computer-implemented method in claim **8** wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

11. The computer-implemented method in claim **8** wherein said tag-based text templates include HTML templates.

12. The computer-implemented method in claim **1** wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

13. The computer-implemented method in claim **1** wherein said step of processing said request further includes the step of processing an object handling extension.

14. The computer-implemented method in claim **13** wherein said object handling extension is an OLE extension.

15. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a page server, said page server receiving said request and releasing said HTTP-compliant device to process other requests wherein said transferring step further includes the steps of:

intercepting said request at said HTTP-compliant device and transferring said request to said page server;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

16. The computer-implemented method in claim **15** wherein said step of transferring said request includes the steps of:

transferring said request from said HTTP-compliant device to a dispatcher; and

dispatching said request to said page server.

17. The computer-implemented method in claim **15** wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

18. The computer-implemented method in claim **15** wherein said step of dynamically generating said page includes the step of dynamically retrieving said data from said one or more data sources.

19. The computer-implemented method in claim **15** wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

20. The computer-implemented method in claim **15** wherein said step of processing said request includes the step of logging into said one or more data sources.

21. The computer-implemented method in claim **15** wherein said step of dynamically generating said page includes the step of maintaining a page cache containing said page.

22. The computer-implemented method in claim **15** wherein said page server includes tag-based text templates for configuring said page.

23. The computer-implemented method in claim **22** wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

24. The computer-implemented method in claim **22** wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

25. The computer-implemented method in claim **22** wherein said tag-based text templates include HTML templates.

26. The computer-implemented method in claim **15** wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

27. The computer-implemented method in claim **15** wherein said step of processing said request further includes the step of processing an object handling extension.

28. The computer-implemented method in claim **27** wherein said object handling extension is an OLE extension.

29. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a dispatcher;

maintaining dynamic information regarding data sources a given page server may access;

dispatching said request to an appropriate page server based on said request and based on said dynamic information, said page server receiving said request and

releasing said HTTP-compliant device to process other requests;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

* * * * *

PATENT APPLICATION SERIAL NO. _____

U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE
FEE RECORD SHEET

01/28/1999 DBUTLER 00000034 09234048

01 FC:101 760.00 OP
02 FC:102 234.00 OP

PTO-1556
(5/87)

02577.P001

PATENT



UNITED STATES PATENT APPLICATION

for

A METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE

Applicant:

Keith Lowery
Andrew B. Levine
Ronald L. Howell

prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
12400 Wilshire Boulevard
Los Angeles, CA 90026-1026
(408) 720-8598

EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number EM 511 190 030 US

Date of Deposit April 23, 1996

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner of Patents and Trademarks, Washington, D.C. 20231.

Rhonda Olmo
(Typed or printed name of person mailing paper or fee)

Rhonda Olmo
(Signature of person mailing paper or fee)

ABSTRACT OF THE DISCLOSURE

The present invention teaches a method and apparatus for creating and managing custom Web sites. Specifically, one embodiment of the present invention claims a computer-implemented method for managing a dynamic Web page generation request to a Web server, the computer-implemented method comprising the steps of routing the request from the Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, processing the request, the processing being performed by the page server concurrently with the Web server, as the Web server processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources.

EPIC000343

a This application is a division of Ser. No. 08/636,477, filed Apr. 23, 1996, now U.S. Pat. No. 5,894,554.

FIELD OF THE INVENTION

The present invention relates to the field of Internet technology. Specifically, the present invention relates to the creation and management of custom World Wide Web sites.

DESCRIPTION OF RELATED ART

The World Wide Web (the Web) represents all of the computers on the Internet that offer users access to information on the Internet via interactive documents or Web pages. These Web pages contain hypertext links that are used to connect any combination of graphics, audio, video and text, in a non-linear, non-sequential manner. Hypertext links are created using a special software language known as HyperText Mark-Up Language (HTML).

Once created, Web pages reside on the Web, on Web servers or Web sites. A Web site can contain numerous Web pages. Web client machines running Web browsers can access these Web pages at Web sites via a communications protocol known as HyperText Transport Protocol (HTTP). Web browsers are software interfaces that run on World Wide Web clients to allow access to Web sites via a simple user interface. A Web browser allows a Web client to request a particular Web page from a Web site by specifying a Uniform Resource Locator (URL). A URL is a Web address that identifies the Web page and its location on the Web. When the appropriate Web site receives the URL, the Web page corresponding to the requested URL is located, and if required, HTML output is generated. The HTML output is

then sent via HTTP to the client for formatting on the client's screen.

Although Web pages and Web sites are extremely simple to create, the proliferation of Web sites on the Internet highlighted a number of problems. The scope and ability of a Web page designer to change the content of the Web page was limited by the static nature of Web pages. Once created, a Web page remained static until it was manually modified. This in turn limited the ability of Web site managers to effectively manage their Web sites.

The Common Gateway Interface (CGI) standard was developed to resolve the problem of allowing dynamic content to be included in Web pages. CGI "calls" or procedures enable applications to generate dynamically created HTML output, thus creating Web pages with dynamic content. Once created, these CGI applications do not have to be modified in order to retrieve "new" or dynamic data. Instead, when the Web page is invoked, CGI "calls" or procedures are used to dynamically retrieve the necessary data and to generate a Web page.

CGI applications also enhanced the ability of Web site administrators to manage Web sites. Administrators no longer have to constantly update static Web pages. A number of vendors have developed tools for CGI based development, to address the issue of dynamic Web page generation. Companies like Spider™ and Bluestone™, for example, have each created development tools for CGI-based Web page development. Another company, Haht Software™, has developed a Web page generation tool that uses a BASIC-like scripting language, instead of a CGI scripting language.

EPIC000345

Tools that generate CGI applications do not, however, resolve the problem of managing numerous Web pages and requests at a Web site. For example, a single company may maintain hundreds of Web pages at their Web site. Current Web server architecture also does not allow the Web server to efficiently manage the Web page and process Web client requests. Managing these hundreds of Web pages in a coherent manner and processing all requests for access to the Web pages is thus a difficult task. Existing development tools are limited in their capabilities to facilitate dynamic Web page generation, and do not address the issue of managing Web requests or Web sites.

Exhibit 1 - Stipulated

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a method and apparatus for creating and managing custom Web sites. Specifically, the present invention claims a method and apparatus for managing dynamic web page generation requests.

In one embodiment, the present invention claims a computer-implemented method for managing a dynamic Web page generation request to a Web server, the computer-implemented method comprising the steps of routing the request from the Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, processing the request, the processing being performed by the page server concurrently with the Web server, as the Web server processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources. Other embodiments also include connection caches to the one or more data sources, page caches for each page server, and custom HTML extension templates for configuring the Web page.

Other objects, features and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description.

EPIC000347

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates a typical computer system in which the present invention operates.

Figure 2 illustrates a typical prior art Web server environment.

Figure 3 illustrates a typical prior art Web server environment in the form of a flow diagram.

Figure 4 illustrates one embodiment of the presently claimed invention.

Figure 5 illustrates the processing of a Web browser request in the form of a flow diagram, according to one embodiment of the presently claimed invention.

EPIC000348

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention relates to a method and apparatus for creating and managing custom Web sites. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent to one of ordinary skill in the art, however, that these specific details need not be used to practice the present invention. In other instances, well-known structures, interfaces and processes have not been shown in detail in order not to unnecessarily obscure the present invention.

Figure 1 illustrates a typical computer system 100 in which the present invention operates. The preferred embodiment of the present invention is implemented on an IBM™ Personal Computer manufactured by IBM Corporation of Armonk, New York. An alternate embodiment may be implemented on an RS/6000™ Workstation manufactured by IBM Corporation of Armonk, New York. It will be apparent to those of ordinary skill in the art that other computer system architectures may also be employed.

In general, such computer systems as illustrated by **Figure 1** comprise a bus 101 for communicating information, a processor 102 coupled with the bus 101 for processing information, main memory 103 coupled with the bus 101 for storing information and instructions for the processor 102, a read-only memory 104 coupled with the bus 101 for storing static information and instructions for the processor 102, a display device 105 coupled with the bus 101 for displaying information for a computer user, an input device 106

EPIC000349

coupled with the bus 101 for communicating information and command selections to the processor 102, and a mass storage device 107, such as a magnetic disk and associated disk drive, coupled with the bus 101 for storing information and instructions. A data storage medium 108 containing digital information is configured to operate with mass storage device 107 to allow processor 102 access to the digital information on data storage medium 108 via bus 101.

Processor 102 may be any of a wide variety of general purpose processors or microprocessors such as the Pentium™ microprocessor manufactured by Intel™ Corporation or the RS/6000™ processor manufactured by IBM Corporation. It will be apparent to those of ordinary skill in the art, however, that other varieties of processors may also be used in a particular computer system. Display device 105 may be a liquid crystal device, cathode ray tube (CRT), or other suitable display device. Mass storage device 107 may be a conventional hard disk drive, floppy disk drive, CD-ROM drive, or other magnetic or optical data storage device for reading and writing information stored on a hard disk, a floppy disk, a CD-ROM a magnetic tape, or other magnetic or optical data storage medium. Data storage medium 108 may be a hard disk, a floppy disk, a CD-ROM, a magnetic tape, or other magnetic or optical data storage medium.

In general, processor 102 retrieves processing instructions and data from a data storage medium 108 using mass storage device 107 and downloads this information into random access memory 103 for execution. Processor 102, then executes an instruction stream from random access memory 103 or read-only memory 104. Command selections and

Exhibit 5 Page 26 of 69

information input at input device 106 are used to direct the flow of instructions executed by processor 102. Equivalent input device 106 may also be a pointing device such as a conventional mouse or trackball device. The results of this processing execution are then displayed on display device 105.

The preferred embodiment of the present invention is implemented as a software module, which may be executed on a computer system such as computer system 100 in a conventional manner. Using well known techniques, the application software of the preferred embodiment is stored on data storage medium 108 and subsequently loaded into and executed within computer system 100. Once initiated, the software of the preferred embodiment operates in the manner described below.

Figure 2 illustrates a typical prior art Web server environment. Web client 200 can make URL requests to Web server 201 or Web server 202. Web servers 201 and 202 include Web server executables, 201(E) and 202(E) respectively, that perform the processing of Web client requests. Each Web server may have a number of Web pages 201(1) - (n) and 202(1) - (n). Depending on the URL specified by the Web client 200, the request may be routed by either Web server executable 201(E) to Web page 201 (1), for example, or from Web server executable 202(E) to Web page 202 (1). Web client 200 can continue making URL requests to retrieve other Web pages. Web client 200 can also use hyperlinks within each Web page to "jump" to other Web pages or to other locations within the same Web page.

Figure 3 illustrates this prior art Web server environment in the form of a flow diagram. In processing block 300, the Web client makes a URL

EPIC000351

request. This URL request is examined by the Web browser to determine the appropriate Web server to route the request to in processing block 302. In processing block 304 the request is then transmitted from the Web browser to the appropriate Web server, and in processing block 306 the Web server executable examines the URL to determine whether it is a HTML document or a CGI application. If the request is for an HTML document 308, then the Web server executable locates the document in processing block 310. The document is then transmitted back through the requesting Web browser for formatting and display in processing block 312.

If the URL request is for a CGI application 314, however, the Web server executable locates the CGI application in processing block 316. The CGI application then executes and outputs HTML output in processing block 318 and finally, the HTML output is transmitted back to requesting Web browser for formatting and display in processing block 320.

This prior art Web server environment does not, however, provide any mechanism for managing the Web requests or the Web sites. As Web sites grow, and as the number of Web clients and requests increase, Web site management becomes a crucial need.

For example, a large Web site may receive thousands of requests or "hits" in a single day. Current Web servers process each of these requests on a single machine, namely the Web server machine. Although these machines may be running "multi-threaded" operating systems that allow transactions to be processed by independent "threads," all the threads are nevertheless on a single machine, sharing a processor. As such, the Web

executable thread may hand off a request to a processing thread, but both threads will still have to be handled by the processor on the Web server machine. When numerous requests are being simultaneously processed by multiple threads on a single machine, the Web server can slow down significantly and become highly inefficient. The claimed invention addresses this need by utilizing a partitioned architecture to facilitate the creation and management of custom Web sites and servers.

Figure 4 illustrates one embodiment of the presently claimed invention. Web client 200 issues a URL request that is processed to determined proper routing. In this embodiment, the request is routed to Web server 201. Instead of Web server executable 201(E) processing the URL request, however, Interceptor 400 intercepts the request and routes it to Dispatcher 402. In one embodiment, Interceptor 400 resides on the Web server machine as an extension to Web server 201. This embodiment is appropriate for Web servers such as Netsite™ from Netscape, that support such extensions. A number of public domain Web servers, such as NCSA™ from the National Center for Supercomputing Applications at the University of Illinois, Urbana-Champaign, however, do not provide support for this type of extension. Thus, in an alternate embodiment, Interceptor 400 is an independent module, connected via an "intermediate program" to Web server 201. This intermediate program can be a simple CGI application program that connects Interceptor 400 to Web server 201. Alternate intermediate programs the perform the same functionality can also be implemented.

In one embodiment of the invention, Dispatcher 402 resides on a

EPIC000353

different machine than Web server 201. This embodiment overcomes the limitation described above, in prior art Web servers, wherein all processing is performed by the processor on a single machine. By routing the request to Dispatcher 402 residing on a different machine than the Web server executable 201(E), the request can then be processed by a different processor than the Web server executable 201(E). Web server executable 201(E) is thus free to continue servicing client requests on Web server 201 while the request is processed "off-line," at the machine on which Dispatcher 402 resides.

Dispatcher 402 can, however, also reside on the same machine as the Web server. The Web site administrator has the option of configuring Dispatcher 402 on the same machine as Web server 201, taking into account a variety of factors pertinent to a particular Web site, such as the size of the Web site, the number of Web pages and the number of hits at the Web site. Although this embodiment will not enjoy the advantage described above, namely off-loading the processing of Web requests from the Web server machine, the embodiment does allow flexibility for a small Web site to grow. For example, a small Web site administrator can use a single machine for both Dispatcher 402 and Web server 201 initially, then off-load Dispatcher 402 onto a separate machine as the Web site grows. The Web site can thus take advantage of other features of the present invention regardless of whether the site has separate machines configured as Web servers and dispatchers.

Dispatcher 402 receives the intercepted request and then dispatches the request to one of a number of Page servers 404 (1) - (n). For example, if Page server 404 (1) receives the dispatched request, it processes the request and

EPIC000354

retrieves the data from an appropriate data source, such as data source 406, data source 408, or data source 410. Data sources, as used in the present application, include databases, spreadsheets, files and any other type of data repository. Page server 404 (1) can retrieve data from more than one data source and incorporate the data from these multiple data sources in a single Web page.

In one embodiment, each Page server 404(1) - (n) resides on a separate machine on the network to distribute the processing of the request. Dispatcher 402 maintains a variety of information regarding each Page server on the network, and dispatches requests based on this information. For example, Dispatcher 402 retains dynamic information regarding the data sources that any given Page server can access. Dispatcher 402 thus examines a particular request and determines which Page servers can service the URL request. Dispatcher 402 then hands off the request to the appropriate Page server.

For example, if the URL request requires financial data from data source 408, dispatcher 402 will first examine an information list. Dispatcher 402 may determine that Page server 404(3), for example, has access to the requisite data in data source 408. Dispatcher 402 will thus route the URL request to Page server 404(3). This "connection caching" functionality is described in more detail below, under the heading "Performance."

Alternately, Dispatcher 402 also has the ability to determine whether a particular Page server already has the necessary data cached in the Page server's page cache (described in more detail below, under the heading

EPIC000355

"Performance"). Dispatcher 402 may thus determine that Page server 404(1) and 404(2) are both logged into Data source 408, but that Page server 404(2) has the financial information already cached in Page server 404(2)'s page cache. In this case, Dispatcher 402 will route the URL request to Page server 404(2) to more efficiently process the request.

Finally, Dispatcher 402 may determine that a number or all Page servers 404(1) - (n) are logged into Data source 408. In this scenario, Dispatcher 402 can examine the number of requests that each Page server is servicing and route the request to the least busy page server. This "load balancing" capability can significantly increase performance at a busy Web site and is discussed in more detail below, under the heading "Scalability".

If, for example, Page server 404(2), receives the request, Page server 404(2) will process the request. While Page server 404(2) is processing the request, Web server executable 201(E) can concurrently process other Web client requests. This partitioned architecture thus allows both Page server 404(2) and Web server executable 201(E) to simultaneously process different requests, thus increasing the efficiency of the Web site. Page server 404(2) dynamically generates a Web page in response to the Web client request, and the dynamic Web page is then either transmitted back to requesting Web client 200 or stored on a machine that is accessible to Web server 201, for later retrieval.

One embodiment of the claimed invention also provides a Web page designer with HTML extensions, or "dyna" tags. These dyna tags provide customized HTML functionality to a Web page designer, to allow the

EPIC000356

designer to build customized HTML templates that specify the source and placement of retrieved data. For example, in one embodiment, a "dynatext" HTML extension tag specifies a data source and a column name to allow the HTML template to identify the data source to log into and the column name from which to retrieve data. Alternatively, "dyna-anchor" tags allow the designer to build hyperlink queries while "dynablock" tags provide the designer with the ability to iterate through blocks of data. Page servers use these HTML templates to create dynamic Web pages. Then, as described above, these dynamic Web pages are either transmitted back to requesting Web client 200 or stored on a machine that is accessible to Web server 201, for later retrieval.

The presently claimed invention provides numerous advantages over prior art Web servers, including advantages in the areas of performance, security, extensibility and scalability.

Performance

One embodiment of the claimed invention utilizes connection caching and page caching to improve performance. Each Page server can be configured to maintain a cache of connections to numerous data sources. For example, as illustrated in **Figure 4**, Page server 404(1) can retrieve data from data source 406, data source 408 or data source 410. Page server 404(1) can maintain connection cache 412(1), containing connections to each of data source 406, data source 408 and data source 410, thus eliminating connect times from the Page servers to those data sources.

EPIC000357

Additionally, another embodiment of the present invention supports the caching of finished Web pages, to optimize the performance of the data source being utilized. This "page caching" feature, illustrated in **Figure 4** as Page cache 414, allows the Web site administrator to optimize the performance of data sources by caching Web pages that are repeatedly accessed. Once the Web page is cached, subsequent requests or "hits" will utilize the cached Web page rather than re-accessing the data source. This can radically improve the performance of the data source.

Security

The present invention allows the Web site administrator to utilize multiple levels of security to manage the Web site. In one embodiment, the Page server can utilize all standard encryption and site security features provided by the Web server. In another embodiment, the Page server can be configured to bypass connection caches 412(1)-(n), described above, for a particular data source and to require entry of a user-supplied identification and password for the particular data source the user is trying to access.

Additionally, another embodiment of the presently claimed invention requires no real-time access of data sources. The Web page caching ability, described above, enables additional security for those sites that want to publish non-interactive content from internal information systems, but do not want real-time Internet accessibility to those internal information systems. In this instance, the Page server can act as a "replication and staging

agent" and create Web pages in batches, rather than in real-time. These "replicated" Web pages are then "staged" for access at a later time, and access to the Web pages in this scenario is possible even if the Page server and dispatcher are not present later.

In yet another embodiment, the Page server can make a single pass through a Web library, and compile a Web site that exists in the traditional form of separately available files. A Web library is a collection of related Web books and Web pages. More specifically, the Web library is a hierarchical organization of Web document templates, together with all the associated data source information. Information about an entire Web site is thus contained in a single physical file, thus simplifying the problem of deploying Web sites across multiple Page servers. The process of deploying the Web site in this embodiment is essentially a simple copy of a single file.

Extensibility

One embodiment of the present invention provides the Web site administrator with Object Linking and Embedding (OLE) 2.0 extensions to extend the page creation process. These OLE 2.0 extensions also allow information submitted over the Web to be processed with user-supplied functionality. Utilizing development tools such as Visual Basic, Visual C++ or PowerBuilder that support the creation of OLE 2.0 automation, the Web site administrator can add features and modify the behavior of the Page servers described above. This extensibility allows one embodiment of the claimed invention to be incorporated with existing technology to develop an

EPIC000359

infinite number of custom web servers.

For example, OLE 2.0 extensions allow a Web site administrator to encapsulate existing business rules in an OLE 2.0 automation interface, to be accessed over the Web. One example of a business rule is the steps involved in the payoff on an installment or mortgage loan. The payoff may involve, for example, taking into account the current balance, the date and the interest accrued since the last payment. Most organizations already have this type of business rule implemented using various applications, such as Visual Basic for client-server environments, or CICS programs on mainframes. If these applications are OLE 2.0 compliant, the Page server "dynaobject" HTML extension tag can be used to encapsulated the application in an OLE 2.0 automation interface. The Page server is thus extensible, and can incorporate the existing application with the new Page server functionality.

Scalability

One embodiment of the claimed invention allows "plug and play" scalability. As described above, referring to **Figure 4**, Dispatcher 402 maintains information about all the Page servers configured to be serviced by Dispatcher 402. Any number of Page servers can thus be "plugged" into the configuration illustrated in **Figure 4**, and the Page servers will be instantly activated as the information is dynamically updated in Dispatcher 402. The Web site administrator can thus manage the overhead of each Page server and modify each Page server's load, as necessary, to improve performance. In this manner, each Page server will cooperate with other Page servers

within a multi-server environment. Dispatcher 402 can examine the load on each Page server and route new requests according to each Page server's available resources. This "load-balancing" across multiple Page servers can significantly increase a Web site's performance.

Figure 5 illustrates the processing of a Web browser request in the form of a flow diagram, according to one embodiment of the presently claimed invention. A Web browser sends a URL request to a Web server in processing block 500. In processing block 502, the Web server receives the URL request, and an interceptor then intercepts the handling of the request in processing block 504. The interceptor connects to a dispatcher and sends the URL request to the dispatcher in processing block 506. In processing block 508, the dispatcher determines which Page servers can handle the request. The dispatcher also determines which Page server is processing the fewest requests in processing block 510, and in processing block 512, the dispatcher sends the URL request to an appropriate Page server. The Page server receives the request and produces an HTML document in processing block 514. The Page server then responds to the dispatcher with notification of the name of the cached HTML document in processing block 516. In processing block 518, the dispatcher responds to the interceptor with the document name, and the interceptor then replaces the requested URL with the newly generated HTML document in processing block 520. The Web server then sends the new HTML document to the requesting client in processing block 522. Finally, the Web browser receives and displays the HTML document created by the Page server at processing block 524.

Thus, a method and apparatus for creating and managing custom Web

EPIC000361

sites is disclosed. These specific arrangements and methods described herein are merely illustrative of the principles of the present invention. Numerous modifications in form and detail may be made by those of ordinary skill in the art without departing from the scope of the present invention. Although this invention has been shown in relation to a particular preferred embodiment, it should not be considered so limited. Rather, the present invention is limited only by the scope of the appended claims.

Case 1:06-cv-00414-SLR Document 246-2 Filed 08/07/2008 Page 39 of 69

CLAIMS

We claim:

1. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing said request from said Web server to a page server, said page server receiving said request and releasing said Web server to process other requests;

processing said request, said processing being performed by said page server concurrently with said Web server, as said Web server processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

2. The computer-implemented method in Claim 1 wherein said step of routing said request includes the steps of:

intercepting said request at said Web server;

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

EPIC000363

3. The computer-implemented method in Claim 2 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

4. The computer-implemented method in Claim 3 wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

5. The computer-implemented method in Claim 4 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

6. The computer-implemented method in Claim 4 wherein said step of processing said request includes the step of logging into said one or more data sources.

7. The computer-implemented method in Claim 4 wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

8. The computer-implemented method in Claim 4 wherein said page server includes custom HTML extension templates for configuring said Web page.

EPIC000364

9. The computer-implemented method in Claim 8 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said custom HTML extension templates.

10. A computer-implemented method for managing a Web site including one or more Web servers, each Web server having one or more Web pages, said computer-implemented method comprising the steps of:

creating a master Web management page wherein said master Web management page includes said one or more Web pages from each of said one or more Web servers at said Web site; and

utilizing said Web management page to recreate said Web site at a new location.

11. A networked system for managing a dynamic Web page generation request, said system comprising:

one or more data sources;

a page server having a processing means;

a first computer system including means for generating said request;

a second computer system including means for receiving said request from said first computer, said second computer system also including a router, said router routing said request from said second computer system to said page server, said page server receiving said request and releasing said

EPIC000365

second computer system to process other requests, said page server processing means processing said request and dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from said one or more data sources.

12. The networked system in Claim 11 wherein said router in said second computer system includes:

an interceptor intercepting said request at said second computer system and routing said request; and

a dispatcher receiving said routed request from said interceptor and dispatching said request to said page server.

13. A Web server for receiving a dynamic Web page generation request, said Web server comprising:

a Web server executable receiving said request; and

intercepting means for intercepting said request and routing said request from said Web server executable to a page server.

14. A page server for receiving a dynamic Web page generation request, said page server comprising;

means for receiving said request;

means for maintaining a connection cache to one or more data sources; and

means for processing said request.

EPIC000366

15. The page server in Claim 14 wherein said means for processing said request further includes means for dynamically retrieving data from said one or more data sources.

16. An article of manufacture comprising a computer readable medium having computer instructions embodied therein for causing a processor to manage a dynamic Web page generation request to a Web server, said computer instructions in said article of manufacture causing said processor to perform the steps of:

routing said request from said Web server to a page server, said page server receiving said request and releasing said Web server to process other requests;

processing said request, said processing being performed by said page server; and

dynamically generating a Web page, said Web page including data retrieved from one or more data sources.

EPIC000367

Attorney's Docket No.: 02577.P001 PATENT

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below, next to my name.

I believe I am the original, first, and sole inventor (if only one name is listed below) or an original, first, and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

A METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE

the specification of which

__X__ is attached hereto.

_____ was filed on _____ as United States Application Number _____ or PCT International Application Number_____ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claim(s), as amended by any amendment referred to above.

I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119(a)-(d), of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under title 35, United States Code, Section 119(e) of any United States provisional application(s) listed below

(Application Number)	Filing Date
(Application Number)	Filing Date

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Number)	Filing Date	(Status -- patented, pending, abandoned)
(Application Number)	Filing Date	(Status -- patented, pending, abandoned)

I hereby appoint Aloysius T. C. AuYeung, Reg. No. 35,432; William Thomas Babbitt, Reg. No. 39,591; Kent D. Baker, Reg. No. 38,822; Jordan Michael Becker, Reg. No. 39,602; Bradley J. Bereznak, Reg. No. 33,474; Michael A. Bernadicou, Reg. No. 35,934; Roger W. Blakely, Jr., Reg. No. 25,831; Gregory D. Caldwell, Reg. No. 39,926; Kent M. Chen, Reg. No. 39,630; Lawrence M. Cho, Reg. No. 39,942; Thomas M. Coester, Reg. No. P39,637; Roland B. Cortes, Reg. No. 39,152; William Donald Davis, Reg. No. 38,428; Daniel M. De Vos, Reg. No. 37,813; Karen L. Feisthamel, Reg. No. 40,264; Scot A. Griffin, Reg. No. 38,167; David R. Halvorson, Reg. No. 33,395; Brian Don Hickman, Reg. No. 35,894; Eric Ho, Reg. No. P39,711; George W Hoover II, Reg. No. 32,992; Eric S. Hyman, Reg. No. 30,139; Jeffrey D. Jacobs, Reg. No. 40,029; Dag H. Johansen, Reg. No. 36,172; Stephen L. King, Reg. No. 19,180; Dolly M. Lee, Reg. No. 39,742; Daniel C. Mallery, Reg. No. 33,532; Michael J. Mallie, Reg. No. 36,591; Kimberley G. Nobles, Reg. No. 38,255; Ronald W. Reagin, Reg. No. 20,340; James H. Salter, Reg. No. 35,668; William W. Schaal, Reg. No. 39,018; James C. Scheller, Reg. No. 31,195; Maria McCormack Sobrino, Reg. No. 31,639; Stanley W. Sokoloff, Reg. No. 25,128; Allan T. Sponseller, Reg. No. 38,318; Steven R. Sponseller, Reg. No. 39,384; David R. Stevens, Reg. No. 38,626; Edwin H. Taylor, Reg. No. 25,129; Lester J. Vincent, Reg. No. 31,460; John Patrick Ward, Reg. No. 40,216; Ben J. Yorks, Reg. No. 33,609; and Norman Zafman, Reg. No. 26,250; my attorneys; and Gary B. Goates, Reg. No. 35,159; Michael Anthony DeSanctis, Reg. No. 39,957; Charles E. Shemwell, Reg. No. 40,171; Edwin A. Sloane, Reg. No. 34,728; and Judith A. Szepesi, Reg. No. 39,393; my patent agents, of BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, with offices located at 12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025, telephone (310) 207-3800, with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

Send correspondence to James H. Salter, BLAKELY, SOKOLOFF, TAYLOR & (Name of Attorney or Agent)
ZAFMAN, 12400 Wilshire Boulevard 7th Floor, Los Angeles, California 90025 and direct telephone calls to James H. Salter, (408) 720-8598.
(Name of Attorney or Agent)

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 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Sole/First Inventor Keith Lowery

Inventor's Signature [signature] Date 4-19-96

Residence Richardson, Texas (City, State) Citizenship U.S.A (Country)

Post Office Address 1702 Drake Drive
Richardson, Texas 75081

Rev. 04/01/96 (D2) cak -2-

Full Name of Second/Joint Inventor Andrew B. Levine

Inventor's Signature _____ Date 4/22/96

Residence Plano, Texas (City, State) Citizenship U.S.A. (Country)

Post Office Address 2628 Courtside Lane
Plano, Texas 75093

Full Name of Third/Joint Inventor Ronald L. Howell

Inventor's Signature Ronald L. Howell Date 4-22-96

Residence Rowlett, Texas (City, State) Citizenship U.S.A. (Country)

Post Office Address P.O. Box 1491
Rowlett, Texas 75030

EXHIBIT 6 PAGE 3 OF 3

Title 37, Code of Federal Regulations, Section 1.56
Duty to Disclose Information Material to Patentability

(a) A patent by its very nature is affected with a public interest. The public interest is best served, and the most effective patent examination occurs when, at the time an application is being examined, the Office is aware of and evaluates the teachings of all information material to patentability. Each individual associated with the filing and prosecution of a patent application has a duty of candor and good faith in dealing with the Office, which includes a duty to disclose to the Office all information known to that individual to be material to patentability as defined in this section. The duty to disclosure information exists with respect to each pending claim until the claim is cancelled or withdrawn from consideration, or the application becomes abandoned. Information material to the patentability of a claim that is cancelled or withdrawn from consideration need not be submitted if the information is not material to the patentability of any claim remaining under consideration in the application. There is no duty to submit information which is not material to the patentability of any existing claim. The duty to disclosure all information known to be material to patentability is deemed to be satisfied if all information known to be material to patentability of any claim issued in a patent was cited by the Office or submitted to the Office in the manner prescribed by §§1.97(b)-(d) and 1.98. However, no patent will be granted on an application in connection with which fraud on the Office was practiced or attempted or the duty of disclosure was violated through bad faith or intentional misconduct. The Office encourages applicants to carefully examine:

(1) Prior art cited in search reports of a foreign patent office in a counterpart application, and

(2) The closest information over which individuals associated with the filing or prosecution of a patent application believe any pending claim patentably defines, to make sure that any material information contained therein is disclosed to the Office.

(b) Under this section, information is material to patentability when it is not cumulative to information already of record or being made or record in the application, and

(1) It establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim; or

(2) It refutes, or is inconsistent with, a position the applicant takes in:

(i) Opposing an argument of unpatentability relied on by the Office, or

(ii) Asserting an argument of patentability.

A prima facie case of unpatentability is established when the information compels a conclusion that a claim is unpatentable under the preponderance of evidence, burden-of-proof standard, giving each term in the claim its broadest reasonable construction consistent with the specification, and before any consideration is given to evidence which may be submitted in an attempt to establish a contrary conclusion of patentability.

(c) Individuals associated with the filing or prosecution of a patent application within the meaning of this section are:

(1) Each inventor named in the application;

(2) Each attorney or agent who prepares or prosecutes the application; and

(3) Every other person who is substantively involved in the preparation or prosecution of the application and who is associated with the inventor, with the assignee or with anyone to whom there is an obligation to assign the application.

(d) Individuals other than the attorney, agent or inventor may comply with this section by disclosing information to the attorney, agent, or inventor.

EPIC000371

PRINT OF DRAWINGS
AS ORIGINALLY FILE.

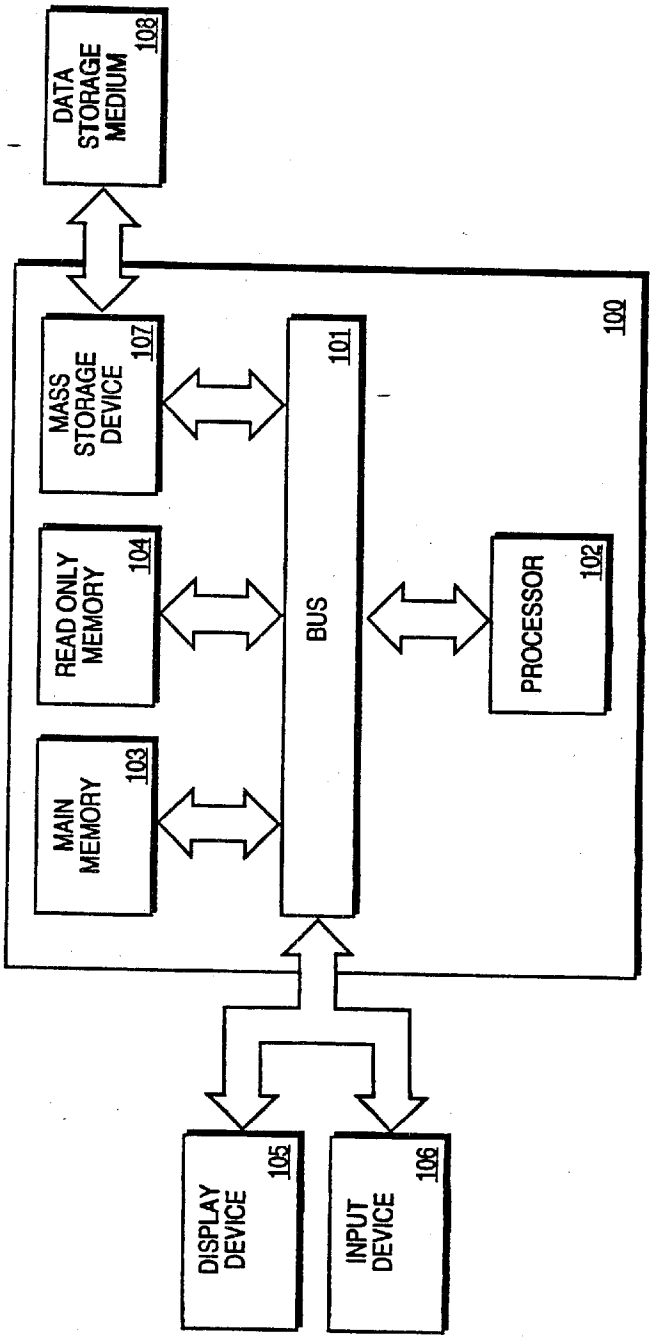


FIG. 1

PRINT OF DRAWINGS
AS ORIGINALLY FILED

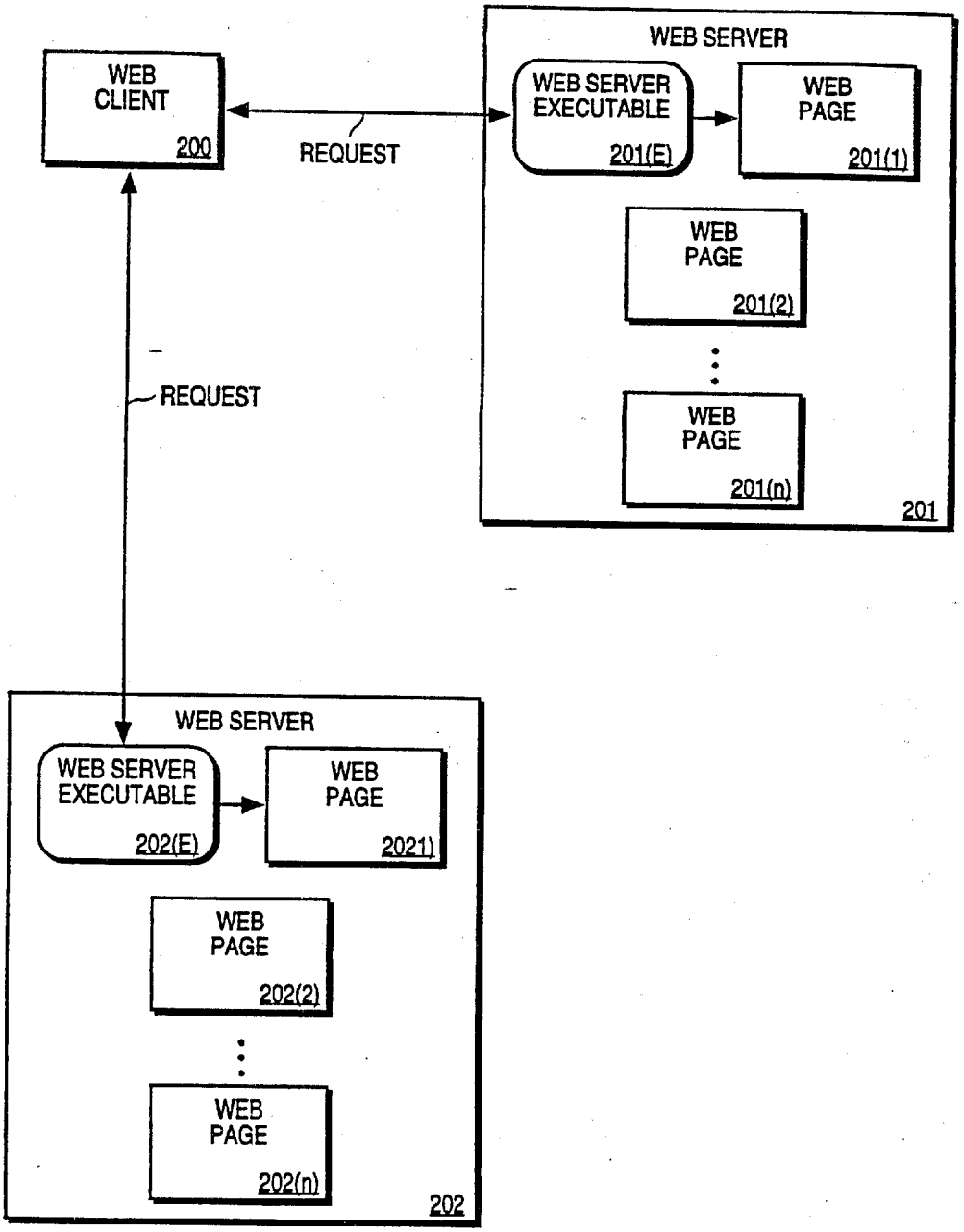


FIG. 2 (PRIOR ART)

PRINT OF DRAWINGS
AS ORIGINALLY FILE

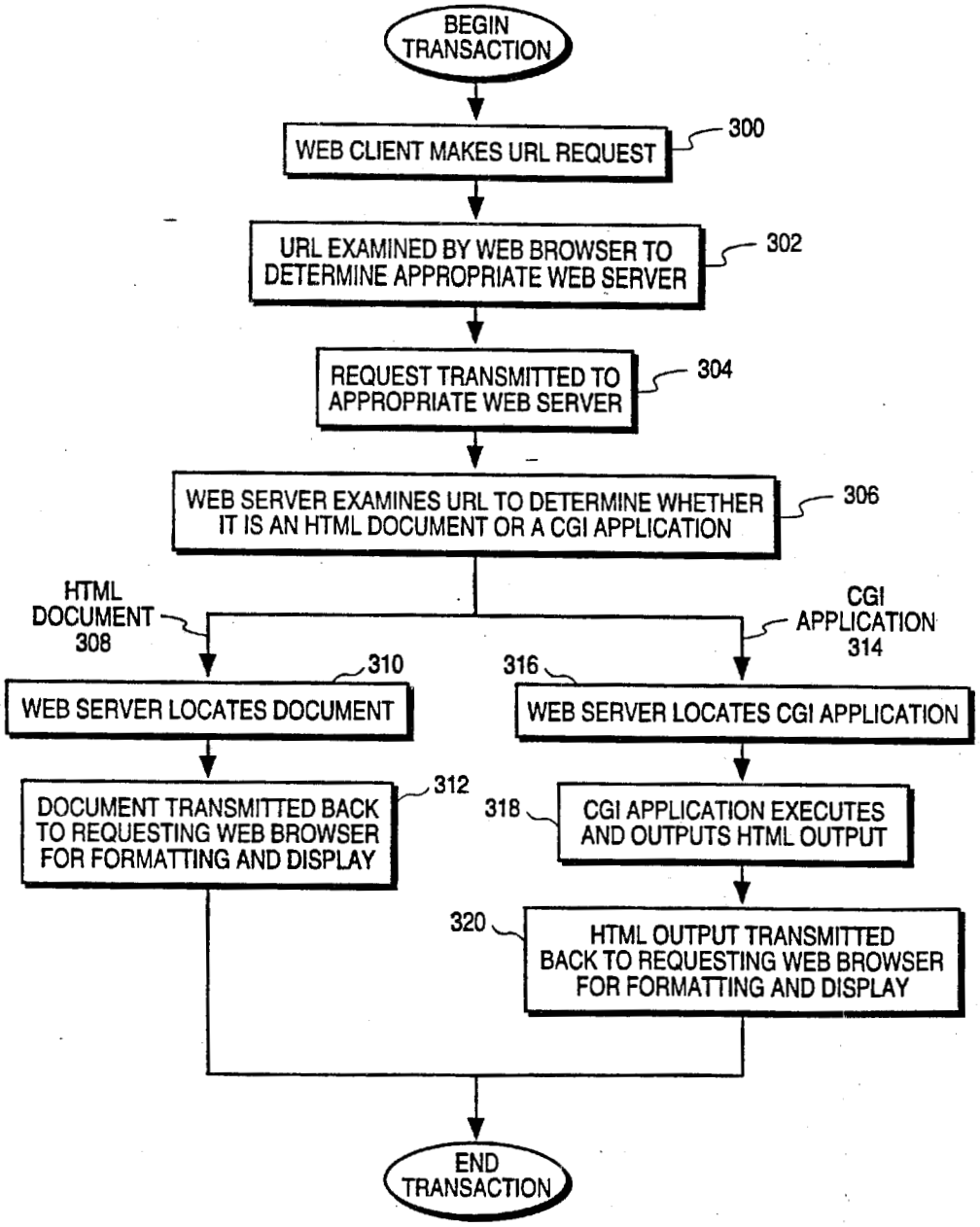


FIG. 3 (PRIOR ART)

PRINT OF DRAWINGS
AS ORIGINALLY FILED

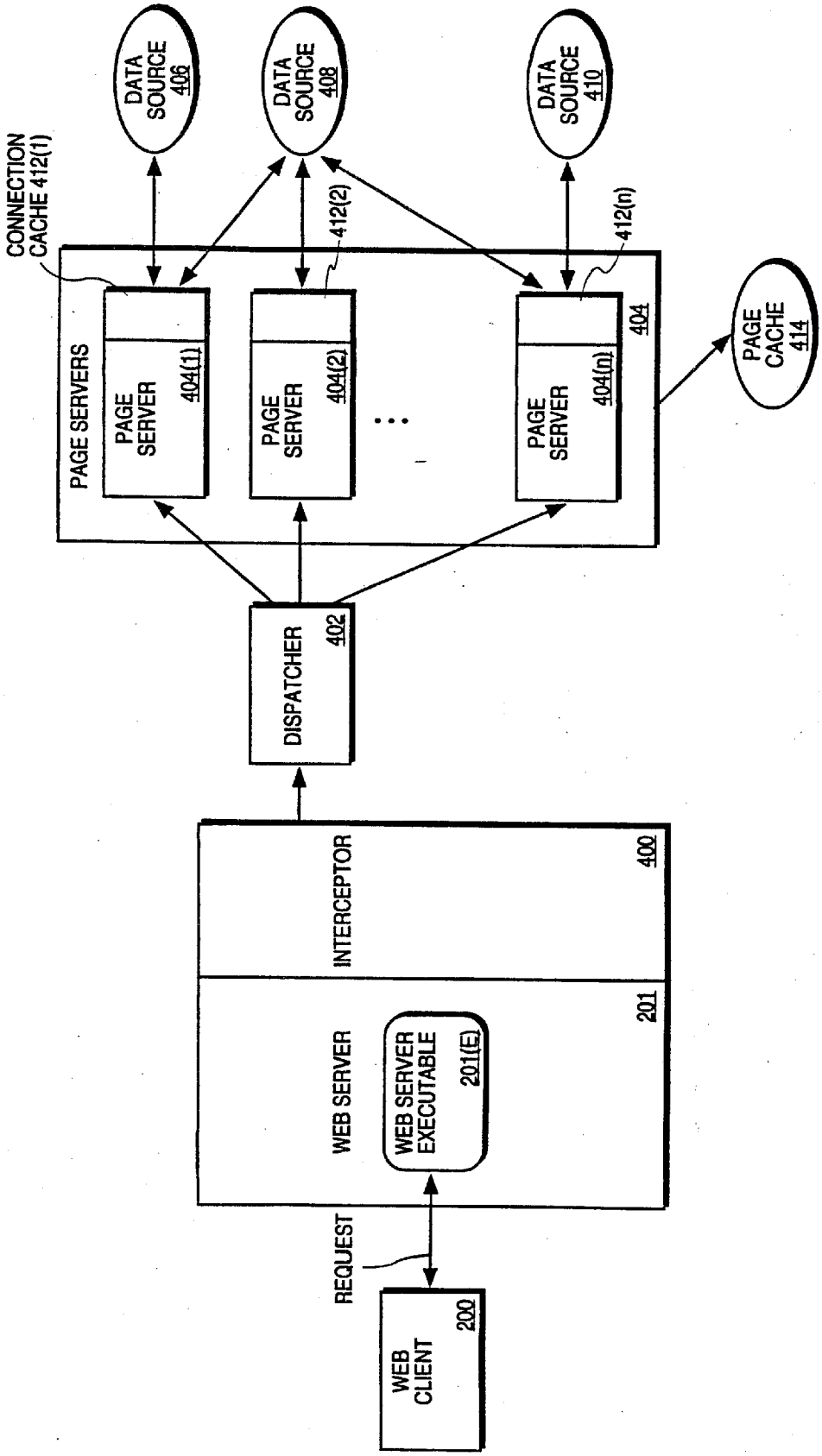


FIG. 4

PRINT OF DRAWINGS
AS ORIGINALLY FILED

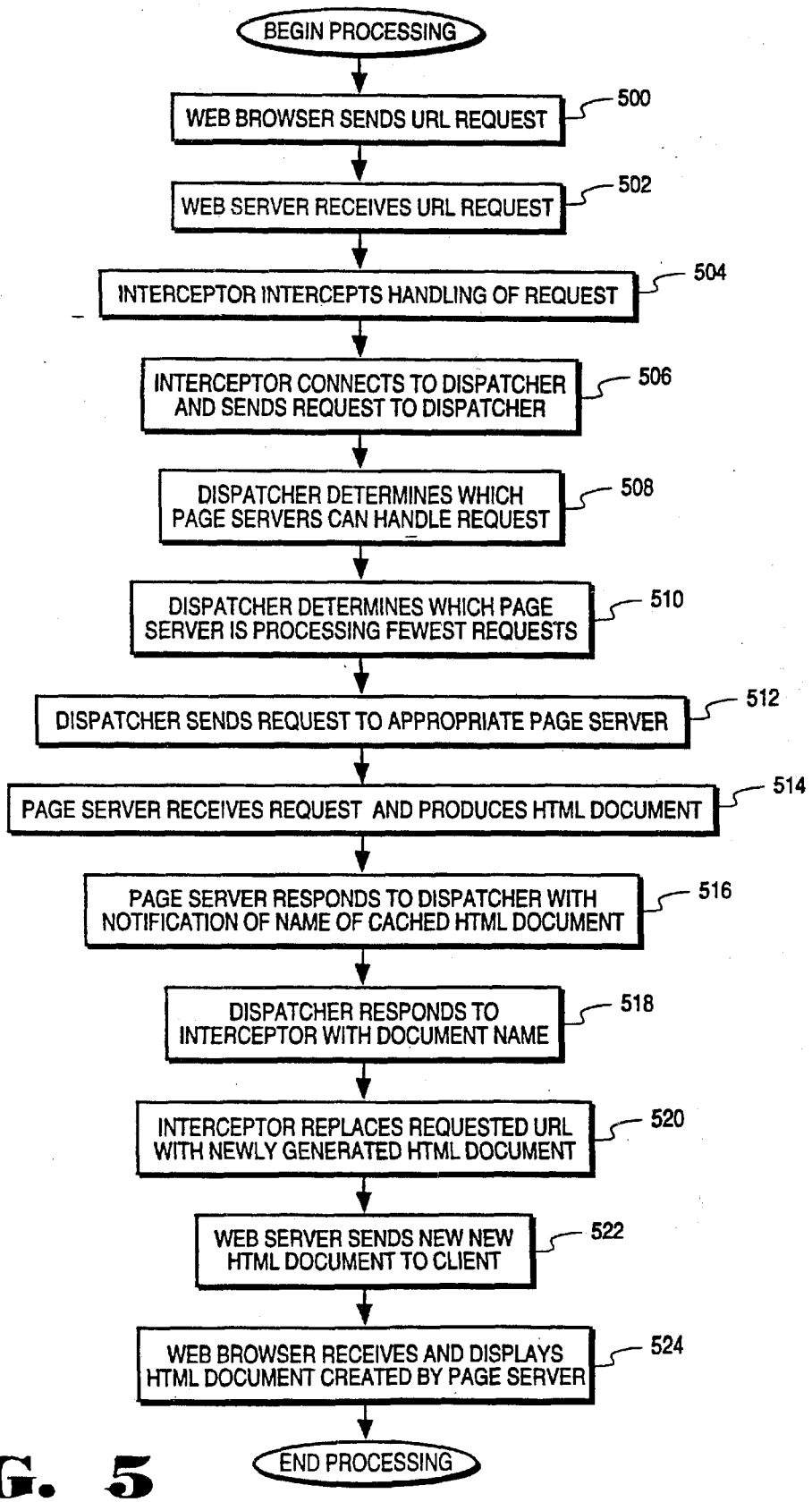


FIG. 5

FIG. 1

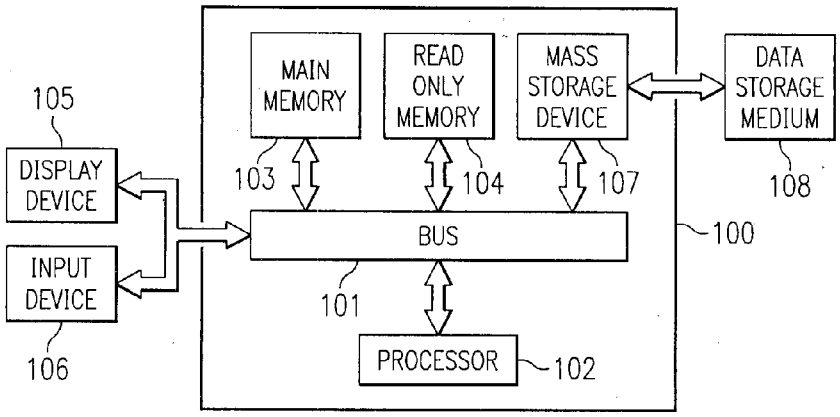


FIG. 2
(PRIOR ART)

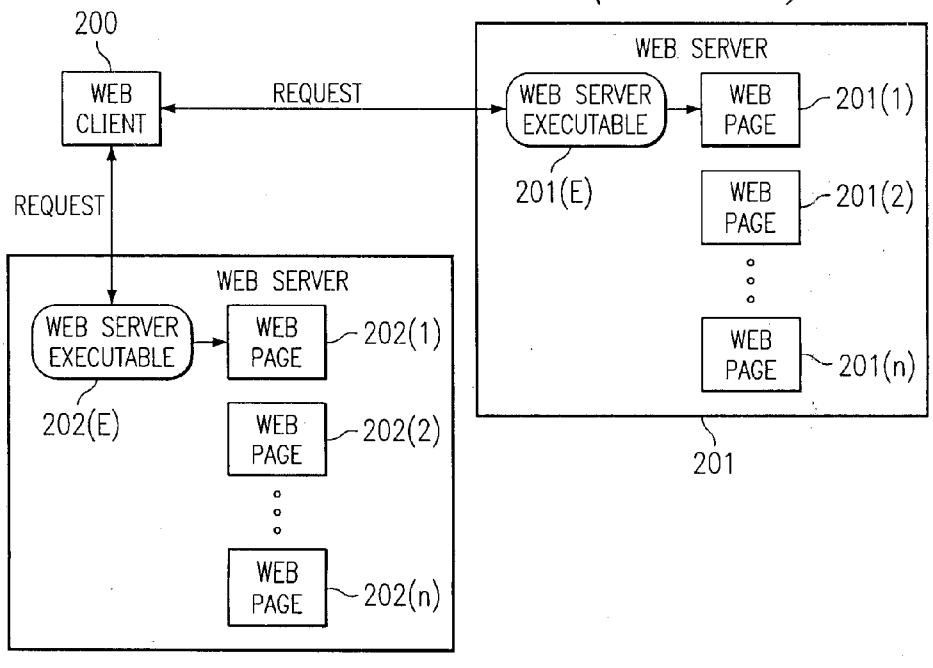
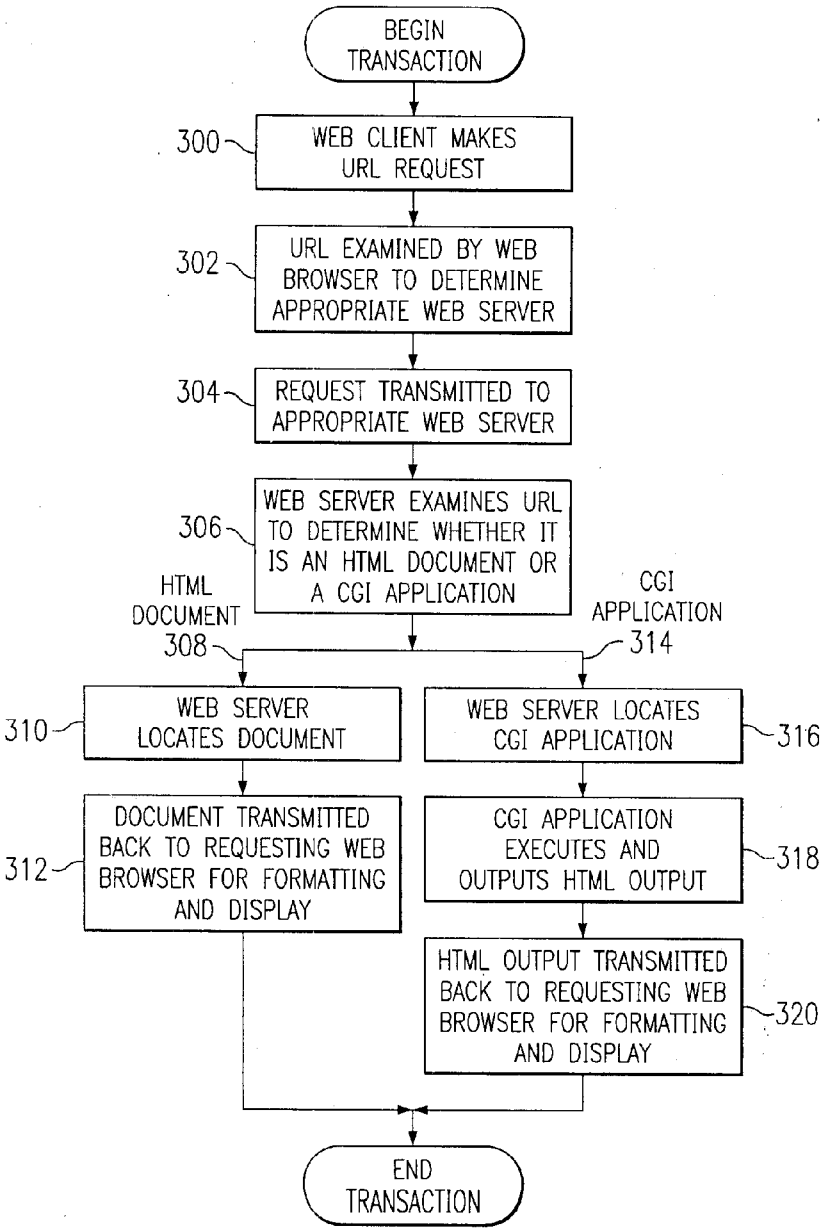


FIG. 3
(PRIOR ART)



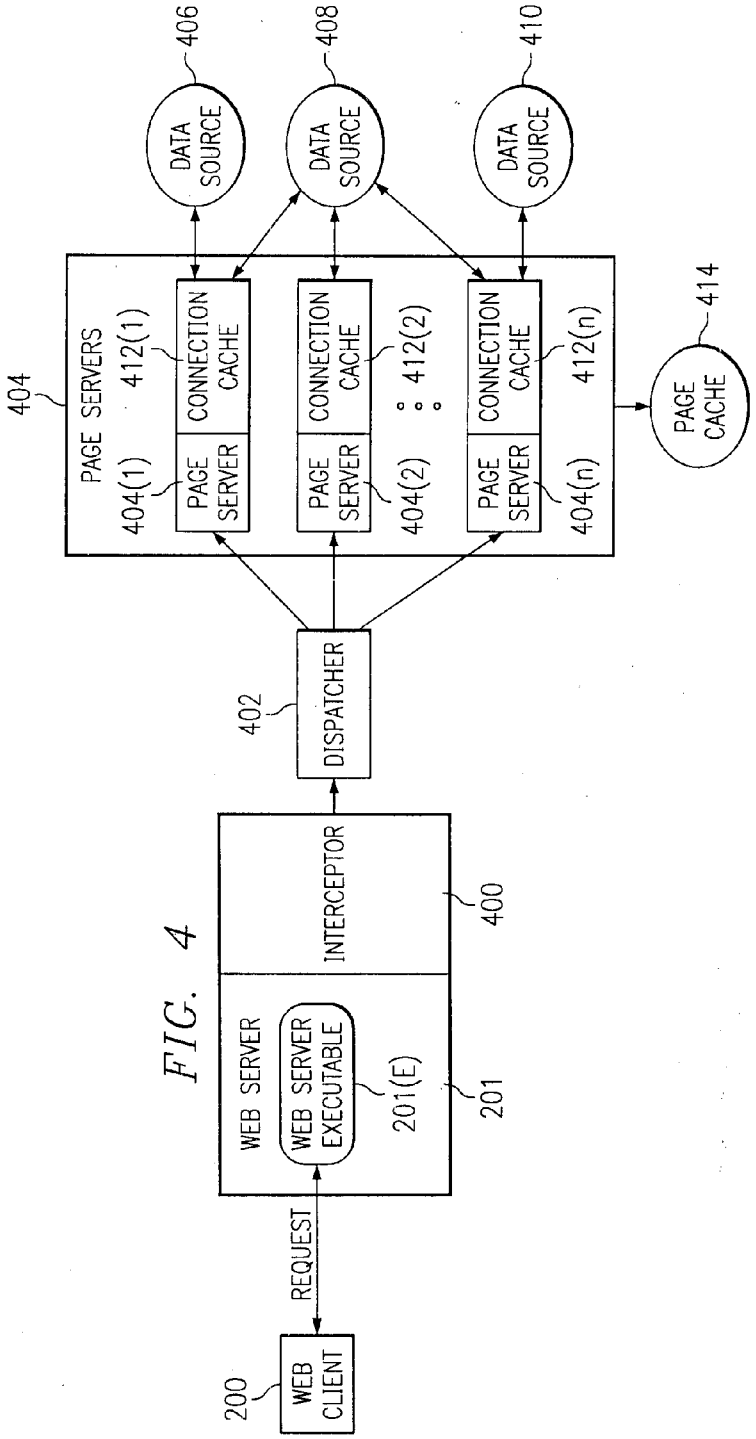
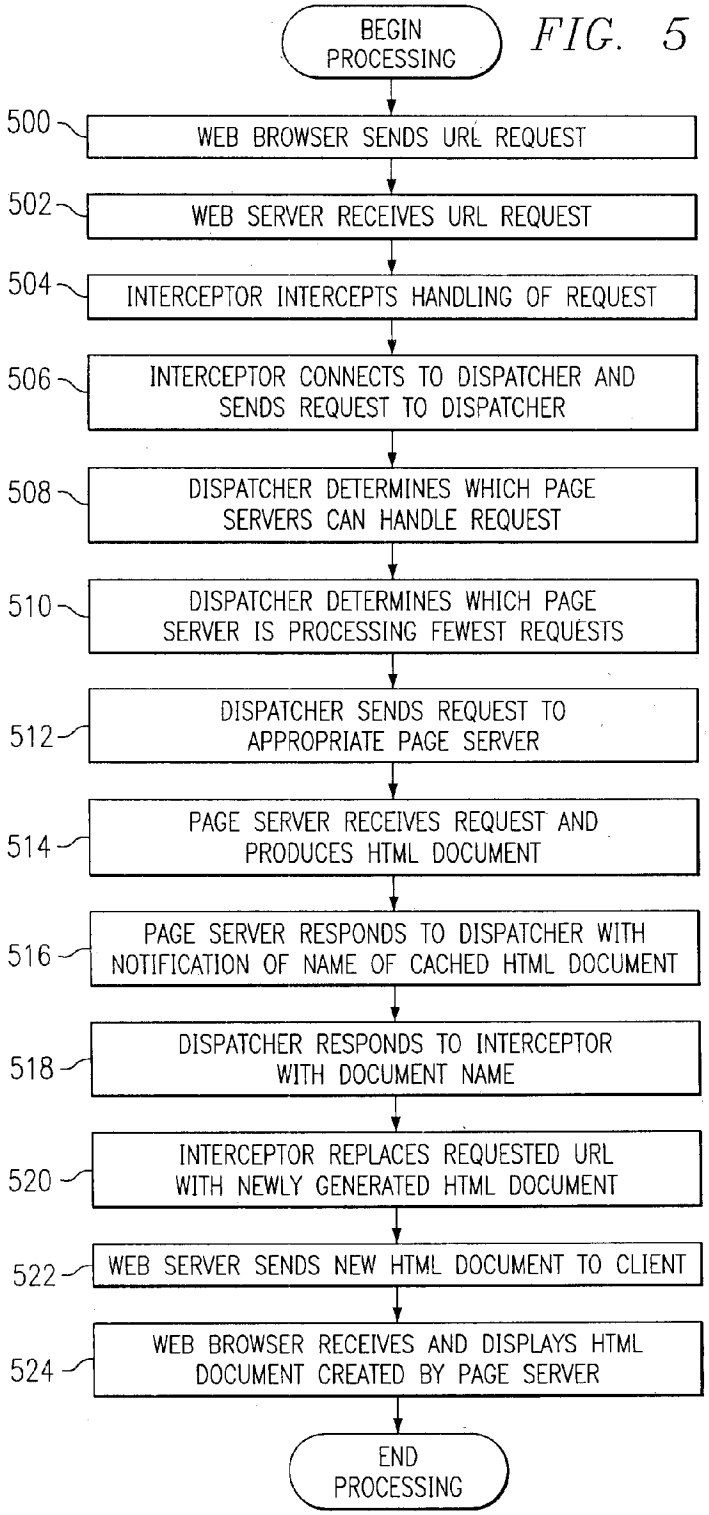

406
408
410
DATA SOURCE
DATA SOURCE
DATA SOURCE
404
PAGE SERVERS
412(1)
CONNECTION CACHE
PAGE SERVER
404(1)
CONNECTION CACHE
PAGE SERVER
404(2)
412(2)
CONNECTION CACHE
PAGE SERVER
404(n)
412(n)
414
PAGE CACHE
402
DISPATCHER
INTERCEPTOR
400
WEB SERVER
WEB SERVER EXECUTABLE
201(E)
201
REQUEST
200
WEB CLIENT

FIG. 4

FIG. 5


BEGIN PROCESSING
500
WEB BROWSER SENDS URL REQUEST
502
WEB SERVER RECEIVES URL REQUEST
504
INTERCEPTOR INTERCEPTS HANDLING OF REQUEST
506
INTERCEPTOR CONNECTS TO DISPATCHER AND SENDS REQUEST TO DISPATCHER
508
DISPATCHER DETERMINES WHICH PAGE SERVERS CAN HANDLE REQUEST
510
DISPATCHER DETERMINES WHICH PAGE SERVER IS PROCESSING FEWEST REQUESTS
512
DISPATCHER SENDS REQUEST TO APPROPRIATE PAGE SERVER
514
PAGE SERVER RECEIVES REQUEST AND PRODUCES HTML DOCUMENT
516
PAGE SERVER RESPONDS TO DISPATCHER WITH NOTIFICATION OF NAME OF CACHED HTML DOCUMENT
518
DISPATCHER RESPONDS TO INTERCEPTOR WITH DOCUMENT NAME
520
INTERCEPTOR REPLACES REQUESTED URL WITH NEWLY GENERATED HTML DOCUMENT
522
WEB SERVER SENDS NEW HTML DOCUMENT TO CLIENT
524
WEB BROWSER RECEIVES AND DISPLAYS HTML DOCUMENT CREATED BY PAGE SERVER
END PROCESSING

PATENT APPLICATION FEE DETERMINATION RECORD

Effective November 10, 1998

Application or Docket Number

CLAIMS AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA
BASIC FEE		
TOTAL CLAIMS	minus 20=	*
INDEPENDENT CLAIMS	minus 3 =	*
MULTIPLE DEPENDENT CLAIM PRESENT		

SMALL ENTITY TYPE ☐		OR	OTHER THAN SMALL ENTITY	
RATE	FEE		RATE	FEE
	380.00	OR		760.00
X$ 9=		OR	X$18=	
X39=		OR	X78=	
+130=		OR	+260=	
TOTAL		OR	TOTAL	

* If the difference in column 1 is less than zero, enter "0" in column 2

CLAIMS AS AMENDED - PART II

AMENDMENT A	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				

SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 9=		OR	X$18=	
X39=		OR	X78=	
+130=		OR	+260=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT B	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 9=		OR	X$18=	
X39=		OR	X78=	
+130=		OR	+260=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT C	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 9=		OR	X$18=	
X39=		OR	X78=	
+130=		OR	+260=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

FORM PTO-875
(Rev 6/98)

Patent and Trademark Office, U.S. DEPARTMENT OF COMMERCE

SERIAL NUMBER	FILING DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
09/234,048	01/19/99	709	2758	02577.P001D

APPLICANT

KEITH LOWERY, RICHARDSON, TX; ANDREW B. LEVINE, PLANO, TX; RONALD L. HOWELL, ROWLETT, TX.

CONTINUING DOMESTIC DATA********
VERIFIED THIS APPLN IS A DIV OF 08/636,477 04/23/96 PAT 5,894,554

371 (NAT'L STAGE) DATA********
VERIFIED

FOREIGN APPLICATIONS********
VERIFIED

IF REQUIRED, FOREIGN FILING LICENSE GRANTED 02/05/99

Foreign Priority claimed / 35 USC 119 (a-d) conditions met	STATE OR COUNTRY	SHEETS DRAWING	TOTAL CLAIMS	INDEPENDENT CLAIMS
☐yes ☑no / ☐yes ☑no ☐Met after Allowance	TX	5	16	6

Verified and Acknowledged _____ Examiner's Initials _____ Initials

ADDRESS

JAMES H SALTER
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
7TH FLOOR
LOS ANGELES CA 90025

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas Texas 75201-2980

11

TITLE

~~METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE~~ SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS.

FILING FEE RECEIVED	FEES: Authority has been given in Paper No. _____ to charge/credit DEPOSIT ACCOUNT NO. _____ for the following:	
$994		☐ All Fees ☐ 1.16 Fees (Filing) ☐ 1.17 Fees (Processing Ext. of time) ☐ 1.18 Fees (Issue) ☐ Other _____ ☐ Credit

jc583 U.S. PTO 01/19/99

Please type a plus sign (+) inside this box [+]

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No. 02577.P001D Total Pages 4

First Named Inventor or Application Identifier Keith Lowery et al.

Express Mail Label No. EL164804887US

jc518 U.S. PTO 09/234048 01/19/99

ADDRESS TO: **Assistant Commissioner for Patents**
Box Patent Application
Washington, D. C. 20231

09/234048 080199 "Express Mail" 080199

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

1. [X] Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)

2. [X] Specification (Total Pages 26)
(preferred arrangement set forth below)
- Descriptive Title of the Invention
- Cross References to Related Applications
- Statement Regarding Fed sponsored R & D
- Reference to Microfiche Appendix
- Background of the Invention
- Brief Summary of the Invention
- Brief Description of the Drawings (if filed)
- Detailed Description
- Claims
- Abstract of the Disclosure

3. [X] Drawings(s) (35 USC 113) (Total Sheets 5)

4. [X] Oath or Declaration (Total Pages 4)

a. [ ] Newly Executed (Original or Copy)

b. [X] Copy from a Prior Application (37 CFR 1.63(d))
(for Continuation/Divisional with Box 17 completed) **(Note Box 5 below)**

i. [ ] DELETIONS OF INVENTOR(S) Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).

5. [X] Incorporation By Reference (useable if Box 4b is checked)
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

6. [ ] Microfiche Computer Program (Appendix)

7. [ ] Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
a. [ ] Computer Readable Copy
b. [ ] Paper Copy (identical to computer copy)
c. [ ] Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

8. ____ Assignment Papers (cover sheet & documents(s))
9. ____ a. 37 CFR 3.73(b) Statement (where there is an assignee)
____ b. Power of Attorney
10. ____ English Translation Document (if applicable)
11. X a. Information Disclosure Statement (IDS)/PTO-1449
X b. Copies of IDS Citations
12. ____ Preliminary Amendment
13. X Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
14. ____ a. Small Entity Statement(s)
b. Statement filed in prior application, Status still proper and desired
15. ____ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. ____ Other: _____

17. **If a CONTINUING APPLICATION,** check appropriate box and supply the requisite information:
___ Continuation X Divisional ____ Continuation-in-part (CIP)
of prior application No: 08/636,477

18. **Correspondence Address**
____ Customer Number or Bar Code Label _____
(Insert Customer No. or Attach Bar Code Label here)
or
X Correspondence Address Below

NAME James H. Salter, Reg. No. 35,668
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
ADDRESS 12400 Wilshire Boulevard
Seventh Floor
CITY Los Angeles STATE California ZIP CODE 90025-1026
Country U.S.A. TELEPHONE (408) 720-8598 FAX (408) 720-9397

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

EPIC000384

Please type a plus sign (+) inside this box [+]

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b)

Attorney Docket No. 02577.P001D Total Pages 4

First Named Inventor or Application Identifier Keith Lowery et al.

Express Mail Label No. EL164804887US

jc518 U.S. PTO
09/234048
01/19/99

ADDRESS TO: **Assistant Commissioner for Patents**
Box Patent Application
Washington, D. C. 20231

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

1. [X] Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)

2. [X] Specification (Total Pages 26)
(preferred arrangement set forth below)
- Descriptive Title of the Invention
- Cross References to Related Applications
- Statement Regarding Fed sponsored R & D
- Reference to Microfiche Appendix
- Background of the Invention
- Brief Summary of the Invention
- Brief Description of the Drawings (if filed)
- Detailed Description
- Claims
- Abstract of the Disclosure

3. [X] Drawings(s) (35 USC 113) (Total Sheets 5)

4. [X] Oath or Declaration (Total Pages 4)
   a. [ ] Newly Executed (Original or Copy)
   b. [X] Copy from a Prior Application (37 CFR 1.63(d))
   (for Continuation/Divisional with Box 17 completed) **(Note Box 5 below)**
   i. [ ] DELETIONS OF INVENTOR(S) Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).

5. [X] Incorporation By Reference (useable if Box 4b is checked)
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

6. [ ] Microfiche Computer Program (Appendix)

7. [ ] Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
   a. [ ] Computer Readable Copy
   b. [ ] Paper Copy (identical to computer copy)
   c. [ ] Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

8. ____ Assignment Papers (cover sheet & documents(s))
9. ____ a. 37 CFR 3.73(b) Statement (where there is an assignee)
____ b. Power of Attorney
10. ____ English Translation Document (if applicable)
11. X a. Information Disclosure Statement (IDS)/PTO-1449
X b. Copies of IDS Citations
12. ____ Preliminary Amendment
13. X Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
14. ____ a. Small Entity Statement(s)
b. Statement filed in prior application, Status still proper and desired
15. ____ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. ____ Other: _____

17. **If a CONTINUING APPLICATION**, check appropriate box and supply the requisite information:
___ Continuation X Divisional ____ Continuation-in-part (CIP)
of prior application No: 08/636,477

18. **Correspondence Address**
____ Customer Number or Bar Code Label _____
(Insert Customer No. or Attach Bar Code Label here)
or
X Correspondence Address Below

NAME James H. Salter, Reg. No. 35,668
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
ADDRESS 12400 Wilshire Boulevard
Seventh Floor
CITY Los Angeles STATE California ZIP CODE 90025-1026
Country U.S.A. TELEPHONE (408) 720-8598 FAX (408) 720-9397

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

PTO/SB/17(10/96)
Approved for use through 09/30/98. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL

TOTAL AMOUNT OF PAYMENT ($) 994.00

Complete if Known:

Application No.	Unassigned
Filing Date	January 19, 1999
First Named Inventor	Keith Lowery et al.
Group Art Unit	2317
Examiner Name	Krick, R.
Attorney Docket No.	02577.P001D

METHOD OF PAYMENT (check one)

1. [x] **The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:**

Deposit Account Number 02-2666
Deposit Account Name

[X] **Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17 and credit any over payments to Deposit Account Number 02-2666**

[ ] **Charge the Issue Fee Set in 37 CFR 1.18 at the Mailing of the Notice of Allowance, 37 CFR 1.131(b)**

2.
- X — **Payment Enclosed**
- X — **Check**
- ____ — **Money Order**
- ____ — **Other**

FEE CALCULATION (fees effective 10/01/97)

1. **FILING FEE**

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	760	201	380	Utility application filing fee	760.00
106	310	206	155	Design application filing fee	
107	480	207	240	Plant filing fee	
108	760	208	380	Reissue filing fee	
114	150	214	75	Provisional application filing fee	
				SUBTOTAL (1)	$ 760.00

2. **CLAIMS**

			Extra		Fee from below		Fee Paid
Total Claims	16	– 20 =	0	X	0	=	0
Independent Claims	6	– 3 =	3	X	78.00	=	234.00
Multiple Dependent Claims			0	X	0	=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
103	18	203	9	Claims in excess of twenty	
102	78	202	39	Independent claims in excess of 3	234.00
104	260	204	130	Multiple dependent claim	
109	78	209	39	Reissue independent claims over original patent	
110	18	210	9	Reissue claims in excess of 20 and over original patent	
				SUBTOTAL (2)	$ 234.00

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for response within first month	
116	380	216	190	Extension for response within second month	
117	870	217	435	Extension for response within third month	
118	1,360	218	680	Extension for response within fourth month	
128	1,850	228	925	Extension for response within fifth month	
119	300	219	150	Notice of Appeal	
120	300	220	150	Filing a brief in support of an appeal	
121	260	221	130	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive unavoidably abandoned application	
141	1,210	241	605	Petition to revive unintentionally abandoned application	
142	1,210	242	605	Utility issue fee (or reissue)	
143	430	243	215	Design issue fee	
144	580	244	290	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Petitions related to provisional applications	
126	240	126	240	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	760	246	380	For filing a submission after final rejection (see 37 CFR 1.129(a))	
149	760	249	380	For each additional invention to be examined (see 37 CFR 1.129(a))	

SUBTOTAL (3) $ 0

*Reduced by Basic Filing Fee Paid

SUBMITTED BY:

Typed or Printed Name: James H. Salter

Signature _____ Date 1/19/99

Reg. Number 35,668 Deposit Account User ID _____ (complete if applicable)

EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number: EL164804887US

Date of Deposit: January 19, 1999

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to BOX APPLICATION, Assistant Commissioner for Patents, Washington, D.C. 20231.

Claire Wallters

Name of Person Mailing Paper or Fee

Claire Wallters

Signature of Person Mailing Paper or Fee

January 19, 1999

Date Signed

D. Johnson
#2 7-14-99
IDS w/Refers

Attorney's Docket No. 02577.P001D PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Patent Application of:

Keith Lowery et al.

Application No.: Unassigned

Filed: January 19, 1999

For: MANAGING WEB SITES AND DYNAMIC WEB PAGE GENERATION REQUESTS

A Rule 1.53(b) Divisional Application of:
U.S. Serial No. 08/636,477
Filed April 23, 1996

Examiner: Krick, R.

Art Unit: 2317



BOX APPLICATION
Assistant Commissioner for Patents
Washington, D.C. 20231

INFORMATION DISCLOSURE STATEMENT

Sir:

Enclosed is a copy of Information Disclosure Citation Form PTO-1449 together with copies of the documents cited on that form. It is respectfully requested that the cited documents be considered and that the enclosed copy of Information Disclosure Citation Form PTO-1449 be initialed by the Examiner to indicate such consideration and a copy thereof returned to applicant(s).

Pursuant to 37 C.F.R. § 1.97, the submission of this Information Disclosure Statement is not to be construed as a representation that a search has been made and is not to be construed as an admission that the information cited in this statement is material to patentability.

Pursuant to 37 C.F.R. § 1.97, this Information Disclosure Statement is being submitted under one of the following (as indicated by an "X" to the left of the appropriate paragraph):

XX 37 C.F.R. §1.97(b).

_____ 37 C.F.R. §1.97(c). If so, then enclosed with this Information Disclosure Statement is one of the following:

____ A statement pursuant to 37 C.F.R. §1.97(e) or

____ A check for $240.00 for the fee under 37 C.F.R. § 1.17(p).

_____ 37 C.F.R. §1.97(d). If so, then enclosed with this Information Disclosure Statement are the following:

(1) A statement pursuant to 37 C.F.R. §1.97(e);

(2) A petition requesting consideration of the Information Disclosure Statement; and

(3) A check for $_____ for the fee under 37 C.F.R. §1.17(i) for submission of the Information Disclosure Statement.

If there are any additional charges, please charge Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 1/19, 1999

James H. Salter
Reg. No. 35,668

12400 Wilshire Blvd.
Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8598

EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number: EL164804887US

Date of Deposit: January 19, 1999

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to BOX APPLICATION, Assistant Commissioner for Patents, Washington, D.C. 20231.

Claire Wallters

Name of Person Mailing Paper or Fee

Claire Wallters

Signature of Person Mailing Paper or Fee

January 19, 1999

Date Signed

Sheet 1 of 1

Form PTO-1449 (REV. 8-83) — U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE

ATTY. DOCKET NO.	SERIAL NO.
02577.P001D	Unassigned

APPLICANT: Keith Lowery et al.

FILING DATE	GROUP
1/19/99	Unassigned

INFORMATION DISCLOSURE CITATION

(Use several sheets if necessary)

JC518 U.S. PTO 09/234048 01/19/99

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
		4 8 6 6 7 0 6	9/12/89	Christophersen et al.	370	85.7	8/27/87
		5 3 4 1 4 9 9	8/23/94	Doragh	395	700	4/2/92
		5 3 9 2 4 0 0	2/21/95	Berkowitz et al.	395	200	7/2/92
		5 4 0 4 5 2 2	4/4/95	Carmon et al.	395	650	10/26/93
		5 4 0 4 5 2 3	4/4/95	DellaFera et al.	395	650	11/10/93
		5 4 0 4 5 2 7	4/4/95	Irwin et al.	395	700	12/31/92
		5 4 5 2 4 6 0	9/19/95	Distelberg et al.	395	700	1/27/93
		5 5 3 2 8 3 8	7/2/96	Barbari	358	400	4/14/95

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATION YES	TRANSLATION NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

			International Search Report; 7 pages; dated August 21, 1997.
			Birman, Kenneth P. and van Renesse, Robbert; Software for Reliable Networks; Scientific American; May 1996; pp. 64-69.
			"Beyond the Web: Excavating the Real World Via Mosaic"; Goldberg et al.; Second International WWW Conference; 10/17/94.

EXAMINER: [signature] DATE CONSIDERED: 2/17/01

*EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

PATENT COOPERATION TREATY

From the INTERNATIONAL SEARCHING AUTHORITY

To:

MR. EDWIN H. TAYLOR
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 WILSHIRE BLVD., SEVENTH FLOOR
LOS ANGELES CA 90025

RECEIVED
AUG 25 1997
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
LOS ANGELES

PCT

NOTIFICATION OF TRANSMITTAL OF THE INTERNATIONAL SEARCH REPORT OR THE DECLARATION

(PCT Rule 44.1)

Applicant's or agent's file reference 002577.P001P	Date of Mailing *(day/month/year)* 21 AUG 1997
International application No. PCT/US97/06840	FOR FURTHER ACTION See paragraphs 1 and 4 below
Applicant INFOSPINNER INC	International filing date *(day/month/year)* 23 APRIL 1997

1. [X] The applicant is hereby notified that the international search report has been established and is transmitted herewith.

Filing of amendments and statement under Article 19:
The applicant is entitled, if he so wishes, to amend the claims of the international application (see Rule 46):

When? The time limit for filing such amendments is normally 2 months from the date of transmittal of the international search report; however, for more details, see the notes on the accompanying sheet.

Where? Directly to the International Bureau of WIPO
34, chemin des Colombettes
1211 Geneva 20, Switzerland
Facsimile No.: (41-22) 740.14.35

For more detailed instructions, see the notes on the accompanying sheet.

2. [ ] The applicant is hereby notified that no international search report will be established and that the declaration under Article 17(2)(a) to that effect is transmitted herewith.

3. [ ] **With regard to the protest** against payment of (an) additional fee(s) under Rule 40.2, the applicant is notified that:

[ ] the protest together with the decision thereon has been transmitted to the International Bureau together with the applicant's request to forward the texts of both the protest and the decision thereon to the designated Offices.

[ ] no decision has been made yet on the protest; the applicant will be notified as soon as a decision is made.

4. **Further action(s):** The applicant is reminded of the following:

Shortly after **18 months** from the priority date, the international application will be published by the International Bureau. If the applicant wishes to avoid or postpone publication, a notice of withdrawal of the international application, or of the priority claim, must reach the International Bureau as provided in rules 90 *bis* 1 and 90 *bis* 3, respectively, before the completion of the technical preparations for international publication.

Within **19 months** from the priority date, a demand for international preliminary examination must be filed if the applicant wishes to postpone the entry into the national phase until 30 months from the priority date (in some Offices even later).

Within **20 months** from the priority date, the applicant must perform the prescribed acts for entry into the national phase before all designated Offices which have not been elected in the demand or in a later election within 19 months from the priority date or could not be elected because they are not bound by Chapter II.

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	Authorized officer THOMAS C LEE
Facsimile No. (703) 305-3230	Telephone No. (703) 305-9717

Form PCT/ISA/220 (January 1994)★ *(See notes on accompanying sheet)*

PATENT COOPERATION TREATY

PCT

INTERNATIONAL SEARCH REPORT

(PCT Article 18 and Rules 43 and 44)

Applicant's or agent's file reference 002577.P001P	FOR FURTHER ACTION	see Notification of Transmittal of International Search Report (Form PCT/ISA/220) as well as, where applicable, item 5 below.
International application No. PCT/US97/06840	International filing date *(day/month/year)* 23 APRIL 1997	(Earliest) Priority Date *(day/month/year)* 23 APRIL 1996
Applicant INFOSPINNER INC		

This international search report has been prepared by this International Searching Authority and is transmitted to the applicant according to Article 18. A copy is being transmitted to the International Bureau.

This international search report consists of a total of 5 sheets.

[X] It is also accompanied by a copy of each prior art document cited in this report.

1. [ ] **Certain claims were found unsearchable** (See Box I).

2. [ ] **Unity of invention is lacking** (See Box II).

3. [ ] The international application contains disclosure of a **nucleotide and/or amino acid sequence listing** and the international search was carried out on the basis of the sequence listing
   - [ ] filed with the international application.
   - [ ] furnished by the applicant separately from the international application,
     - [ ] but not accompanied by a statement to the effect that it did not include matter going beyond the disclosure in the international application as filed.
   - [ ] transcribed by this Authority.

4. With regard to the **title**,
   - [ ] the text is approved as submitted by the applicant.
   - [X] the text has been established by this Authority to read as follows:

SYSTEM FOR DYNAMICALLY CREATING AND MANAGING A CUSTOM WEB SITE

5. With regard to the **abstract**,
   - [ ] the text is approved as submitted by the applicant.
   - [X] the text has been established, according to Rule 38.2(b), by this Authority as it appears in Box III. The applicant may, within one month from the date of mailing of this international search report, submit comments to this Authority.

6. The figure of the drawings to be published with the abstract is:

Figure No. 4

- [X] as suggested by the applicant.
- [ ] because the applicant failed to suggest a figure.
- [ ] because this figure better characterizes the invention.
- [ ] None of the figures.

Form PCT/ISA/210 (first sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06840

Box III TEXT OF THE ABSTRACT (Continuation of item 5 of the first sheet)

The present invention teaches a system for creating and managing custom Web sites, specifically, managing a dynamic Web page generation request from a Web client (200) to a Web server (201). Instead of Web server executable (201(E)) processing the request, Interceptor (400) intercepts the request and routes it to Dispatcher 402. Dispatcher (402) receives the intercepted request, examines the request, and dispatches the request to one of a number of Page servers (404). The specified Page server (404) processes the request while Web sever executable (201(E)) concurrently process other Web client requests.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06840

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :H04N 1/00; G06F 17/00
US CL :358/400, 403, 442,444; 395/200.01, 671, 675
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 358/400, 403, 442,444; 395/200.01, 671, 675

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

	Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
✓	Y,P	US, 5,532,838 A (BARBARI) 02 July 1996, the abstract, col. 11 line 39 - col. 12 line 54, and col. 17 line 28 - col. 18 line 50	1-16
✓	A	US, 5,452,460 A (DISTELBERG ET AL) 19 September 1995, the abstract and col. 5 line 5 - col. 6 line 12	1-16
✓	A	US, 5,404,523 A (DELLAFERA et al) 04 April 1995, the abstract and col. 4 line 33 - col. 6 line 24	1-16
✓	A	US, 5,392,400 A (BERKOWITZ ET AL) 21 February 1995, the abstract and col. 1 line 62 - col. 2 line 59	1-16

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)		
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed	"&"	document member of the same patent family

Date of the actual completion of the international search 30 JUNE 1997	Date of mailing of the international search report 21 AUG 1997
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer THOMAS C LEE Telephone No. (703) 305-9717

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06840

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US, 5,341,499 A (DORAGH) 23 August 1994, the abstract and col. 3 line 12 - col. 4 line 23	1-16
Y	GOLDBERG ET AL, Second International WWW Conference, Oct 17-21, 1994, "Beyond the Web: E xcavating the Real World Via Mosaic", pages 1-15, especially pages 5-7	1-16

Form PCT/ISA/210 (continuation of second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06840

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

APS
search terms: generat? (3a) request#; rout? (3a) request#; (dynamic? or automatic?) (3a) creat? (3a) (document# or page# or information#); process? (3a) request#

NOTES TO FORM PCT/ISA/220

These Notes are intended to give the basic instructions concerning the filing of amendments under Article 19. The Notes are based on the requirements of the Patent Cooperation Treaty and of the Regulations and the Administrative Instructions under that Treaty. In case of discrepancy between these Notes and those requirements, the latter are applicable. For more detailed information, see also the PCT Applicant's Guide, a publication of WIPO.

In these Notes, "Article", "Rule" and "Section" refer to the provisions of the PCT, the PCT Regulations and the PCT Administrative Instructions, respectively.

INSTRUCTIONS CONCERNING AMENDMENTS UNDER ARTICLE 19

The applicant has, after having received the international search report, one opportunity to amend the claims of the international application. It should however be emphasized that, since all parts of the international application (claims, description and drawings) may be amended during the international preliminary examination procedure, there is usually no need to file amendments of the claims under Article 19 except where, e.g. the applicant wants the latter to be published for the purposes of provisional protection or has another reason for amending the claims before international publication. Furthermore, it should be emphasized that provisional protection is available in some States only.

What parts of the international application may be amended ?

The claims only.

The description and the drawings may only be amended during international preliminary examination under Chapter II.

When ? Within 2 months from the date of transmittal of the international search report or 16 months from the priority date, whichever time limit expires later. It should be noted, however, that the amendments will be considered as having been received on time if they are received by the International Bureau after the expiration of the applicable time limit but before the completion of the technical preparations for international publication (Rule 46.1).

Where not to file the amendments ?

The amendments may only be filed with the International Bureau and not with the receiving Office or the International Searching Authority (Rule 46.2).

Where a demand for international preliminary examination has been/is filed, see below.

How ? Either by cancelling one or more entire claims, by adding one or more new claims or by amending the text of one or more of the claims as filed.

A replacement sheet must be submitted for each sheet of the claims which, on account of an amendment or amendments, differs from the sheet originally filed.

All the claims appearing on a replacement sheet must be numbered in Arabic numerals. Where a claim is cancelled, no renumbering of the other claims is required. In all cases where claims are renumbered, they must be renumbered consecutively (Administrative Instructions, Section 205(b)).

What documents must/may accompany the amendments ?

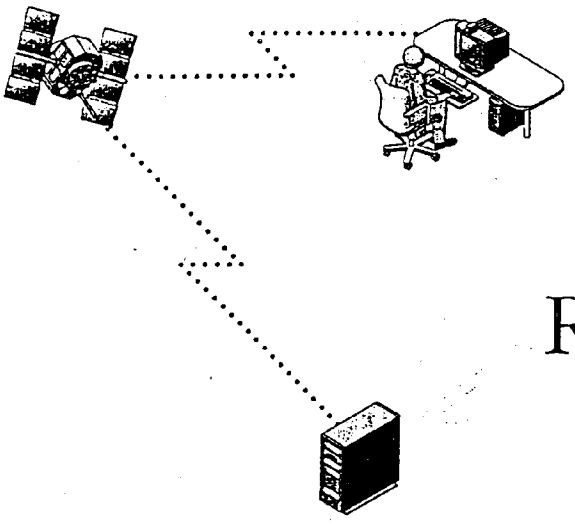
Letter (Section 205(b)):

The amendments must be submitted with a letter.

The letter will not be published with the international application and the amended claims. It should not be confounded with the "Statement under Article 19(1)" (see below, under "Statement under Article 19(1)").

The letter must indicate the differences between the claims as filed and the claims as amended. It must, in particular, indicate, in connection with each claim appearing in the international application (it being understood that identical indications concerning several claims may be grouped), whether:

(i) the claim is unchanged;
(ii) the claim is cancelled;
(iii) the claim is new;
(iv) the claim replaces one or more claims as filed;
(v) the claim is the result of the division of a claim as filed.



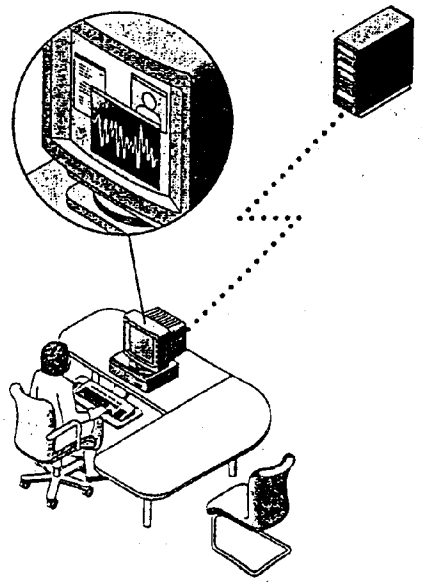
REMOTE CONSULTATION
Videoconferencing server allows doctors to consult other experts for additional information about cases.

Software for Reliable Networks

Techniques that enable distributed computing systems to reorganize themselves can restore operation when one part crashes

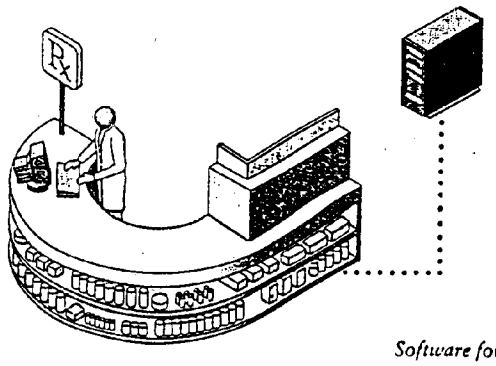
by Kenneth P. Birman and Robbert van Renesse

PHYSICIAN
From a private office in another building, a physician can monitor patients in the hospital. The physician can access vital signs, such as breathing and heart rate, laboratory results and current medical records.



Surfing the Internet is no longer just a seductive pastime. In rising numbers, organizations of all kinds, from computer companies to publishing firms, are turning to on-line services that operate much like the World Wide Web. These services can help manage important information, speed decision making and improve efficiency. But as more and more enterprises become dependent on this new technology, many are also exposed to the downside of computer networking. The drawbacks are particularly evident to users of distributed computing systems, which link programs, servers and data files dispersed across an extended network of computers and terminals.

As every computer user knows, programs that operate across networks are prone to failure. Indeed, Leslie B. Lamport, a pioneer in distributed computing at Digital Equipment Corporation, defined a distributed computing system as "one in which the failure of a computer you didn't even know existed can render your own computer unusable." The Web is certainly not exempt from breakdowns [*see box on page 68*]. During late 1995, users of the Web reported several "brownouts," when communication on the Internet was largely impossible. Such lapses have been variously attributed to software errors, excessive traffic on transmission lines, and overload or complete failure of the Web servers, which are computers that



PHARMACY
The hospital pharmacist also adds information to patients' records, noting when requested medications were dispensed. Accurate data about all medications a patient receives help to prevent dangerous drug interactions.

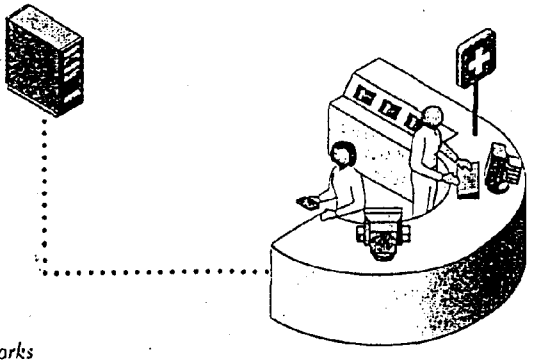
store the documents users access from their workstations. Most likely, a combination of factors contributes to brownouts. Unfortunately, similar events will multiply as computer networks—not just the World Wide Web but also distributed computing systems serving banks, schools and many offices—continue to expand.

When computers crash, sometimes the only casualties are the user's time and temper. If the automated bank teller nearest you is not working, the one across the street from it may be. But the shutdown of very complicated networks can have dire consequences. On July 15, 1994, the NASDAQ stock exchange, an exclusively electronic stock market, opened two hours late because of a mysterious problem that compromised the entire system. Initially, workers thought a software bug triggered the shutdown, but the error was ultimately traced to a malfunctioning disk. Because trading was delayed only for a few hours, little revenue was lost. Yet the event could have been a catastrophe: the market would have faced enormous losses had trading not resumed when it did.

In another example, from January 1990, the AT&T telephone system experienced a large-scale outage when an electronic switch within the system failed. Calls were automatically shifted to a backup switch, but this switch also crashed because of a software bug. The failure rippled through the network, resulting in a nationwide shutdown that lasted for nine hours, during which 60,000 people lost all telephone service and 70 million telephone calls could not be completed. To anyone familiar with the challenges of managing even a simple network, the surprise is not that these mishaps occur but rather that they are not more frequent.

Building Reliable Electronic Bridges

Even a brief distributed systems failure can pose a significant problem for applications that require around-the-clock operation. Air-traffic-control and financial computer networks must be exceedingly reliable and constantly updated. A message that a "host is not responding" or a misleading display that shows out-of-date information about an airplane's flight path or a stock's price could easily provoke an accident or financial misadventure. As the way people live and work continues to be transformed, the



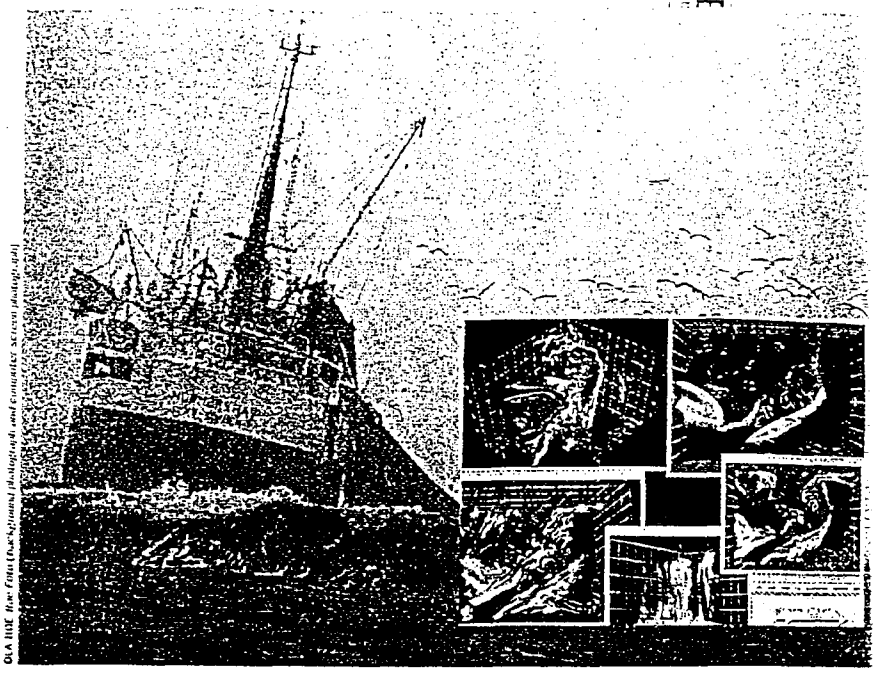
LABORATORY
To ensure t ysicians and nurses do n dvertently make decisions based on out-of-date records, the server storing laboratory results is replicated on several servers across the system. If the original server becomes unreachable for any reason, requests for records are rerouted to other sites.

BEDSIDE MONITORING
The medical-records server stores information on vital signs, such as heart rate and blood pressure, as well as when a patient received medication. This information must always be available to doctors and nurses; the system ensures its accessibility by replicating the data and the programs that manage them.

NURSES' STATION
Nurses frequently update patients' records, feeding new information into the system either at terminal stations throughout the hospital or with the aid of handheld electronic notepads.

FUTURE HOSPITAL COMPUTER SYSTEM connects patients with medical personnel throughout the building or anywhere in the world. Because the crash of one computer can endanger a patient's well-being, such systems may employ active replication, described in the article, to cope with failures.

JARED SCHNEIDMAN/ISD



WEATHER MONITORING NETWORK alerts Norwegian fishermen to dangerous storms (*left*) or hazardous oil spills (*right*). The computer system StormCast links re-

security and stability of their finances, property and even health will increasingly depend on distributed computer systems. Thus, although it is easy to talk about the potential benefits of the information superhighway, we believe the bridges that link computers must be inspected more closely. Various computer scientists, including the two of us, have been working since the late 1970s on developing software to improve distributed computer networks, making them more secure and resistant to failure—an activity that people in the field refer to as designing robust distributed computing systems.

Why do distributed systems crash? If we exclude systems that fail because they were mismanaged or poorly designed, the most common scenario involves an isolated problem at one site that triggers a chain of events in which program after program throughout the network eventually shuts down. One response to this threat might be to strengthen individual components—incorporating computers and disks specially designed to tolerate faults, for example. But ceilings can still leak, causing short circuits; power can fluctuate; and communications connections can be inadvertently cut. Acts of sabotage by hackers or disgruntled employees can also endanger distributed systems. Although engineers and computer programmers can improve the durability of hardware and software, no computer can ever be made completely reliable.

Even if every component of a system were extremely dependable, the story would not end there. Merely interconnecting reliable computers and bug-free programs does not yield a robust distributed system. Instead it produces a network that works well under most conditions. Electronic-mail programs, bulletin boards and the Web were designed using components that, considered individually, are very trustworthy. Yet these systems frequently freeze when anything unexpected happens to an individual component of the system; for instance, the system may crash when one machine or a communications line becomes overloaded. Some additional form of protection is therefore needed.

During the past two decades, programmers have attacked the dependability problem by developing fault-tolerant software—programs that allow computer systems to restore normal operation even when problems occur. The technique eliminates the chains of internal dependencies that link the operation of a system as a whole to the operation of any single component. The resulting systems do not need to shut down even if some sites go off-line. Instead they resume service by rapidly reconfiguring to work around crashed servers.

Saved by the Backup

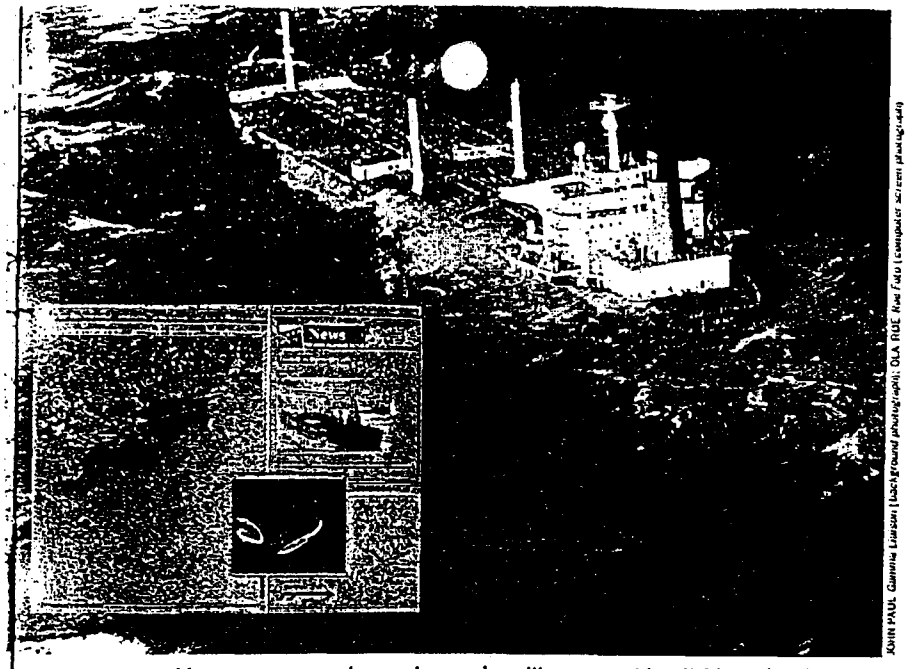
Computer scientists refer to these arrangements as highly available distributed systems. Because these systems are designed to replicate critical information continuously and to distribute multiple backup copies among their individual computers, they can adapt to changing conditions—a malfunctioning disk drive at one site, an overload at another, a broken communications connection and so forth. As long as failures do not occur so often that the software lacks time to react, these systems can respond by pulling up from elsewhere a duplicate copy of a needed file or a replica of an on-line program. In this way, a system as a whole remains available and, ideally, provides uninterrupted service to the users still connected.

A simple and popular method of building a highly available distributed system involves a primary and a backup system. If the primary machine fails, the backup can be called into service. Switching between the two is easy if the data never change. The conversion becomes difficult, however, if data or files change while the system is running. And in an extensive network of servers, data, files and programs, it can be difficult to distinguish between a system that has genuinely crashed and one that is merely experiencing communications difficulties.

Suppose that a computer is trying to update information on both the primary and backup servers, but one of them stops responding to messages. If the problem is merely in the communications lines, the messages will get through, given enough time. But if the server has actually failed, the computer doing the updating would wait indefinitely; in the meantime, the system would be unavailable. If the computer trying to carry out the update inappropriately stops waiting and sends the update to only one server, however, the primary and the backup will no longer be identical. Errors will arise if the system attempts to use the outdated server.

The NASDAQ financial market illustrates one way to resolve this conundrum. The network has two central trading servers. To prevent confusion, only one is active at any given time. The NASDAQ operators themselves decide when to switch to the replacement server. Unfortunately, very few distributed systems can rely on the wisdom of a human operator to detect failures and then to switch the entire network from one server to another. Rather programmers must automate this decision so that the transition can occur seamlessly.

EPIC000401



JOHN PAUL Gamma Liaison (background photograph); OLA RUDE Ken Futo (computer screen photograph)

mote video cameras, weather stations and satellites to provide reliably updated reports. StormCast can be accessed on the World Wide Web at http://www.cs.uit.no/

Moreover, highly available distributed systems often have large numbers of servers and programs. Consequently, these systems typically maintain a membership list, which keeps track of every program, noting whether it is working or not. If a program is unresponsive for any reason, it is marked as faulty. By recognizing a failure at one site, the system can then reconfigure itself and redirect work to operational sites.

The NASDAQ system also demonstrates a second concern about reliability in distributed systems. The two-hour trading delay in 1994 could have been avoided if the operators had switched immediately to the backup. They opted to wait, however, because of concerns that a software bug might have caused the primary system to malfunction. If such a bug were present, the backup might also crash, just as the AT&T backup system did. Because it is impossible to guarantee that software is completely free of bugs, some form of protection is needed to reduce the risk that backup versions of a critical server will crash following the failure of a primary server.

Programmers have responded to this challenge with an approach known as active replication. In active replication, a system's software establishes redundant copies of vital programs or servers through the use of so-called process groups. A process group links a set of programs that cooperate closely. A distributed system may contain many process groups, and programs can belong to several of these groups. Each group is known by a name much like a file name and has its own list of current members. Most important, the process group provides a means for sending messages to its members. This message-passing function ensures that each member of the group receives every message in the same order, even if the sender crashes while transmitting the message.

If a particular program is necessary for maintaining availability, the system introduces a group of programs, each of which replicates the original. To update the data managed by the replicated program, the system sends a message to the process group. Each member reacts by updating its particular replica. Because all the programs see the same updates in the same order, they will remain in mutually consistent states.

Active replication enables a system to tolerate faults because any group member can handle any request: if one machine crashes, work can be redirected to an operational site. Furthermore, if a request does not alter data, one site can process the query rather than tie up the entire system. In this way, multiple tasks can be worked on at once by different programs, speeding up the application by employing parallel processing.

Of course, if all members of a process group handle an incoming message in the same erroneous manner, all the members could, in theory, crash simultaneously. Although it would seem that active replication should be vulnerable to such failures, this turns out not to be the case. Programmers have often observed that the errors most likely to be missed in testing software are those involving the order in which data are received. These bugs can be provoked only by unlikely sequences or timings of events. When a system employs active replication, the replicas do see the same updates in the same order; however, updates are only a small part of the requests a program sees. Most of the time, replicated programs work in parallel, with each program handling its own set of queries in a unique order. Thus, even if a software bug slips through testing and interferes with a few parts of a networked application, it is unlikely to cause all the members of any particular process group to crash at the same time.

The idea behind active replication is simple, but the software needed to support it is not. Managing dynamically changing membership lists and communicating with process groups is difficult, particularly in the face of inevitable crashes and lost messages. Although distributed computing has become commonplace over the past decade, active replication has only recently emerged from the laboratory.

Tool Kits for Robust Networks

Over the past few years, more than a dozen software teams have developed packages for robust distributed computing systems. All provide high availability through active replication, although they each differ somewhat in their emphasis. Some packages focus on speed, for example; others on the need for security.

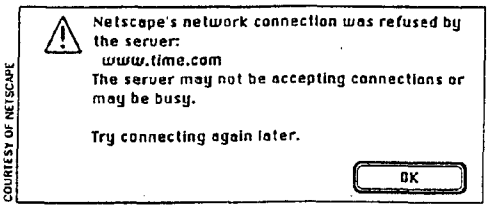
Our research efforts at Cornell University contributed two such packages. One of us (Birman) headed the team that introduced Isis in 1987; more recently, the two of us worked on Horus, introduced in 1994. The names "Isis" and "Horus" allude to Egyptian mythology. The goddess Isis helped to revive the god Osiris after he was torn to pieces in a battle with the war god Set; Horus was Isis's son, who eventually triumphed over Set. By analogy, the Isis and Horus packages can help restore a distributed system that has been disrupted by a failure.

Packages such as Isis employ a set of software functions, or "tools," that replicate and update data, keep track of

EPIC000402

The Tangled Web

Most of the World Wide Web is invisible. For many users, the Web appears to have only two components: a browser program and remote servers where documents to be downloaded are stored. But the Web—and the Internet, which provides the medium of communication that connects Web sites—consists of millions of additional programs and servers; dozens cooperate to fetch a single document. For example, to fetch an item from a server at Cornell University, the name "www.cornell.edu" must be translated, or mapped, to a numerical address that the software recognizes. This task may be handled by a series of mapping programs before the correct address is located. Furthermore, a request to connect to a Web site typically passes through a number of so-called proxies—programs that save copies of frequently accessed documents as a way of reducing the load on remote Web servers. If a nearby proxy has stored a needed document, the user can avoid a lengthy file transfer over the Internet.



COURTESY OF NETSCAPE

Such hidden dependence on intermediate programs is common in distributed systems, but it can contribute to system failures. If a name-mapping program does not respond, if a proxy fails or if the Web server crashes, the initial request will not go through. So the ubiquitous error message that the Web server has failed or is busy (*above*) can be misleading: the overload or failure of any number of intermediate programs can result in such an outcome.

The Internet as a whole can experience "brownouts" somewhat akin to a telephone or power outage. For instance, in late 1995 a major Internet name service, located in Atlanta, became intermittently overloaded. During these periods, no one on the Web could fetch documents from servers whose addresses were not already known to the local system. This type of brownout can affect large numbers of people worldwide. Even when a connection is made to a server, errors can still occur. Copies saved by Web proxies are not updated when the original document is, so there is no guarantee that users will see the most up-to-date version of a Web page. In many situations, this possibility does not cause significant problems, but some time-sensitive applications can become untrustworthy if the necessary documents are not kept current. Web proxies improve the Internet's reliability in one sense, by reducing the processing load on the network. But proxies decrease reliability by creating the possibility that people will see outdated information.

Critical projects on the Web or other distributed computing systems will require stronger guarantees that information retrieved from the system is accurate, current and always available when needed. One way to avoid such mistakes is to arrange for saved copies of vital information to be managed by the process of active replication described in the accompanying article. —*K.P.B.*

process groups and assist in handling membership changes. Isis can also parcel out data processing among servers (a procedure known as load sharing). Distributed systems that make use of load sharing exhibit many of the advantages of parallel computing but without requiring special-purpose parallel computers. By dividing up incoming work among multiple servers functioning in concert, Isis enables systems to manage large tasks quickly. Also, if a particular application requires additional computing power, one can add an extra server, and the load-sharing technique will adapt itself to the new group size. The possibility, offered by such tool kits as Isis and Horus, of improving both performance and reliability often surprises developers: they tend to assume that making a system more robust will also make it slower and more expensive.

Active replication has been applied in a number of settings, including several telecommunications networks, stock markets, banks and brokerages. In Norway, researchers have developed an environmental monitoring system based on the technology [*see illustrations on preceding two pages*]. The French air-traffic-control agency is also exploring the technique for use in a new generation of air-traffic-control software. And manufacturing plants have used process groups to coordinate work and to reconfigure assembly lines when equipment is taken off-line to be serviced.

As computer scientists look to ever more demanding applications, however, they discover that active replication has important limitations. Load sharing is not always possible or desirable: some systems (notably, those in which data stored at a server change very rapidly) slow down when components are replicated. For example, in videoconferencing technology, active replication does improve the fault tolerance of the network of servers that must keep running even when some participants are cut off. But the technique would slow down the system—without improving dependability—if applied to the transmission of video data to remote users.

The need for flexibility motivated us to develop Horus. Like the Isis tool kit, Horus supports active replication, but it also provides much greater versatility. The basic strategy behind Horus is modularity, resembling that of a child's set of Legos: different building blocks of Horus can fit together in any combination to support the specific needs of a particular process group. One block might encrypt data so that hackers cannot break into the system. Another block might address potential communications failures that can arise when messages are lost or corrupted. Programmers using Horus decide which properties their system actually needs, permitting them to customize the system for its intended use. Furthermore, Horus can be extended with custom-designed blocks for special needs that we may not have encountered or anticipated in our own work.

Horus has a growing group of users worldwide. At Cornell, Brian C. Smith has used it to build a videoconferencing system for "groupware" applications. Horus information is available from http://www.cs.cornell.edu/Info/Projects/HORUS/

A Crisis of Will?

Our work on Isis and Horus has convinced us that careful planning can ensure the dependability of computer networks. But making the information superhighway robust may take more

EPIC000403

time and money than computer makers and users are willing to commit. Software for distributed applications is typically built with existing technology that was not designed for dependability. Moreover, researchers need to seek better methods for designing large-scale systems that are robust and that provide very high performance: a system that is extremely robust when accessed by 50 users simultaneously may turn out to be unacceptably slow and hence unreliable if 5,000 people do so.

Although programmers have applied the technology for robust distributed computing successfully in some instances, the public hears more about failures of nonrobust systems. For example, over the past few years, there have been dozens of reports on the problems with the current air-traffic-control system. In the fall of 1995 the Los Angeles system failed, leaving controllers unable to communicate with aircraft; a midair collision was avoided by seconds.

To make matters worse, updated air-traffic-control software, commissioned in 1982 by the Federal Aviation Administration, has been repeatedly delayed and scaled back. The FAA selected the original proposal precisely because of its innovative approach to distributed computing; now it seems the highly available and distributed aspects of the proposed software have been almost entirely eliminated. Yet air-traffic controllers criticize the existing system as dangerously inadequate, particularly because it lacks a distributed software architecture and has become undependable with age. Highly publicized fiascoes such as these have fueled a common perception that there is a crisis in computer software [see "Software's Chronic Crisis," by W. Wayt Gibbs; SCIENTIFIC AMERICAN, September 1994].

But if we are really in the midst of a software development crisis, it is perhaps as much a crisis of will as of means. Not all developers are concerned with making their networking software robust, and the public pressure for reliability does not seem to extend beyond a few especially sensitive applications. Indeed, companies that market distributed computing packages often state in product licenses that their technologies may not be dependable enough for use in critical applications—implying that reliability is not a reasonable objective. In our opinion, this situation is analogous to the unlikely prospect of automakers selling cars with the warning that vehicles are unsafe for use on highways. The computer equivalents of safety belts and air bags are infrequently applied to software development. And the desire for sophisticated, user-friendly interfaces as well as improved speed and performance tends to dominate the attention both of the software developers and the people who use the programs.

Reliability often conjures up an image of slow, ponderous computer systems that is incompatible with the allure of effortless and instantaneous access to information on the data superhighway. Yet robust technology does not have to be slow and unpleasant to use: the Golden Gate Bridge is a model of stability as well as grace. With each passing hour, more and more uses are being found for the information bridges that link computers. Our enthusiasm to incorporate elegant electronic bridges in every conceivable application should not overshadow a reasonable degree of concern about whether or not such bridges will be able to support the resulting traffic of information. We believe that robust distributed systems provide a valuable tool for connecting computers quickly and dependably, creating opportunities for business and pleasure in the information society. But we also believe that in many cases, unless a distributed system can be engineered to function robustly, it may be better not to build—or use—one at all. ■



AIRLINE PASSENGERS waited for delayed flights in New York City's La Guardia Airport in May 1995 after a power failure triggered a shutdown of the local air-traffic-control system. The effort to rejuvenate aging U.S. air-traffic-control software using distributed systems dates back to the 1980s.

The Authors

KENNETH P. BIRMAN and ROBBERT VAN RENESSE have worked together on distributed computing systems for the past five years. Birman is professor of computer science at Cornell University. After developing the Isis tool kit in the 1980s, he founded a company to commercialize the technology. Isis Distributed Systems now operates as a division of Stratus Computer, Inc. Van Renesse entered the field of distributed computing after deciding not to pursue a career as a circus acrobat. He is now a senior research associate at Cornell and is the primary architect and developer of the Horus system.

Further Reading

FAULT TOLERANCE IN TANDEM COMPUTER SYSTEMS. Jim Gray, Joel Bartlett and Robert W. Horst in *The Evolution of Fault-Tolerant Computing*. Edited by A. Avizienis, H. Kopetz and J. C. Laprie. Springer-Verlag, 1987.

FATAL DEFECT: CHASING KILLER COMPUTER BUGS. Ivars Peterson. Random House, 1995.

GROUP COMMUNICATION. Special section in *Communications of the ACM*, Vol. 39, No. 4, pages 50–97; April 1996.

EPIC000404

Beyond the Web: Excavating the Real World Via Mosaic

THE MERCURY PROJECT.

- Ken Goldberg, Assistant Professor, Computer Science
- Michael Mascha, Assistant Professor, Anthropology and
- Steven Gentner, M.S. Candidate, Computer Science
- Juergen Rossman, Graduate Student, University of Dortmund, Germany
- Nick Rothenberg, PhD Candidate, Visual Anthropology
- Carl Sutter, Senior Programmer/Analyst, Center for Scholarly Technology
- Jeff Wiegley, PhD Candidate, Computer Science

University of Southern California. Los Angeles, CA.

(To appear in the Second International WWW Conference, Chicago, IL, Oct 17-21, 1994.)

Abstract

This paper describes a Mosaic server that allows users to "leave the Web" and interact with the real world. An interdisciplinary team of anthropologists, computer scientists and electrical engineers collaborated on the project, desigining a system which consists of a robot arm fitted with a CCD camera and a pneumatic system. By clicking on an ISMAP control panel image, the operator of the robot directs the camera to move vertically or horizontally in order to obtain a desired position and image. The robot is located over a dry-earth surface allowing users to direct short bursts of compressed air onto the surface using the pneumatic system. Thus robot operators can "excavate" regions within the environment by positioning the arm, delivering a burst of air, and viewing the image of the newly cleared region. This paper describes the system in detail, addressing critical issues such as robot interface, security measures, user authentication, and interface design. We see this project as a feasibility study for a broad range of WWW applications.

Goals of the Project

WWW and Mosaic[1]-like servers provide a multi-media interface that spans all major platforms. Thousands of sites have been set up in the past year. Our goal with this project was to provide public access to a teleoperated robot, thus allowing users to reach beyond the digital boundaries of the WWW.

Such a system should be robust as it must operate 24 hours a day and it should be low in cost (we had an extremely limited budget). It is worth noting that the manufacturing industry uses the same criteria to evaluate robots for production. Thus our experience with RISC robotics (see below) proved helpful.

Our secondary goal was to create an evolving WWW site that would encourage repeat visits by users to collectively solve a puzzle. As of this writing we do not have sufficient data to report on the success of the "puzzle" component; therefore this paper focuses on the details of the implementation. We also speculate on how Mosaic might be used for other tele-operated applications.

Related Work

The first "teleoperated robots" were developed over 30 years ago. The basic objective has always been to develop systems capable of working in inhospitable environments (such as radiation sites). Teleoperation began with very simple mock-ups in nuclear power plants [Mos], progressing to more versatile setups for teleoperation of robots in space [Miz]. Over the last 20 years, the development of intuitively operable teleoperation tools has continued to play an important role in the development of

robotics in general. The basic objectives have remained the same, even though the methods and technical limitations have changed.

Today, sophisticated "Telerobot Operator Control Stations" [Kan] are equipped with stereoimage-displays, "force reflecting hand controllers" and comprehensive video graphics support. The development of teleoperation stations is currently being pushed further with the help of latest graphics workstations to provide so-called "telepresence." Modern telepresence systems, considered to be pushing the frontier of research in this field, are defined as follows [Aki]: "At the worksite, the manipulator has the dexterity to allow the operator to perform normal human functions. At the control station, the operator receives sufficient quantity and quality of sensory feedback to provide a feeling of actual presence at theworksite."

The Mercury Project does not achieve this level of telepresence but provides a limited level of teleoperation. One of our goals was to provide "teleoperation for the masses." Instead of developing a highly sophisticated, multi-million-dollar testbed, we opted for a simple and reliable end-effector on a commercial robot. Combined with an intuitively operable man-machine-interface, the system gives all WWW users access to teleoperation.

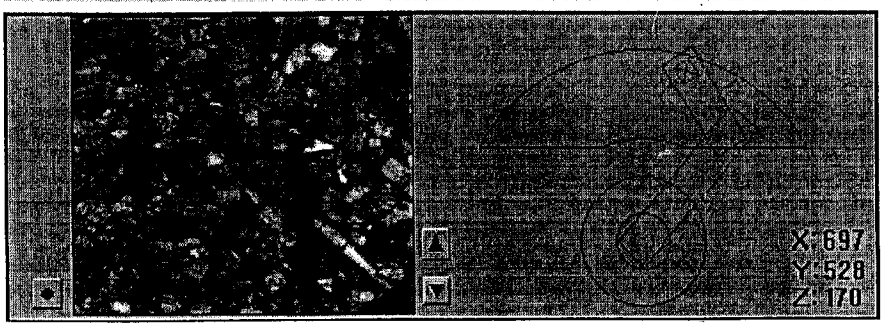
In the Discussion section, we describe a number of other WWW sites that offer interactive capabilities.

User Interface and Environment Design

The interface design for the system was challenging due to the limitations of the HTML/HTTP environment, as well as network traffic considerations. An effective system was created within such limitations by carefully designing the physical environment for the robot, and by fine-tuning the user-machine interface. For example, the initial idea of a live video feed from the camera was dropped in order to maintain compatibility with all visual clients on the Web. (Although we could have implemented some custom clients [2], we decided to stay within the limits of HTML/HTTP to reach as large a user base as possible, making this a truly global system.) In addition, initial simulations using a robot fitted with grippers (simulated in VIRTUS WALKTHROUGH) revealed a high degree of complexity in control functions [3], not suitable for the anticipated 5-10 seconds per frame page loading time, a 2D Mosaic window and a naive/untrained user.

The team chose instead to use a simple environment which would allow relatively easy control of the robot. Here the analogy taken from real world archaeology - using a dry-earth environment and compressed air bursts - allowed us to simplify the robot control dramatically. Thus users could be quickly trained in the operation of the system, through a simple "OperatorÕs Orientation" and a "Level 1 Clearance Test."

Even with a simplified system, users are still able to choose between fine and gross movements of the arm. Fine pitch movements are executed by clicking in the camera image, with the robot moving to center the arm over the X,Y coordinates of the click-point. Crude navigation is provided by clicking on a schematic picture of the robot and it's workspace, with the robot moving to center the arm over the click-point. Two buttons allow navigation in the Z axis (between "up" and "down" positions), with a button to blow air only active when at the Z=O (i.e., "down") position.



(Click to see an animated robot operation session in MPEG - 175K)

Other features of the system were designed to balance functionality with user needs. All HTML documents sent to the clients are carefully designed to minimize network traffic in order to get a high refresh rate. For example, control panel functions are clearly distinguished from text-based information documents. The ÒOperator's LogÓ was implemented to create a forum for collaborative efforts to solve the puzzle/problem regarding the underlying logic which links the artifacts. (The ÒOperator's LogÓ is readable throughout the system but only writeable after completing an operating session.) A second entry path was also created to the system, which provides a "back-story" explaining the project while also hinting at possible "real world" uses of the system.

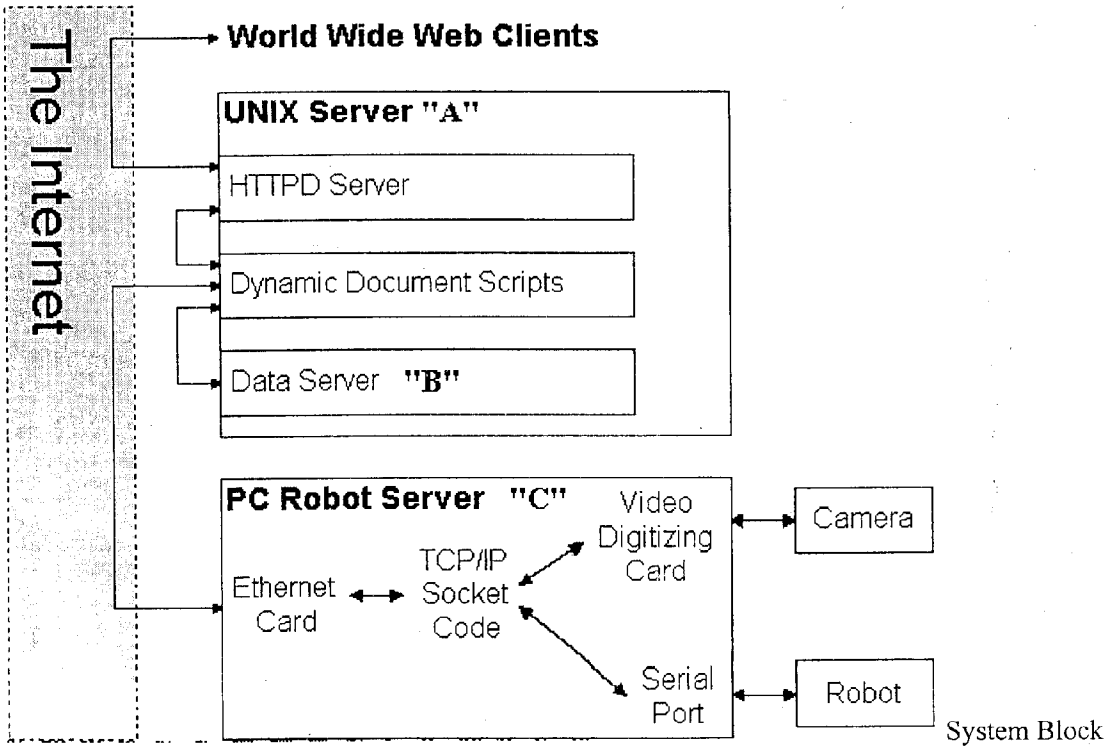
Access to the Robot

Most of the HTML documents seen by the user on our site are generated by a script running on the WWW server. Using a random token scheme described below, the system tracks each user as he or she proceeds through the interface and generates appropriate HTML documents. This allows the system to discriminate between "observers" and "operators" so that it presents only accessible options to each.

To operate the robot, the user must read the information on how to use the control panel, and then complete a level-1 clearance test to get a password. Since only one person can operate the robot at a time, the system maintains a queue of pending operators. A typical user will enter his/her password, and then add him/herself to the queue. Each time update button is clicked, the system updates the queue and returns a current status page. When the user's turn arrives, the screen returned is the live operators' control screen.

System Architecture

Below is a Block Diagram for the system. We start with an overview that necessarily glosses over many interesting details.



System Block

Diagram

At one end are WWW clients from around the world; at the other end is a robot arm combined with a camera. The robot and camera provide an updated image of the environment, which is combined with a schematic of the robot arm/workspace and control buttons to produce the final GIF image that is send to users.

At any given time there may be dozens of clients interacting with the system. Since there can only be one Operator at a time, one challenge is to keep track of which client is the operator.

The Mercury system is comprised of three communicating servers. The first, call it A, is a standard Mosaic server (NCSA httpd v.1.3, currently running on a Sun SPARCserver 1000, with SunOS Release 5.3. When the RTE Site is requested by an observer, the most recent image, which is stored on server A, is simply returned.

The database of registered users is handled by another server, call it B. In our case, Server B runs on the same machine as server A. The database server is custom programmed for this project, but performs fairly standard database functions.

When a client request comes in, Server A communicates with server B. If that client is an Operator, Server A must then communicate with a third server, call it C, that controls the robot. Server C runs on a Pentium-based PC and communicates with servers A and B via the Internet. Server A decodes the ISMAP X&Y mouse coordinates, and sends them to server C.

On server C, a custom program decodes these coordinates into a robot command and verifies that the command is legal, e.g., within the robot workspace. If it is, this command is then converted into a robot command format which is sent to the robot over a serial line. Once the robot move is completed server C uses the CCD camera to capture a stable 8 bit 192x165 image of the workspace.

Using a simple set of equations for inverse kinematics server C then generates a schematic view of the robot in its new configuration. This schematic is combined with the camera image, and the up, down, and air control buttons to form a new composite image. Server C then compresses this image into GIF format and returns it to Server A, which updates the most recent image and returns it to the Operator client.

Subsystems

Random Tokens for Cache Avoidance and User Tracking

Following some complex and unwieldy tests, we implemented a random token scheme for tracking users as they use the system. Each time a URL is returned, a large random number is added to the path (which the NCSA HTTPD 1.3 server splits into the PATH_INFO environment variable). This "token" serves two purposes:

The first is to prevent the WWW client from caching the robot view. When a document is requested a second time during a session, it is much faster to swap in a local copy of the document rather than going back over the net to retrieve it a second time. Most implementations of Mosaic support such caching at various memory levels. However in our case we want to repeatedly retrieve the URL containing the robot image because it is updated continuously. In brief, we DON'T WANT users to cache this url. The random token makes each request look different and tricks the client into retrieving a fresh version of the document.

The second use for the token is to identify Operators. When an operator logs in with a correct password, the system begins tracking him/her as he/she moves from viewing the robot to being on the operators' queue to operating the robot. Since the same script is used for all views, the token allows the system to customize the result for every user depending on his/her position in the system.

Scripts

The robot view screen is controlled for the most part by one script at the HTTPD server. Each call to the main script has a token attached to the URL. The token is decoded by the WWW server, and placed in the PATH_INFO environment variable. The main script then checks the token with the database server to determine the status of the user. Each check of the database generates a system update to keep the queue moving. The user's status is used to generate the custom system status page.

The robot image itself is only changed by the operator when he or she makes a move. Each image is date and time stamped, so WWW clients that cache the image will only retrieve the image when it changes (since its filename will be different, due to a different time stamp).

Due to the client-server architecture of the World Wide Web HTTPD protocol, The robot system (server) has no way to contact the client except at the client's prompting. From the user's point of view, once he or she gets the robot view screen, there is no way for the server to keep sending updates automatically as the robot is moved by the operator. The screen updates must be driven by the user. Since the user must trigger each update, we wanted to provide a button for doing so, since each web client handles reloading the page differently. Some sites have a "reload" hypertext link to the same page, but this doesn't work for any client that caches pages. If a page is being viewed, hitting reload will just re-display the page from the cache, thus not obtaining a new view from the system. Asking the user to disable his/her cache is also problematic, since not all clients allow this option.

One attempt was made to use a mini-form, since the submit button always calls a script and is not cached. that scheme was eventually dropped, since passing registered user identification information to the server via hidden fields only worked on some clients. Using the random token allows for an elegant interface.

Since the robot can only be controlled by one person at a time, a registration scheme was implemented to allow the server to track operators as they move on to the waiting queue and progress to controlling the robot. Since the server only knows the IP address of each user, some user information had to be incorporated into the HTML robot view document itself for re-transmission to the system when the user hits "reload." There are various techniques used by many sophisticated web systems to accomplish user identification between document requests, but we found some problems in many of the standard solutions. In the end, the random token served excellently as a means of identifying registered users.

A preliminary attempt was made to use a small form to identify the user. Hidden fields could hold the user id, but once again, many clients do not implement the hidden field attributes so the interface is cluttered by unnecessary fields. Putting the user's id information into the ACTION field of the form tag is also client dependent. Unfortunately, some clients strip that data before adding the encoded field information.

Since random tokens were already being passed with each update, the system was extended to track the tokens of each registered user. Each time the script is called, the token is exchanged for a new one, and the database is updated with the new token for registered users. One side effect is that the user can not use the client re-load button, since this will not use the new URL (it is embedded in the update HREF).

The Data Server

The data server ("B") is a custom Perl script that handles all of the database work for the project. It continuously runs as a TCP/IP listener, waiting for database transactions from the other system scripts. The data server runs as a single process, handling requests serially to maintain internal data integrity. Typically, transactions are very short, since the data is kept in main memory. The data server could be replaced by an off-the-shelf transaction based DB system in the future. A time-out is set to close the connection if there is too much time elapsed between commands. This was implemented because some WWW clients would crash in the middle of a document request, leaving the system waiting for the

EPIC000409

connection to be closed.

Internal Network Interface

The networking functionality required by the project was defined by two factors. On one hand, the camera that we purchased required a PC-based platform running an MicroSoft DOS or compatible operating system to run on Server C. On the other hand, the expected load of client requests required a machine capable of more heavy networking duties such as a Sun workstation (Server A). Currently Server A is located across campus from server C.

These servers are connected via Ethernet. Each machine has its own IP address and resides in the usc.edu domain. Communication is achieved using a socket connection between the two machines. The implementation on Server A was done using the standard BSD socket functions provided with the SunOS 4.1 operating system and Perl. On Server C we used a publicly available socket package called Waterloo TCP and Borland C. The Waterloo TCP package was obtained from the ftp site dorm.rutgers.edu in the file /pub/msdos/wattcp/wattcp.zip.

With this software Server A can request a socket connection to Server C to establish a connection. The first step in obtaining a new image is for Server A to write a command consisting of thirty bytes which encodes the (xy) coordinates of the ISMAP event. After Server C completes the moves and generates the new image, it writes the size of the new image to server A so that server A knows exactly how many bytes to expect. Server C then proceeds to write the entire image to the socket and waits for the socket to close to insure deliver of the data. Once server A has read all the specified bytes it closes the socket. Server C is now ready and waiting for another socket connection. Server A is free to continue processing the Mosaic actions of the current users.

Current throughput is approximately 20 Kbytes/second, which is poor compared to the 0.5 megabyte per second rate that can be achieved between two Sun workstations in close proximity on the campus network. At this time we feel that the delays are being imposed by the MS-DOS operating system because of it's inability to support networking operations and its lack of multitasking abilities, which necessitates busy waiting cycles in the PC software to obtain concurrence between the robotic/camera operations and the networking duties.

Our low data rate is somewhat tolerable because the time for communication between Servers A and C is small compared with Internet delays between clients and server A. One way to speed communication would be to use different methods of image compression such as JPEG to reduce the size of the image. However this may introduce latency due to encoding.

The IBM Robot and Server "C"

The robot we're using is an IBM SR5427 SCARA arm, built around 1980.

SCARA stands for "Selective Compliance Assembly Robot Arm" . Robots with SCARA kinematics are common in industrial assembly for "pick-and-place" operations because they are fast, accurate and have a large 2.5D workspace. However, the SCARA arm can only rotate its gripper about the vertical (Z) axis. We selected this robot over other robots in our lab due to it's excellent durability, large workspace, and because it was gathering dust in the Robot Education Lab.

The IBM SCARA robot is controlled through a 4800 baud serial port by a custom written C library constructed with reference from IBM's BASIC library distributed along with the robot. The commands sent by the library are simple instructions consisting of instruction id, length, data and checksum. The data length and content varies depending on instruction id. The IEEE floating point format is used to represent the necessary data. This command string is then sent over the serial line to the robot to issue the command.

Unfortunately IBM no longer supports this robot and we were forced to read two antiquated BASIC

programs and monitor their serial line transmissions to decipher the protocols needed for serial control of the robot. The robot accepts XYZ and Theta commands using IEEE format and checksums. Server C now runs on a Pentium based PC with all custom code written in Borland C.

The first step was implementing a local graphical user interface to control robot movements and monitor subsequent functions such as network flow. We chose two views of the workspace: a global schematic view for coarse motions, and a local camera view for fine motions. Note that a click on the camera image requires a different relative move if the camera is in the up or down position. To handle it, we implemented an empirical calibration program.

The major difficulty in implementing Server C was to schedule response to the network, the local mouse, and the image capture board. At first we discussed a multi-tasking environment but, upon further study, we realized we could achieve this cooperation within a single DOS task. Another problem, inherent to DOS based applications, is memory management. This complication was solved by careful usage of memory and by utilizing the screen itself as a memory buffer. This careful usage of memory enabled the custom written GIF encoder to use more memory which, combined with an appropriate hash function, sped the GIF encoding process up to a few microseconds.

In future versions of Mercury, we plan to incorporate a more sophisticated PC-based robot simulation system based on COSIMIR [Fre] from the University of Dortmund.

Camera

We are using an EDC 1000 digital CCD camera from Electrim Inc. This camera was chosen based on size and cost. Image data is sent from the camera back through a serial line into a video capture card. The picture captured is always 192 by 165 pixels with 256 shades of gray. The image size and gray shades are fixed. Focus and contrast are manually adjusted. Exposure time can be changed by software to range between 1/200 th to 1/64 th of a second. 1/150th exposure was used to reduced light streaking that the camera is prone to.

Although the robot's control system quickly dampens oscillation about the destination point, dynamic effects can cause image blur. Two solutions were implemented. First the robot was slowed down enough as to reduce some of the vibration but not to hinder the robot access speed considerably. Second, once the robot responds positively to an issued command, the camera captures two pictures each at 1/64 th of a second. These two images are compared to determine a factor of similarity. If this factor is below some set value the image is presumed to be stable, otherwise subsequent pictures are taken until the image pair is determined to be stable. More then 5 trials results in a time-out in which case the most current image is used and the program continues. This image comparison procedure reduces movement streaks seen in pictures of moving objects.

Lighting the workspace has been problematic. The work space is primarily luminated by standard florescent ceiling fixtures and augmented by two additional florescent lamps to reduce shadows and raise the overall ambient light levels. We tested a contrast enhancement routine to normalize the lighting of each image captured from the camera. This increased the visual aesthetics of the image but subjected it to drastic light and dark changes as the robot moved onto different objects with different light reflecting qualities. In response, a global lighting adjustment was implemented but found to reduce certain areas to unacceptable darkness. Certainly a better lighting system is required.

Due to the manual focus adjustment of the camera, the focus adjustment could not be changed between the up and down position of the camera. This resulted in a compromise focus adjustment that is not perfect for the up or down position of the robot arm, but accepatable in both positions.

To decrease compressed image size and thus increase network transfer rate the image is reduced from 256 to 64 gray scales since most systems available can only display 256 colors or 64 shades of gray. Thus the gray scale reduction did not reduce image quality but reduced compressed image size by about 10K.

Robustness and Soft Resets

All robot motions are monitored by Server C. Each command sent to the robot is verified to be within the robot's workspace. Acknowledgments from the robot are monitored to detect errors. When an error is detected, Server C automatically resets the robot controller, recalibrates, and returns the robot to its previous position.

Performance

History and Statistics

Daily statistics are available and may be correlated with project milestones. As of the writing of this paper, the system has been online for over 4 weeks and there are approximately 100 users per day. There is also a list of all hosts that have visited the system. As of this writing the system has been visted by hosts from all of the major continents except the polar caps.

Refresh Rates via Ethernet

System response time seems to be mostly dependent on network link speeds. Locally, we get screen refreshes at rates of 5-10 seconds per page. Similar response times have been reported from Europe. Obviously, a slow local link or SLIP connection will drastically affect the update speed, since the robot control image is essential to the system. Updates are also strongly affected by the speed of the WWW client application.

Uptime

The system is designed for 24 hour use. The WWW server scripts are generally modified, tested and then loaded into the running system. Background programs monitor the system and notify the team members if there are problems.

Operators' Logs

When an operator has finished driving, he or she is prompted to make a textual entry into an "Operator's log". The Operator's log provides an ongoing forum for discussion of the system and record of artifacts discovered in the sand.

For example, several skeptics have claimed that the system is an elaborate hoax where all images are taken from a prestored library (much like the celebrated Apollo Moonwalk hoax of 25 years ago). We have had encouraging comments from the robotics community, including several researchers at NASA.

Discussion and Future Applications:

This project is an initial step in an ongoing educational and research project at the University of Southern California. It brings together faculty and students of different backgrounds to collaborate in the design and implementation of a networked system that combines robotics with archaeology and interactive art.

This system exemplifies RISC Robotics, which advocates Reduced Intricacy in Sensing and Control. The SCARA-type robot requires only 4 axes, is relatively inexpensive and robust, and it is easy to avoid singularities. The end effector we've used here is also about the minimum. For more on RISC as applied to industrial robotics, please see RISC for Industrial Robots: Recent Results and Open Problems, (with J. Canny),1994 IEEE Conference on Robotics and Automation.

We see the project leading in several directions. For Mosaic and the WWW, the required interface design prompted new developments related to several issues, including user authentication, user queuing

and interface security (as discussed above).

For this project we chose a very simple application. The server can be extended to a variety of platforms that permit remote inspection and manipulation of objects -- for example, providing unique and unedited access and views of priceless and otherwise inaccessible resources (a Grecian urn, a Gutenberg Bible, etc.), thus providing an alternative to pre-stored libraries which are limited in terms of perspective, depth of resolution, etc. (Cite Recent NY Times article on the Metropolitan Museum of Art).

Further extensions for this project might include: the robot could be placed out in the field, in a remote anthropological site or on the moon; the camera could be replaced with a scanning electron microscope; or the remote operator could be a doctor examining a patient or a specialist performing remote inspection or manufacturing. All of these areas also have significant implications for education, as they present the opportunity for virtual "field trips" to a live site while permitting remote manipulation from the classroom.

Anthropologists have conventionally recorded the diverse cultural heritage of humankind by means of varied media: written text, graphics, film, sound and still images. The advantage of a system like the one described in this paper lies in the fact that you do not have to rely on prerecorded media. It enables the user to view and possibly record her or his own "slice of reality". We see the Web as a perfect medium for updating pre-recorded media as described in [Mas] Interactive Education: Transitioning CD-ROMs to the WEB, a paper presented at the First WWW Conference, Geneva 94. Furthermore, we now have the possibility to combine updateable prerecorded media of all sorts with live recordings and live remote interactions. The possibilites of a system that combines global access to up-datable prerecorded media and combines it with the possibility of live remote interactions are just beginning to unfold, and are a central focus of interest for the anthropologists from the E-LAB involved in this project.

In view of other interactie WWW sites, we propose a three-tiered system describing interactively on the WWW. Under Level I, interaction is solely between digital information stored on computers or created by scripts running on such machines, and connected or communicating with the WWW and Mosaic clients. In Level II sites, the clients are able to observe the "real world" by means of a camera observing and digitizing visual and, hopefully soon, audio-visual information. The camera acts as an "eye" for the Web, providing multiple "real world images" from a global theater. A number of Web sites fall into this category, such as the Coffee Pot and the Fishtank sites. All have the same characteristic of passively observing the real world. We also know of one restricted site that allows the user to alter the user's point of view (see LEGO pan and tilt site.)

The Mercury Project introduces a third level. Level III sites reach beyond the digital domain to allow users to alter a remote physical environment. We envision this project as a first glimpse into the possibilities available at Level III. We might also speculate about other levels, which might allow remote users to control a mobile robot and thus "tele-ambulate".

Footnotes

[1]
To simplify we mention only Mosaic as a WWW client but we are aware of the fact that there are different WWW clients similar to Mosaic, e.g. MacWeb, Cello, etc.

[2]
MBONE broadcasting, REFERENCE to MIT LIVE VIDEO SITE [diversion - possible fixes to client refresh problem to show we know about the X stuff etc.] There are two possible fixes to this problem, One is to release specially modified clients that set up a two-way communication, the second is to use some other software to display the current system on the user's client workstation. Since many clients are used to view the WWW, making modifications would be difficult, especially since they are being updated all the time. Even if source code could be obtained for every major client, changes would have to made to every release of all these be on each release of these applications, The second possibility is to write a separate program to run on the clients' workstation. The problem here is to write a robots client that can be released for enough platforms

to be useful, Since this would be an esoteric piece of the system, it is not likely that other sites would customize the software for different systems like is done for the major systems. One technique is to use the X windows protocol to display a client application on the users workstation running an X server. (weather, movies) We felt that this would be a limited audience, however. It also may compromise security from the user's point of view. Both these approaches may be attempted in version 2.0 of the system to allow more enhanced use of the system for some users. The HTTPD protocol could be extended to allow these sort of connections, though - maybe we need a new protocol for passing media only back that doesn't have all the hooks into system calls like X Windows and Display PostScript.

[3]
3D control of a robot needs: 3 dimensions of spatial movement, 3 dimensions of orientation and 1 to 3 dimensions of gripper control.

Acknowledgments

- Rick Lacy, Mark Brown, and the Center for Scholarly Technology,
- Depts. of CS and Anthro,
- Prof. Peter Danzig (CS)
- The alpha and beta testers
- The Laika Foundation
- The Los Angeles Museum of Miniatures

References

[Aki]
D.L. Akin, M. Minsky, e.a.:"Space Applications of automation, robotics, and machine intelligence systems (ARAMIS)",Phase II, Vol. 3, M.I.T. Contract NASA 8-34381, NASA Marshall Space Flight Center.

[Kan]
E.Kan, J.Tower e.a.:"The JPL Telerobot Operator Control Station: Part 1 - Hardware", Proceedings of the NASA Conference on Space Telerobotics, 1989.

[Mak]
B.J. Makinson:"Research and Development Prototype for Machine Augmentation of Human Strength and Endurance", Report S-71-1065, General Electric Company, Schenectady, NY, 1971

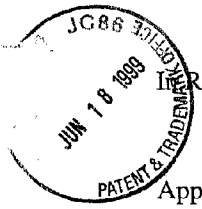
[Mos]
R.S. Mosher: "Force Reflecting Electrohydraulic Servomanipulator", Electro-Technology, pp. 138, 1960

[Miz]
N.J. Mizen: "Preliminary Design for the Shoulders and Arms of a Powered Exosceletal Structure", Cornel Aeronautical Laboratory Report VO-1692-V-4, 1965

[Fre]
Freund, E.; Rossmann, J.; Uthoff, J.; van der Valk, U. : "Towards realistic Simulation of Industrial Robots", Proceedings of the IEEE/RSJ/GI Intelligent Robots and Systems, IROS, Muenchen, 1994

D. Johnson
#3 7-15-99
105 Meyers

Attorney's Docket No. 02_ _.P001D PATENT



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Patent Application of:
Keith Lowery et al.
Application No.: 09/234,048
Filed: January 19, 1999
For: A Method And Apparatus For Creating And Managing A Custom Web Site

Examiner: Not Yet Assigned
Art Unit: 2755

RECEIVED
JUN 25 1999
Group 2700

Assistant Commissioner for Patents
Washington, D.C. 20231

INFORMATION DISCLOSURE STATEMENT

Sir:

Enclosed is a copy of Information Disclosure Citation Form PTO-1449 together with copies of the documents cited on that form. It is respectfully requested that the cited documents be considered and that the enclosed copy of Information Disclosure Citation Form PTO-1449 be initialed by the Examiner to indicate such consideration and a copy thereof returned to applicant(s).

Pursuant to 37 C.F.R. § 1.97, the submission of this Information Disclosure Statement is not to be construed as a representation that a search has been made and is not to be construed as an admission that the information cited in this statement is material to patentability.

Pursuant to 37 C.F.R. § 1.97, this Information Disclosure Statement is being submitted under one of the following (as indicated by an "X" to the left of the appropriate paragraph):

XX 37 C.F.R. §1.97(b).

_____ 37 C.F.R. §1.97(c). If so, then enclosed with this Information Disclosure Statement is one of the following:

____ A statement pursuant to 37 C.F.R. §1.97(e) or

____ A check for $240.00 for the fee under 37 C.F.R. § 1.17(p).

_____ 37 C.F.R. §1.97(d). If so, then enclosed with this Information Disclosure Statement are the following:

LJV/cak (08/28/98)

(1) A [illegible]ement pursuant to 37 C.F.R. §1.97(e);

(2) A petition requesting consideration of the Information Disclosure Statement; and

(3) A check for $_____ for the fee under 37 C.F.R. §1.17(i) for submission of the Information Disclosure Statement.

If there are any additional charges, please charge Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 6/15, 1999

James H. Salter
Reg. No. 35,668

RECEIVED
JUN 25 1999
Group 2700

12400 Wilshire Blvd.
Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8598

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231 on June 15, 1999
(Date of Deposit)

Claire Wallters
(Typed or printed name of person mailing correspondence)

Claire Wallters
(Signature of person mailing correspondence)

LJV/cak (08/28/98)

Sheet 1 of 1

OIPE JC86 JUN 18 1999 PATENT & TRADEMARK OFFICE

Form PTO-1449 (REV. 8-83) — U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE

INFORMATION DISCLOSURE CITATION

(Use several sheets if necessary)

ATTY. DOCKET NO.	SERIAL NO.
02577.P001D	09/234,048
APPLICANT: Keith Lowery et al.	
FILING DATE: 1/19/99	GROUP: 2755

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
		5,774,660	6/30/98	Brendel et al.	395	200.31	8/5/96
		5,774,668	6/30/98	Choquier et al.	395	200.53	6/7/95

RECEIVED
JUN 25 1999
Group 2700

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATION YES	TRANSLATION NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

Dias, Daniel M., et al.; A Scalable and Highly Available Web Server; IBM Research Division; T.J. Watson Research Center; 7 pages.

EXAMINER: [signature] DATE CONSIDERED: 2/17/01

*EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

A Scalable and Highly Available Web Server

Daniel M. Dias William Kish* Rajat Mukherjee and Renu Tewari

IBM Research Division
T. J. Watson Research Center
P.O. Box 704, Yorktown Heights, NY 10598

{dias, c1kish, rajatm, c1renu}@watson.ibm.com

Abstract

We describe a prototype scalable and highly available web server, built on an IBM SP-2 system, and analyze its scalability. The system architecture consists of a set of logical front-end or network nodes and a set of back-end or data nodes connected by a switch, and a load balancing component. A combination of TCP routing *and Domain Name Server (DNS) techniques are used to balance the load across the front-end nodes that run the Web (httpd) daemons. The scalability achieved is quantified and compared with that of the known DNS technique. The load on the back-end nodes is balanced by striping the data objects across the back-end nodes and disks. High availability is provided by detecting node or daemon failures and reconfiguring the system appropriately. The scalable and highly available web server is combined with parallel databases, and other back-end servers, to provide integrated scalable and highly available solutions.*

1 Introduction

With the explosion of traffic on the World Wide Web [3, 12], the load on popular servers on the web is increasing rapidly. In this paper we describe and analyse various methods for building a scalable web server on a cluster of computing nodes, in which the throughput of the cluster can be increased by growing the size of the clustered system. We analyze the limits to scalability and the efficiency of these methods, and the trade-offs between the alternatives. As both the traffic and the commercial applications on the popular web servers increases, high availability of these servers becomes increasingly important. Given a clustered web server, we describe how the server can be made highly available, by taking over the load of a failed node by the remaining nodes in the cluster. Based on these methods, we describe a prototype of a scalable and highly available web server that we have built on an IBM Scalable Parallel SP-2 system.

*Consultant from Brown Cow Engineering Inc. He can be reached at kish@browncow.com

1.1 Relation to Previous Work

The NCSA prototype scalable HTTP server is described in [8, 10]. The method they use, illustrated in Figure 1, consists of having a set of HTTP servers on nodes in a cluster, that use the Andrew File System (AFS) [6] for sharing a set of HTML documents, and using the round-robin Domain Name Server (RR-DNS) [2, 13] for distributing accesses among the nodes in the cluster. Essentially, clients are presented a single name associated with the set of HTTP servers, and the RR-DNS maps this single name to the different IP address of the HTTP servers, in a round-robin manner; thus different clients will (ideally) be mapped to different server nodes in the cluster. In this way, the load is distributed among the servers, each of which access the same set of URLs through the distributed file system (AFS in this case).

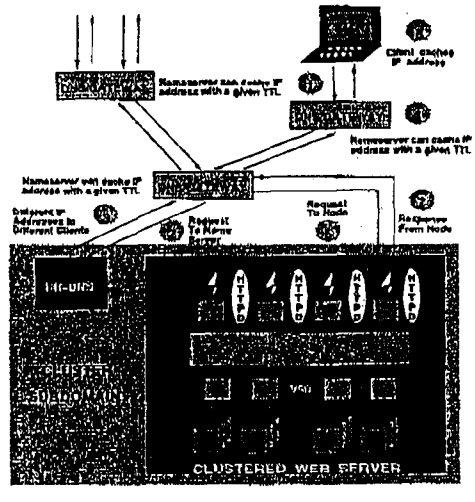


Figure 1: Domain Name Server: Flow

There are several problems that arise with this method. First, as illustrated in Figure 1, there are typically several name servers between clients and the RR-DNS that cache the resolved name-to-IP address

mapping. In order to force a mapping to different server IP addresses, the RR-DNS can specify a *time-to-live* (TTL) for a resolved name, such that requests made after the specified TTL are not resolved in the local nameserver, but are forwarded to the authoritative RR-DNS to be re-mapped to the IP address of a different HTTP server. Multiple name requests made during the TTL period will be mapped to the same HTTP server. Thus, bursts of requests from new clients can appear at the same HTTP server, leading to significant load imbalance. If the TTL is made very small, there is a significant increase in network traffic for name resolution. Therefore, name servers often impose their own minimum TTL, and ignore very small TTLs (e.g., 0) given by the RR-DNS. A second problem is that clients cache the resolved name-to-IP address mapping (henceforth referred to as address caching). Since the clients may make future requests at any time, the load on the HTTP servers cannot be controlled subsequently and will vary due to statistical variations in client access patterns. Further, clients make requests in bursts as each web page typically involves fetching several objects including text and images, and this burst is directed to a single server node, increasing the skew. We show that these effects can lead to significant dynamic load imbalance, requiring that the cluster be operated at lower mean loads in order to be able to handle peak loads. Another issue relates to load balancing is the distributed file server used by the HTTP servers. Typically, there is a skew in the file accesses, with some URLs being more popular than others, leading to a skew in the file server nodes. This becomes more significant when large documents are accessed and file system caching does not suffice.

The remainder of the paper is organized as follows: In Section 2 we present a scheme that combines the RR-DNS with a set of so called *TCP routers* [14] to better balance the load among the HTTP servers. This combination also allows for better scalability. We also describe a method that stripes the files across (data server) nodes in the cluster [5, 4, 16] thereby balancing the data access load. In Section 3, we quantitatively examine load balancing using the RR-DNS approach, and quantify the performance and scalability of the web server that we have prototyped. In Section 4 we describe how high availability is achieved in the prototype. Finally, concluding remarks appear in Section 5.

2 Scalable Web Server Architectures

The scalable web server architecture is illustrated in Figure 2, and consists of multiple web servers running on a clustered architecture. A clustered architecture consists of a group of nodes connected by a fast interconnection network, such as a switch. Each node in the cluster has a local disk array attached to it. The disks of a node can either maintain a local copy of the web documents or a share it among the nodes. For better performance and load balancing, the shared web documents are striped across multiple nodes. As shown in Figure 2, the nodes of a cluster are divided into two logical categories: (i) front-end

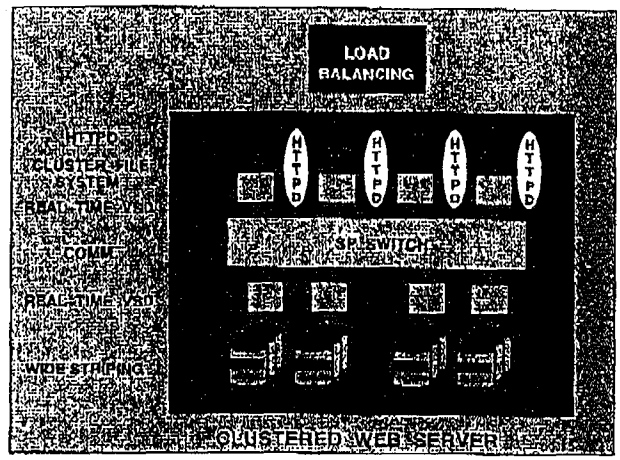


Figure 2: Scalable Web Server Architecture

(delivery) nodes and (ii) back-end (storage) nodes. A load balancing component is used to distribute incoming requests from the external network to the front-end nodes, which also run httpd daemons. The (logical) front-end node then forwards the required commands to the back-end nodes that have the data (document), using a shared file-system. We assume an underlying software layer (e.g., virtual shared disk) which makes the interconnection architecture transparent to the nodes [1]. The results are then sent back from the back-end node to the front-end node through the switch and transmitted to the user.

The clustered architecture outline above can be applied to several applications, and is similar to the design of a scalable video server. Essentially, the same architecture applies to a video server as well, with the httpd daemon replaced by a daemon that reads video objects, and plays them out to clients across the network at the required rate. For both web and video servers, two possible configurations can be used: (i) two-tier and (ii) flat. In the two-tier architecture of Figure 3, the logical front-end and back-end nodes are mapped to different physical nodes of the cluster and are distinct. In a flat architecture, on the other hand, each physical node can serve as both the logical front-end and back-end [16]. All nodes are identical, performing both storage and delivery functions. For a clustered web server we will principally consider the two tier architecture only.

2.1 Load Balancing

A key requirement in order to achieve scalability of the web server is that of balancing the load across the multiple front-end nodes and the back-end nodes.

2.1.1 Front-End

The front-end nodes run the web daemons and are connected to the external network. To balance the load among them, the client requests need to be spread evenly across nodes. Several methods can be used to

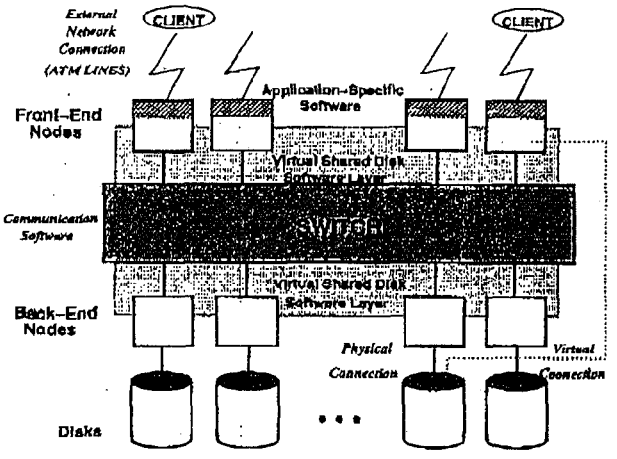


Figure 3: Two-tier Server Architectures

achieve this, with varying degrees of effectiveness. In Section 1, we outlined the round-robin domain name server (RR-DNS) based solution similar to the NCSA approach. To summarize, in this scheme all the front-end nodes are given a single logical name, and the RR-DNS maps the name to multiple IP addresses. However, due to address caching in the client, the bursty nature of client accesses, and TTL values imposed by nameservers, the load balancing achieved is coarse, as quantified in Section 3.

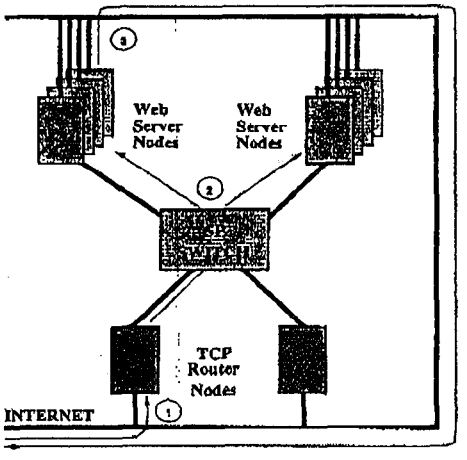


Figure 4: Scalable Web Server (Router): Flow

Another method for achieving load balancing is based on routing, and is illustrated in Figure 4. One or more nodes of the cluster serve as *TCP router(s)*, forwarding client requests to the different front-end nodes in the cluster in a round-robin order. The name and IP address of the router is public, while the addresses of the other nodes in the cluster is private. (If there is more than one router node, a single name is used and RR-DNS is used to map the name to the multiple routers.) This has been referred to as the *encapsulated cluster* in [14]. The client sends requests to the router node which in turn forwards all packets belonging to a particular TCP connection to one of the server front-end nodes. The router can use different algorithms based on load to select which node to route to, or use a simple round-robin scheme. In the prototype (see below), the router resides just above the IP layer as a kernel extension. The server nodes directly send the request back to the client without using the router. However, the server nodes change the source address on the packets sent back to the client to be that of the router node. This makes the entire routing action transparent to the clients. Note that, the response packets are large compared to the request packets, especially for images and video clips, and these bypass the router. Thus, the overhead added by the router is small (and is quantified later).

One advantage of the router scheme over the DNS based solutions is that fine grained load balancing can be achieved and there is no problem of client or name-server caching. One potential issue with both the RR-DNS and routing approaches is that the router node could become a bottleneck, as could the domain name server. These performance aspects are quantified in Section 3.

To further improve scalability, the router and DNS schemes can be combined in various ways. First, a number of router nodes can be used, as shown in Figure 4, and the RR-DNS method can be used to map different clients to different router nodes. This hybrid scheme can tolerate the coarse grained load balancing achieved using RR-DNS because the corresponding router will route any burst of requests, that were mapped by the RR-DNS to the same router, to different front-end server nodes. It achieves good scalability because (i) a long TTL can be used so that the node running the RR-DNS does not become a bottleneck, and (ii) several router nodes can be used achieving scaling beyond that of a single router.

Another hybrid router-DNS scheme combines the logical router and front-end nodes: The DNS based solution is used to get the coarse grained load balancing. Each front-end server node then acts as a router; if the load on the server is low, then the request is handled locally at that server, otherwise the router forwards the request to another node based on its load or a global load balancing algorithm. This scheme eliminates the forwarding by the router unless the load on the front-end node requires it.

2.1.2 Back-End

The back-end nodes function as the server nodes for the shared file-system (e.g., AFS, NFS etc.) that is used by the front-ends to access the data. For instance, the NCSA prototype [8] uses AFS for the data server. With traditional shared file systems, the total document set is partitioned among the different back-end servers. The partitioned approach could lead to load imbalances based on the access skew among the documents. Back-end servers that store the *hot*

documents will get overloaded, resulting in hot spots. The file-system caching at the front-end could alleviate this problem for small documents that fit in the local cache. Only the initial requests need to go to the back-end server and the remaining can be serviced from the local front-end cache. For larger documents, however, the back-end server nodes will be unevenly hit.

To balance the load among the back-end nodes different methods ranging from replication to striping can be used. If the data is replicated across each back-end nodes, the logical front-end and back-end can be mapped to same physical node making it a flat architecture. The scheme described for front-end load balancing can be used directly. Full replication of data is expensive in terms of space utilization. Also, if the data is modified frequently, some form of consistency control is required.

Instead of replication, each document is divided into logical blocks, which are then distributed among the disks in the system. The logical block represents the size of the striping unit, and could be mapped to multiple packets within the switch communication layer and to multiple physical blocks on disk (we use the term block, to refer to a logical block). The blocks belonging to a document may either span all the disks in the system (referred to as *wide striping*) or may be confined to a smaller subset of disks (referred to as *short striping*). The block sizes are typically large (> 64KB) to minimize the access latencies. Successive blocks of an object are allocated to consecutive disks using a sequential placement. The shared file system has additional support to access the striped documents.

Wide striping implicitly achieves higher disk-arm bandwidth and load balancing [5, 16]. However, small documents on the server cannot be striped across all the disks in the system. Such documents are striped only on a subset of the disks in the system and replicated on the remaining disks sets. Unlike short striping, however, in a server that employs wide striping, a single disk or node failure affects all the streams being accessed. Back-end node failures can be masked using the availability schemes described in Section 4.

3 Performance and Scalability

In this section, we study the performance issues relating to the scalability of cluster-based solutions using the RR-DNS and router. The performance of the RR-DNS was measured independently from that of the web server. Web server performance was derived from the Webstone benchmark [18], and the RR-DNS performance was measured via a stand-alone RR-DNS benchmark. Studies of the load imbalances using the RR-DNS and router strategies were performed via CSIM [15] simulations of a networked system.

3.1 Load Imbalance Due to Client and Network Effects

We have qualitatively discussed the factors that affect load balancing across nodes in Section 2. To quantitatively study the effect of dynamic load imbalance we simulate a networked system and vary the number of web page requests per client session, TTL at the nameservers and the number of nodes in the server. The simulation uses a burst model and a single level of caching nameservers between the clients and the server. For each page requested by a client a burst of small requests, representing the objects within a page, are sent to the server. Each client resolves the name of the server only once during a session. The number of remote nameservers is typically much larger than the number of nodes in the server. We illustrate the impact of the TTL with a single level of caching nameservers. The simulation parameters are shown in Table 1.

Parameter	Default
Number of Nodes	16
Number of Nameservers	150
Levels of Caching Nameservers	1
Clients per node	80
Server node capacity (requests are of the same size)	50 req./sec
Mean interarrival time (between page requests per client)	15 secs.
Mean number of bursts per page	10
Mean # of Page Requests/client session	20
TTL	300 secs.

Table 1: Simulation Parameters

Figure 5 shows the effect of varying the number of requests per client session on the dynamic load imbalance at the server for different TTL values at the nameservers. Each request to the server is assumed to be of the same size so that the load imbalance observed is not due to varying request size. The dynamic load imbalance is measured as a ratio of the maximum number (99% percentile) of requests serviced by a node in any sampling interval to the mean computed over the total simulation time period. (Note that, both the RR-DNS and router schemes lead to good average load balance across the nodes over an extended period of time. We focus on the dynamic load balance which can lead to very different peak loads on nodes in the cluster using the two schemes.) The inter-arrival times between page requests per client session is derived from an exponential distribution with a mean of 15 seconds. For a TTL value of 0, as the mean number of page requests per client session increases from 1 to 20, the effect of client address caching becomes more pronounced, resulting in a load imbalance increase from 8% to about 30%. Note, each page requests consist of a burst of requests (mean of 10) to the server. With an increase in the TTL value from 0 to 300 seconds, without client address caching (number of page request per client is one), the load imbalance increases from 8% to about 33%, sharply at first and then gradual. The effect of increasing TTL at the nameservers or that of increasing the duration of a client session is adverse on the load imbalance

at the server nodes. The router approach results in a much lower load imbalance of about 8% caused by the bursty nature of client requests and random arrivals.

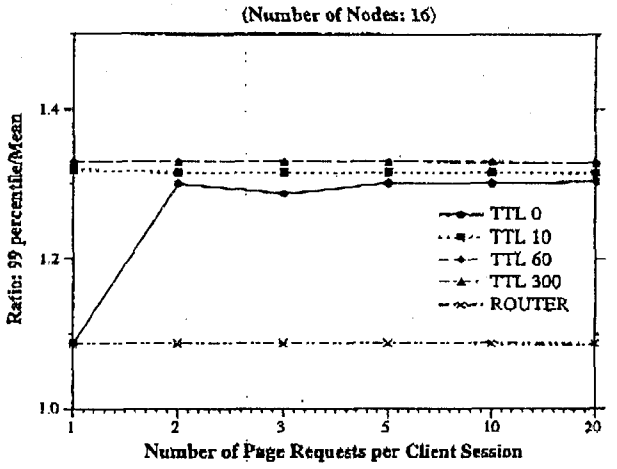


Figure 5: Load Imbalance of RR-DNS

The load imbalance is also a function of the number of nodes in the web server. As the number of nodes increases the load imbalance using the router decreases. Figure 6 shows the effect of varying the number of nodes on the load imbalance using the RR-DNS and router approaches. When the number of nodes is small (e.g., 2) the load imbalance due to the bursty nature of the requests and client random arrivals is high for both the router (22%) and the RR-DNS (30%). As the number of nodes increases, the effect diminishes for the router as it forwards the requests to different nodes. For the RR-DNS approach however, the load imbalance due to client and nameserver address caching contribute largely to the load imbalance and it gradually increases (31%) as the nodes are increased. Thus, for a large scale system a router based approach is recommended as the load imbalance is significantly smaller for the router than the DNS approach. For smaller systems both approaches are viable. Also in the simulation study each client randomly selects a nameserver, whereas in practice some nameservers may be much more utilised than others, causing larger imbalances using the RR-DNS approach.

3.2 Scalability of RR-DNS and Router

Figure 7 shows the scalability achievable via the RR-DNS solutions for different workloads (file sizes) for different ratios of new accesses to total accesses made to the web server. The RR-DNS is accessed only upon requests from new clients, since the address of the web server node is cached at the client after the first access. We varied the percentage of new accesses from 100% (every access is from a new client) to 1% (a client makes a hundred requests to the web site before retiring). Since every document typically consists of several constituent objects, the latter number reflects a few pages accessed per client.

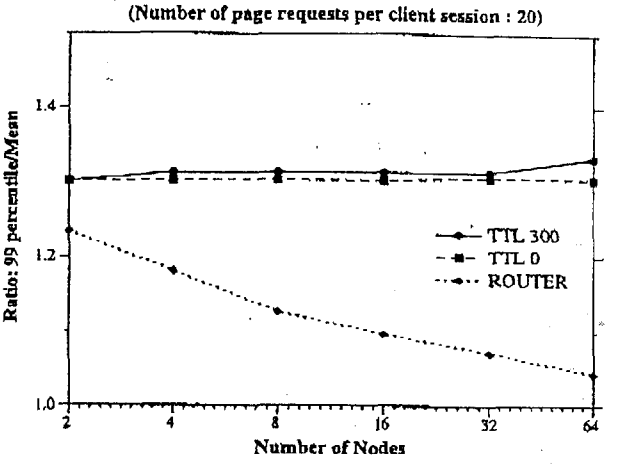


Figure 6: Effect of System Size on Load Imbalance

Since each node can handle fewer requests per second when the documents are large, we see that a single RR-DNS can handle the request to a larger number of server nodes when the average file size is larger. In absolute terms, the RR-DNS scales to a large number of nodes (100-200) even at smaller file sizes, assuming that 10-20% of the RR-DNS accesses are from new clients.

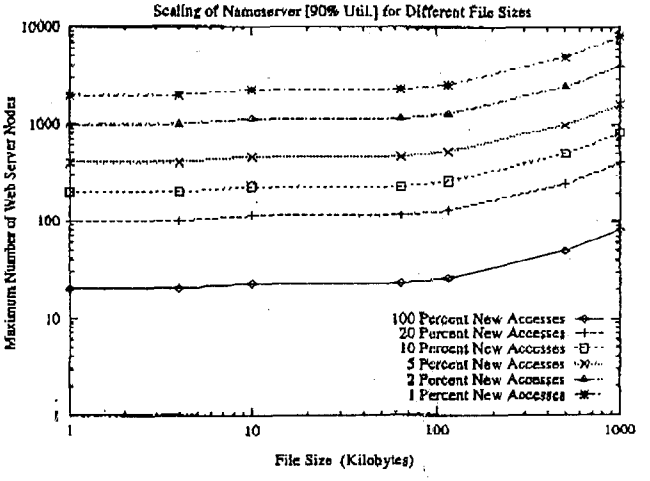


Figure 7: Scalability of RR-DNS

For the router, we measured the CPU utilization during the execution of the Webstone benchmark with four server nodes configured. For a given server node at 90-95% utilization, the CPU utilization of the router node is about 2.2% when the file-size is small (1 KByte) and 1.2% for large files (1 MByte). Thus, based on the average file size of the web server workload, a single router node (operating at 90% utilization) can support between 40 and 75 nodes, more than sufficient for practical systems. Figure 8 shows

the scaling of a single router node for documents with different file sizes. If larger configurations are desired, multiple router nodes can be used, with load balancing across the routers being achieved via RR-DNS. As shown in section 3.1, the router solution has better load balancing properties compared to the RR-DNS solution.

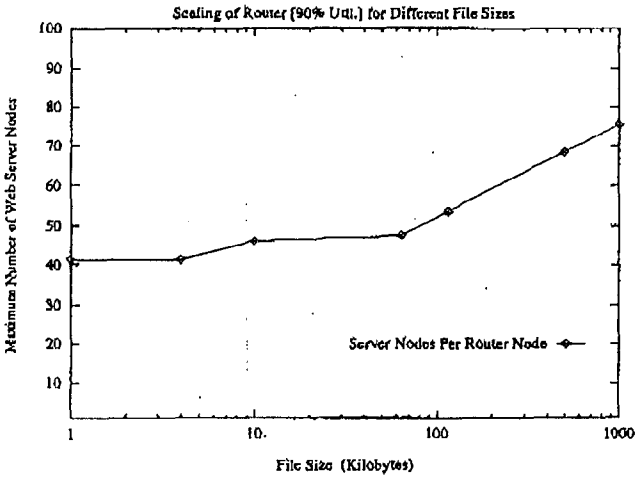


Figure 8: Scalability of a Single Router Node

4 High Availability

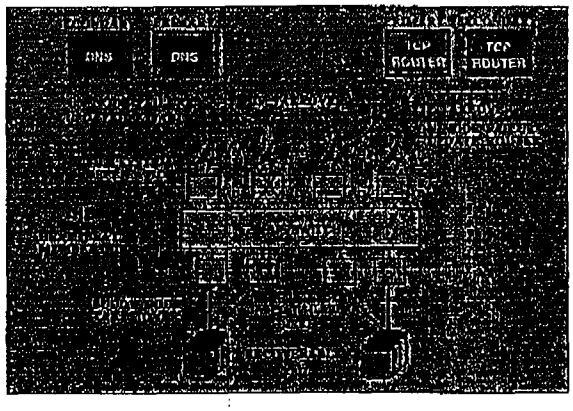


Figure 9: High Availability

As the traffic and commercial applications on popular web servers increase, providing highly available web servers becomes increasingly important. One of the advantages of a clustered system is the possibility of providing high availability by masking node and other failures. This is done by detecting failures and reconfiguring the system such that the workload is taken over by the remaining nodes in the cluster. For the scalable web server, in order to guarantee high availability, we need to mask both front-end and back-end node failure. The method we use for making the web server highly available is illustrated in Figure 9. Node failures are detected by using a heart-beat and membership protocol [7], that essentially eliminates nodes from the membership group in a consistent manner when (several) heartbeat messages from the node are not received. Network failures are detected by using heartbeats on multiple networks. Other resource failures are detected by monitors. On detection of node or other resource failures, a distributed recovery driver component [11] initiates recovery by coordinating a series of recovery commands or scripts across the remaining nodes in the system. The actions taken for specific failures is outlined below.

For front-end node failure, if the DNS or router solution is used for load balancing, the failed node is removed from either the name server tables or the router configuration table. Once these tables are updated to reflect the failure, all remote clients that try to make a connection after the failure will not be directed to the failed node. However, in the DNS approach some clients may have cached the address of the failed node and will continue to try connecting to it. Similarly, with the router based approach a router node may fail causing clients to see a failure. To handle these scenarios each front-end node and router node has a "buddy" which takes over its IP address after failure. Remote clients that have cached the address of the failed node will connect to the buddy transparently. For IP takeover the nodes have a redundant adapter. One of the adapters is used for servicing all the external IP requests and one for booting. The IP address of the main adapter is taken over by a spare adapter in the buddy node. The switching is done in software and is orchestrated by the recovery driver, as outlined earlier.

To handle the back-end server failure we use a hardware approach, namely twin-tailed disks. Other approaches like replication and software-RAID can be used for large scale servers with real-time requirements [17]. The disks attached to a node are twin-tailed to a buddy node. When a node fails the buddy node takes over the disk. The front-end nodes forward the requests to the buddy. Twin-tailing requires either special disks or extra SCSI adapters on the nodes that connect to the shareable disks. With twin-tailing the load on the buddy can double in the worst case. Multi-tailing can reduce the overload to some extent.

5 Conclusions and Future Work

Based on the analysis and design described above we have built a prototype scalable and highly available web server on an IBM SP2 system, illustrated in Figure 10. This system is essentially an instantiation of the architecture shown in Figure 2, and is a two-tier system with four front-end and four back-end nodes. The disks are connected to the back-end nodes. The disks are (logically) shared among the front-end nodes using the Virtual Shared Disk software [1]. The data for web files are striped across the back-end nodes and disks, thereby achieving load balancing across the back-end nodes based on work by [4]. The combination of the TCP router and RR-DNS described earlier is used for load balancing across HTTP daemons run-

ning all the front-end nodes. The TCP router is based on the encapsulated cluster prototype [14]. The scalability data presented in Section 3 is based on measurements on the prototype. The methods for achieving high availability outlined in the previous section have been implemented in the prototype.

The hardware and software infrastructure is shared between the scalable web server and a prototype scalable video server: the video objects are striped across the same nodes and disks, the HTTP daemons are replaced by video push daemons, and load balancing is achieved by a control server [9] that assigns the client streams to front-end nodes. We present forms with video selections on the web, and allow the web client to start up (out of band) play-out of video objects over separate high bandwidth channels. Currently, the constrained bandwidth to the client and also in the network prevent any significant real-time video access on the web. Over time, we believe that real-time video over the web will become feasible; the prototype's ability to support scalability for both web and video will then become increasingly important.

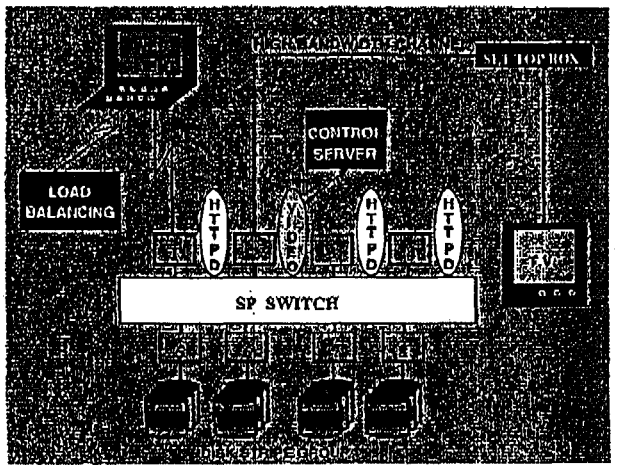


Figure 10: SP Video and Web Server Prototype

We quantified the scalability of this solution, and examined the performance of the (known) RR-DNS approach. We showed through simulations that the pure DNS approach can be significantly impacted by dynamic load imbalance among the nodes in the cluster. As a specific example, due to client caching of the mapping from the name to IP address, the peak load on nodes in the cluster can be 30% to 40% higher than the mean load on the nodes, because bursts of requests from clients lead to dynamic load imbalance. With higher values of the TTL at nameservers, the load dynamic load imbalance with RR-DNS gets even worse. This means that, using RR-DNS, the cluster nodes would need to be configured such that the mean load is less than 60% in order to be able to sustain the peak loads that occur. We showed that the router solution largely eliminates this problem, and provides excellent dynamic load balancing; this allows the cluster nodes to operate at much higher mean loads, and still sustain the peak loads.

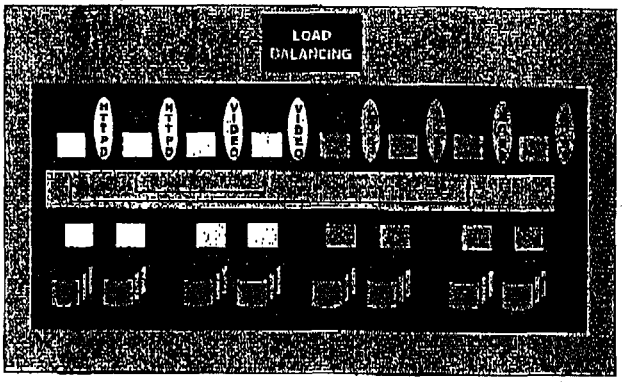


Figure 11: Integrated Scalable Solutions

In this paper we have focused on a scalable and highly available web server, with web pages stored on a shared file system. For many applications, access to back-end servers such as databases and on-line transaction processing via the web is required. The scalable web server described above can be integrated with parallel/scalable back-end servers, to create integrated scalable solutions, as illustrated in Figure 11. The figure illustrates the scalable web-server integrated in our prototype with a scalable video server and a parallel database, all in one management and high-availability domain.

Acknowledgments: The prototype draws on components developed by many people, especially C. R. Attanasio, S.C. Smith, C. Polyzois, R. Haskin, D. McNabb, F. Schmuck, J. Wyllie, M. Devarakonda and S. Fakhouri.

References

[1] C.R. Attanasio, M. Butrico, C.A. Polyzois, S.E. Smith, and J.L. Peterson. Design and Implementation of a Recoverable Virtual Shared Disk. IBM Research Report RC 19843, T.J Watson Research Center, Yorktown Heights, New York, 1994.

[2] T. Brisco. DNS Support for Load Balancing. RFC 1794, Rutgers University, April 1995.

[3] Berners-Lee T. et al. The World-Wide Web. *Communications of the ACM*, 37(8):76 – 82, August 1994.

[4] Roger Haskin. Personal Communication, 1994.

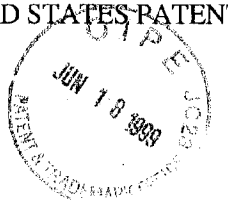
[5] Roger Haskin and Frank D. Stein. A System for Delivery of Interactive Television Programming. In *COMPCON*. IEEE, March 1995.

[6] J.H. Howard, M.L. Kazar, S.G. Menees, D.A. Nichols, M. Satyanarayanan, R.N. Sidebotham, and M.J. West. Scale and Performance in a

#64

Atty Docket No. 02577.P001D PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE



In re Application of:

Keith Lowery et al.

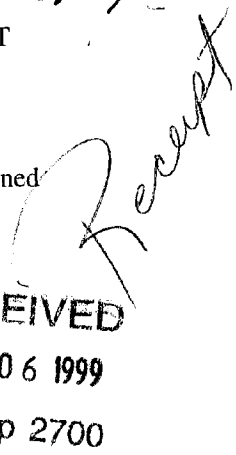
Application No.: 09/234,048

Filed: January 19, 1999

For: A METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE

Examiner: Not Yet Assigned

Art Unit: 2755



Assistant Commissioner for Patents
Washington, D.C. 20231

REQUEST FOR CORRECTION OF FILING RECEIPT

Dear Sir:

In the Filing Receipt for the above-referenced patent application filed January 19, 1999, the continuing data was not recorded.

The continuing data as claimed by applicant is as follows:

THIS APPLN IS A DIV OF 08/636,477 4/23/96

Please have the Filing Receipt changed to reflect the continuing data. Enclosed is a copy of the incorrect Filing Receipt with the change noted thereon.

In support of this change, enclosed please find a copy of the Patent Application Transmittal that was submitted with the patent application. This Patent Application Transmittal is included herewith as Exhibit A. Please note that elements 5 and 17 are marked. Please have the Filing Receipt changed to reflect the continuing data.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Dated: 6/15 1999

James H. Salter
Reg. No. 35,668

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408)720-8598

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C 20231

on June 15, 1999
Date of Deposit

Claire Walters
Name of Person Mailing Correspondence

Claire Walters 6/15/99
Signature Date

PTO-103X
Rev. 8-95)

FILING RECEIPT

ENTERED
FEB 18 1999
STATUS DB-LA

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY AND COMMISSIONER
OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NUMBER	FILING DATE	GRP ART UNIT	FIL FEE REC'D	ATTORNEY DOCKET NO.	DRWGS	TOT CL	IND CL
09/234,048	01/19/99	2755	$994.00	02577.P001D	5	16	6

JUN 18 1999

JAMES H SALTER
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
7TH FLOOR
LOS ANGELES CA 90025

RECEIVED
FEB 16 1999
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
LOS ANGELES

hfo Spinner
FHS

Receipt is acknowledged of this nonprovisional Patent Application. It will be considered in its order and you will be notified as to the results of the examination. Be sure to provide the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. If an error is noted on this Filing Receipt, please write to the Application Processing Division's Customer Correction Branch within 10 days of receipt. Please provide a copy of the Filing Receipt with the changes noted thereon.

Applicant(s)

KEITH LOWERY, RICHARDSON, TX; ANDREW B. LEVINE, PLANO, TX; RONALD L. HOWELL, ROWLETT, TX.

THIS APPLN IS A DIV OF 08/636,477 4/23/96

RECEIVED
JUL 06 1999
Group 2700

FOREIGN FILING LICENSE GRANTED 02/05/99
TITLE
METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE

PRELIMINARY CLASS: 395

ENTERED JHS DB

DATA ENTRY BY: MARTIN, DIANE TEAM: 04 DATE: 02/05/99

(see reverse)

EXHIBIT A

Please type a plus sign (+) inside this box [+]

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b)

Attorney Docket No. 02577.P001D Total Pages 4

First Named Inventor or Application Identifier Keith Lowery et al.

Express Mail Label No. EL164804887US

ADDRESS TO: **Assistant Commissioner for Patents**
Box Patent Application
Washington, D. C. 20231

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

1. X Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)

2. X Specification (Total Pages 26)
(preferred arrangement set forth below)
- Descriptive Title of the Invention
- Cross References to Related Applications
- Statement Regarding Fed sponsored R & D
- Reference to Microfiche Appendix
- Background of the Invention
- Brief Summary of the Invention
- Brief Description of the Drawings (if filed)
- Detailed Description
- Claims
- Abstract of the Disclosure

3. X Drawings(s) (35 USC 113) (Total Sheets 5)

4. X Oath or Declaration (Total Pages 4)

a. ___ Newly Executed (Original or Copy)

b. X Copy from a Prior Application (37 CFR 1.63(d))
(for Continuation/Divisional with Box 17 completed) **(Note Box 5 below)**

i. ___ DELETIONS OF INVENTOR(S) Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).

5. X Incorporation By Reference (useable if Box 4b is checked)
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

6. _ Microfiche Computer Program (Appendix)

7. _ Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
a. ___ Computer Readable Copy
b. ___ Paper Copy (identical to computer copy)
c. ___ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

8. _____ Assignment Papers (cover sheet & documents(s))
9. _____ a. 37 CFR 3.73(b) Statement (where there is an assignee)
_____ b. Power of Attorney
10. _____ English Translation Document (if applicable)
11. X a. Information Disclosure Statement (IDS)/PTO-1449
X b. Copies of IDS Citations
12. _____ Preliminary Amendment
13. X Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
14. _____ a. Small Entity Statement(s)
b. Statement filed in prior application, Status still proper and desired
15. _____ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. _____ Other: _____

17. **If a CONTINUING APPLICATION**, check appropriate box and supply the requisite information:
___ Continuation X Divisional ___ Continuation-in-part (CIP)
of prior application No: 08/636,477

18. **Correspondence Address**
_____ Customer Number or Bar Code Label _____ (Insert Customer No. or Attach Bar Code Label here)
or
X Correspondence Address Below

NAME James H. Salter, Reg. No. 35,668
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
ADDRESS 12400 Wilshire Boulevard
Seventh Floor
CITY Los Angeles STATE California ZIP CODE 90025-1026
Country U.S.A. TELEPHONE (408) 720-8598 FAX (408) 720-9397

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

OIPE JC62 JUL 12 1999 PATENT & TRADEMARK OFFICE

02577.P001D

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:

Keith Lowery, et al.

Serial No.: 09/234,048

Filed: January 19, 1999

For: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS (AS AMENDED)

A Rule 1.53(b) Divisional Application of:
U.S. Serial No. 08/636,477
Filed April 23, 1996

Examiner: Unassigned

Art Unit: 2755

RECEIVED
JUL 16 1999
TC 2700 MAIL ROOM

Assistant Commissioner for Patents
Washington, D.C. 20231

PRELIMINARY AMENDMENT

Sir:

Applicants respectfully request that this Preliminary Amendment be entered prior to examination of the above-identified patent application.

IN THE TITLE

Please amend the title to read --SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS--.

IN THE CLAIMS

Please cancel claims 1-16 and enter new claims 17-45 as follows.

17. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing a request from a Web server to a page server, said page server receiving said request and releasing said Web server to process other requests wherein said routing step further includes the steps of:

intercepting said request at said Web server and routing said request to said page server;

processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

18. The computer-implemented method in Claim 17 wherein said step of routing said request includes the steps of:

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

19. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

20. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

21. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

22. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of logging into said one or more data sources.

23. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

24. The computer-implemented method in Claim 17 wherein said page server includes tag-based text templates for configuring said Web page.

25. The computer-implemented method in Claim 24 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

26. The computer-implemented method in Claim 24 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one

or more data sources in response to said request.

27. The computer-implemented method in Claim 24 wherein said tag-based text templates include HTML templates.

28. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

29. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of processing an object handling extension.

30. The computer-implemented method in Claim 29 wherein said object handling extension is an OLE extension.

31. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a page server, said page server receiving said request and releasing said HTTP-compliant device to process other requests wherein said transferring step further includes the steps of:

intercepting said request at said HTTP-compliant device and transferring said request to said page server;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

16 15
32. The computer-implemented method in Claim 31 wherein said step of transferring said request includes the steps of:

transferring said request from said HTTP-compliant device to a dispatcher; and

dispatching said request to said page server.

17 15
33. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

18 15
34. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of dynamically retrieving said data from said one or more data sources.

19 15
35. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

20 15
36. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of logging into said one or more data sources.

21 15
37. The computer-implemented method in Claim 31 wherein said step of

dynamically generating said page includes the step of maintaining a page cache containing said page.

38. The computer-implemented method in Claim 31 wherein said page server includes tag-based text templates for configuring said page.

39. The computer-implemented method in Claim 38 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

40. The computer-implemented method in Claim 38 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

41. The computer-implemented method in Claim 38 wherein said tag-based text templates include HTML templates.

42. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

43. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of processing an object handling extension.

28 44. The computer-implemented method in Claim 43 wherein said object handling extension is an OLE extension.

29 45. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a dispatcher;

maintaining dynamic information regarding data sources a given page server may access;

dispatching said request to an appropriate page server based on said request and based on said dynamic information, said page server receiving said request and releasing said HTTP-compliant device to process other requests;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

REMARKS

Consideration of this application in view of the foregoing amendments is respectfully requested. Claims 1-16 have been cancelled and new claims 17-45 have been added to more distinctly claim the present invention. Claims 17-45 are presented for examination. Applicants respectfully submit that the claims are in condition for allowance.

An Information Disclosure Statement accompanies this Preliminary Amendment. Applicant respectfully submits that none of the references cited in the Information Disclosure Statement anticipate or render obvious the claims of the present application.

If there are any additional charges, please charge Deposit Account No. 02-2666. If a telephone interview would in any way expedite the prosecution of this application, the Examiner is invited to contact James H. Salter at (408) 720-8598.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 7/9, 1999

James H. Salter (Reg. No. 35,668)

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA. 90025-1026
(408) 720-8598

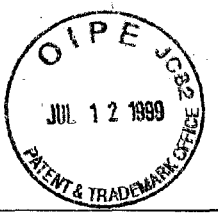
FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters — 7/9/99
Signature — Date

#6 GAU 2755



PTO/SB/17(10/96)
Approved for use through 09/30/98. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL

TOTAL AMOUNT OF PAYMENT ($) 162.00

Complete if Known:

Application No.	09/234,048
Filing Date	January 19, 1999
First Named Inventor	Keith Lowery, et al.
Group Art Unit	2755
Examiner Name	Unassigned
Attorney Docket No.	02577.P001D

METHOD OF PAYMENT (check one)

1. [ ] **The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:**

   Deposit Account Number _____
   Deposit Account Name _____

   [X] **Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17 and credit any over payments to Deposit Account Number 02-2666**

   [ ] **Charge the Issue Fee Set in 37 CFR 1.18 at the Mailing of the Notice of Allowance, 37 CFR 1.131(b)**

2. X **Payment Enclosed**
   X **Check**
   _____ **Money Order**
   _____ **Other**

RECEIVED
JUL 16 1999
TC 2700 MAIL ROOM

FEE CALCULATION (fees effective 10/01/97)

1. **FILING FEE**

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	760	201	380	Utility application filing fee	
106	310	206	155	Design application filing fee	
107	480	207	240	Plant filing fee	
108	760	208	380	Reissue filing fee	
114	150	214	75	Provisional application filing fee	

SUBTOTAL (1) $ 0

2. **CLAIMS**

			Extra		Fee from below		Fee Paid
Total Claims	29	- 20 =	9	X	18	=	162
Independent Claims	3	- 3 =	0	X	0	=	0
Multiple Dependent Claims			0	X	0	=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
103	18	203	9	Claims in excess of twenty	162.00
102	78	202	39	Independent claims in excess of 3	
104	260	204	130	Multiple dependent claim	
109	78	209	39	Reissue independent claims over original patent	
110	18	210	9	Reissue claims in excess of 20 and over original patent	

SUBTOTAL (2) $ 162.00

A

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for response within first month	
116	380	216	190	Extension for response within second month	
117	870	217	435	Extension for response within third month	
118	1,360	218	680	Extension for response within fourth month	
128	1,850	228	925	Extension for response within fifth month	
119	300	219	150	Notice of Appeal	
120	300	220	150	Filing a brief in support of an appeal	
121	260	221	130	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive unavoidably abandoned application	
141	1,210	241	605	Petition to revive unintentionally abandoned application	
142	1,210	242	605	Utility issue fee (or reissue)	
143	430	243	215	Design issue fee	
144	580	244	290	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Petitions related to provisional applications	
126	240	126	240	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	760	246	380	For filing a submission after final rejection	
149	760	249	380	For each additional invention to be examined	
				Other: _____	

SUBTOTAL (3) $ 0

*Reduced by Basic Filing Fee Paid

SUBMITTED BY:

Typed or Printed Name: James H. Salter

Signature _____ Date 7/9/99

Reg. Number 35,668 Deposit Account User ID _____ (complete if applicable)

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters 7/9/99
Signature Date

A

OIPE JC82 JUL 12 1999 PATENT & TRADEMARK OFFICE

PTO/SB/17(10/96)
Approved for use through 09/30/98. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

RECEIVED JUL 16 1999 TC 2700 MAIL ROOM

FEE TRANSMITTAL

TOTAL AMOUNT OF PAYMENT ($) 162.00

Complete if Known:

Application No.	09/234,048
Filing Date	January 19, 1999
First Named Inventor	Keith Lowery, et al.
Group Art Unit	2755
Examiner Name	Unassigned
Attorney Docket No.	02577.P001D

METHOD OF PAYMENT (check one)

1. [ ] **The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:**

   Deposit Account Number _____
   Deposit Account Name _____

   [X] **Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17 and credit any over payments to Deposit Account Number 02-2666**

   [ ] **Charge the Issue Fee Set in 37 CFR 1.18 at the Mailing of the Notice of Allowance, 37 CFR 1.131(b)**

2. X **Payment Enclosed**
   X **Check**
   _____ **Money Order**
   _____ **Other**

FEE CALCULATION (fees effective 10/01/97)

1. **FILING FEE**

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	760	201	380	Utility application filing fee	
106	310	206	155	Design application filing fee	
107	480	207	240	Plant filing fee	
108	760	208	380	Reissue filing fee	
114	150	214	75	Provisional application filing fee	

SUBTOTAL (1) $ 0

2. **CLAIMS**

			Extra		Fee from below		Fee Paid
Total Claims	29	- 20 =	9	X	18	=	162
Independent Claims	3	- 3 =	0	X	0	=	0
Multiple Dependent Claims			0	X	0	=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
103	18	203	9	Claims in excess of twenty	162.00
102	78	202	39	Independent claims in excess of 3	
104	260	204	130	Multiple dependent claim	
109	78	209	39	Reissue independent claims over original patent	
110	18	210	9	Reissue claims in excess of 20 and over original patent	

SUBTOTAL (2) $ 162.00

A

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for response within first month	
116	380	216	190	Extension for response within second month	
117	870	217	435	Extension for response within third month	
118	1,360	218	680	Extension for response within fourth month	
128	1,850	228	925	Extension for response within fifth month	
119	300	219	150	Notice of Appeal	
120	300	220	150	Filing a brief in support of an appeal	
121	260	221	130	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive unavoidably abandoned application	
141	1,210	241	605	Petition to revive unintentionally abandoned application	
142	1,210	242	605	Utility issue fee (or reissue)	
143	430	243	215	Design issue fee	
144	580	244	290	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Petitions related to provisional applications	
126	240	126	240	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	760	246	380	For filing a submission after final rejection	
149	760	249	380	For each additional invention to be examined	
		Other:			

SUBTOTAL (3) $ 0

*Reduced by Basic Filing Fee Paid

SUBMITTED BY:

Typed or Printed Name: James H. Salter

Signature _____ Date 7/9/99

Reg. Number 35,668 Deposit Account User ID _____ (complete if applicable)

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters 7/9/99
Signature Date

02577.P001D

PATENT

OIPE JC82 JUL 12 1999 PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:

Keith Lowery, et al.

Examiner: Unassigned

Serial No.: 09/234,048

Art Unit: 2755

Filed: January 19, 1999

For: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS (AS AMENDED)

A Rule 1.53(b) Divisional Application of:
U.S. Serial No. 08/636,477
Filed April 23, 1996

Assistant Commissioner for Patents
Washington, D.C. 20231

SIR: Transmitted herewith is a Preliminary Amendment for the above application.

_____ Small entity status of this application under 37 C.F.R. §§ 1.9 and 1.27 has been established by
a verified statement previously submitted.

_____ A verified statement to establish small entity status under 37 C.F.R. §§ 1.9 and 1.27 is enclosed.

_____ No additional fee is required.

The fee has been calculated as shown below:

	(Col. 1) Claims Remaining After Amd.		(Col. 2) Highest No. Previously Paid For	(Col. 3) Present Extra	SMALL ENTITY Rate	SMALL ENTITY Additional Fee	OTHER THAN A SMALL ENTITY Rate	OTHER THAN A SMALL ENTITY Additional Fee
Total Claims	* 29	Minus	** 20	9	x9	$ 0	x18	$ 162
Indep. Claims	* 3	Minus	*** 6	0	x39	$ 0	x78	$ 0
_____ **First Presentation of Multiple Dependent Claim(s)**					**+130**	**$ 0**	**+260**	**$ 0**
					Total Add. Fee	$ 0	Total Add. Fee	$ 162

* If the entry in Col. 1 is less than the entry In Col. 2, write "0" in Col. 3.

** If the "Highest No. Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.

*** If the "Highest No. Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space. The "Highest No. Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

A

02577.P001D *PATENT*

__X__ A check in the amount of $__162.00__ is attached for presentation of additional claim(s).

_____ Applicant(s) hereby Petition(s) for an Extension of Time of ___( ) month(s) pursuant to 37 C.F.R. § 1.136(a).

_____ A check for $_____ is attached for processing fees under 37 C.F.R. § 1.17.

_____ Please charge my Deposit Account No. 02-2666 the amount of $_____.
A duplicate copy of this sheet is enclosed.

__X__ The Commissioner of Patents and Trademarks is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account No. 02-2666 **(a duplicate copy of this sheet is enclosed)**:

__X__ Any additional filing fees required under 37 C.F.R. § 1.16 for presentation of extra claims.

__X__ Any extension or petition fees under 37 C.F.R. § 1.17.

Respectfully submitted,
BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP

Date: 7/9, 1999

James H. Salter (Reg. No. 35,668)

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408) 720-8598

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters
Signature

7/9/99
Date

A

OIPE JC82 JUL 12 1999 PATENT & TRADEMARK OFFICE

02577.P001D

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:

Keith Lowery, et al.

Examiner: Unassigned

Serial No.: 09/234,048

Art Unit: 2755

Filed: January 19, 1999

For: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS (AS AMENDED)

A Rule 1.53(b) Divisional Application of:
U.S. Serial No. 08/636,477
Filed April 23, 1996

RECEIVED JUL 16 1999 TC 2700 MAIL ROOM

Assistant Commissioner for Patents
Washington, D.C. 20231

SIR: Transmitted herewith is a Preliminary Amendment for the above application.

_____ Small entity status of this application under 37 C.F.R. §§ 1.9 and 1.27 has been established by
a verified statement previously submitted.

_____ A verified statement to establish small entity status under 37 C.F.R. §§ 1.9 and 1.27 is enclosed.

_____ No additional fee is required.

The fee has been calculated as shown below:

	(Col. 1) Claims Remaining After Amd.		(Col. 2) Highest No. Previously Paid For	(Col. 3) Present Extra
Total Claims	* 29	Minus	** 20	9
Indep. Claims	* 3	Minus	*** 6	0
	First Presentation of Multiple Dependent Claim(s)			

SMALL ENTITY Rate	Additional Fee	OTHER THAN A SMALL ENTITY Rate	Additional Fee
x9	$ 0	x18	$ 162
x39	$ 0	x78	$ 0
+130	**$ 0**	**+260**	**$ 0**
Total Add. Fee	$ 0	Total Add. Fee	$ 162

* If the entry in Col. 1 is less than the entry In Col. 2, write "0" in Col. 3.

** If the "Highest No. Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.

*** If the "Highest No. Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space. The "Highest No. Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

__X__ A check in the amount of $ 162.00 is attached for presentation of additional claim(s).

_____ Applicant(s) hereby Petition(s) for an Extension of Time of ____( ) month(s) pursuant to 37 C.F.R. § 1.136(a).

_____ A check for $_____ is attached for processing fees under 37 C.F.R. § 1.17.

_____ Please charge my Deposit Account No. 02-2666 the amount of $_____.
A duplicate copy of this sheet is enclosed.

__X__ The Commissioner of Patents and Trademarks is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account No. 02-2666 **(a duplicate copy of this sheet is enclosed)**:

__X__ Any additional filing fees required under 37 C.F.R. § 1.16 for presentation of extra claims.

__X__ Any extension or petition fees under 37 C.F.R. § 1.17.

Respectfully submitted,
BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP

Date: 7/9, 1999

James H. Salter (Reg. No. 35,668)

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408) 720-8598

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters — 7/9/99
Signature — Date



D. Johnson
#6 7-28-99
IDS w/ Refs

Attorney's Docket No. 02577.P001D

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Patent Application of:
Keith Lowery et al.

Application No.: 09/234,048

Filed: January 19, 1999

For: System and Method for Managing Dynamic Web Page Generation Requests (As Amended)

Examiner: Unassigned

Art Unit: 2755

RECEIVED
JUL 16 1999
TC 2700 MAIL ROOM

Assistant Commissioner for Patents
Washington, D.C. 20231

INFORMATION DISCLOSURE STATEMENT

Sir:

Enclosed is a copy of Information Disclosure Citation Form PTO-1449 together with copies of the documents cited on that form. It is respectfully requested that the cited documents be considered and that the enclosed copy of Information Disclosure Citation Form PTO-1449 be initialed by the Examiner to indicate such consideration and a copy thereof returned to applicant(s).

Pursuant to 37 C.F.R. § 1.97, the submission of this Information Disclosure Statement is not to be construed as a representation that a search has been made and is not to be construed as an admission that the information cited in this statement is material to patentability.

Pursuant to 37 C.F.R. § 1.97, this Information Disclosure Statement is being submitted under one of the following (as indicated by an "X" to the left of the appropriate paragraph):

XX 37 C.F.R. §1.97(b).

____ 37 C.F.R. §1.97(c). If so, then enclosed with this Information Disclosure Statement is one of the following:

____ A statement pursuant to 37 C.F.R. §1.97(e) or

____ A check for $240.00 for the fee under 37 C.F.R. § 1.17(p).

LJV/cak (08/28/98)

_____ 37 C.F.R. §1.97(d). If so, then enclosed with this Information Disclosure Statement are the following:

(1) A statement pursuant to 37 C.F.R. §1.97(e);

(2) A petition requesting consideration of the Information Disclosure Statement; and

(3) A check for $_____ for the fee under 37 C.F.R. §1.17(i) for submission of the Information Disclosure Statement.

If there are any additional charges, please charge Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 7/9, 1999

James H. Salter
Reg. No. 35,668

12400 Wilshire Blvd.
Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8598

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231 on July 9, 1999
(Date of Deposit)

Claire Wallters
(Typed or printed name of person mailing correspondence)

Claire Wallters
(Signature of person mailing correspondence)

LJV/cak (08/28/98)

Please type a plus sign (+) inside this box →

PTO/SB/08A (10-96)
Approved for use through 10/31/99. OMB 0651-0031
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

Substitute for form **1449A/PTO**
(Modified by BSTZ 6/30/99)

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(use as many sheets as necessary)

PATENT & TRADEMARK OFFICE JUL 12 1999

Sheet 1 of 1

Complete if Known	
Application Number	09/234,048
Filing Date	1/19/99
First Named Inventor	Keith Lowery
Group Art Unit	2755
Examiner Name	Unassigned
Attorney Docket Number	02577.P001D

U.S. PATENT DOCUMENTS

Examiner Initials *		U.S.Patent Document Number	Name of Patentee or Applicant of Cited Document	Date of Publication of Cited Document MM-DD-YYYY	Filing Date if Appropriate

RECEIVED JUL 15 1999 TC 2700 MAIL ROOM

FOREIGN PATENT DOCUMENTS

Examiner Initials *		Foreign Patent Document: Office or Country	Number	Date	Name of Patentee or Applicant of Cited Document	Date of Publication of Cited Document MM-DD-YYYY	Translation? Yes/No

OTHER DOCUMENTS

Examiner Initials *		Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published (if known).	Translation? Yes/No
		ANDRESEN, DANIEL, ET AL.; Scalability Issues for High Performance Digital Libraries on the World Wide Web; Department of Computer Science; University of California at Santa Barbara; 10 pages.	
		ANDRESEN, DANIEL, ET AL.; SWEB: Towards a Scalable World Wide Web Server on Multicomputers; Department of Computer Science; University of California at Santa Barbara; 7 pages.	
		HOLMEDAHL, VEGARD; ET AL.; Cooperative Caching of Dynamic Content on a Distributed Web Server; Department of Computer Science; University of California at Santa Barbara; 8 pages.	
		OVERSON, NICOLE; NeXT Ships WebObjects--On Time--As Promised; Deja.com: NeXT Ships WebObjects--On Time--As Promishttp://X28..deja.com/=dnc/[ST_m=ps...EXT=927585438.1744765032&hitnum=33.	

Examiner Signature		Date Considered	2/17/01

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

Scalability Issues for High Performance Digital Libraries on the World Wide Web

Daniel Andresen, Tao Yang, Omer Egecioglu, Oscar H. Ibarra, and Terence R. Smith
Department of Computer Science
University of California
Santa Barbara, CA 93106
{dandrese, tyang, omer, ibarra, smithtr}@cs.ucsb.edu

OIPE JC82 JUL 12 1999 PATENT & TRADEMARK OFFICE

Abstract

We investigate scalability issues involved in developing high performance digital library systems. Our observations and solutions are based on our experience with the Alexandria Digital Library (ADL) testbed under development at UCSB. The current ADL system provides on-line browsing and processing of digitized maps and other geo-spatially mapped data via the World Wide Web (WWW). A primary activity of the ADL system involves computation and disk I/O for accessing compressed multi-resolution images with hierarchical data structures, as well as other duties such as supporting database queries and on-the-fly HTML page generation. Providing multi-resolution image browsing services can reduce network traffic but impose some additional cost at the server. We discuss the necessity of having a multi-processor DL server to match potentially huge demands in simultaneous access requests from the Internet. We have developed a distributed scheduling system for processing DL requests, which actively monitors the usages of CPU, I/O channels and the interconnection network to effectively distribute work across processing units to exploit task and I/O parallelism. We present an experimental study on the performance of our scheme in addressing the scalability issues arising in ADL wavelet processing and file retrieval. Our results indicate that the system delivers good performance on these types of tasks.

1. Introduction

The number of digital library (DL) projects is increasing rapidly at both the national and the international levels (see, for example, [6, 2]). Many of the current projects are moving rapidly towards their goals of supporting on-line retrieval and processing of major collections of digitized documents over the Internet.

Performance and scalability issues are especially important for the Alexandria Digital Library (ADL) project [2, 14]. The fundamental goal of this project is to provide users with the ability to access and process broad classes of spatially-referenced materials from the Internet. Materials that are currently in the collections of ADL and accessible through the ADL World Wide Web (WWW) server include *geographically*-referenced items such as digitized maps, satellite images, digitized aerial photographs, and associated metadata. When fully developed, ADL will comprise a set of nodes distributed over the Internet supporting such library components as collections, catalogs, interfaces, and ingest facilities. ADL is currently building collections that will involve millions of items requiring terabyte levels of storage. Many collection items have sizes in the gigabyte range while others require extensive processing to be of value in certain applications. The catalog component alone contains a metadatabase of significant size.

Before such goals can be achieved, however, major issues of performance and scalability must be resolved, particularly for DLs supporting extensive collections or collections with large data items. Critical performance bottlenecks that must be overcome to assure adequate access over the Internet involve server processing capability and network bandwidth. While we expect network communication technology to improve steadily, particularly with the advent of ATM and B-ISDN, we still need to consider the minimization of network traffic in the design of the current system. Additionally, the server performance must scale to match expected demands.

A strategy used in the ADL system for reducing network traffic is to provide the service of progressive image browsing and the subregion retrieval, to avoid unnecessary large image transmission. The trade-off, however, is that more processing is required at the server site. Considering that popular WWW sites such as Alta Vista, Lycos and Yahoo have been receiving over two million accesses per day (or 20-30 requests per second), and the ADL server involves much more intensive I/O and heterogeneous CPU activities, a multi-processor server becomes indispensable [2].

In this paper, we investigate the network bandwidth requirement in the ADL system using progressive image browsing retrieval and the computational and I/O demands for supporting such activities. We study the use of networks of existing, inexpensive workstations and disks to augment the processing and storage capabilities of DL servers. In particular, we have investigated to what extent recycling the idle cycles of processing units in networks of workstations, as well as retrieving files in parallel from inexpensive disks can significantly improve the scalability of a DL server responding to many simultaneous requests.

RECEIVED JUL 16 1999 TECH CENTER 2800 MAIL ROOM

We have implemented our scheme on a cluster of SUN SPARC nodes connected by a Meiko CS-2 Elan network, and clusters of SUN and DEC workstations connected by Ethernet. Each processing unit (e.g. a SUN SPARC node) is linked to a local disk and is capable of handling a user request. There are a variety of resource constraints that can affect the performance of the server. These constraints include: the CPU speed and memory size of a single processing unit; the current system load; the transmission bandwidth between the processing unit and its local disk; the network latency and bandwidth between a processing unit and a remote disk when the accessed files are not stored in the local disk; and disk contention when multiple I/O requests access the same disk. By understanding these effects, we can achieve server scalability. In particular, by actively monitoring the run-time CPU, disk I/O, and network loads of system resource units, we can dynamically schedule user requests to nodes in a manner that provides the greatest overall processing efficiency.

Our strategies are based on our work for a scalable WWW server called SWEB [1]. We assume that the system contains a set of networked workstations (see Figure 3). Some of the workstations are connected to SCSI-II disks or mass storage subsystems. In this paper, the terms workstation unit, node, and processor are interchangeable. We assume that each CPU unit may be used by other applications and can leave and join the resource pool at any time.

The paper is organized as follows: Section 2 briefly describes the ADL Project. Section 3 discusses the scalability issues addressed in the ADL system for ameliorating network and server bottlenecks. Section 4 discusses scheduling and resource monitoring strategies in our multi-processor system. Section 5 presents experimental studies concerning multi-node performance and analyzing overhead and the effectiveness of resource scheduling. Section 6 discusses related work. Section 7 discusses conclusions and future work.

2. The Alexandria Digital Library on the WWW

Key aspects of the development strategy for ADL involve:

- developing a library whose components are distributed over the Internet and are accessible to many classes of users;
- following an evolutionary and incremental approach to both design and implementation;
- focusing on the design of *digitally supportable extensions* to traditional library functionality, and making ADL consistent with requirements of the library community;
- providing the user with access to the implicit information available in the DL collections, as well as to the explicit information;
- developing collections of *spatially-referenced* materials.

We comment briefly on these key strategic points.

Since we are initially focusing on users who access ADL from the WWW, primary access to ADL is from WWW browsers connected to the ADL WWW server. Z39.50 clients are also able to connect with the ADL SQL/Z39.50 query engines. The WWW is based on three critical components: the Uniform Resource Locator (URL), the HyperText Markup Language (HTML), and the HyperText Transfer Protocol (HTTP). The URL defines which resource the user wishes to access, the HTML language allows the information to be presented in a platform-independent but still well-formatted manner, while the HTTP protocol is the application-level mechanism for achieving the transfer of information [8]. The WWW supports general types of multimedia information systems while a DL system provides more advanced features for browsing, searching, and delivering digitized documents.

A major reason for adopting an evolutionary and incremental approach to design and development stems from the rapidity of developments in Internet technology. The first increment in the development of ADL involved the design and construction of a stand-alone "rapid prototype" (RP) system [5]. A second, and now completed, increment provided an augmented version of the functionality of the RP over WWW. The third increment is focused on developing a greatly enhanced catalog component based on a general model of metadata and supporting catalog interoperability with other DLs.

The approach of providing digitally-supportable extensions to traditional libraries is represented in the high-level architecture of ADL shown in Figure 1. Each of these components may be distributed over the Internet. This architecture involves the four major

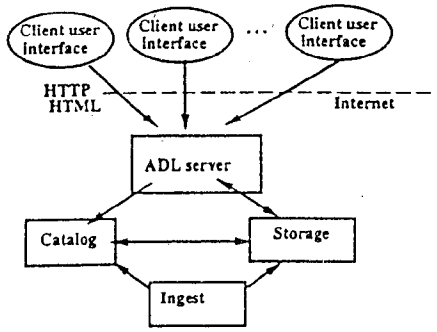


Figure 1. The ADL architecture.

components of traditional libraries, namely a catalog component, a collection component, an ingest component, and a user interface component.

- The *storage component* of ADL contains a collection of *digital objects*. The collections on which ADL is initially focused include spatially-referenced materials, such as digitized maps, digitized aerial photographs, and images from many domains of application [5]. An important aspect of the ADL collection is that individual items are typically very large. Satellite images are frequently 100 MB in size, and sizes of up to two GB are not uncommon.

- The *catalog component* of ADL permits users to make a mapping between their requirements for information and the most appropriate set of information that can be accessed from the library's collection of items. Since it is important that spatially-referenced items be accessible by means of a spatial reference, each item is represented in the catalog by a *spatial footprint*, which is a pointset characterizing the spatial extent of the item in the space over which it is defined. Footprints are represented in an extensible metadata model for spatially-referenced information (currently combining the FGDC and USMARC standards [5]) and are indexed to support efficient search over the catalog holdings. The metadata model also incorporates extensions involving gazetteers (i.e. mappings between named geographic features and the footprints of their spatial extent) and preselected image "textures features". Both the gazetteer and the image texture features are used to support content-based search.
- The *ingest component* involves the digitization of non-digital items; the extraction of catalog metadata from items that are admitted to the collections (which initially may be in either digital or non-digital form); and the application of transformations, such as wavelet decompositions, to ingested items. The metadata extraction is in accordance with the metadata model of the catalog. Currently available metadata includes approximately 450K frame-level records for a NASA/Ames database; approximately 350K sheet-level records for Geodex topographic series; approximately 100K USMARC map records from MELVYL; and catalog records for selected items, such as WWW sites for spatial data in digital form and aerial photographs for four local counties in California.
- The goal of the *interface component* is to provide easy access to a core set of functionality for a heterogeneous user population. This component contains a browser based on HTTP/HTML to support user access via the WWW. Current implementations of HTTP/HTML impose significant limitations on browsers, such as statelessness and the general reliance on small, fast transactions. In particular, HTML lacks mechanisms for presenting spatial data in vector form, and provides weak support for the entry of spatially-indexed information. The development of the WWW interface component has involved attempts to overcome such limitations, and the system provides external viewers and "helper apps" for the display of spatially-indexed materials of both raster and vector type. These limitations are being overcome with the current generation of programmable browsers.

3. Scalability of the ADL

As noted above, digitized data objects in ADL are typically very large. With current network speeds, it is quite infeasible to consider sending the full contents of an image file to users for the browsing purposes. An image data file of size 100 MB will take about 8.5 minutes over a full T1 (1.544Mb/sec) connection. For the next generation of Internet, e.g. T3 (45Mb/sec), TV set-top boxes (10Mb/sec), ATM and vBNS (155Mb/sec), the transmission time will significantly decrease but the demands for larger image files will continue increasing, especially when there are millions of users on the Internet. The ADL has adopted progressive multi-resolution and subregion browsing strategies to reduce Internet traffic in accessing map images. This approach is based on the following ideas:

- Users often make the selection of materials of interest without browsing the image information at fine-grain levels of resolution; in particular, they initially need information only at coarse levels of resolution. Delivering images at coarse grain resolution substantially reduces the size of data transferred between the client and the ADL server.

  For current computer monitors, it is reasonable to assume that one would usually view an image of resolution 512×512, or at most $1K \times 1K$ to fit a screen. Compressed 512×512 color images have size of 100K-300KBytes and take around ten seconds to transfer over a T1 link.
- Users should be able to rapidly view higher-resolution versions of those images already being viewed to assist their selection. It is desirable to have a method that can construct a higher resolution image from the lower resolution image with only a small amount of additional data. If such construction can be performed at the client site, then since the lower solution image is already available at the client site, only the difference data needs to be transferred from the ADL server over the Internet. Note that the size of difference data is usually small, taking less than 1 second to deliver in a T1 link.
- Satellite map images usually have high resolutions (e.g. $2K \times 2K$ and $10K \times 10K$), and such high resolutions can not be viewed on a regular screen. A reasonable user requirement is to browse a subregion of an image to identify details of interest. Popular subregion resolutions are likely to be around 512×512. Thus supporting subregion browsing can also significantly reduce network bandwidth requirements.

To support these features, the ADL system is using a wavelet-based hierarchical data representation for multi-resolution decomposition of images. Images and their subregions can be browsed in different levels of resolution and can be delivered in a progressive manner [2]. We briefly describe the techniques of wavelet image data retrieval and transformation below.

Given an image, a forward wavelet transform produces a subsampled image of lower resolution called a "thumbnail", and three additional coefficient data sets. More formally, for the given quantized image I_1 of resolution $R \times R$[1], we specify the input and output of the forward wavelet transform as follows.

$$(I_2, C_1, C_2, C_3) = Forward_Wavelet(I_1).$$

I_2 is the thumbnail of resolution $\frac{R}{2} \times \frac{R}{2}$, C_1, C_2 and C_3 are of resolution $\frac{R}{2} \times \frac{R}{2}$. Fig. 2 depicts the result of wavelet transform.

[1]Rectangular shapes can also be supported while square images are used here for demonstration.



Figure 2. *left* A map image I_1.(right) The thumbnail I_2 after applying the forward wavelet transform.(left)

The inverse wavelet transform can be performed to re-construct the original image on-the-fly from the coefficient data sets and the thumbnail.

$$I_1 = Inverse_Wavelet(I_2, C_1, C_2, C_3).$$

If image thumbnail I_2 is available at the client site, then by requesting that ADL sends C_1, C_2, C_3, image I_1 can be reconstructed at the client site. The image reconstruction is not time consuming, taking about 1.5 seconds for a 512×512 image on a SUN SPARC 5. The size of compressed data C_1, C_2, C_3 to be transferred is in the range of 10 to 100KBytes, which takes less than 1 second over a T1 link.

If a user wishes to access subregions of an image I_1, then the corresponding subregions in thumbnail I_2, C_1, C_2, C_3 can be retrieved and the reconstruction performed accordingly. We model such a process as follows.

$$subregion(I_1) = Inverse_Wavelet(subregion(I_2), subregion(C_1), subregion(C_2), subregion(C_3)).$$

A detailed definition of forward and inverse wavelet functions can be found in [4]. The time complexity of wavelet transforms is proportional to the image size. The wavelet transform can be applied recursively, namely the thumbnail I_2 can be decomposed further to produce smaller thumbnails $I_2, I_3, \cdots$. The ingest component of the ADL performs the forward transformation to decompose data images into thumbnails and the coefficient data set.

The ADL system uses the above wavelet technique and at run-time, the following operations will be frequently invoked on the ADL server.

- *Retrieval of regular files and thumbnail images:* when a client requests a thumbnail for the initial browsing, the server retrieves the corresponding file from the ADL storage.
- *Retrieval of image coefficient data for progressive image browsing:* when a client further requests images with higher resolutions, the server retrieves compressed image coefficient data from permanent storage and delivers it to the client as the client machine performs the inverse wavelet to construct images of higher resolutions from the existing thumbnail and new coefficient data. If the client machine does not have such a capability, the server performs the image reconstruction.
- *Retrieval of image subregions:* after a client identifies an interesting point in an image, the request is made for an enhanced resolution view of the subregion surrounding that point. The appropriate image data is then retrieved and sent to the client.

If a user finds it necessary to access the original large image, e.g., for scientific applications, the ADL will direct this request to a large storage server (currently a 1TB robotic tape storage device at the San Diego Supercomputer Center). This file will be delivered via ftp since large images (with size 10-100MB), take a long time to transfer over the Internet. We do not address the issue of ftp delivery of large documents in this paper.

It should be noted that there are other operations performed in the ADL server. For example, content-based database queries to find suitable images are important, so the speed of the database server and its supporting mass storage is vital [2, 12]. We assume that the database functionality is provided by a separate computer within ADL, and so focus our attention on the problem of delivering data, whether simple files or wavelet data, to the user as quickly as possible over the Net. We also focus on scalability issues in supporting the multi-resolution and subregion browsing of images.

While we have addressed issues relating to the network bandwidth bottleneck, the ADL server itself can be another bottleneck in document delivery. For example, the computation and disk I/O performed in the ADL for decompressing and accessing subregions of images involve a substantial amount of time and occupy disk I/O channels for long periods.

There are several aspects in assessing the scalability of a DL system. When there are many requests coming in, the typical situation (such as one in the current WWW servers) is that the server's response for an individual request becomes slow. For a single-workstation server, there is an upper bound for the number of requests per second (RPS) that the server can handle. For example, a SPARC 10 can handle 4 requests per second for delivering files of size 1MB-2MB. When the system limit is reached, the requests fail due to congestion at the server. Our overall objective for the system is to reduce and sustain the response time under large numbers of simultaneous requests. We define the response time for a request as the length of time from when a request is initiated until all requested information arrives at the client for that transaction. Another performance goal is to have the RPS limit of the system as large as possible since the access activities of current popular WWW sites already indicate that $\geq$ 20 requests per second can be expected.

We performed an experiment to determine the network bandwidth requirements after adopting the progressive image browsing strategy, and examine the ADL server requirements. Table 1 gives the compressed size for a number of greyscale images. Each image

was eight bits per pixel, and was a square image. We use the compression algorithm developed in [11] whose compression ratio is approximately 90%. The compressed full images are still sizeable. Using progressive image delivery can reduce network demands since full resolution images are not always required. For example, with the "pentagon" image (a satellite photo of the Pentagon), the 128 × 128 pixel thumbnail requires just 3.3K, and only 6.2K of additional data is needed to view this image at a 256 × 256 resolution. This indicates our multi-resolution/subregion browsing strategy significantly reduces network bandwidth requirements. But retrieving subregions from the compressed images imposes processing cost. On a Sun SPARC station 10, the current implementation takes 2 seconds of CPU time for extracting 8KB of compressed coefficient data from a 2K x 2K pixel greyscale image. To support 20 users per second, at least 40 high-end workstations need to be employed.

Image	*Dim.*	*Full*	*Thumb*	×2	×4
lena	256^2	7K	1K	1.5K	2.3K
textures	512^2	47K	3.3K	6K	14K
pentagon	1024^2	102K	3.3K	6.2K	16.4K
spot_reg1	2048^2	440K	13K	27K	75K

Table 1. Wavelet compressed data size for progressive delivery. "Full" is the compressed full image size. The thumbnail size is 1/8 of the original image dimensions.

To reduce and sustain the response time under large numbers of simultaneous requests, our strategies involve:

- Utilizing multiple networked commodity workstations and disks to build a scalable server. The computing environment can be heterogeneous and workstation/processor units with different speeds and different loads at any time.
- Developing sophisticated dynamic scheduling algorithms for exploiting task and I/O parallelism adaptive to the run-time change of system resource loads and availabilities. The system needs to provide good system resource estimation to assist the scheduler. The scheduler needs to incorporate multiple system performance parameters to assign user requests to a proper unit for efficient processing. The overhead involved for current resource load assessment and scheduling should be minimized.

4. System Resource Monitoring and Request Scheduling

In this section we first discuss the monitoring of system resources, introduce the functional modules of our scheduler, and we then present an algorithm for determining the processor assignment of a given ADL request.

There are several factors that affect the response time in processing ADL requests. These include loads on CPU, disk, and

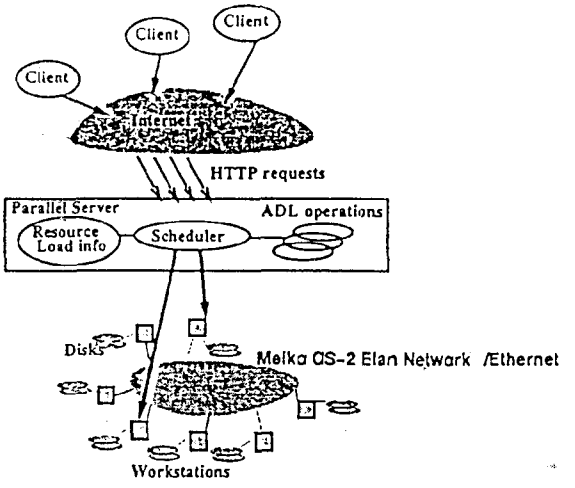


Figure 3. ADL multi-node server architecture.

network resources. The load of a processing unit must be monitored so that requests can be distributed to relatively lightly loaded processors. An ADL request following the HTTP protocol is handled through a TCP/IP connection and then subsequently through a forked subprocess. This subprocess may retrieve a file or invoke a Computational Gateway Interface (CGI) program implementing an ADL-specific operation. In processing most ADL requests, image data needs to be retrieved from disks, so disk channel usage must be observed. Simultaneous user requests accessing different disks can utilize parallel I/O to achieve higher throughput. The local interconnection network bandwidth affects the performance of file retrieval since many files may not reside in the local disk of a processor. Therefore remote file retrieval through the network file system will be involved. Local network traffic congestion could dramatically slow the request processing.

Thus a critical component of our system is a load daemon running at each processor to detect its own CPU, disk, and network load, and periodically broadcasts this information to other processors. Our experiments show that such overhead is insignificant.

4.1. Internal scheduler structure

There are two approaches to designing the scheduler. One is to have a centralized scheduler running on one processor such that all requests go through this processor. The scheduler monitors the usages of all system resources, makes assignment decisions based on this information, and routes requests to appropriate processors. Our main reason for not adopting this approach is that the central distributor becomes a single point of failure, making the entire system vulnerable.

The current version of our system uses a distributed scheduler. The user requests are first evenly routed to processors via Domain Name System (DNS) rotation. DNS rotation provides the initial

assignment of HTTP requests and is used in the NCSA multi-workstation server [10]. In this scheme, multiple real machines are mapped to the same IP name. When a client requests the network ID of the machine name (e.g., www.cs.ucsb.edu), the DNS at the server site rotates the network IDs, picking one (e.g., 1.1.1.1) to send back to the client. The rotation on available workstation network IDs is in a round-robin fashion. This functionality is available in current DNS systems. The major advantages of this technique are simplicity and ease of implementation [10].

The DNS is subject to caching problems when attempting to do dynamic load balancing, and it assigns requests without consulting dynamically-changing system load information. Thus our scheduler conducts a further re-direction of requests. Each processor in the ADL server contains a scheduler and those processors collaborate with each other to exchange system load information. After a request is routed to a processor via DNS, the scheduler in that processor makes a decision regarding whether to process this request or redirect it to another processor. An HTTP request is not allowed to be re-routed more than once in order to avoid the ping-pong effect.

The functional structure of the scheduler at each processor is depicted in Fig. 4. It contains a daemon based on NCSA httpd code for handling httpd requests, with a *broker* module that determines the best possible processor to handle a given request. The broker consults with two other modules, the *oracle* and the *loadd*. The *oracle* is a miniature expert system, which uses a user-supplied table to characterize the CPU and disk demands for a particular task. The *loadd* daemon is responsible for updating the system CPU, network and disk load information periodically (every 2-3 seconds), and marking those processors which have not responded in a preset period of time as unavailable. When a processor leaves or joins the resource pool, the *loadd* daemon will be aware of the change.

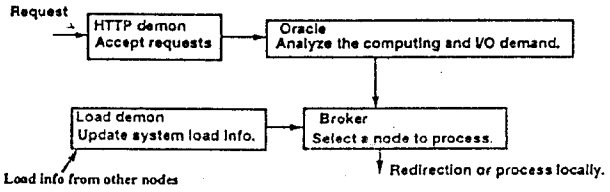


Figure 4. The functional modules of a scheduler for each processor.

After a processor accepts a HTTP request through TCP connection, the system assigns a request to a particular processor. We discuss how this request can be *transparently* routed to this processor. The best situation would be to modify the UNIX sockets package to change the semantics of the "accept" system call, but such modification requires a substantial change of the UNIX operating system kernel. The approach that we implemented is based on the fact that the HTTP protocol allows for a response called "URL Redirection". When client C sends a request to server S_0, S_0 returns a rewritten URL r' and a response code indicating that the information is located at r'. C then follows r' to retrieve the resulting data. Most Net browsers and clients automatically query the new location, so redirection is virtually transparent to the user.

The primary advantages of URL redirection are the simplicity of implementation and universal compatibility. An simple flow-of-control modification can be made within a WWW server, where the main complexity lies in the routines to determine the optimal server for a particular request. Furthermore, the approach lies well within our design parameters; it does not require a modification of the HTTP protocol, is reasonably efficient, and is able to support sophisticated optimization algorithms. The primary disadvantage of URL redirection in practice is the added overhead of an additional connect/pass request/parse/respond cycle after the redirection occurs. We will show that such overhead is more than negated by improved performance overall.

4.2. The processor assignment of ADL requests

In [1], we designed an algorithm that decides the routing for a general HTTP request. In this subsection, we discuss the strategies for the ADL system.

In the previous work on load balancing (e.g. [13]), usually one factor (CPU load) is considered. A processor can be classified as lightly loaded and heavily loaded based on the CPU load. One purpose of such a classification is to update load information *only* when a classification changes. Such a strategy reduces unnecessary overhead. In our problem context, it is hard to classify a processor as heavily or lighted loaded since there are several load parameters. A processor could have a light CPU load but its local disk may receive many access requests from the network file system.

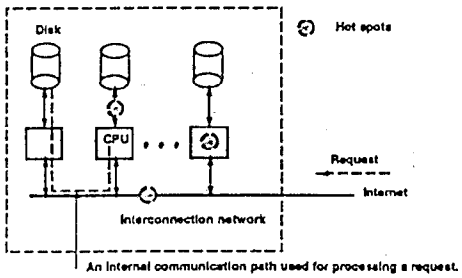


Figure 5. Resource hot spots monitored.

As listed above, several system load parameters affect such a scheduling decision and should be considered together. Figure 5 shows hot spots in system resources to be monitored and depicts a communication path used in processing an ADL request. This path starts from one CPU and goes through the local interconnection network and then through another disk (remote file accessing). We discuss two cases in explaining how the aggregated resource load factors affect the choice of processor assignment. Assume that a request is to access a file on the disk of Processor A (called disk A in short).

- Given a slow interconnection network, this request should be assigned to processor A if A is not heavily loaded. If this request is assigned to another processor B, the remote

file access from B to disk A takes a long time, which slows down the response performance.

But if processor A is heavily loaded, then this request should be assigned to a node whose CPU is lightly loaded since computation cost takes a part of the response time. Also if disk A's channels are fully occupied, the available bandwidth may be smaller than the network bandwidth. In this case disk locality becomes less significant, since remote accessing speed is not the key factor dominating the performance.

- Given a fast interconnection network, remote access will not create a bottleneck for request processing, thus a processor which has a light CPU load should be selected. Of course the load of network, disk channels and CPUs should be consistently monitored to let the scheduler be aware of the change.

Multiple system load parameters can be used together in deciding the assignment of a request. Since our goal is to minimize the response time for each request, we design the decision-making heuristic based on the following estimated response time function for processing each request:

$$T_h = T_{redirection} + T_{data} + T_{CPU} + T_{net}$$

$T_{redirection}$ is the overhead to redirect the request to another processor, if required. T_{data} is the time to transfer the required data from the disk drive, or from the remote disk if the file is not local. T_{CPU} is the time to fork a process and perform disk I/O to handle a HTTP request, plus any known associated computational cost. In the ADL environment, we identify a set of wavelet-related functions and characterize their CPU demands. For example, accessing a wavelet data subregion costs CPU cycles proportional to the subregion size in the ADL implementation. T_{net} is the cost for transferring the processing results over the Internet. Since we are interested in getting the result out of the server as soon as possible, T_{net} is considered constant in our scheduling scheme. Furthermore, this time is likely to be the same across the set of nodes and can be ignored.

Having determined the estimated time for each node to fill the request using the formula above, the broker selects a node with the minimum estimated completion time. If the chosen node x is not the processor for this broker, the request is redirected to x. At node x, this request is processed in the normal HTTP manner, with any CGI's executed as needed.

It should be noted that it is not easy to model the cost associated with processing an ADL request accurately. We still need to investigate further the design of such a function. Our experiment shows that the current cost function does reflect the impact of multiple parameters on the overall system response performance, and that the multi-node system delivers acceptable performance based on such a heuristic function, adapting to dynamically-changing system resource loads. In our experiments, we show that the overhead for scheduling and system monitoring is insignificant compared to the system resources used for request fulfillment and other activities.

5. Experimental Studies

Our primary experimental testbed consists of a workstation cluster (Meiko CS-2) at UCSB. Each node has a scalar processing unit (a 40Mhz SPARC Viking chip) with 32MB of RAM running Solaris 2.3. We mainly use six CS-2 nodes, each of which is connected to a SCSI-II 1GB disk on which test data files reside. The reason that we use 6 Meiko nodes is that these were the available resource with disks dedicated to our research. Under these configurations our experiments still demonstrate the scalability of the system and effectiveness of our scheduling techniques. Disk service is available to all other nodes via NFS mounts. These nodes are connected via a modified fat-tree network called Elan with a peak bandwidth of 40MB/s. The complete ADL SWEB server is based on NCSA httpd 1.3 source with extensive modifications to support scheduling functionality. Each custom client accesses the broker, and is then redirected (if necessary) to the appropriate node to fulfill the request. All software uses the sockets library built on TCP/IP. Because current Meiko CS-2 communication routines are not optimized for TCP/IP we were only able to achieve approximately 5-15% of the peak communication performance on the Meiko.

We consider two major types of requests for the ADL: file fetches (e.g. thumbnails) and wavelet data retrieval for multi-resolution/subregion browsing. The second type usually involves a certain amount of CPU activity in locating appropriate image data and performing on-the-fly compression. We believe that these types of requests will consume the majority of resources within the current ADL WWW prototype. As we discussed before, direction of database queries (farmed out to a dedicated database machine) and dynamically generated interface pages for the client (which may prove significant) consume other CPU cycles. We will address these activities in our future work, but we expect that our results in this paper shall be valid for those activities as well.

Our clients were primarily situated within UCSB. This was due to the instability of available Internet bandwidth making it difficult to obtain consistent results. Furthermore, our goal is to get the data out of the server as fast as possible, and providing bandwidth is outside the scope of our research. We expect that over the next few years, the available bandwidth of the Internet will increase substantially and will be comparable to our current local environment.

It should be noted that the results we report involve the average performance of multiple iterations over an extended period. The test performance is affected by dynamically-changing system loads since the machines are shared by many active users at UCSB.

We report our experiments to examine the performance of our system in the following aspects: scalability of overall performance, the effectiveness of scheduling strategies, and a discussion of scheduling overhead costs imposed.

5.1. The maximum number of requests per second

The first experiment was run to determine how many requests per second could be processed in delivering regular files such as thumbnails or text. This depends on the average file sizes requested and the number of nodes. The maximum number of serviced RPS (or MRPS in short) is determined by fixing the average file size and increasing the RPS until requests start to fail, which indicates that the system limit is reached. The duration of the test which simulates the burst of simultaneous requests also affects the experimental results. Requests arriving in a short period can be queued and processed gradually. But requests continuously generated in a long period cannot be queued without actively processing them since new requests are continuously arriving. We have chosen 30 seconds as the test duration since in practice bursts of requests arrive periodically.

File sizes in bytes	1K	1.5M
Single server	45	9
Multi-node server	82	45

Table 2. MRPS for a test duration of 30s on six nodes clustered by Meiko CS-2 Elan.

Table 2 shows the MRPS numbers obtained for a test duration of 30 seconds on the 6-node Meiko. For small files, the MRPS can reach 45 for a single node, but only 9 for 1.5MB files. We also conducted a test for a duration of 120 seconds, in which MRPS drops to 4 for a single node. The multi-node server can significantly speedup the MRPS as shown in Table 2. We also tested the MRPS on the workstations clustered by the Ethernet. Effective bandwidth of our Ethernet is much smaller than the CS-2 Elan network, the MRPS for processing 1.5 MB files is about twice as small as on the Meiko.

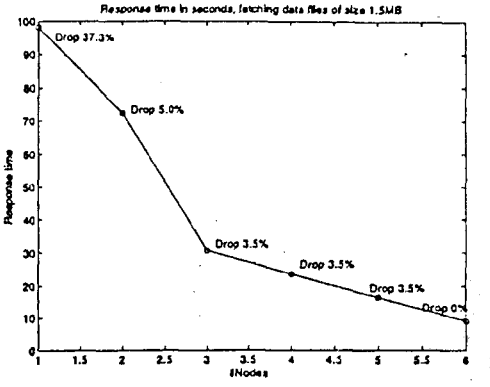
No. servers	1	2	4	6
MRPS	2	4	8	12

Table 3. MRPS for accessing 512×512 image subregion data at a duration of 30s on Meiko.

File accesses do not involve additional computations on the server. Accessing subregion data involves some computation to locate the correct location of data in compressed large images. We examine the performance of accessing 512×512 subregion data from a set of $2K \times 2K$ map images at the server site and delivering the results to the client for constructing a 512×512 image. Figure 3 shows the MRPS for a duration of 30 seconds on Meiko for such browsing activities. Linear speedups are obtained using multiple nodes. Again, it should be noted that the single node performance reaches 2 MRPS for a test of 30 seconds. For a longer test, the requests cannot be queued and thus the MRPS drops.

5.2. Response time and drop rates

We examine the response time when we vary the number of server nodes. The system starts to drop requests if the RPS reaches the limit. For small file requests (1K) with RPS=16, the multi-node server performs much better than one-node server but the response remains constant (around 0.5 second) when using 2 or more processors, because none of the theoretical or practical limits on bandwidth or processing power have been reached.

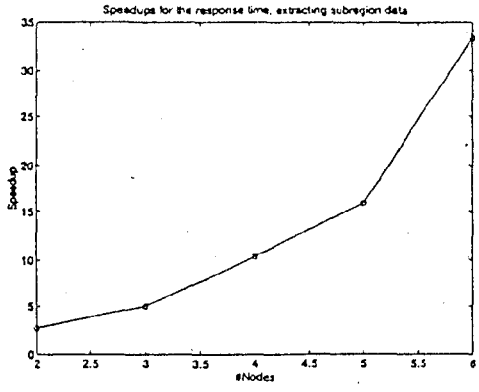


$\times 512$ subregions from $2K \times 2K$ satellite images) we again note a super linear speedup as shown in Figure 7.

The super linear speedup in terms of response times reflect the fact that the total size of memory in the multi-node server is much larger than on a one-node server. Furthermore, the multi-node server accommodates more requests within main memory while one-node server spends more time in swapping between memory and the disk. The network overhead for accessing remote files is distributed among multiple nodes rather than concentrated at a single node.

5.3. The effectiveness and overhead of scheduling

Our scheduling strategy takes into consideration the locality of requested files, and also the current resource loads. We compare our approach with a strategy that uniformly distributes the requests



$\geq$ 2 nodes.

to nodes (called round robin). Through such a comparison, we examine the benefits of scheduling in an ADL environment with heterogeneous computing and I/O activities.

We tested the ability of the system to handle ADL requests for accessing image and text files with mixed sizes from 100 bytes to 1.5MB. Fig 8 shows the performance improvement in terms of response times. For heavily loaded situations, our scheduling strategy considers the aggregative impact of system resources over the ADL operations and has an advantage of 18-54% over round robin for heavily-loaded situations with varying request requirements. For lightly loaded cases, the round robin performs well since CPU/disk/network resources are not saturated. For such cases, load imbalance does not affect the performance of processing. The added overhead in our strategy only slows down transfers by 1%, which is insignificant. A series of tests with wavelet subregion image data accesses were performed, also indicating similar results.

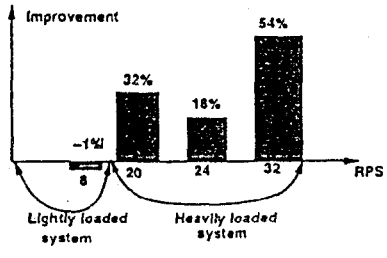


Figure 8. Effectiveness of scheduling. Improvement ratio of intelligent scheduling over round-robin. Meiko, 30 sec.

In order to determine the overhead of scheduling imposed on the system, we instrumented our code to determine the overhead cost for a request. For the case of fetching a 1.5MB image file over a fairly heavily loaded system, the total time from request initiation to receiving the last byte of information was approximately 5.4 seconds. Of that time, the server spent 4.9 seconds transmitting data, with approximately 10% of the total time spent acquiring the connection, handling the redirection, and spent merely 0.005 seconds in the scheduling algorithm. The algorithm requires approximately 0.004 seconds for analysis when the URL has not been previously redirected, and about 0.001 seconds when it has. For wavelet subregion image extraction, the numbers are virtually identical, with the exception that the time spent actually fulfilling the request is typically larger. The results indicate that the overall overhead introduced by the scheduling algorithm and load monitoring is insignificant.

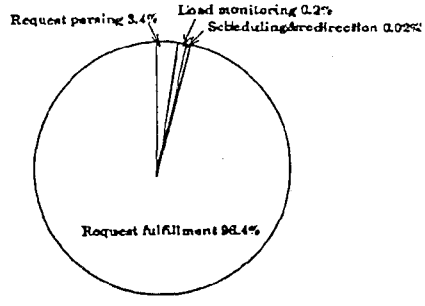


Figure 9. CPU time distribution at a server node. The test is for 512 x 512 subregion extraction.

We also investigated the distribution of CPU cycles at the server site to identify the amount contributed for scheduling decisions and resource information collection. Figure 9 shows the server load distribution for processing requests that access subregions of $2K \times 2K$ images from the ADL map collection in the previous experiment (see Figure 7). 3.4% of the CPU is used for parsing the HTML commands, but less than 0.2% time is used for collecting load information and making scheduling decisions. We investigated the overhead for fetching files, and the percentage of overhead for load information collection and scheduling again is very small (less than 0.1%).

6. Related Work

Numerous other initiatives to create high-performance HTTP servers have been reported. The *Inktomi* server at UC Berkeley is based on NOW technology [3], and the research focus is on information searching and indexing. NCSA [10] has built a multi-workstation HTTP server based on round-robin domain name resolution (DNS) to assign requests to workstations. The round-robin technique is effective when HTTP requests access HTML information of relatively uniform size and the load and computing power of workstations is relatively comparable. Our assumption is that the computing power and other machine resources can be heterogeneous. They can be used for other computing needs, and can leave and join the system resource pool at any time. Thus scheduling techniques which are adaptive to the dynamic change of system

load and configuration are desirable. Heterogeneous computing is studied in [15]. We have not addressed scheduling on processors with different architectures in this paper; however, our techniques can be extended to such cases.

Our dynamic scheduling scheme is closely related to the previous work on load balancing on distributed systems, for which a collection of papers is available in [13]. In these studies, task arrivals may temporarily be uneven among processors and the goal of load balancing is to adjust the imbalance between processors by appropriately transferring tasks from overloaded processors to underloaded processors. The task resource requirements are unknown, and the criteria for task migration are based on a single system parameter, i.e., the CPU load. We call them single-faceted scheduling strategies. In WWW applications, there are multiple parameters that affect the system performance, including CPU loads, interconnection network performance and disk channel usages. The optimal HTTP request assignment to processors does not solely depend on CPU loads. Thus we develop a multi-faceted scheduling scheme that can effectively utilize the system resources by considering the aggregate impact of multiple parameters on system performance. In [7], resource requirements are predicted and suggested to guide the load sharing. They discuss multiple factors, but utilize only the CPU factor in predicting response times. With ADL applications we can take advantage of specific domain knowledge to accurately predict multiple resource requirements for each request.

7. Conclusions and Future Work

We have discussed issues involved for developing a scalable DL system on the WWW. Multi-resolution and subregion image browsing substantially reduces the network bandwidth requirements, but imposes additional data indexing complications and computational demands at the server site. We have implemented a multi-node scheme to strengthen the processing capabilities of the ADL server. Our experiments indicate that our scheduling scheme effectively monitors and utilizes the multiple system resources, while the overhead required to support such a scheme is extremely low.

In future work, we plan to migrate the multi-node scheme into the production environment as the primary server for the Alexandria Digital Library project. We feel this will introduce many interesting research problems relating to scheduling wildly varying demands in a heterogeneous environment. We will also investigate the incorporation of other Alexandria-specific functions such as map image querying, searching and on-the-fly HTML generation.

Acknowledgments

This work was supported in part by funding from NSF, ARPA, and NASA under NSF IRI94-11330 and a startup fund from University of California at Santa Barbara. Thanasis Poulakidas and Ashok Srinivasan provided the wavelets implementation and compression data used. We would like to thank the Alexandria Digital Library team for many valuable discussions and suggestions. We would also like to acknowledge the student researchers who contributed to this project, including Vegard Holmedahl and David Watson.

References

[1] D. Andresen, T. Yang, V. Holmedahl, O. Ibarra, SWEB: Towards a Scalable WWW Server on MultiComputers, *Proc. of Intl. Symp. on Parallel Processing*, IEEE, April, 1996. HTTP://www.cs.ucsb.edu/Research/rapid_sweb/SWEB.html.

[2] D.Andresen, L.Carver, R.Dolin, C.Fischer, J.Frew, M.Goodchild, O.Ibarra, R.Kothuri, M.Larsgaard, B.Manjunath, D.Nebert, J.Simpson, T.Smith, T.Yang, Q.Zheng, "The WWW Prototype of the Alexandria Digital Library", *Proceedings of ISDL'95: International Symposium on Digital Libraries*, Japan August 22 - 25, 1995.

[3] E. Brewer, Personal communication, http://inktomi.berkeley.edu, Jan., 1996.

[4] E.C.K. Chui, Wavelets: A Tutorial in Theory and Applications, Academic Press, 1992.

[5] C. Fischer, J.Frew, M. Larsgaard, T.R. Smith and Q.Zheng. Alexandria Digital Library: Rapid Prototype and Metadata Schema. Proceedings of Advances in Digital Libraries 1995, Vol. ,pp. , 1995.

[6] E. Fox, Akscyn, R., Furuta, R. and Leggett, J. (Eds), Special issue on digital libraries, *CACM*, April 1995.

[7] K. Goswami, M. Devarakonda, R. Iyer, Prediction-based Dynamic Load-sharing Heuristics, *IEEE Trans. on Parallel and Distributed Systems*, 4:6, pp. 638-648, 1993.

[8] Hypertext Transfer Protocol(HTTP): A protocol for networked information, http://www.w3.org/hypertext/WWW/Protocols/HTTP/HTTP2.html, June, 1995.

[9] Lycos Usage: Accesses per Day, http://lycos.cs.cmu.edu/usage-day.html, June 17, 1995.

[10] E.D. Katz, M. Butler, R. McGrath, A Scalable HTTP Server: the NCSA Prototype, *Computer Networks and ISDN Systems*. vol. 27, 1994, pp. 155-164.

[11] A. Poulakidas, A. Srinivasan, O. Egecioglu, O. Ibarra, and T. Yang, Experimental Studies on a Compact Storage Scheme for Wavelet-based Multiresolution Subregion Retrieval, Proceedings of NASA 1996 Combined Industry, Space and Earth Science Data Compression Workshop, Utah, April 1996.

[12] S. Prabhakar, D. Agrawal, A. El Abbadi, A. Singh, T. Smith, Content based placement and browsing, CS Tech Report, UCSB, 1995.

[13] B. A. Shirazi, A. R. Hurson, and K. M. Kavi (Eds), Scheduling and Load Balancing in Parallel and Distributed Systems, IEEE CS Press, 1995.

[14] T. R. Smith and J. Frew. Alexandria digital library. *CACM*, 38(4):61-62, April 1995.

[15] R. Wolski, C. Anglano, J. Schopf, F. Berman, Developing Heterogeneous Applications Using Zoom and HeNCE, *Proceedings of the Heterogeneous Computing Workshop, HCW '95*, pp. 12-21, Santa Barbara, CA, IEEE, April, 1995.

OIPE JC82 JUL 12 1999 PATENT & TRADEMARK OFFICE

SWEB: Towards a Scalable World Wide Web Server on Multicomputers

Daniel Andresen Tao Yang Vegard Holmedahl Oscar H. Ibarra
Department of Computer Science
University of California
Santa Barbara, CA 93106
{dandrese, tyang, veho, ibarra}@cs.ucsb.edu

Abstract

We investigate the issues involved in developing a scalable World Wide Web (WWW) server on a cluster of workstations and parallel machines. The objective is to strengthen the processing capabilities of such a server by utilizing the power of multicomputers to match huge demands in simultaneous access requests from the Internet. We have implemented a system called SWEB on a distributed memory machine, the Meiko CS-2, and networked workstations. The scheduling component of the system actively monitors the usages of CPU, I/O channels and the interconnection network to effectively distribute HTTP requests across processing units to exploit task and I/O parallelism. We present the experimental results on the performance of this system.

1 Motivation

The Scalable Web server (SWEB) project grew out of the needs of the Alexandria Digital Library (ADL) project at UCSB [A96]. Digital library systems, which provide the on-line retrieval and processing of digitized documents through Internet, have increasingly turned into a topic of national importance. The Alexandria Project is focused on the design, implementation, and deployment of a distributed digital library for spatially-indexed information. The collections of the library currently involve geographically-referenced materials, such as maps, satellite images, digitized aerial photographs, and associated metadata. A rapid prototype system for the Alexandria digital library has already been developed which allows users to browse spatially-indexed maps and images [AC95+]. To expose the system to a broad user community, the next version of the system (available in early 1996) will be connected to the World Wide Web (WWW) using the Hypertext Transport Protocol (HTTP) [A96].

Our work is motivated by the fact that the Alexandria digital library WWW server has a potential to become the bottleneck in delivering digitized documents over high-speed Internet. Popular WWW sites such as the *Lycos* and *Yahoo* [LYC95] receive over one million accesses a day. For WWW-based network information systems such as digital libraries, the servers involve much more intensive I/O and heterogeneous CPU activities. To meet such demands, the SWEB project makes use of existing networked computers and inexpensive disk resources to strengthen the processing and storage capabilities of WWW servers. We expect that using the idle cycles of those processing units and retrieving files in parallel from inexpensive disks can significantly improve the scalability of the server in response to a large amount of simultaneous HTTP requests.

Numerous other initiatives to create high-performance HTTP servers have been reported. The *Inktomi* server at UC Berkeley is based on the NOW technology [BR96]. NCSA [KBM94] has built a multi-workstation HTTP server based on round-robin domain name resolution (DNS) to assign requests to workstations. The round-robin technique is effective when HTTP requests access HTML information of relatively uniform size and the load and computing powers of workstations are relatively comparable. Our assumption is that the computing powers of workstations and parallel machine resources can be heterogeneous. They can be used for other computing needs, and can leave and join the system resource pool at any time. Thus scheduling techniques which are adaptive to the dynamic change of system load and configuration are desirable. The DNS in a round-robin fashion cannot predict those changes. Another weakness of the technique is the degree of name caching which occurs. DNS caching enables a local DNS system to cache the name-to-IP address mapping, so that most recently accessed hosts can quickly be mapped. The downside is that all requests for a period of time from a DNS server's domain will go to a particular IP address.

We have developed the preliminary version of a scalable WWW server (SWEB) [AC95+] running on a Meiko CS-2 distributed memory machine and a network of workstations (NOW) such as SUN and DEC machines. Each processing unit is capable of handling a user request following the HTTP protocol. The distinguishing feature of SWEB is effective resource utilization by close collaboration of multiple processing units. Server scalability is achieved by actively monitoring the run-time CPU, disk I/O, and network loads of system resource units, and dynamically scheduling user HTTP requests to a proper node for efficient processing.

Our dynamic scheduling scheme is closely related to the previous work on load balancing on distributed systems, for which a collection of papers is available in [SHK95]. In these studies, tasks arrivals may temporarily be uneven among processors and the goal of load balancing is to adjust the imbalance between processors by appropriately transferring tasks from overloaded processors to underloaded processors. The task resource requirements are unknown, and the criteria for task migration are based on a single system parameter, i.e., the CPU load. We call them single-faceted

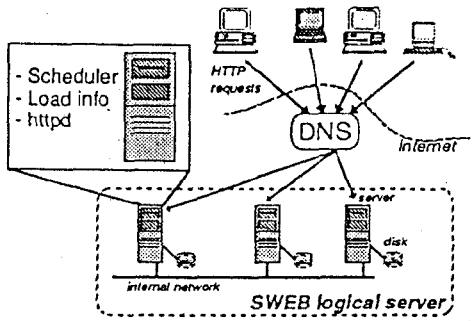


Figure 2: *The computing and storage architecture of SWEB.* DNS directs tasks to servers running task scheduler.

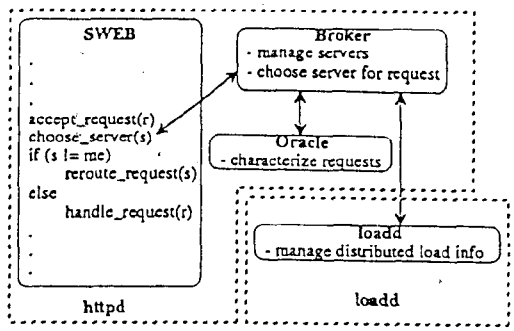


Figure 3: The functional modules of a SWEB scheduler in a processor.

the single central distributor becomes a single point of failure, making the entire system more vulnerable.

The current version of SWEB uses a distributed scheduler. The user requests are first evenly routed to SWEB processors via the DNS rotation, as shown in Figure 2. The rotation on available workstation network IDs is in a round-robin fashion. This functionality is available in current DNS systems. The major advantages of this technique are simplicity, ease of implementation, and reliability [KBM94].

The DNS assigns the requests without consulting dynamically-changing system load information. Then SWEB conducts a further assignment of requests. Each processor in SWEB contains a scheduler and those processors collaborate with each others to exchange system load information. After a request is routed to a processor via DNS, the scheduler in that processor makes a decision regarding whether to process this request or assign it to another processor. Two approaches, URL redirection or request forwarding, could be used to achieve reassignment and we use the former. Request forwarding is very difficult to implement within HTTP. URL redirection gives us excellent compatibility with current browsers and near-invisibility to users. Any HTTP request is not allowed to be redirected more than once to avoid the ping-pong effect.

The functional structure of the scheduler at each processor is depicted in Fig. 3. It contains a httpd daemon based on NCSA httpd code for handling httpd requests, with a *broker* module which determines the best possible processor to handle a given request. The broker consults with two other modules, the *oracle* and the *loadd*. The *oracle* is a miniature expert system, which uses a user-supplied table to characterize the CPU and disk demands for a particular task. The *loadd* daemon is responsible for updating the system CPU, network and disk load information periodically (every 2-3 seconds), and marking those processors which have not responded in a preset period of time as unavailable. When a processor leaves or joins the resource pool, the *loadd* daemon will be aware of the change.

3.2 The multi-faceted scheduling algorithm

In this subsection we present the algorithm that decides where a HTTP request should be routed. As we discussed before, several system load parameters affect such a decision. In a single-faceted scheduling system, a processor can be classified as lightly loaded and heavily loaded based on one parameter, e.g. CPU load. One purpose of such a classification is to update load information only when a classification changes to reduce unnecessary overhead, e.g. [SHK95]. In our problem context, it is hard to classify a processor as heavily or lightly loaded since there are several load parameters. A processor could have a light CPU load but its local disk may receive many access requests from the network file system. Thus each processor in SWEB has a load daemon to detect its own CPU, disk, and network load, periodically broadcasting such information to other processors. In our experiments we will show that such overhead is insignificant.

We address how multiple system load parameters can be used together in deciding the assignment of HTTP requests. Since our goal is to minimize the response time for each request, we design the heuristic based on the estimated cost for processing each request using the following formula:

$$t_s = t_{redirection} + t_{data} + t_{CPU} + t_{net}$$

$t_{redirection}$ is the cost to redirect the request to another processor, if required. t_{data} is the time to transfer the required data from the disk drive, or from the remote disk if the file is not local. t_{CPU} is the time to fork a process, perform disk reading to handle a HTTP request, plus any known associated computational cost if the request is a CGI operation. t_{net} is the cost for transferring the processing results over the Internet. We discuss and model these individual cost terms as follows.

- $$t_{data} = \begin{cases} \frac{\text{Requested file size}}{b_{disk} \times \delta_1} \\ \frac{\text{Requested file size}}{min(b_{disk} \times \delta_1, b_{net} \times \delta_2)} \end{cases}$$

  If the file is local, the time required to fetch the data is simply the file size divided by the available bandwidth of the local storage system, b_{disk}. We also measure the disk channel load δ_1. If there are many requests, the disk transmission performance degrades accordingly.

  If the data is remote, then the file must be retrieved through the interconnection network. The local network bandwidth, b_{net}, and load δ_2 must be incorporated. Experimentally,

scheduling strategies. In WWW applications, there are multiple parameters that affect the system performance, including CPU loads, interconnection network performance and disk channel usages. The optimal HTTP request assignment to processors does not solely depend on CPU loads. Thus we need to develop a multi-faceted scheduling scheme that can effectively utilize the system resources by considering the aggregate impact of multiple parameters on system performance. In [GDI93], resource requirements are predicted and suggested to guide the load sharing. They mention multiple factors, but utilize only the CPU factor in predicting response times.

The paper is organized as follows: Section 2 gives the background and problem definition. Section 3 discusses the possible approaches for building a scalable WEB server and the SWEB organization and load balancing strategies. Section 4 presents the experimental results showing the performance of SWEB and compares our approach with others. Section 5 discusses conclusions and future work.

2 Background: The World Wide Web

The World Wide Web is based on three critical components: the Uniform Resource Locator (URL), the HyperText Markup Language (HTML), and the HyperText Transfer Protocol (HTTP). The URL defines which resource the user wishes to access, the HTML language allows the information to be presented in a platform-independent but still well-formatted manner, and the HTTP protocol is the application-level mechanism for achieving the transfer of information [HT95].

A simple HTTP request would typically activate a sequence of events from initiation to completion as shown in Figure 1. First, the client determines the host name from the URL, and uses the local Domain Name System (DNS) server to determine its IP address. The local DNS may not know the IP address of the destination, and may need to contact the DNS system on the destination side to complete the resolution. After receiving the IP address, the client then sets up a TCP/IP connection to a well-known port on the server where the HTTP process is listening. The request is then passed in through the connection. After parsing the request, the server sends back a response code (e.g., 202 in the HTTP protocol stands for "OK. File found.", and 404 is "File not found.") followed by the results of the query. The connection is then closed by either the client or the server.

3 SWEB: A scalable WWW server

We call a system *scalable* if the system response time for individual requests is kept as small as theoretically possible when the the number of simultaneous HTTP requests increases, while maintaining a low request drop rate and achieving a high peak request rate. We define the response time for a request as the length of the time period from when a request is initiated until all the requested information arrives at the client.

For a single-workstation server, there is an upper bound for the number of requests per second (rps) that the server can handle. For example, the NCSA has performed a number of tests using high-end workstations, and discovered in their working environment approximately 5-10 rps could be dealt with using the NCSA httpd server [KBM94], which cannot match the current and future loads (e.g. a digital library server). Thus multiple servers are needed for achieving scalable performance.

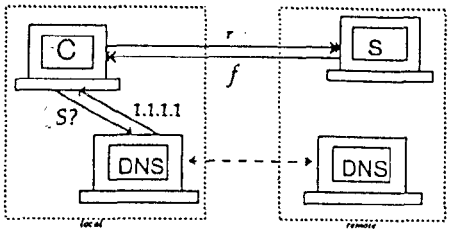


Figure 1: *A simple HTTP transaction.* Client C looks up the address of server S, sends over request r, and receives response f.

Our overall objective of the system is to reduce and sustain the response time under large numbers of simultaneous requests. Goals considered in designing this system are twofold. First, we hope to demonstrate how to utilize existing inexpensive commodity networks, heterogeneous workstations, and disks to build a scalable WWW server. Second, we attempt to develop dynamic scheduling algorithms for exploiting task and I/O parallelism adaptive to the run-time change of system resource loads and availabilities. The scheduler needs to incorporate multiple system performance parameters to assign user HTTP requests to a proper unit for efficient processing, while adding minimal overhead.

There are several performance factors that affect the response time in processing HTTP requests. These include processor load, caused by the overhead necessary to send bytes out on the network properly packetized and marshaled; disk I/O, which limits how quickly data can be delivered to the processor; the local network, over which remote data requests must be fetched; and the internet connection to the client, which often is a severe bottleneck.

Our goal is to get the requested information out of the server as fast as possible. Thus our scheduler primarily monitors the above first three performance factors. The Internet bandwidth information is used partially in request re-direction. Our scheduling algorithm is multi-faceted in the sense that the proper decision on routing HTTP requests needs to aggregate the impact of the above multiple system parameters on the overall system response time. We will present the architecture for SWEB and its scheduling algorithm next.

3.1 SWEB Architecture

There are two approaches in designing the scheduler. One is to have a centralized scheduler running on one processor such that all HTTP requests go through this processor. The scheduler monitors the usages of all system resources, makes assignment decisions based on this information, and routes requests to appropriate processors. We did not take this approach mainly because

we found on the Meiko approximately a 10% penalty for a remote NFS access, and on the SUN workstations connected by Ethernet the cost increases by 50%-70%.

- $t_{CPU} = CPU_{load} \frac{\text{No. of operations required}}{CPU_{speed}}$.
  The t_{CPU} term estimates the amount of processing time required to complete the task. This is based on the speed of the server, the estimated load on a destination node (CPU_{load}), and the estimated number of operations required for the task. The computation requirement for a particular request is estimated by the *oracle* component of the system (see Figure 3). The parameters for different architectures are saved in a configuration file. It should be noted that some estimated CPU cycles may overlap with network and disk time and the overall cost may be overestimated slightly, but this conservative estimation works well in our experience.
  The load estimation of remote processors is based on the periodic updating of information given by those remote processors. It is possible that a processor p_x is incorrectly believed to be lightly loaded by other processors, and many requests will be redirected to it. To avoid this unsynchronized overloading, we conservatively increase the CPU load of p_x by σ. This strategy is found to be effective in [SHK95]. We use $\sigma = 30\%$.
- $t_{net} = \frac{\#_bytes_required}{net_bandwidth}$
  This term is used to estimate the time necessary to return the results back to the client over the network. When the scheduler compares processors, we assume all processors will have basically the same cost for this term, so it is not estimated. Our research goal is to produce the query result for a HTTP request as fast as possible in the server site.
- $$t_{redirection} = \begin{cases} 0 \\ 2 * t_{client-server\ latency} + t_{connect} \end{cases}$$
  This term measures the time necessary to move a HTTP request from one server to another. This is set to twice the estimated latency of the connection between the server and the client ($t_{client-server\ latency}$) plus the time for a server to set up a connection ($t_{connect}$). The conceptual model is that of a very short reply going back to the client browser, who then automatically issues another request to the new server address. The transfer time is zero if the task is already local to the target server. The estimate of the link latency is available from the TCP/IP implementation, but in the initial implementation is hand-coded into the server.

Given the arrival of HTTP request r at processor x, the scheduler at processor x goes through the following steps.

1. *Preprocess a request* The server parses the HTTP commands, and completes the pathname given, determining appropriate permissions along the way. It also determines whether the requested document exists, has moved, or is a CGI program to execute.
2. *Analyze request* The server then determines whether itself or another server should fulfill the request. It does so by checking the results from the preprocessing phase. If r is already determined to be a redirection, does not exist, or is not a retrieval of information[1], then the request is always completed at x. Then the request is passed to the broker for analysis. The broker then:
   (a) Determines the server on whose local disk the file resides (if any).
   (b) Calculates an estimated time for each available server-node for request r.

   Having determined the estimated time for each server to fill the request, the broker indicates its choice (determined by the minimum time to completion) to the main process.
3. *Redirection* If the chosen server is not x, the request is redirected appropriately.
4. *Fulfillment* At this point the request is processed in the normal HTTP server manner, with any CGI's executed as needed, and any client responses shuttled out the appropriate socket.

It should be noted that modeling the cost associated with processing a HTTP request accurately is not easy. We still need to investigate further the design of such a function. Our experiments show that the current cost function does reflect the impact of multiple parameters on the overall system response performance, and that SWEB delivers acceptable performance based on such a heuristic function, adapting to dynamically-changing system resource loads. In our experiments, we will show that the overhead of SWEB scheduling is insignificant compared to the system load used for request fulfillment and other activities.

3.3 Performance Analysis

We provide an analysis on the maximum sustained rps which are achievable within our schema. We use the following parameters: p is the number of nodes, r^* is the theoretical max sustained rps, F is the average file size requested, b_1 is the bandwidth of local disk accessing, b_2 is the bandwidth of remote disk accessing, d is the average redirection probability, A is overhead in preprocessing a request, O is the overhead of redirection, and H is the average processing time for each request. Then we can show the maximum sustained rps for file fetches is

$$r^* \approx \frac{1}{(\frac{1}{p}+d)\frac{F}{b_1} + (1-\frac{1}{p}-d)\frac{F}{min(b_1,b_2)} + A + d(A+O)}.$$

The full analysis is discussed in [AY95+].

For example, if $b_1 = 5MB/s$ and $b_2 = 4.5MB/s$, $O \approx 0$, $p = 6$, $r^* = 2.88$, then the maximum sustained rps is 17.3 for 6 nodes. We will show this is close to our experimental results in Section 4.

[1] SWEB currently focuses on GET and related commands used in the HTTP protocol. Other commands (e.g., POST) are not handled, but SWEB could be extended to do so in the future.

4 Experimental Results

Our primary experimental testbed consists of a Meiko CS-2 distributed memory machine at UCSB. Each node has a scalar processing unit (a 40Mhz SuperSparc chip) with 32MB of RAM running Solaris 2.3. We mainly use six CS-2 nodes, each of which is connected to a dedicated 1GB hard drive on which the test files reside. Disk service is available to all other nodes via NSF mounts. These nodes are connected via a modified fat-tree network with a peak bandwidth of 40MB/s. Our secondary testbed is a network of 4 SparcStation LX's running Solaris 2.4 connected via standard 10 Mb/s Ethernet. Each LX has a local 525 MB hard drive and 16MB of RAM. The effective bandwidth of this Ethernet is low since it is shared by other UCSB machines.

The complete SWEB system is based on NCSA httpd 1.3 source with modifications to support the distributed functionality. We plan to move to version 1.5 in the immediate future. All software uses the sockets library built on the Solaris TCP/IP streams implementation. The use of the built-in TCP/IP library was a deliberate decision based on several factors, including the following:

1. *Compatibility.* We wanted to use current WWW/HTTP compatible clients for testing purposes, as well as our own programs. Current browsers typically use TCP/IP and sockets or the equivalent.
2. *Portability.* With an implementation built on standard libraries available on virtually any UNIX system, our software can easily be moved to a heterogeneous NOW with no modifications.

There are, of course, disadvantages. The primary disadvantage is certainly performance: we were only able to achieve approximately 5-15% of the peak communication performance on the Meiko, where a DMA transfer handled by the built-in Elan communications co-processor can achieve speeds near the peak bandwidth. We also felt that additional code complexity was caused by not using the built-in messaging libraries such as NX/2 or Active Messages.

We ran a series of tests where a burst of requests would arrive nearly simultaneously, simulating the action of a graphical browser such as Netscape where a number of simultaneous connections are made, one for each graphics image on the page. The clients were primarily situated within UCSB. We also tested SWEB via requests from the East coast of the US (Rutgers University) to examine the benefits of scheduling in a high network latency situation. We concentrate on the UCSB data on the scalability of the system believing this more accurately reflects the high-bandwidth networks we anticipate.

It should be noted that the results we report are average performances by running the same tests multiple times. The test performance is affected by dynamically-changed system loads since the machines are shared by many active users at UCSB.

We report our experiments to examine the performance of our system in the following aspects: scalability of overall performance, scheduling strategy comparison, and overhead costs.

4.1 Scalability of overall performance

Maximum rps on Meiko CS-2 and NOW. The first experiment was run to determine how many requests per second SWEB could process. This depends on the average file sizes requested and the number of nodes. In [KBM94], it is reported that a high-end workstation running NCSA httpd could fulfill approximately 5 rps. We examine how a one-node NCSA httpd 1.3 server performs, and compare it with the 6-node SWEB on Meiko CS-2 and the 4-node SWEB on NOW. The maximum rps is determined by fixing the average file size and increasing the rps until requests start to fail, which indicates that the system limit is reached. The duration of the test which simulates the burst of simultaneous requests also affects the experimental results. The requests coming in a short period can be queued and processed gradually. But the requests continuously generated in a long period cannot be queued without actively processing them since there are new requests coming after each second. We use two types of tests. One is a short period as a duration of 30 seconds and at each second a constant number of requests are launched. The long period has 120 seconds, in order to obtain the sustained maximum rps.

It can be seen from Table 1 that the maximum rps of a single-node server is improved significantly by the multi-node server. The speedups vary for different file sizes. The maximum rps obtained in a short period is much higher than that in a long period because requests accumulated in a short period can be queued. For NOW results with 1.5MB file size, 11 rps is reached for duration of 30s but only 1 is achieved for sustained maximum rps. This is because the maximum disk and Ethernet bandwidth limit is reached for this case. These results have been confirmed via the analysis in Section 3.3, which gave an analytical maximum sustained 17.8 rps for 1.5M files on the Meiko, consistent with the 16 rps achieved in practice. It should be noted that in practice requests come in periodic bursts. Thus we use a 30-second test period in the rest of experiments, representing a non-trivial but limited burst of requests.

Response time and drop rate on Meiko CS-2 and NOW. In Table 2, we report the response time (the time from after the client sends a request until the completion of this request) when we vary the number of server nodes. The system starts to drop requests if the server reaches its rps limit. For small file requests (1K), the multi-node server performs much better than one-node server but the response remains constant when using 2+ processors. This is because none of the theoretical or practical limits on bandwidth or processing power have been reached for small files. The response time does increase to a certain degree after the number of processors exceeds 2, which reflects the overhead required by the scheduling algorithm and distributed file system.

For relatively large files (1.5MB), the processing time is substantially longer. When the number of processors increases, the SWEB provides substantially better performance. These results are consistent with those of NCSA, and also strongly confirm the notion that under heavy loads a distributed server solution can achieve significant speedups. Under especially heavy loads, which would tend to occur during peak hours at popular sites, a single server tends to drop almost half or more of the connections made, whereas a distributed server might have a larger overall average response time and fill every request. The superlinear

	Short period (30s)				*Sustained (120 s)*			
	Meiko		NOW		Meiko		NOW	
file sizes	1K	1.5M	1K	1.5M	1K	1.5M	1K	1.5M
Single server	45	9	24	4	16	2	10	< 1
SWEB	82	45	76	11	45	16	28	1

Table 1: *Maximum rps* for a test duration of 30s and 120s on Meiko CS-2 and NOW.

speedup we obtain reflects the fact that the total size of memory in SWEB is much larger than on a one-node server, and that the multi-node server accommodates more requests within main memory while one-node server spends more time in swapping between memory and the disk. Additionally, the network overhead is distributed among multiple servers rather than concentrated at a single node.

4.2 Comparison between different scheduling strategies

Our scheduling strategy takes into consideration the locality of requested files, and also the current resource loads. We compare our approach with others: one is the NCSA approach that uniformly distributes requests to nodes (round robin); another is to purely exploit the file locality by assigning requests to the nodes that own the requested files. We present experimental results in retrieving a set of files with uniform and non-uniform sizes.

Request with non-uniform file size. With non-uniform file sizes, the load distribution between processors by the initial DNS assignment is heterogeneous. We expect that the round-robin approach cannot adapt such load variations. We tested the ability of the system to handle requests with sizes varying from short, approximately 100 bytes, to relatively long, approximately 1.5MB. Table 3 shows the actual response time in seconds for this case. For lightly loaded systems, SWEB performs comparably with the others. For heavily loaded systems (rps $\geq$ 20), SWEB has an advantage of 15-60% over round robin and file locality.

RPS	8	16	20	24	32
Round Robin	1.5	2.4	5.2	6.9	15.3
File Locality	1.7	2.5	4.4	7.9	9.3
SWEB	1.7	2.6	3.5	5.9	6.2

Table 3: *Performance under non-uniform requests.* 1.5MB file size, 30 sec. duration, 0% drop rate, Meiko CS-2.

It is important to exploit file locality and also consider CPU load. We performed a skewed test to illustrate the fundamental weakness of the file locality heuristic where each client accessed the same file located on a single server, effectively reducing the parallel system to a single server. In this situation, round-robin handily outperforms file locality, with and average response times of 3.7s and 81.4s, respectively. This test was performed with six servers, 8 rps, for 45s, and file size of 1.5MB.

We also ran clients on the east coast (Rutgers University, New Jersey), and the tests results show a performance gain of over 10% using file locality instead of round robin from an Ethernet-linked server, in spite of the poor bandwidth and long latency over the connection from the east coast to the west coast.

Requests with uniform file size. For uniform file sizes, the DNS scheme assigns HTTP requests evenly to nodes, which leads to a homogeneous distribution of CPU load, but the network communication load is not minimized. We expect that the advantages of the re-scheduling will manifest in a high communication cost environment. On Meiko CS-2, we conducted several tests and the result shows that three strategies have similar performance. This is because NFS is implemented in Meiko CS-2 through the fast Elan fat tree network and the chance of network contention is much smaller than on Ethernet. In a relatively slow, bus-type Ethernet in a NOW environment, the advantage of exploiting file locality is more clear. This is verified in an experiment shown in Table 4.

RPS	2	4	8
Round Robin	48.1	128.3	243.7
File locality	42.25	124.2	219.0
SWEB	41.6	124.8	210.4

Table 4: *Performance under uniform requests on NOW.* File size 1.5MB, 0% drop rate.

4.3 Overhead of SWEB

Overhead distribution from the client point of view. SWEB code was instrumented to determine the overhead cost for a request. Table 5 shows the case of a 1.5MB file fetched over a fairly heavily loaded system. The results indicate that the overall overhead introduced by SWEB analysis and scheduling algorithm is insignificant. For small files (approximately 1K), the data transfer time is smaller, but the overhead of SWEB compared to the preprocessing time for parsing HTTP commands is still very small. The direct cost of analysis is typically about 1-4 ms. for cost estimation and 4 ms. to generate a redirection if necessary. Indirect costs which affect the time required to complete a redirection depend heavily on the latency between client and server. For a client fetching a 1.5M file on the Meiko, of the 5.4 sec. total time, well over 90% is spent doing data transfer.

Overhead distribution from the server point of view. From

	Meiko						NOW			
#nodes	1	2	3	4	5	6	1	2	3	4
1K file time	9.8	0.40	0.41	0.44	0.49	0.62	10.9	0.83	0.40	0.26
1K drop rate	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
1.5M file time	98.2	72.5	30.7	23.7	16.5	9.9	> 600	201.4	170.6	167.6
1.5M drop rate	37.3%	5.0%	3.5%	3.5%	3.5%	0.0%	*	20.5%	0%	0%

Table 2: *Performance in terms of response times and drop rates.* On Meiko CS-2, 16 rps, 30 sec. duration. On NOW, 16 rps for 1k file, 8 rps for 1.5M file. The duration is 30s. Time units are in seconds, averaged over all requests. *Single server test timed out after no responses were received.

Activity	*Time*
Preprocessing Req.	70 msec.
Analysis (SWEB)	1 or 4 msec.
Redirection (SWEB)	4 msec.
Data Transfer	4.9 sec.
Network Costs	0.5 sec.
Total Client Time	5.4 sec.

Table 5: *Cost distribution in average response time.* 1.5M file size, Meiko CS-2. Items marked "SWEB" are introduced by the SWEB system. All other times are due to standard HTTPd functionality.

the server's point of view, we need to examine how much load the SWEB contributes for scheduling decisions and resource information collection. Our data shows that in processing requests for files of sizes 1.5MB when 16 rps, 4.4% of CPU cycles are used for parsing the HTML commands, but less than 0.01% time is used for collecting load information and making scheduling decisions. Approximately 0.2% of the available CPU is used for load monitoring. We have also tested the overhead with small files (1K), and the percentage of overhead for load information collection and scheduling again is insignificant. The relative percentages would change as the processing/fulfillment phase decreased with the file size.

5 Conclusions and Future Work

We have implemented a WWW server running on a distributed memory machine and a NOW. We show that our scheduling algorithm effectively monitors and utilizes the multiple system resources, and the overhead required to support such a scheme is extremely low. The resulting performance gains show a decided advantage over other approaches such as round-robin. We are currently migrating the system to be the primary server for the ADL and investigating its performance in a heterogeneous environment [WA95+].

Acknowledgments

This work was supported in part by funding from NSF, ARPA, and NASA under NSF IRI94-11330 and NSF CCR-9409695, and a startup fund from University of California at Santa Barbara. We would like to thank Omer Egecioglu, Terry Smith, Cong Fu and the Alexandria Digital Library team for many valuable discussions and suggestions. We would also like to acknowledge the anonymous referees, and the student researchers who contributed to this project, including Dona Agustin and Hongbing Wang.

References

[A96] The Alexandria Digital Library Project, http://alexandria.sdc.ucsb.edu/

[AC95+] D.Andresen, L.Carver, R.Dolin, C.Fischer, J.Frew, M.Goodchild, O.Ibarra, R.Kothuri, M.Larsgaard, B.Manjunath, D.Nebert, J.Simpson, T.Smith, T.Yang, Q.Zheng, "The WWW Prototype of the Alexandria Digital Library", *Proceedings of ISDL'95: International Symposium on Digital Libraries*, Japan August 22 - 25, 1995.

[AY95+] D.Andresen, T.Yang, V.Holmedahl, O.Ibarra, "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", *Dept. of Computer Science Tech Rpt. TRCS95-17* U.C. Santa Barbara, Sept., 1995, http://www.cs.ucsb.edu/ Research/rapid_sweb/SWEB.html.

[SHK95] B. A. Shirazi, A. R. Hurson, and K. M. Kavi (Eds), Scheduling and Load Balancing in Parallel and Distributed Systems, IEEE CS Press, 1995.

[BR96] E. Brewer, Personal communication, http://inktomi.berkeley.edu, Jan., 1996.

[GDI93] K. Goswami, M. Devarakonda, R. Iyer, Prediction-based Dynamic Load-sharing Heuristics, *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 6, pp. 638-648, June, 1993.

[HL95+] J. Hsieh, M. Lin, J. Liu, T. Ruwart, and D. H.C. Du, Performance of A Mass Storage System for Video-On-Demand, Submitted to *Journal of Parallel and Distributed Computing*, Special issue on Multimedia Processing and Technology.

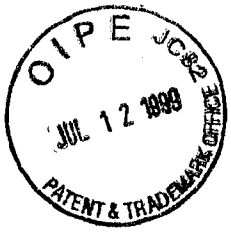
[HT95] Hypertext Transfer Protocol(HTTP): A protocol for networked information, http://www.w3.org/hypertext/WWW/Protocols/, June 26, 1995.

[LYC95] Lycos Usage: Accesses per Day, http://lycos.cs.cmu.edu/ usage-day.html, June 17, 1995.

[KBM94] E.D. Katz, M. Butler, R. McGrath, A Scalable HTTP Server: the NCSA Prototype, *Computer Networks and ISDN Systems*, vol. 27, 1994, pp. 155-164.

[WA95+] R. Wolski, C. Anglano, J. Schopf, F. Berman, Developing Heterogeneous Applications Using Zoom and HeNCE, *Proceedings of the Heterogeneous Computing Workshop, HCW '95*, pp. 12-21, Santa Barbara, CA, IEEE, April, 1995.

Cooperative Caching of Dynamic Content on a Distributed Web Server



Vegard Holmedahl, Ben Smith, Tao Yang
Department of Computer Science
University of California
Santa Barbara, CA 93106
{veho, besmith, tyang}@cs.ucsb.edu

Abstract

In this paper we propose a new method for improving the average response time of Web servers by cooperatively caching the results of requests for dynamic content. The work is motivated by our recent study of access logs from the Alexandria Digital Library server at UCSB, which demonstrates that approximately a 30 percent decrease in average response time could be achieved by caching dynamically generated content. We have developed a distributed Web server called Swala, in which the nodes cooperatively cache the results of CGI requests, and the cache meta-data is stored in a replicated global cache directory. Our experiments show that the single-node performance of Swala without caching is comparable to the Netscape Enterprise server, that considerable speedups are obtained using caching, and that the cache hit ratio is substantially higher with cooperative cache than with stand-alone cache.

1 Introduction

World Wide Web usage has experienced explosive growth in the last few years. In order to accommodate the growth, advances in server technology are needed to improve the response time to the client under heavy load conditions. Typical methods for improving performance are file caching with proxies [5, 6], and load balancing multi-node Web servers [2, 7, 13]. Research shows that for file fetches on the Web, the network is responsible for a significant portion of the response time. Web proxy caching is effective because it reduces the network bottleneck by keeping copies of files closer to clients. We take a different approach to improving Web access performance, recognizing that for some Web sites, processor utilization rather than network bandwidth is the bottleneck. This applies especially to sites making extensive use of requests for dynamic content, such as CGI requests. Our solution is a distributed Web server, called Swala, which cooperatively caches the results of CGI requests.

Our work is motivated, in part, by our experience with the Alexandria Digital Library (ADL) system [1] developed at UCSB. The current ADL system provides on-line browsing and processing of digitized maps and other geospatially mapped data through the Web. The activity of the ADL server includes computation and disk I/O for accessing compressed multi-resolution images with hierarchical data structures, as well as spatial database queries and on-the-fly HTML page generation. The limiting factor for this workload is the CPU rather than the network. Our study of the access logs of the ADL reveals that dynamic requests typically take at least an order of magnitude longer to complete than file fetches. Furthermore, we find that dynamic requests often are repeated, so caching the results of these requests would yield a considerable performance improvement. Our work is applicable to any Web site making extensive use of dynamic requests with long execution times, given a reasonable amount of repetition.

2 Related work

Numerous papers [5, 6] discuss caching of Web content; however, they focus on caching static data and intentionally avoid caching dynamic data. Furthermore, they concentrate on proxy caching rather than server-side caching. Proxy caching of dynamic content may not be feasible because it does not permit caching of authenticated content and it hinders cache replacement algorithms based on job execution time.

In a project currently in progress at UC Berkeley, Vahdat and Anderson propose a framework for monitoring source files, automatically invalidating the result whenever the source changes [16]. An example application that uses the monitoring system is dynamic content caching: they have modified a version of the Apache HTTP server to cache CGI results, and use their monitoring system to invalidate cached entries. Their model does not provide any method for limiting the size of the cache and intelligently replacing

EPIC000466

cache entries. Nor does it permit other methods of invalidation. It also requires file system access on every cache lookup, because they do not maintain an in-memory cache table.

Caching the results of dynamic requests has been studied recently by IBM [12]. They have written a cache server and rewritten their server applications to insert and delete cache items. There are two main drawbacks to this approach. First, they require that a server application be rewritten to take advantage of the cache, and this can be a nontrivial task. Secondly, for every dynamic request, their Web server still must invoke the application, even if it only returns a cache hit. For call mechanisms such as CGI, the operating system overhead for this call is significant, as demonstrated in one of our experiments. Our work overcomes both limitations, since our cache is built into the Web server.

3 Access log analysis

In this section we illustrate the benefits of dynamic request results caching by analyzing an access log from the Alexandria Digital Library (ADL) at UCSB [1]. We have studied its log for September and October 1997, which contains a total of 69,990 requests. After filtering out HEAD and POST requests [4], we have re-sent the requests to the server and timed them. Illegal requests have been removed from the result file before analyzing the statistics, so the total number of requests studied is 69,337, of which 28,663 (41.3%) require execution of a CGI program. The longest running request among those studied is 127.7 seconds. We have found that the total service time for the 69,337 requests is 46,156 seconds, with an average response time of 0.67 seconds. The average response time for a file fetch is 0.03 seconds, and the average response time for a CGI is 1.6 seconds, giving two orders of magnitude difference between the execution of a CGI program and a file fetch. Moreover, the execution of the CGI programs constitutes 97% of the total service time for this set of requests, although the programs account for only 41.3% of the total number of requests. This motivates our further study of the log, since it implies that there is significant potential for a reduction in response time by optimizing CGI requests.

Our analysis results are summarized in Table 1. The first column shows the lower time threshold for requests included in the detailed study. The second column shows the number of requests taking longer than that threshold. The third column shows the total number of requests that were a repeat of a previous request. The fourth column shows the number of entries needed in the cache to exploit all repetition. The fifth column shows the potential time saving by fetching the repeated requests from cache. The sixth column shows the percentage of the total service time that could have been saved by CGI caching. For instance, by assuming that we could cache any CGI request taking more than 1 second to complete, we determined that we would need 189 cache entries to avoid all repeated execution, and that these 189 entries would result in 2,899 cache hits. By adding up the execution time for the requests that could have been cache hits, we found that 13,241 seconds could be saved. With a total service time of 46,156 seconds, this yields a potential reduction in average response time for this workload of almost 29 percent.

Time threshold	#long requests	Total # repeats	# uniq. repeats	Time saved	Saved %
0.5 sec.	15153	7164	364	15957	34.6
1.0 sec.	7581	2899	189	13241	28.7
2.0 sec.	6721	2673	155	12840	27.8
3.0 sec.	6627	2666	151	12812	27.8

Table 1. Potential time saving by caching CGI.

Our analysis of different time thresholds illustrates one of the trade-offs in caching. If we cache too many short requests, we risk having a working set that exceeds our cache size, resulting in cache thrashing and no performance improvement. On the other hand, if we only cache very long requests, we will not realize as much of the benefit of caching. The threshold needs to be selected carefully, based on the system workload. More advanced cache replacement methods can alleviate some of the problem, by keeping the most important requests (in terms of execution time, access frequency, time of access, size etc.) in the cache. For a discussion of the five cache replacement methods implemented in Swala, we refer the reader to [10].

While our log analysis has been limited to a digital library, we expect that other Web sites that make extensive use of dynamic requests also can benefit from dynamic content caching. This is verified by recent work [12].

4 Design of the Swala distributed Web server

Swala is a multi-threaded, distributed Web server that runs on a cluster of workstations and shares cache information and cache data between nodes. The server saves the execution results of CGIprograms and stores information (meta-data) about the cached data in the cache directory. Each node communicates with the others to exchange cache data and meta-data. Multi-threading raises consistency issues, which we discuss in section 4.2. We use memory-mapped I/O whenever possible to minimize the number of system calls and eliminate double-buffering. Multi-threading and memory-mapped I/O are important components of efficient Web servers [11].

4.1 Module design

As illustrated in Figure 1, every Swala node contains two primary runtime modules:

The *HTTP module* starts and shuts down all other threads. During startup, it initializes the thread-pool responsible for handling HTTP requests (hereafter called *request threads*). Thus, this module would comprise the entire Web server if we did not perform caching. The request threads in the HTTP module take turns listening on the main port for incoming connections and handling the requests. After receiving a new connection, the request thread is responsible for the request from parsing to completion.

The *cache module* maintains the local cache directory (which contains information about all entries cached on all nodes), and provides information to request threads. It contains three daemon threads. The first thread receives information about cache insertions and deletions from the other nodes, and updates the local directory. The second thread listens for cache data requests from the other nodes and starts a separate thread for each request to return the cache contents. The third thread wakes up every few seconds and deletes expired cache entries.

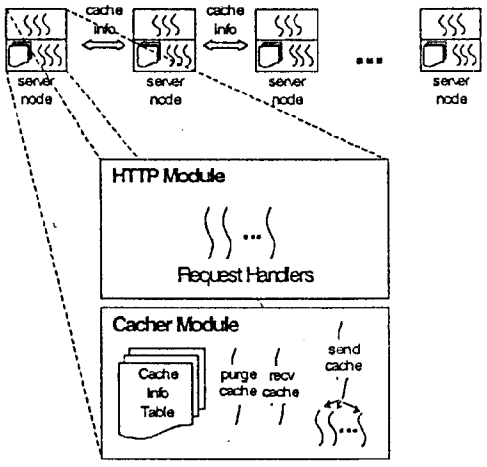


Figure 1. Components of the Swala architecture.

Our cache only contains the results of CGI requests; files are not cached. There are two reasons for this. First, our access log analysis in Section 3 shows that execution of dynamic requests generally takes orders of magnitude more time than file fetches. Secondly, previous work on Web file caching [5] has determined that for file requests, the network is the bottleneck, so file caching should occur as close to the client as possible, i.e. at a proxy server rather than at the base Web server.

We store only the cache directory in main memory, and use a separate operating system file to store the results of each cached request. Thus, every cache fetch in effect becomes a file fetch. There are two reasons why we have chosen this design. A disk cache is simple to maintain, and can contain orders of magnitude more cache data than a memory cache for the same price. Furthermore, our implementation runs on a UNIX platform, which does extensive file system caching, so we expect any recently used, reasonably-sized file to be available in memory. Even if the file is not in the disk cache, fetching from disk will be at least an order of magnitude faster than re-executing the CGI.

Not all CGI requests can or should be cached. For example, CGI scripts that return different results for different users should not be cached. Swala uses a configuration file, loaded at startup, to provide the system administrator with a flexible way to control which requests are cache-able. Thus, on an incoming request, the request thread asks the cache manager to analyze it and determine whether it is (1) uncacheable, (2) cache-able but not cached, or (3) cached. The following explains the action of the request thread in each case:

An *uncacheable* request is executed without any more communication with the cache manager.

If the cache manager determines that a request is *cacheable but not yet cached*, it prepares a file to receive the request results and hands the file handle to the request thread. The CGI is executed, and its results are written to the cache file and returned to the user. The CGI return value is checked. If the execution has been successful and the execution time is longer than a runtime-defined limit, the request handler asks the cache manager to add the information to the directory and broadcast it to the other nodes in the group. Otherwise, the cache manager is asked to discard the results of the request.

A *cached* request is fetched from the (local or remote) cache in which it resides, and returned to the client. After a cache fetch, the cache manager on the node that owns the item updates meta-data statistics, to provide information used during cache replacement.

Figure 2 illustrates the control flow for a CGI request on the Swala server. Note that the Figure is considerably simplified compared with the actual algorithm of request execution; including all the possible error conditions would render the flowchart unreadable.

4.2 Cache Consistency Issues

We need to address two types of consistency in our Web server design: content consistency and cache table consistency. *Content consistency* is concerned with returning the most recent copy (or result) of a document or script. This issue is similar to cache invalidation in a shared memory sys-

EPIC000468

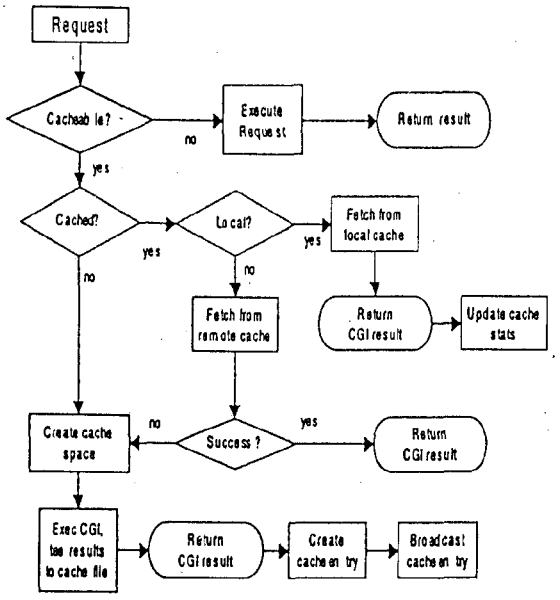


Figure 2. Swala control flow for handling CGI requests.

tem. In the context of caching dynamic content, strong content consistency requires that if the CGI is to execute again, the new result is identical to the cached result. Weaker consistency relaxes this requirement, allowing access to "old" data for a period of time. The second consistency issue is a result of the distributed, multi-threaded design of our system. We call this *cache table consistency*, and it is concerned with avoiding concurrent updates (which might cause data corruption), as well as maintaining consistent information in all nodes' cache directories.

We implement a relatively weak consistency protocol for content consistency by allowing the system administrator to set a Time To Live (TTL) field for different CGIs. With this value set, the cached results will expire after a predetermined interval. Our scheme is reasonable for Web sites with infrequent updates of data and CGI scripts. This is true in digital library applications, where the material available through the Web server normally is read-only. For Web sites that need stronger content consistency, we plan to investigate other cache entry invalidation methods in future versions of Swala, for example by receiving invalidation messages from applications [12], or by monitoring the input of the CGI programs whose output is being cached, to detect invalidation [16].

We address cache table consistency with a two-level consistency protocol: intra-server and inter-server consistency. In the following section we describe our cache table consistency protocol.

Our *intra-node consistency protocol* protects the internal cache directory against simultaneous updates, avoiding potential data corruption. Every node contains a directory with one table for each node in the group, and each table contains information about what is cached at the corresponding node. There are three possible levels of locking that can be used to prevent data corruption in the cache directory: we could lock the whole directory for each access, lock only a table at a time, or lock each individual cache entry. (A fourth option is multi-granularity locking, for instance using entry locks on one table while using table lock on the other tables. We have not considered multi-granularity locking in this implementation.) We implement locking at the table level, with read- and write-locks to protect the cache table, in order to minimize lock contention while maximizing scalability. The process involved in satisfying a typical cache-able request motivates our choice of locking strategy: for every cache-able request received, we check the cache table for each node to see if any node has the request cached. If there were only one lock for the whole directory, we would experience unacceptable lock contention. If we used the third option, a typical cache lookup would involve a significant number of locks and unlocks, since the lookup would search a portion of each table in a directory. This would in turn seriously hamper scalability, since every added server would increase the number of locks & unlocks on lookup by the cache size.

With our *inter-node consistency protocol*, we are forced to compromise between strong consistency and minimal overhead. For instance, a strong inter-node protocol which ensures consistency between the cache directories on all nodes at all times would be quite expensive, since it would require a variation of a two-phase commit protocol between the nodes with synchronization on every cache update. Our choice of consistency protocol is motivated by a desire to maximize scalability and minimize critical path delay. This motivation has resulted in a relatively weak consistency protocol in the sense that cache updates are done asynchronously among the nodes without any global locks. More specifically, when a cache entry is inserted or deleted on a node, this information is broadcast to the other nodes, and they update their directories accordingly.

With the protocol described above, it is possible that a node may redo a cached request or attempt to fetch a deleted cache entry. We explain these situations below.

We define it as a *false cache miss* when the node reruns a CGI program instead of fetching the results from the cache. There are two situations when this could happen. If a node receives an uncached request, and an identical request arrives at the same node again before it has

EPIC000469

completed the first request, the node will redo the request, rather than wait for the cached results of the first request. The second situation where a false cache miss could occur is caused by our loose inter-node consistency protocol. If a node called A caches a request, and another node called B receives an identical request before it receives the cache update information from A, node B will redo the request instead of fetching the cached data from A. In this situation, the same information will be cached at two nodes. Both situations will occur rarely, since it is highly improbable that two identical requests will arrive within the relatively small time window that it takes to execute the CGI.

We define it as a *false cache hit* when a node requests a cache entry that has been deleted from the cache of a remote node. This happens if node A asks node B for an entry after B deleted it and before A was aware of the deletion. The time interval in which this can happen is quite small, since the latency between the nodes is expected to be low. Furthermore, the cost of a cache miss on remote cache is small: when node A receives the cache miss response, it will execute the CGI request locally, as shown in Figure 2. The client will only experience the added delay of a request/reply session between the two nodes, and this delay will in most cases be more than an order of magnitude smaller than the execution time of the request.

5 Experiments

We have implemented and tested the Swala distributed Web server in a Unix environment, using TCP/IP communication between the nodes. Our testbed consists of six Sun 143-MHz Ultra 1 and two Sun Ultra 2 (dual 167-MHz CPU) workstations, with 64 or 128 MB of memory, connected by a fast (100 Mbit) Ethernet. To avoid the impact of using multiple processors during speedup experiments, we use only one CPU on the Ultra 2 nodes during the tests. Thus, the CPU power is roughly equivalent on all nodes. We present three sets of experiments. The first set compares the single-node performance of Swala with two popular Web servers and examines the overhead and benefit of caching on a single node. The second set of experiments concerns scalability; we examine the performance on an increasing number of nodes and study the benefit and overhead of caching in the clustered environment. The last set examines the cache hit ratio in Swala and illustrates the advantage of cooperative caching over stand-alone caching. For all these experiments, clients are located within the campus network to avoid Internet bandwidth fluctuations.

5.1 Single-node performance of Swala and overhead of cache fetch

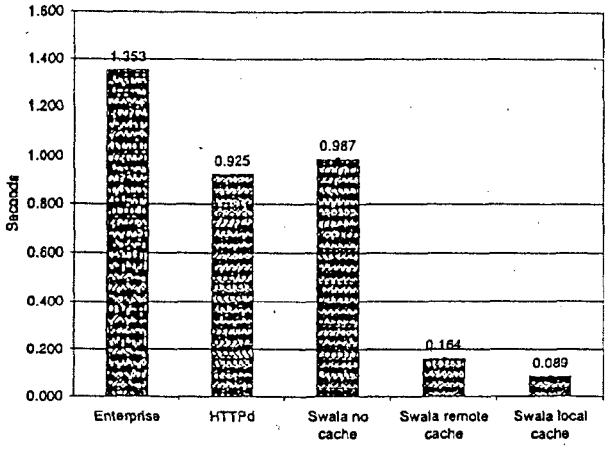
We present the results of a series of experiments comparing Swala with NCSA's HTTPd Version 1.5.1 [15] and Netscape Corporation's Enterprise Server [14]. The purpose of these experiments is twofold. We compare the efficiency of Swala with popular research and commercial servers, for a workload consisting of file fetches and CGI execution. We also measure the overhead of a local and remote cache fetch imposed by our cache system on a cached request. We use NCSA HTTPd because it is a widely used research package, and we use Netscape Enterprise because it is one of the most efficient commercial Web servers available [11]. We use WebStone [17], a standard Web benchmarking tool, for the experiments in this sub-section.

# clients	HTTPd	Enterprise	Swala
6	0.207	0.032	0.058
12	0.410	0.068	0.073
18	0.604	0.105	0.099
24	0.838	0.140	0.133
30	1.146	0.174	0.164
36	1.304	0.212	0.199
42	1.709	0.248	0.235

Table 2. File fetch average response time in seconds measured using WebStone.

The first set of experiments is limited to file fetches: a 500 byte file is requested 35% of the time; a 5 Kb file is requested 50%; a 50Kb file is requested 14%; a 500Kb file is requested 0.9%, and a 1Mb file is requested 0.1% of the time. The average response times of Swala, Netscape and HTTPd are summarized in Table 2. The results show that the average response time of Swala is between 2 and 7 times faster than HTTPd. One reason for HTTPd's low performance is that it uses processes rather than threads. Enterprise is slightly faster than Swala on a small number of clients and slightly slower on a higher numbers of clients.

We further compare the performance of Netscape, HTTPd and Swala for CGI program execution. For these experiments, we use WebStone on three client machines, with 24 client processes sending the same request: a CGI program called `nullcgi` that does no work and produces less than a hundred bytes of output. This enables us to measure the overhead of CGI program execution, and compare it with the overhead of a cache fetch. For the Enterprise, HTTPd, Swala no-cache, and Swala local-fetch experiments, we run the Web server on a single node. For the Swala remote-fetch test, we start Swala on two nodes.



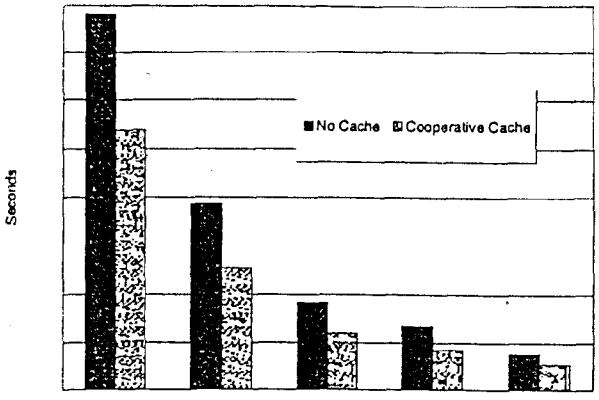


Figure 4. Multi-node performance of Swala with and without caching.

tion 3. The workload therefore contains the same number of repeats and the same amount of temporal locality as the original log. In this experiment, each of two clients starts eight threads, and every thread launches requests to a single server node. Figure 4 shows the average response times of Swala with and without caching, where the number of server nodes varies from one to eight. The results show that Swala with cooperative caching enabled yields a much lower average response time than Swala with caching disabled. For instance, the decrease in response time with caching on eight nodes is about 25%. The results also show that the performance of Swala scales well when the number of nodes increases. With eight nodes, the average speedup is about nine.

We have also measured the overhead of cooperative cache usage. We want to ensure that our handling of a cache miss followed by insertion of a new entry, broadcast of the cache information to the other nodes, and the replicated cache directory update is efficient enough not to destroy the advantage offered by cooperative caching. We have created a workload where each request is unique and cacheable. This forces every new request to generate a cache miss and a cache insert. We perform two different experiments to measure the overhead of cooperative cache maintenance. The first experiments measures the cost of local insert and broadcast, while the second experiment quantifies the cost of maintaining the global replicated cache directory. In the first experiment, we start Swala on two through eight servers, in non-caching and cooperative caching mode. We send 180 requests, each of which will run for one second on an unloaded CPU, to one of the nodes in the group, and the response time from this node is measured. As table 3

EPIC000471

shows, the cache miss and insert overhead is insignificant and independent of the number of server nodes. We therefore conclude that the overhead of cache insert and cache information broadcast is negligible.

# nodes	No Cache (s)	Coop. Cache (s)	Increase (s)
2	0.965	0.992	0.027
4	0.963	0.995	0.032
6	0.961	0.994	0.033
8	0.967	0.993	0.026

Table 3. Response time overhead of cache insertion and cache information broadcast.

In preparation of the second overhead experiment, we have created a program which only sends cache directory updates to a Swala node. This enables us to simulate a complete eight-node Swala execution with minimal network disturbance: we start Swala on only one node, telling it that other nodes are running (to prepare its cache directory for cache information from eight nodes); we start the pseudo-server program to act as the other seven nodes. The Swala node receives 180 uncacheable requests during the test, and each request takes approximately one second on an unloaded CPU. In our tests, cache information update messages are sent from the pseudo-server program, imposing the extra load of updating the local cache directory. The average response time for different levels of updates, with the base case as the first row, is reported in Table 4. The first column lists the number of updates per second (UPS) that the node receives in addition to its own work on uncacheable requests. The second column gives the average response time on requests for this rate of UPS, and the third column shows the increase in response time in seconds from the base case. As the table shows, the increase in response time on the one-second requests is insignificant.

UPS	Avg. Response time (s)	Increase (s)
0	0.958	0.000
10	0.970	0.012
20	0.976	0.018
30	0.976	0.018
40	0.990	0.032
50	1.005	0.047
60	1.018	0.060

Table 4. Response time overhead of replicated cache directory maintenance.

5.3 A comparison between cooperative and stand-alone caching

Our third set of experiments isolates and measures the advantage of sharing cache data. We have used the analysis from Section 2 to determine the theoretical upper bound on cache hits for the requests issued: by counting the exact number of unique requests and repeats, we know how many cache hits are theoretically possible on a cache of infinite size. This upper bound on cache hits is compared with the actual number of cache hits on stand-alone caching and cooperative caching. The results are summarized in the two tables below. During each of these tests, 1600 requests are issued, 1122 of which are unique. In the cooperative tests, the nodes constituting the server are aware of each other and exchange cache information and cache data, while in the stand-alone test, each node caches what it receives and is unaware of any other node. Thus, a request may be cached several times in the stand-alone tests, whereas in the cooperative cache test, no cache entry is repeated (except for false cache misses). As the experiments show, the ability to reuse another node's cache entry rather than re-executing the script accounts for a large portion of the advantage of cooperative caching. The first column of tables 5 and 6 shows the number of nodes used. The second and third columns show the number of cache hits achieved for stand-alone and cooperative caching respectively. The fourth and fifth columns show how close to optimal caching this group of nodes comes.

The difference between the two tests is the cache size. In Table 5, the cache size is 2000, so that even a single node can cache every request, if necessary. Thus the only advantage that the cooperative cache has is that once *one* node has cached a request, no one else will have to execute it again (barring false cache misses). The results in Table 5 show that cooperative caching reaches a near-optimal performance. Even though each individual node has sufficient cache space for all requests, cooperative caching still substantially outperforms stand-alone caching in terms of cache hit ratios.

	# Cache hits		% of Upper Bound	
# nodes	Stand.	Coop.	Stand. %	Coop. %
1	466	N/A	97.5%	N/A
2	423	475	88.5%	99.4%
4	383	466	80.1%	97.5%
6	352	473	73.6%	99.0%
8	327	473	68.4%	99.0%

Table 5. Cache hit ratios, stand-alone and cooperative caching, cache size 2000.

Table 6 presents the results of a test with cache size 20. The consequence of the small cache size is that stand-alone caching (and cooperative caching with few nodes) is going to suffer considerably from cache overflow, having to continually replace entries in the cache. The table shows that even with a (cooperatively combined) cache size smaller than 14% of the unique cacheable requests received (on 8 nodes), the cooperative cache achieves over 70% of the upper bound of cache hits, while the stand-alone cache configuration achieves less than 40%.

	# Cache hits		% of Upper Bound	
# nodes	Stand.	Coop.	Stand. %	Coop. %
1	137	N/A	28.7%	N/A
2	140	208	29.3%	43.5%
4	167	283	34.9%	59.2%
6	177	320	37.0%	66.9%
8	183	352	38.3%	73.6%

Table 6. Cache hit ratios, stand-alone and cooperative caching, cache size 20.

6 Conclusions

In this paper, we have discussed a new approach to improving Web response time by cooperatively caching CGI results on a cluster of workstations. We have analyzed the access log from a digital library Web site and found significant potential for time saving through caching of dynamic content. Our experimental results demonstrate that the single-node performance of Swala is competitive with commonly used Web servers, the overhead of cooperative cache management is relatively small, and cooperative caching of CGI results substantially improves the average client response time.

Acknowledgments

We are grateful for valuable suggestions from Anurag Acharya and Ambuj Singh at UCSB and Michael Quinn at Oregon State University. We thank the anonymous referees for their comments, the members of the Parallel Systems Laboratory at UCSB for their patience during our intensive Web server tests, and Dan Andresen for collaborative research on the early versions of the Web server (the SWEB project). This work was supported in part by the Norwegian State Educational Loan Fund, NSF IRI94-11330, NSF CCR-9702640, NSF CDA-9529418 and UC MICRO/SUN.

References

[1] D. Andresen, L. Carver, R. Dolin, C. Fischer, J. Frew, M. Goodchild, O. Ibarra, R. Kothuri, M. Larsgaard, B. Manjunath, D. Nebert, J. Simpson, T. Smith, T. Yang, Q. Zheng, "The WWW Prototype of the Alexandria Digital Library", *Proceedings of ISDL'95: International Symposium on Digital Libraries*, 1995.

[2] D. Andresen, T. Yang, V. Holmedahl, O. Ibarra, "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", *Proc. of 10th IEEE International Symp. on Parallel Processing (IPPS'96)*, pp. 850-856. April, 1996.

[3] D. Andresen, T. Yang, O. Egecioglu, O. Ibarra, T. Smith, "Scalability Issues for High Performance Digital Libraries on the World Wide Web", Proc. of the 3rd IEEE ADL96 (Advances in Digital Libraries), pp. 139-148, 1996.

[4] T. Berners-Lee, R. Fielding, and H. Frystyk, Hypertext Transfer Protocol – HTTP/1.0, RFC 1945, HTTP Working Group, May, 1996.

[5] P. Cao, S. Irani, "Cost-Aware WWW Proxy Caching Algorithms", Proc. of the USENIX Symposium on Internet Technologies and Systems, pp. 193-206, December, 1997.

[6] A. Chankhunthod, P. Danzig, C. Neerdaels, M. Schwartz and K. Worrell, "A Hierarchical Internet Object Cache", Technical Report 95-611, Computer Science Department, University of Southern California, Los Angeles, California, March 1995.

[7] D. Dias, W. Kish, R. Mukherjee, R. Tewari, "A Scalable and Highly Available Web Server", Proc. of COMPCON 1996, Forty-First IEEE Computer Society International Conference: Technologies for the Information Superhighway, Santa Clara, California, February, 1996.

[8] S. Gadde, M. Rabinovich, J. Chase, "Reduce, Reuse, Recycle: An Approach to Building Large Internet Caches", Workshop on Hot Topics in Operating Systems (HotOS), May 1997.

[9] James Gwertzman, Margo Seltzer, "World Wide Web Cache Consistency," Proceedings of the 1996 USENIX Technical Conference, San Diego, CA, Jan 1996.

[10] V. Holmedahl, B. Smith, T. Yang, "Cooperative caching of dynamic content on a distributed Web server," University of California at Santa Barbara technical report #TRCS98-12.

[11] J. Hu, S. Mungee, D. Schmidt, "Techniques for Developing and Measuring High-performance Web Servers over ATM Networks," Washington University technical report #WUCS-97-09.

[12] A. Iyengar, J. Challenger, "Improving Web Server Performance by Caching Dynamic Data", Proc. of the USENIX Symposium on Internet Technologies and Systems, pp. 49-60, December, 1997.

[13] E.D. Katz, M. Butler, R. McGrath, A Scalable HTTP Server: the NCSA Prototype, *Computer Networks and ISDN Systems*. vol. 27, 1994, pp. 155-164.

[14] "Netscape Server Central Index Page", http://home.netscape.com/comprod/server_central/.

[15] "The NCSA HTTPd Home Page", http://hoohoo.ncsa.uiuc.edu/.

[16] A. Vahdat, T. Anderson, "Transparent Result Caching," draft accepted for publication in USENIX '98, http://now.cs.berkeley.edu/WebOS/papers/trec.ps.

[17] "WebStone", http://www.sgi.com/Products/WebFORCE/WebStone/index.html.

OIPE JUL 12 1999 PATENT & TRADEMARK OFFICE

deja.com
Get more out of Deja.com
TO REGISTER FOR my deja
Explore by clicking here
Share what you know. Learn what you don't.

message

Message 1 of 1 help

NeXT Ships WebObjects -- On Time -- As Promised

Author: Nicole Overson <Nicole_Overson@next.com>
Date: 1996/03/28
Forum: comp.sys.next.announce

more headers author posting history

FOR IMMEDIATE RELEASE

Contact:
Nicole Overson
NeXT Software, Inc.
415-780-3731
nicole_overson@next.com

NeXT SOFTWARE SHIPS WebObjects PRODUCT LINE

Cross-Platform, Web Application Development Environment to be Distributed to Customers Today Via the Web

Redwood City, Calif.-March 28, 1996-NeXT Software, a leading producer of software development environments for the Internet and Enterprise, announced that the WebObjects product family release 1.0 ships to customers across the globe today right on schedule. Already over 15,000 copies of the free, entry-level beta product have been downloaded off NeXT's website at http://www.next.com. Creating further momentum for the product, NeXT's WebConstructor Group, an innovative consulting group leveraging WebObjects technology, reports that more than 15 consulting projects have been started for leading organizations since its inception less than two months ago.

The WebObjects product family offers a truly cross-platform and language independent development environment for building interactive server-based web applications for both the Internet and the Enterprise. NeXT's WebObjects product suite is designed to meet a variety of business needs- WebObjects, the free entry-level product, allows organizations to easily start building dynamic websites, WebObjects Pro gives developers the ability to compile reusable components, scale across multiple machines, and access databases, and WebObjects Enterprise brings a corporation's entire computing infrastructure to the Web.

"We're very impressed with the functionality WebObjects brings to corporations building dynamic, web-based applications," said Jonathan Schwartz, president of Lighthouse Design. "In less than two months of working with WebObjects, Lighthouse has already delivered

WebVision, a server-based developer component designed to augment NeXT's technology, delivering data-driven graphics to WebObjects developers."

"Demonstrating the cross-platform interest, over half of all the 15,000 WebObjects downloads have been for the Windows NT platform while the other half is split by versions for SunSoft Solaris and NeXT's NEXTSTEP operating systems", said Linda Siener, director of software product marketing, NeXT Software.

Additionally, NeXT's WebConstructors report over 15 contracts signed by customers to date. Comprised of a highly skilled and creative team of developers, the WebConstructor group is dedicated to building web applications that incorporate a corporation's existing code base and legacy database information. The goals of the team are to construct these applications quickly while bringing innovation and creativity to the development process and providing proof of concept.

WebObjects is Open, Open, Open
WebObjects is designed to work with existing infrastructure enabling corporations to expand the reach of their systems to the Web without having to rewrite data and applications.

* WebObjects is operating system-independent spanning both UNIX and Windows environments including Microsoft's Windows NT, SunSoft Solaris, and NeXT's NEXTSTEP with more coming soon.

* WebObjects works with industry standard data sources such as Oracle, Informix, and Sybase databases and, through third party software, can support mainframe and legacy applications.

* WebObjects is browser independent working with standard browsers such as the Netscape Navigator, Microsoft Internet Explorer, and Mosaic.

* WebObjects is HTTP server independent, OLE compatible, takes advantage of native server APIs, such as Netscape's NSAPI, and can use existing programming languages such as PERL, C, and C++ languages.

WebObjects Supports Current and Emerging Web Technologies
WebObjects offers highly interactive, dynamic web applications and is designed to interoperate with emerging technologies such as VRML, Shockwave, Real Audio, Java, and JavaScript standards. WebObjects allows you to reach people with real-time information about flights, personal banking accounts, and product demonstrations.

WebObjects Quickly Delivers Interactive Web Solutions
WebObjects allows web developers to deliver dynamic web applications faster than with traditional development tools. Customers can reuse components and can modify several applications at once by editing a single component. This environment translates into a large competitive advantage that leads directly to increased revenues and greater marketshare.

WebObjects also enables corporations to scale and manage their websites. Webmasters simply plug in additional resources without having to take their machines and applications off line-this is critical for corporations experiencing "grid lock" on their site due to increased traffic. WebObjects facilitates this by managing incoming HTTP requests and distributing web application processes across multiple machines.

WebObjects Covers Security

WebObjects offers a secure solution for corporations building commerce-enabled applications for the Web. NeXT has leveraged existing security standards such as Netscape's SSL on the HTTP server. NeXT also provides authentication services via Security Dynamics' SecurID token cards, offering enterprise network security through one-time password identity verification.

The WebObjects Product Family

WebObjects
The base product, WebObjects, offers an easy entry point for building dynamic Web-based applications. With WebScript you can easily build interactive applications using the pre-built components supplied with the product. This free download, is designed to introduce developers to NeXT's powerful environment and includes the Dodge Virtual Showroom application, allowing Web developers to get started today.

The WebObjects is available today in its production release 1.0 for free download via NeXT's website at http://www.next.com. Future components and upgrades will also be available via the Web.

WebObjects Pro
WebObjects Pro answers the needs of Web developers wanting to move beyond script-based development to build more complex applications. The WebObjects Pro version includes the ability to incorporate C, C++, and Objective C code into Web-based applications and, with OLE connectivity and database client library support, the Pro product enables developers to access OLE objects and relational databases. WebObjects Pro also provides for distributed Web application servers allowing organizations to add more power as their system grows.

WebObjects Pro is available for $2,999 through NeXT's telesales organization who can be reached at 1-800-TRY-NeXT.

WebObjects Enterprise
WebObjects Enterprise is NeXT's complete dynamic Web development solution. Targeted to customers who want a solution that encapsulates their existing data and applications, the Enterprise solution allows corporations to preserve their technology investment while building new internal and external Web applications.

WebObjects Enterprise is sold through NeXT's direct salesforce for prices starting at $24,999. Contact NeXT at 1-800-TRY-NeXT to help you determine what you need and how to get started.

Distribution
In addition to CD, the WebObjects products are available via on-line distribution.

NeXT Software, Inc.

NeXT delivers an open development environment for building custom applications for both the World Wide Web and the Enterprise. NeXT's development software adds value to today's business by shortening the development cycle, working with an organization's existing technology, supporting current and emerging standards, and enabling organizations to grow and change with ease.

Providing its customers with a combination of award-winning software and professional services, NeXT delivers real-world solutions.

NeXT Software, Inc. is headquartered in Redwood City, California, and has offices in North America, London, Paris, Munich, and Tokyo.

For more information please visit NeXT's Web site at http://www.next.com
or call 1-800-TRY-NeXT.

Trademark Info:
NeXT, the NeXT logo, WebObjects, Enterprise Objects Framework, and NEXTSTEP are trademarks or registered trademarks of NeXT Software, Inc. All other trademarks mentioned belong to their respective owners.

###

---

view for bookmarking
text only

mail this message to a friend
Sponsored by Fatbrain.com (*)

post reply << prev - next >>

subscribe to comp.sys.next.announce

Arts & Entertainment Automotive Computing & Tech Health
Money News Recreation People Sports Travel

SHOPPING - Yellow Pages - Long Distance Deals - Free Stuff - Trade with Datek - Go to Gigabuys! - GET IT NOW @ NECX - FREE downloads! - Get FREE Health Info@drkoop.com - Apartments.com - eBay Auctions

Copyright © 1995-99 Deja.com, Inc. All rights reserved. Conditions of use.
Site privacy statement reviewed by TRUSTe.

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
09/234,048	01/19/99	LOWERY	K 02577.P001D

EXAMINER	
PERVEEN, R	
ART UNIT	PAPER NUMBER
2182	7

TM21/0223

JAMES H SALTER
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
7TH FLOOR
LOS ANGELES CA 90025

DATE MAILED: 02/23/01

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)

1- File Copy

U.S. G.P.O. 2000 ; 465-188/25268

Office Action Summary	Application No. 09/234,048	Applicant(s) LOWERY et al
	Examiner Rehana Perveen	Group Art Unit 2182

☒ Responsive to communication(s) filed on *Jul 12, 1999*.

☐ This action is **FINAL**.

☐ Since this application is in condition for allowance except for formal matters, **prosecution as to the merits is closed** in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11; 453 O.G. 213.

A shortened statutory period for response to this action is set to expire *3* month(s), or thirty days, whichever is longer, from the mailing date of this communication. Failure to respond within the period for response will cause the application to become abandoned. (35 U.S.C. § 133). Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

Disposition of Claims

☒ Claim(s) *17-45* is/are pending in the application.

Of the above, claim(s) _____ is/are withdrawn from consideration.

☐ Claim(s) _____ is/are allowed.

☒ Claim(s) *17-45* is/are rejected.

☐ Claim(s) _____ is/are objected to.

☐ Claims _____ are subject to restriction or election requirement.

Application Papers

☒ See the attached Notice of Draftsperson's Patent Drawing Review, PTO-948.

☐ The drawing(s) filed on _____ is/are objected to by the Examiner.

☐ The proposed drawing correction, filed on _____ is ☐approved ☐disapproved.

☐ The specification is objected to by the Examiner.

☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

☐ All ☐ Some* ☐ None of the CERTIFIED copies of the priority documents have been

☐ received.

☐ received in Application No. (Series Code/Serial Number) _____.

☐ received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

*Certified copies not received: _____.

☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

☒ Notice of References Cited, PTO-892

☒ Information Disclosure Statement(s), PTO-1449, Paper No(s). *2, 3, & 6*

☐ Interview Summary, PTO-413

☒ Notice of Draftsperson's Patent Drawing Review, PTO-948

☐ Notice of Informal Patent Application, PTO-152

THOMAS LEE
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

--- SEE OFFICE ACTION ON THE FOLLOWING PAGES ---

Part III DETAILED ACTION

1. Claims 17-45 are presented for examination.

Specification

2. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

Double Patenting

3. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. See *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970);and, *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321© may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent is shown to be commonly owned with this application. See 37 CFR 1.130(b).

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

4. Claims 17-45 are rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1-11 of U.S. Patent No. 5,894,554. Although the conflicting claims are not identical, they are not patentably distinct from each other.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless --

(e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371© of this title before the invention thereof by the applicant for patent.

6. Claims 17-45 are rejected under 35 U.S.C. 102(e) as being anticipated by Leaf, patent no. 5,754,772.

7. As to claim 17, Leaf teaches routing a request from a Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, intercepting the request at the Web server and routing the request to the page server, processing the request by the page server while the Web server concurrently

processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources (col. 2 lines 30-60 and col. 4 line 10 - col. 5 line 50).

8. As to claim 18, Leaf teaches routing the request from the Web server to a dispatcher (Transaction Gateway Client), and dispatching the request to the page server (Transaction processing System 26) (col. 4 lines 55-67).

9. As to claims 19 and 20, Leaf teaches identifying said one or more data sources and dynamically retrieving data from the one or more data sources (col. 4 lines 55-67).

10. As to claim 21, Leaf, inherently, teaches maintaining a connection cache to one or more data sources.

11. As to claim 22, Leaf teaches logging into the one or more data sources (inherent, col. 4 lines 10-67).

12. As to claim 23, Leaf, inherently, teaches maintaining a page cache containing the Web page.

13. As to claims 24-26, Leaf teaches tag-based text templates for configuring the Web page, inserting the dynamically retrieved data from the one or more data sources into the tag-based text templates, at least one of said tag-based text templates drives a format of the data dynamically retrieved from the one or more data sources in response to the request (col. 4 line 10 - col. 5 line 50).

14. As to claim 27, Leaf teaches the tag-based text templates include HTML templates (col. 4 lines 55-67).

15. As to claim 28, Leaf, inherently, teaches dynamically updating data at the one or more data sources.

16. As to claims 29 and 30, Leaf teaches processing an object handling extension, the object handling extension being an OLE extension (col. 4 line 10 - col. 5 line 45).

17. Claims 31-45 are different variations of claims 17-30, and therefore, are rejected under the same rationale.

18. Further references of interest are cited on Form PTO-892 which is an attachment to this office action.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks
Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry)

Or:

(703) 306-5404 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington, VA., Sixth Floor (Receptionist).

19. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Rehana Perveen, whose telephone number is (703) 305-8476. The examiner can normally be reached Monday through Friday from 8:00 AM - 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thomas C. Lee, can be reached at (703) 305-9717. The fax phone number for this Group is (703) 306-5404.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Rehana Perveen
February 18, 2001

Notice of References Cited

Application No.	Applicant(s)		
09/234,048	LOWERY et al		
Examiner	Group Art Unit	Page 1 of 1	
Rehana Perveen	2182		

U.S. PATENT DOCUMENTS

	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
A	5,761,673	6/1998	Bookman et al	707	104
B	5,754,772	5/1998	Leaf	395	200.33
C	5,404,527	4/1995	Irwin et al	395	700
D					
E					
F					
G					
H					
I					
J					
K					
L					
M					

FOREIGN PATENT DOCUMENTS

	DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
N						
O						
P						
Q						
R						
S						
T						

NON-PATENT DOCUMENTS

	DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
U		
V		
W		
X		

Form PTO 948 (Rev. 8-98) U.S. DEPARTMENT OF COMMERCE - Patent and Trademark Office Application No. 09/234048

NOTICE OF DRAFTSPERSON'S PATENT DRAWING REVIEW

The drawing(s) filed (insert date) 1/19/99 are:
A. ☐ approved by the Draftsperson under 37 CFR 1.84 or 1.152.
B. ☒ objected to by the Draftsperson under 37 CFR 1.84 or 1.152 for the reasons indicated below. The Examiner will require submission of new, corrected drawings when necessary. Corrected drawing must be sumitted according to the instructions on the back of this notice.

1. DRAWINGS. 37 CFR 1.84(a): Acceptable categories of drawings: Black ink. Color.
   ___ Color drawings are not acceptable until petiton is granted. Fig(s) _____
   ___ Pencil and non black ink not permitted. Fig(s) _____
2. PHOTOGRAPHS. 37 CFR 1.84 (b)
   ___ 1 full-tone set is required. Fig(s) _____
   ___ Photographs not properly mounted (must use brystol board or photographic double-weight paper). Fig(s) _____
   ___ Poor quality (half-tone). Fig(s) _____
3. TYPE OF PAPER. 37 CFR 1.84(e)
   ___ Paper not flexible, strong, white, and durable. Fig(s) _____
   ___ Erasures, alterations, overwritings, interlineations, folds, copy machine marks not accepted. Fig(s) _____
   ___ Mylar, velum paper is not acceptable (too thin). Fig(s) _____
4. SIZE OF PAPER. 37 CFR 1.84(f): Acceptable sizes:
   ☒ 21.0 cm by 29.7 cm (DIN size A4)
   ☒ 21.6 cm by 27.9 cm (8 1/2 x 11 inches)
   ___ All drawing sheets not the same size. Sheet(s) _____
   ___ Drawings sheets not an acceptable size. Fig(s) 1-6
5. MARGINS. 37 CFR 1.84(g): Acceptable margins:

   Top 2.5 cm Left 2.5cm Right 1.5 cm Bottom 1.0 cm
   SIZE: A4 Size
   Top 2.5 cm Left 2.5 cm Right 1.5 cm Bottom 1.0 cm
   SIZE: 8 1/2 x 11

   Margins not acceptable. Fig(s) _____
   _____ Top (T) _____ Left (L)
   _____ Right (R) _____ Bottom (B)
6. VIEWS. 37 CFR 1.84(h)
   REMINDER: Specification may require revision to correspond to drawing changes.
   Partial views. 37 CFR 1.84(h)(2)
   ___ Brackets needed to show figure as one entity. Fig(s) _____
   ___ Views not labeled separately or properly. Fig(s) _____
   ___ Enlarged view not labeled separetely or properly. Fig(s) _____
7. SECTIONAL VIEWS. 37 CFR 1.84 (h)(3)
   ___ Hatching not indicated for sectional portions of an object. Fig(s) _____
   ___ Sectional designation should be noted with Arabic or Roman numbers. Fig(s) _____
8. ARRANGEMENT OF VIEWS. 37 CFR 1.84(i)
   ___ Words do not appear on a horizontal, left-to-right fashion when page is either upright or turned so that the top becomes the right side, except for graphs. Fig(s) _____
9. SCALE. 37 CFR 1.84(k)
   ___ Scale not large enough to show mechanism without crowding when drawing is reduced in size to two-thirds in reproduction. Fig(s) _____
10. CHARACTER OF LINES, NUMBERS, & LETTERS. 37 CFR 1.84(i)
    ___ Lines, numbers & letters not uniformly thick and well defined, clean, durable, and black (poor line quality). Fig(s) _____
11. SHADING. 37 CFR 1.84(m)
    ___ Solid black areas pale. Fig(s) _____
    ___ Solid black shading not permitted. Fig(s) _____
    ___ Shade lines, pale, rough and blurred. Fig(s) _____
12. NUMBERS, LETTERS, & REFERENCE CHARACTERS. 37 CFR 1.84(p)
    ___ Numbers and reference characters not plain and legible. Fig(s) _____
    ___ Figure legends are poor. Fig(s) _____
    ___ Numbers and reference characters not oriented in the same direction as the view. 37 CFR 1.84(p)(1) Fig(s) _____
    ___ English alphabet not used. 37 CFR 1.84(p)(2) Figs _____
    ___ Numbers, letters and reference characters must be at least .32 cm (1/8 inch) in height. 37 CFR 1.84(p)(3) Fig(s) _____
13. LEAD LINES. 37 CFR 1.84(q)
    ___ Lead lines cross each other. Fig(s) _____
    ___ Lead lines missing. Fig(s) _____
14. NUMBERING OF SHEETS OF DRAWINGS. 37 CFR 1.84(t)
    ___ Sheets not numbered consecutively, and in Arabic numerals beginning with number 1. Sheet(s) _____
15. NUMBERING OF VIEWS. 37 CFR 1.84(u)
    ___ Views not numbered consecutively, and in Arabic numerals, beginning with number 1. Fig(s) _____
16. CORRECTIONS. 37 CFR 1.84(w)
    ___ Corrections not made from prior PTO-948 dated _____
17. DESIGN DRAWINGS. 37 CFR 1.152
    ___ Surface shading shown not appropriate. Fig(s) _____
    ___ Solid black shading not used for color contrast. Fig(s) _____

COMMENTS

REVIEWER A.D DATE 2/20/99 TELEPHONE NO. _____

ATTACHMENT TO PAPER NO. 7

ATTORNEY DOCKET NC
066241.0119

P ENT APPLICATION
09/234,048

OIPE MAY 2 3 2001 PATENT & TRADEMARK OFFICE

RECEIVED MAY 30 2001 Technology Center 2100

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Assistant Commissioner
for Patents
Washington, DC 20231

Dear Sir:

INFORMATION DISCLOSURE STATEMENT

Applicants respectfully request, pursuant to 37 C.F.R. §§ 1.56, 1.97 and 1.98, that the references listed on the attached PTO-1449 form be considered and cited in the examination of the above-identified patent application. Copies of the references are enclosed for the convenience of the Examiner. No representation is made that a search has been made, that the references are material to the patentability of the present application, or that the references qualify as prior art.

The Information Disclosure Statement is being submitted pursuant to 37 C.F.R. § 1.97(c). The stipulated fee of $180.00 is enclosed herewith. The Commissioner is hereby authorized to charge any fees or credit any overpayments to Deposit Account No. 02-0384 of Baker Botts L.L.P.

05/29/2001 KZEWDIE 00000014 09234048
02 FC:126 180.00 OP

Respectfully submitted,

Matthew B. Talpis
Reg. No. 45,152

Correspondence Address:
Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas, Texas 75201-2980
Phone: (214) 953-6984
Fax: (214) 661-4984

Date: May 23, 2001

DAL01:605096.1
066241.0119

PTO-1449 Information Disclosure Citation in an Application	Application No. 09/234,048	Applicant(s) Keith Lowery, et al.	
	Docket Number 066241.0119	Group Art Unit 2182	Filing Date January 19, 1999

OIPE MAY 23 2001 PATENT & TRADEMARK OFFICE

U.S. PATENT DOCUMENTS

		DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE
	A	5,751,956	05/12/98	Kirsch	395	200.33	02/21/96
	B						
	C						
	D						
	E						
	F						
	G						

RECEIVED MAY 30 2001 Technology Center 2100

FOREIGN PATENT DOCUMENTS

		DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATION YES	TRANSLATION NO
	H							
	I							
	J							
	K							
	L							

NON-PATENT DOCUMENTS

		DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
	M		
	N		
	O		
	P		
	Q		
	R		
	S		
	T		

EXAMINER	DATE CONSIDERED
	8/10/01

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP § 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to the applicant.

U.S. Patent and Trademark Office

DAL01:605098.1
066241.0119

05-25-01

GP/2182

ATTORNEY DOCKET NO
066241.0119

PATENT APPLICATION
09/234,048

OIPE JC20 MAY 23 2001 PATENT & TRADEMARK OFFICE

#9/Resp
5/23/01

RECEIVED MAY 30 2001 Technology Center 2100

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

BOX: Patent Application
Honorable Assistant Commissioner
of Patents
Washington, D.C. 20231

Dear Sir:

RESPONSE TO OFFICE ACTION

In response to the Office Action mailed February 23, 2001, Applicants respectfully request the Examiner to reconsider the rejection of the claims in view of the following comments as set forth below. The Claims have not been amended.

IN THE SPECIFICATION

In the Title, please delete "SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS" and replace with -- SYSTEM AND METHOD FOR RESPONDING TO REQUESTS ASSOCIATED WITH DYNAMIC WEB PAGE GENERATION --.

IN THE CLAIMS

For the convenience of the Examiner, all pending claims of the Application are reproduced below.

Claims 1-16 have previously been cancelled.

17. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing a request from a Web server to a page server, said page server receiving said request and releasing said Web server to process other requests wherein said routing step further includes the steps of:

intercepting said request at said Web server and routing said request to said page server;

processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

18. The computer-implemented method in Claim 17 wherein said step of routing said request includes the steps of:

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

19. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

20. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

21. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

22. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of logging into said one or more data sources.

23. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

24. The computer-implemented method in Claim 17 wherein said page server includes tag-based text templates for configuring said Web page.

25. The computer-implemented method in Claim 24 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

26. The computer-implemented method in Claim 24 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

27. The computer-implemented method in Claim 24 wherein said tag-based text templates include HTML templates.

28. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

29. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of processing an object handling extension.

30. The computer-implemented method in Claim 29 wherein said object handling extension is an OLE extension.

31. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a page server, said page server receiving said request and releasing said HTTP-compliant device to process other requests wherein said transferring step further includes the steps of:

intercepting said request at said HTTP-compliant device and transferring said request to said page server;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

32. The computer-implemented method in Claim 31 wherein said step of transferring said request includes the steps of:

transferring said request from said HTTP-compliant device to a dispatcher; and

dispatching said request to said page server.

33. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

34. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of dynamically retrieving said data from said one or more data sources.

35. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

36. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of logging into said one or more data sources.

37. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of maintaining a page cache containing said page.

38. The computer-implemented method in Claim 31 wherein said page server includes tag-based text templates for configuring said page.

39. The computer-implemented method in Claim 38 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

40. The computer-implemented method in Claim 38 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

41. The computer-implemented method in Claim 38 wherein said tag-based text templates include HTML templates.

42. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

43. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of processing an object handling extension.

44. The computer-implemented method in Claim 43 wherein said object handling extension is an OLE extension.

45. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a dispatcher;

maintaining dynamic information regarding data sources a given page server may access;

dispatching said request to an appropriate page server based on said request and based on said dynamic information, said page server receiving said request and releasing said HTTP-compliant device to process other requests;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

REMARKS

This Application has been carefully reviewed in light of the Office Action mailed February 23, 2001. At the time of the Office Action, Claims 17-45 were pending in this patent application. The Examiner rejected Claims 17-45. Thus, Claims 17-45 are now pending in the Application. Applicants respectfully request reconsideration and favorable action in this case.

IN THE TITLE

The Examiner has rejected the title of the invention as being non-descriptive of the invention to which the claims are directed. Applicants respectfully traverse this rejection, but in the interest of advancing prosecution have amended the title to read "SYSTEM AND METHOD FOR RESPONDING TO REQUESTS ASSOCIATED WITH DYNAMIC WEB PAGE GENERATION". Applicants respectfully submit that the title as amended is descriptive of the invention to which the claims are directed.

IN THE DRAWINGS

Applicants respectfully request that submission of formal drawings be deferred until prior to issuance.

DOUBLE PATENTING

The Examiner has rejected claims 17-45 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over Claims 1-11 of Lowery et al., U. S. Pat. No. 5,894,554, ("Lowery"). Applicants have included a Terminal Disclaimer with respect to the double patenting rejection. Applicants respectfully request withdrawal of this rejection.

SECTION 102 REJECTION

Claims 17-45 stand rejected under U.S.C. § 102(e) as being anticipated by Leaf, U.S. Pat. No. 5,754,772 ("Leaf"). Applicants respectfully traverse this rejection for at least the following reasons.

Claim 17 recites, in part, "intercepting said request at said Web server and routing said request to said page server". Leaf does not teach or suggest "intercepting said request".

Instead, Leaf teaches that the web server routes the request directly to the transaction gateway client. Leaf, col. 4, lines 55-57. Leaf does not teach or suggest "intercepting said request at said Web server" because merely routing a request from a web server to the transaction gateway does not involve interception.

Therefore, for at least this reason, Claim 17 is patentable over Leaf. Thus, Applicants respectfully request allowance of Claim 17.

Dependent Claims 18-30 depend from independent Claim 17, shown above to be allowable. Claims 18-30 are allowable as depending from an allowable base claim and as defining further distinctions over the cited reference.

In particular, Claim 21 recites, in part, "maintaining a connection cache to said one or more data sources" and Claim 23 recites, in part, "maintaining a page cache containing said Web page". The Examiner states that the elements of Claims 21 and 23 are inherent in Leaf. Office Action, p. 4. Leaf does not involve either a connection cache or a page cache. Applicants respectfully submit that neither a connection cache nor a page cache are inherent in Leaf. Applicants respectfully request that the Examiner indicate some teaching or suggestion of Leaf with respect to a page cache and a connection cache. Therefore, Claims 21 and 23 are patentable over Leaf. Thus, Applicants respectfully request allowance of Claims 21 and 23.

Also, Claim 24 recites "wherein said page server includes tag-based text templates for configuring said Web page." Applicants respectfully submit that Leaf does not teach or suggest this element of Claim 24. In contrast to the Examiner's assertion, Leaf merely teaches that the Transaction Gateway Client may format the data into an HTML document. Leaf, col. 4, lines 63-67; See also Leaf, col. 4, lines 25-27. Mere formatting of data into an HTML document does not teach or suggest a "tag-based text template for configuring said Web page". Therefore, Claim 24 is patentable over Leaf. Thus, Applicants respectfully request allowance of Claim 24.

Independent Claim 31 is also allowable at least for the reasons discussed above. In particular, Claim 31 recites "intercepting said request at said HTTP-compliant device and transferring said request to said page server". Accordingly, Applicants respectfully request that Claim 31 be allowed.

Dependent Claims 32-44 depend from independent Claim 31, shown above to be allowable. Claims 32-44 are allowable as depending from an allowable base claim and as defining further distinctions over the cited reference. In particular, Claim 35 recites "wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources." Claim 37 recites "wherein said step of dynamically generating said page includes the step of maintaining a page cache containing said page." Claim 38 recites "wherein said page server includes tag-based text templates for configuring said page." Thus, Applicants respectfully request allowance of Claims 32-44.

Independent Claim 45 recites, in part, "maintaining dynamic information regarding data sources a given page server may access". Applicants respectfully submit that Leaf does not teach or suggest this element of Claim 45. Leaf provides no teaching or suggestion of "maintaining dynamic information regarding data sources". Indeed, Leaf notes that "each transaction gateway client pre-establishes a static connection with the n-line transaction processing system." Leaf, col. 2, lines 37-40. Leaf's suggestion of static connections teaches away from "dynamic" information. Therefore, for at least this reason, Claim 45 is patentable over Leaf. Thus, Applicants respectfully request allowance of Claim 45.

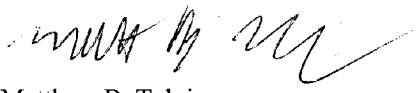
CONCLUSION

Applicants have now made an earnest attempt to place this case in condition for immediate allowance. For the foregoing reasons and for other reasons clearly apparent, Applicants respectfully request reconsideration and allowance of Claims 17-45.

Although Applicants believe that no other fees are due, the Commissioner is hereby authorized to charge any fees or credit any overpayments to Deposit Account No. 02-0384 of Baker Botts L.L.P.

If there are matters that can be discussed by telephone to further the prosecution of this application, Applicants respectfully request that the Examiner call its attorney at the number listed below.

Respectfully submitted,
BAKER BOTTS L.L.P.
Attorneys for Applicants



Matthew B. Talpis
Reg. No. 45,152

Correspondence Address:
Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas, Texas 75201-2980
Phone: 214-953-6984
Fax: 214-661-4984

Date: May 23, 2001

ATTORNEY DOCKET NUMBER:
066241.0119

TENT APPLICATION
09/234,048

OIPE JC26
MAY 23 2001
PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
MAY 30 2001
Technology Center 2100

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Box: Patent Application
Assistant Commissioner for
Patents
Washington, D.C. 20231

CERTIFICATE OF MAILING BY EXPRESS MAIL

I hereby certify that the attached Response to Office Action (12 pages), an Information Disclosure Statement (1 page), a PTO Form-1449 (1 page with 1 reference), a check in the amount of $180.00 for the stipulated filing fee, a Terminal Disclaimer (2 pages with 4 attachments), a check in the amount of $110.00 to cover the Terminal Disclaimer filing fee, a Baker Botts L.L.P. return receipt postcard (1 card), and this Certificate of Mailing (1 page) is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 C.F.R. § 1.10 on this 23rd day of May, 2001 and is addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

Willie Jiles

Willie Jiles

Express Mail Receipt
No. EL759179317US

DAL01:605946.1
066241.0119

OIPE
MAY 23 2001
PATENT & TRADEMARK OFFICE

ATTORNEY DOCKET NUMBER
066241.0119

PATENT APPLICATION
09/234,048

#10
jam
6-1-01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
MAY 30 2001
Technology Center 2100

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Honorable Assistant Commissioner
of Patents
Washington, D.C. 20231

Dear Sir:

TERMINAL DISCLAIMER TO OBVIATE A DOUBLE PATENTING REJECTION (37 1.321 (c)) AND CERTIFICATE UNDER 37 C.F.R. §3.73 (b)

I, Bradley A. Carl, Vice President and General Counsel, of epicRealm Operating Inc., represent that epicRealm Operating Inc. is the assignee and the exclusive owner of the entire right, title and interest of, in and to application Serial No. 09/234,048, filed on January 19, 1999, for System and Method for Managing Dynamic Web Page Generation Requests, as indicated by a copy of the Change of Name documents filed concurrently herewith (copies attached); and certify that to the best of assignee's knowledge and belief, title is in the assignee seeking to take action; and that I am empowered to act on behalf of assignee.

I 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 are 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, and that such willful false statements may jeopardize the validity of the Application or any patent issuing thereon.

05/25/2001 KDEMESS1 00000009 09234048
01 FC:146 110.00 OP

epicRealm Operating Inc. hereby disclaims the terminal part of any patent granted on the above-identified application, which would extend beyond the expiration date of U.S. Patent No. 5,894,554 granted April 13, 1999, also assigned to and owned by said epicRealm Operating Inc. as indicated by the Change of Name documents filed concurrently herewith (copies attached), and hereby agree that any patent so granted on the above-identified application shall be enforceable only for and during such period that the legal title to said patent shall be the same as the legal title to United States Patent No. 5,894,554, this agreement to run with any patent granted on the above-identified application and to be binding upon the grantee, its successor or assigns.

Petitioner does not disclaim any terminal part of any patent granted on the above-identified application prior to the expiration of the full statutory term of the above-referenced U.S. Patent No. 5,894,554, in the event that one or more of the following occurs: U.S. Patent No. 5,894,554 expires for failure to pay a maintenance fee, is held unenforceable, is found invalid, is statutorily disclaimed in whole or terminally disclaimed under 37 C.F.R. 1.321(a), has all claims canceled by a reexamination certificate or is otherwise terminated prior to expiration of its statutory term as presently shortened by any terminal disclaimer, except for the separation of legal title stated above.

The fee required by 37 C.F.R. 1.20(d) is submitted herewith and believed to be correct. However, the Commissioner is hereby authorized to charge any underpayment or credit any overpayment of fees to Deposit Account No. 02-0384 of Baker Botts, L.L.P.

Respectfully submitted,

23 May 01
Date

Bradley A. Carl
Vice President and General Counsel
of epicRealm Operating Inc.

State of Delaware

Office of the Secretary of State

I, EDWARD J. FREEL, SECRETARY OF STATE OF THE STATE OF DELAWARE, DO HEREBY CERTIFY THAT THE SAID "EPICREALM INC.", FILED A CERTIFICATE OF MERGER, CHANGING ITS NAME TO "EPICREALM OPERATING INC.", THE TWENTY-NINTH DAY OF DECEMBER, A.D. 2000, AT 9:02 O'CLOCK A.M.

AND I DO HEREBY FURTHER CERTIFY THAT THE EFFECTIVE DATE OF THE AFORESAID CERTIFICATE OF MERGER IS THE THIRTY-FIRST DAY OF DECEMBER, A.D. 2000, AT 11:58 O'CLOCK P.M.

AND I DO HEREBY FURTHER CERTIFY THAT THE AFORESAID CORPORATION IS DULY INCORPORATED UNDER THE LAWS OF THE STATE OF DELAWARE AND IS IN GOOD STANDING AND HAS A LEGAL CORPORATE EXISTENCE NOT HAVING BEEN CANCELLED OR DISSOLVED SO FAR AS THE RECORDS OF THIS OFFICE SHOW AND IS DULY AUTHORIZED TO TRANSACT BUSINESS.

AND I DO HEREBY FURTHER CERTIFY THAT THE FRANCHISE TAXES HAVE BEEN PAID TO DATE.

Edward J. Freel

Edward J. Freel, Secretary of State

2558538 8320

001656594

AUTHENTICATION: 3004715

DATE: 12-29-00

CERTIFICATE OF AMENDMENT OF CERTIFICATE OF INCORPORATION

InfoSpinner, Inc., a corporation organized and existing under the General Corporation Law of the State of Delaware (the "Corporation"),

DOES HEREBY CERTIFY:

FIRST: That this corporation was originally incorporated on November 8, 1995, pursuant to the General Corporation Law of the State of Delaware (the "General Corporation Law").

SECOND: That the Company's Certificate of Incorporation is amended by deleting in its entirety existing ARTICLE I and inserting in lieu thereof a new ARTICLE I, reading as follows:

"ARTICLE I

The name of this corporation is EpicRealm Inc."

THIRD: The foregoing amendment was approved by the holders of the requisite number of shares of the Corporation in accordance with Section 228 of the General Corporation Law and written notice of such approval of this Certificate of Amendment of the Company's Certificate of Incorporation has been given in accordance with the provisions of Section 228 of the General Corporation Law to those holders of the Corporation's stock who have not so approved this Certificate of Amendment.

* * * * *

d-736569.1

IN WITNESS WHEREOF, this Certificate of Amendment of Certificate of Incorporation has been signed by the President of this corporation this 3rd day of January, 2000.

INFOSPINNER, INC.
(to be renamed hereby
EPICREALM INC.)

By: _____
John D. Ferguson
President and CEO

State of Delaware

Office of the Secretary of State

I, EDWARD J. FREEL, SECRETARY OF STATE OF THE STATE OF DELAWARE, DO HEREBY CERTIFY THE ATTACHED IS A TRUE AND CORRECT COPY OF THE CERTIFICATE OF AMENDMENT OF "INFOSPINNER, INC.", CHANGING ITS NAME FROM "INFOSPINNER, INC." TO "EPICREALM INC.", FILED IN THIS OFFICE ON THE FOURTH DAY OF FEBRUARY, A.D. 2000, AT 1 O'CLOCK P.M.

A FILED COPY OF THIS CERTIFICATE HAS BEEN FORWARDED TO THE NEW CASTLE COUNTY RECORDER OF DEEDS.

Edward J. Freel, Secretary of State

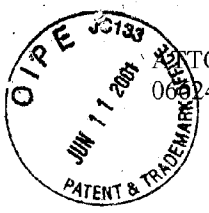
2558538 8100

001058164

AUTHENTICATION: 0239571

DATE: 02-04-00

02/04/00 FRI 15:20 [TX/RX NO 6490]



ATTORNEY DOCKET NO.:
066241.0119

PATENT APPLICATION
09/234,048

2182
#11

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
JUN 13 2001
Technology Center 2100

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Box: Patent Application
Assistant Commissioner for
Patents
Washington, D.C. 20231

CERTIFICATE OF MAILING BY FIRST CLASS MAIL

I hereby certify that the attached Revocation of Attorney and Appointment of New Attorneys for Non-Provisional Application, with Certificate Under 37 C.F.R. 3.73(b) (3 pages and 10 pages of attachments), a Baker Botts L.L.P. return receipt postcard (1 card), and this Certificate of Mailing (1 page) is being deposited with the United States Postal Service as first class mail on this 8th day of June, 2001 and is addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

Willie Jiles
Wille Jiles

DAL01:607002.1
066241.0119

ATTORNEY DOCKET NUMBER
066241.0119

PATENT APPLICATION
09/234,048

OIPE JC133 JUN 11 2001 PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED JUN 13 2001 Technology Center 2100

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Assistant Commissioner
for Patents
Washington, D.C. 20231

Dear Sir:

REVOCATION OF ATTORNEY AND APPOINTMENT
OF NEW ATTORNEYS FOR NON-PROVISIONAL APPLICATION, WITH
CERTIFICATE UNDER 37 C.F.R. 3.73(b)

epicRealm Operating Inc., a Delaware corporation, certifies that it is the assignee in the patent application identified above by virtue of a chain of title from the inventors of the patent application identified above to the current assignee as shown below:

1. From: Inventors
   To: InfoSpinner, Inc.
   Assignment recorded at Reel 7977, Frame 0511, on April 28, 1996

2. From: InfoSpinner, Inc.
   To: epicRealm Inc.
   Filed May 23, 2001, a copy of which is attached.

3. From: epicRealm Inc.
   To: epicRealm Operating Inc.
   Filed May 23, 2001, a copy of which is attached.

DAL01:606996.1
066241.0119

I hereby revoke all prior powers of attorney in the subject application and appoint the following as principal attorneys with full power to prosecute this application and transact all business in the United States Patent and Trademark Office connected therewith:

Jerry W. Mills	Reg. No. 23,005
Robert M. Chiaviello, Jr.	Reg. No. 32,461
Ann C. Livingston	Reg. No. 32,479
Thomas R. Felger	Reg. No. 28,842
Charles S. Fish	Reg. No. 35,870
Kevin J. Meek	Reg. No. 33,738
T. Murray Smith	Reg. No. 30,222
Barton E. Showalter	Reg. No. 38,302
David G. Wille	Reg. No. 38,363
Bradley P. Williams	Reg. No. 40,227
Terry J. Stalford	Reg. No. 39,522
Christopher W. Kennerly	Reg. No. 40,675
Harold E. Meier	Reg. No. 22,428
Douglas M. Kubehl	Reg. No. 41,915
Thomas R. Nesbitt, Jr.	Reg. No. 22,075
James J. Maune	Reg. No. 26,946
Roger J. Fulghum	Reg. No. 39,678
Scott F. Partridge	Reg. No. 28,142
James B. Arpin	Reg. No. 33,470
Jay B. Johnson	Reg. No. 38,193
Robert W. Holland	Reg. No. 40,020
Tara D. Knapp	Reg. No. 43,723
Michael R. Barré	Reg. No. 44,023
William R. Borchers	Reg. No. 44,549
Brian W. Oaks	Reg. No. 44,981
Luke K. Pedersen	Reg. No. 45,003
Matthew B. Talpis	Reg. No. 45,152
David M. Doyle	Reg. No. 43,596
Keiko Ichiye	Reg. No. 45,460
Jeffery D. Baxter	Reg. No. 45,560
Thomas A. Beaton	Reg. No. 46,543
Kurt M. Pankratz	Reg. No. 46,977
Brian E. Szymczak	Reg. No. 47,120
Thomas J. Frame	Reg. No. 47,232
Scott F. Wendorf	Reg. No. 48,029
Chad C. Walters	Reg. No. 48,022
Patent Agents:	
Brian A. Dietzel	Reg. No. 44,656

Please address all correspondence to the following:

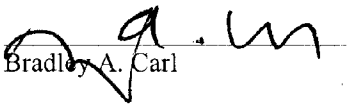
Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas, TX 75201-2980
Phone: 214.953.6984
Fax: 214.661.4984

I, Bradley A. Carl, Vice President and General Counsel of epicRealm Operating Inc., a Delaware corporation, am empowered to sign this certificate on behalf of the assignee.

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 are made with the knowledge that willful false statements, and the like so made, are punishable by fine or imprisonment, or both, under Section 1001, Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

epicRealm Operating Inc.

June 4, 2001
Date

By: 
Bradley A. Carl

Its: Vice President and General Counsel

OIPE JC139 JUN 11 2007 PATENT & TRADEMARK OFFICE

FF

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY AND COMMISSIONER
OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SEPTEMBER 02, 1996

JAMES H. SALTER
12400 WILSHIRE BLVD.
SEVENTH FLOOR
LOS ANGELES, CA 90025

PTAS
RECEIVED
SEP 09 1996
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
LOS ANGELES

100210280A

02577 P001
JHS SN

UNITED STATES PATENT AND TRADEMARK OFFICE
NOTICE OF RECORDATION OF ASSIGNMENT DOCUMENT

THE ENCLOSED DOCUMENT HAS BEEN RECORDED BY THE ASSIGNMENT DIVISION OF THE U.S. PATENT AND TRADEMARK OFFICE. A COMPLETE MICROFILM COPY IS AVAILABLE AT THE ASSIGNMENT SEARCH ROOM ON THE REEL AND FRAME NUMBER REFERENCED BELOW.

PLEASE REVIEW ALL INFORMATION CONTAINED ON THIS NOTICE. THE INFORMATION CONTAINED ON THIS RECORDATION NOTICE REFLECTS THE DATA PRESENT IN THE PATENT AND TRADEMARK ASSIGNMENT SYSTEM. IF YOU SHOULD FIND ANY ERRORS OR HAVE QUESTIONS CONCERNING THIS NOTICE, YOU MAY CONTACT THE EMPLOYEE WHOSE NAME APPEARS ON THIS NOTICE AT 703-308-9723. PLEASE SEND REQUEST FOR CORRECTION TO: U.S. PATENT AND TRADEMARK OFFICE, ASSIGNMENT DIVISION, BOX ASSIGNMENTS, NORTH TOWER BUILDING, SUITE 10C35, WASHINGTON, D.C. 20231.

RECORDATION DATE: 04/28/1996
REEL/FRAME: 7977/0511
NUMBER OF PAGES: 2

BRIEF: ASSIGNMENT OF ASSIGNOR'S INTEREST (SEE DOCUMENT FOR DETAILS).

ASSIGNOR:
LOWERY, KEITH
DOC DATE: 04/22/1996

ASSIGNOR:
LEVINE, ANDREW B.
DOC DATE: 04/22/1996

ASSIGNOR:
HOWELL, RONALD L.
DOC DATE: 04/22/1996

ASSIGNEE:
INFOSPINNER, INC.
1222 E. ARAPAHO
SUITE 320
RICHARDSON, TEXAS 75081

SERIAL NUMBER: 08636477
FILING DATE: 04/23/1996
PATENT NUMBER:
ISSUE DATE:

DIANE RUSSELE, EXAMINER
ASSIGNMENT DIVISION
OFFICE OF PUBLIC RECORDS

OIPE JC133 JUN 11 2001 PATENT & TRADEMARK OFFICE
MAIL ROOM APR 29 1996 TRADEMARK

06-28-1996

100210280

3463677A/D

FORM PTO-1595
(Rev. 6-93)
OMB No. 0651-0011 (exp. 4/94)
Tab settings ⇨ ⇨ ⇨

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

To the Honorable Commissioner of Patents and Trademarks. Please record the attached original documents or copy thereof.

MRD 4-28-96

1. Name of conveying party(ies):

Keith Lowery
Andrew B. Levine
Ronald L. Howell

Additional name(s) of conveying party(ies) attached? ☐ Yes ■ No

2. Name and address of receiving party(ies):

Name: InfoSpinner, Inc.
Internal Address:
Street Address: 1222 E. Arapaho
Suite 320
City Richardson State TX ZIP 75081

Additional name(s) & address(es) attached? ☐ Yes ■ No

3. Nature of conveyance:

■ Assignment ☐ Merger
☐ Security Agreement ☐ Change of Name
☐ Other

Execution Date: April 22, 1996

4. Application number(s) or patent number(s):

If this document is being filed together with a new application, the execution date of the application is: April 23, 1996

A. Patent Application No.(s)

B. Patent No.(s)

67479 U.S. PTO
06/01/96

Additional numbers attached? ☐ Yes ■ No

5. Name and address of party to whom correspondence concerning document should be mailed:

Name: James H. Salter
Internal Address:
Street Address: 12400 Wilshire Blvd.
Seventh Floor
City: Los Angeles State: CA ZIP: 90025

6. Total number of applications and patents involved: 1

7. Total fee (37 CFR 3.41).....$ 40.00

■ Enclosed
☐ Authorized to be charged to deposit account

8. Deposit account number:

(Attach duplicate copy of this page if paying by deposit account)

DO NOT USE THIS SPACE

9. Statement and signature.

To the best of my knowledge and belief, the foregoing information is true and correct and any attached copy is a true copy of the original document.

James H. Salter — Name of Person Signing
Signature
4/23/96 — Date

Total number of pages comprising cover sheet, attachments, and document: 2

Mail documents to be recorded with required cover sheet info
Comissioner of Patents & T

08/636477

Attorney's Docket No.: 02577.P001

ASSIGNMENT

(For Execution Prior To Filing Patent Application)

PATENT

In consideration of good and valuable consideration, the receipt of which is hereby acknowledged, we, the undersigned, Keith Lowery, Andrew B. Levine, and Ronald L. Howell,

hereby sell, assign, and transfer to InfoSpinner Inc.,

a corporation of Delaware, having a principal place of business at 1222 E. Arapaho, Suite 320 Richardson, Texas 75081, ("Assignee"), and its successors, assigns, and legal representatives, the entire right, title, and interest for the United States and all foreign countries, in and to any and all improvements that are disclosed in the application for the United States patent that has been executed by the undersigned prior hereto or concurrently herewith on the dates indicated below and is entitled A METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE,

and in and to said application and all divisional, continuing, substitute, renewal, reissue, and all other patent applications that have been or shall be filed in the United States and all foreign countries on any of said improvements; and in and to all original and reissued patents that have been or shall be issued in the United States and all foreign countries on said improvements; and in and to all rights of priority resulting from the filing of said United States application;

agree that said Assignee may apply for and receive a patent or patents for said improvements in its own name; and that, when requested, without charge to, but at the expense of, said Assignee, its successors, assigns, and legal representatives, to carry out in good faith the intent and purpose of this Assignment, the undersigned will execute all divisional, continuing, substitute, renewal, reissue, and all other patent applications on any and all said improvements; execute all rightful oaths, assignments, powers of attorney, and other papers; communicate to said Assignee, its successors, assigns, and representatives all facts known to the undersigned relating to said improvements and the history thereof; and generally assist said Assignee, its successors, assigns, or representatives in securing and maintaining proper patent protection for said improvements and for vesting title to said improvements, and all applications for patents and all patents on said improvements, in said Assignee, its successors, assigns, and legal representatives; and

covenant with said Assignee, its successors, assigns, and legal representatives that no assignment, grant, mortgage, license, or other agreement affecting the rights and property herein conveyed has been made to others by the undersigned, and that full right to convey the same as herein expressed is possessed by the undersigned.

Each Inventor: Please Sign and Date Below:		**Each Inventor: Please also list the date that you signed the accompanying DECLARATION AND POWER OF ATTORNEY:**
4-22, 1996 Date	Name: Keith Lowery	4-22, 1996 Date
4-22, 1996 Date	Name: Andrew B. Levine	4-22, 1996 Date
4-22, 1996 Date	Name: Ronald L. Howell	4-22, 1996 Date
, 19 Date	Name:	, 19 Date
, 19 Date	Name:	, 19 Date
, 19 Date	Name:	, 19 Date

State of: _____ } SS.
County of: _____ }

Assignment Document Return Address:
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
12400 Wilshire Blvd., Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8598

On this ____ day of _____ 19____, before me, _____, the undersigned Notary Public, personally appeared _____

[___] personally known to me [___] proved to me on the basis of satisfactory evidence to be the person(s) whose name(s) _____ subscribed to the within instrument, and acknowledged that _____ executed it.

WITNESS my hand and official seal.

_____
Notary's Signature

Rev. 03/13/95 (A1)

PTO JC132 PATENT & TRADEMARK OFFICE JUN 11

Attorney Docket 066421.0101

FORM PTO 1595
1-31-92

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

RECORDATION FORM COVER SHEET
PATENTS ONLY

To the Honorable Commissioner of Patents and Trademarks. Please record the attached original documents or copy thereof.

1. Name and Address of Conveying Party(ies) INDIVIDUAL:

epicRealm Inc.
2435 N. Central Expressway
Palisades Central II
Suite 400
Richardson, Texas 75080

Additional name(s) of conveying party(ies) attached?
___ Yes _X_ No

2. Name and Address of receiving Party(ies):

Name: epicRealm Operating Inc.

Internal Address:

Street Address: 2435 N. Central Expressway
Palisades Central II
City: Suite 400
Richardson
State:
Texas Zip: 75080

Additional name(s) & Address(es) attached? [ ] Yes [X] No

3. Nature of conveyance:

[ ] Assignment
[ ] Security Agreement
[ ] Other
[ ] Merger
[X] Change of Name

Effective Date: December 31, 2000

4. Application number(s) or patent number(s):
If this document is being filed together with a new application, the execution date of the application is: _____.

A. Patent Application No.(s):

See Attached

B. Patent No.(s):

Additional Numbers [X] Yes [ ] No

5. Name and address of party to whom correspondence concerning document should be mailed to:

Name: Kevin J. Meek
Internal Address: BAKER BOTTS L.L.P.
Street Address: 2001 Ross Avenue, Suite 600
City: Dallas
State: Texas Zip 75201-2980

6. Total number of application and patents involved: One

7. Total Fee (37C.F.R. 3.41): $280.00

[X] Enclosed
[ ] Authorized to be charged to deposit account

8. Deposit Account Number:

(Attach Duplicate Copy of this page if paying by deposit account)

DO NOT USE THIS SPACE

9. Statement and signature.
To the best of my knowledge and belief, the foregoing information is true and correct and any attached copy is a true copy of the original document.

Kevin J. Meek
Signature

4/3/01
Date

Total number of pages including cover sheet 2

OMB No. 0651-0011 (exp. 4/94)

Do not detach this portion

Mail documents to be recorded with required cover sheet information:

Commissioner of Patents and Trademarks
Box Assignments
Washington, D.C. 20231

Public burden reporting for this sample cover sheet is estimated to average about 30 minutes per document to be recorded, including time for reviewing the document and gathering the data needed, and completing and reviewing the sample cover sheet. Send comments regarding this burden estimate to the U.S. Patent and Trademark Office, Office of Information Systems, PK2-1000C, Washington, D.C. 20231, and to the Office of Management and Budget, Paperwork Reduction Project, (0651-0011), Washington, D.C. 20503

DAL01:595338.1

State of Delaware

Office of the Secretary of State

I, EDWARD J. FREEL, SECRETARY OF STATE OF THE STATE OF DELAWARE, DO HEREBY CERTIFY THAT THE SAID "EPICREALM INC.", FILED A CERTIFICATE OF MERGER, CHANGING ITS NAME TO "EPICREALM OPERATING INC.", THE TWENTY-NINTH DAY OF DECEMBER, A.D. 2000, AT 9:02 O'CLOCK A.M.

AND I DO HEREBY FURTHER CERTIFY THAT THE EFFECTIVE DATE OF THE AFORESAID CERTIFICATE OF MERGER IS THE THIRTY-FIRST DAY OF DECEMBER, A.D. 2000, AT 11:58 O'CLOCK P.M.

AND I DO HEREBY FURTHER CERTIFY THAT THE AFORESAID CORPORATION IS DULY INCORPORATED UNDER THE LAWS OF THE STATE OF DELAWARE AND IS IN GOOD STANDING AND HAS A LEGAL CORPORATE EXISTENCE NOT HAVING BEEN CANCELLED OR DISSOLVED SO FAR AS THE RECORDS OF THIS OFFICE SHOW AND IS DULY AUTHORIZED TO TRANSACT BUSINESS.

AND I DO HEREBY FURTHER CERTIFY THAT THE FRANCHISE TAXES HAVE BEEN PAID TO DATE.

2558538 8320

001656594

Edward J. Freel

AUTHENTICATION: 3864715

DATE: 12-29-00

PATENT & TRADEMARK OFFICE JC16 JUN 11 2001

Docket No. 066241.0119

FORM PTO-1595
1-31-92

RECORDATION FORM COVER SHEET
PATENTS ONLY

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

To the Honorable Commissioner of Patents and Trademarks. Please record the attached original documents or copy thereof.

1. Name and Address of Conveying Party(ies):

InfoSpinner, Inc.

Corporation/State: Delaware

Additional name(s) of conveying party(ies) attached?
___ Yes __X__ No

3. Nature of conveyance:

	Assignment		Merger
	Security Agreement	X	Change of Name
	Other		

Effective Date: February 4, 2000

2. Name and Address of receiving Party(ies):

Name: epicRealm Inc.

Internal Address: Suite 400

Street Address: Palisades Central II, 2435 N. Central Expwy.
City: Richardson
State: Texas Zip: 75080

Corporation/State: Delaware

Additional name(s) & address(es) attached? [ ] Yes [X] No

4. Application number(s) or patent number(s):

If this document is being filed together with a new application, the execution date of the application is: _____

A. Patent Application No.(s) 09/234,048

B. Patent No.(s) 5,894,554

Additional Numbers attached? [ ] Yes [X] No

5. Name and address of party to whom correspondence concerning document should be mailed:

Name: Matthew B. Talpis, Esq.
Internal Address: Baker Botts L.L.P.
Street Address: 2001 Ross Avenue, Suite 600
City: Dallas
State: Texas Zip: 75201-2980

6. Total number of applications and patents involved: One (1)

7. Total Fee (37 CFR 3.41): $ 80.00

[X] Enclosed
[ ] Authorized to be charged to deposit account

8. Deposit account number:

(Attach Duplicate Copy of this page if paying by deposit account)

DO NOT USE THIS SPACE

9. Statement and signature.

To the best of my knowledge and belief, the foregoing information is true and correct and any attached copy is a true copy of the original document.

Matthew B. Talpis — Name of Person Signing



Signature

May 23, 2001 — Date

Total number of pages including cover sheet: 4

OMB No. 0651-0011 (exp.4/94)

Do not detach this portion

Mail documents to be recorded with required cover sheet information:

Commissioner of Patent and Trademarks
Box Assignments
Washington, D.C. 20231

Public burden reporting for this sample cover sheet is estimated to average about 30 minutes per document to be recorded, including time for reviewing the document and gathering the data needed, and completing and reviewing the sample cover sheet. Send comments regarding this burden estimate to the U.S. Patent and Trademark Office, Office of Information Systems, PK2-1000C, Washington, D.C. 20231, and to the Office of Management and Budget, Paperwork Reduction Project, (0651-0011), Washington, D.C. 20503.

L01:605850.1
241.0119

04/04/2000 15:05 FAX
FROM CORPORATION TRUST, DOVER DE. 302-674-3340 (FRI) 02. 04' 00 16:22/ST. 16:21/NO. 3560598878 P 2

CERTIFICATE OF AMENDMENT
OF
CERTIFICATE OF INCORPORATION

InfoSpinner, Inc., a corporation organized and existing under the General Corporation Law of the State of Delaware (the "Corporation"),

DOES HEREBY CERTIFY:

FIRST: That this corporation was originally incorporated on November 8, 1995, pursuant to the General Corporation Law of the State of Delaware (the "General Corporation Law").

SECOND: That the Company's Certificate of Incorporation is amended by deleting in its entirety existing ARTICLE I and inserting in lieu thereof a new ARTICLE I, reading as follows:

"ARTICLE I

The name of this corporation is EpicRealm Inc."

THIRD: The foregoing amendment was approved by the holders of the requisite number of shares of the Corporation in accordance with Section 228 of the General Corporation Law and written notice of such approval of this Certificate of Amendment of the Company's Certificate of Incorporation has been given in accordance with the provisions of Section 228 of the General Corporation Law to those holders of the Corporation's stock who have not so approved this Certificate of Amendment.

* * * * *

d-736369.1

02/03/00 THU 18:05 [TX/RX NO 6424]
02/04/00 FRI 15:20 [TX/RX NO 6490]

IN WITNESS WHEREOF, this Certificate of Amendment of Certificate of Incorporation has been signed by the President of this corporation this 3/ day of January, 2000.

INFOSPINNER, INC.
(to be renamed hereby
EPICREALM INC.)

By: _____
John D. Ferguson
President and CEO

State of Delaware

Office of the Secretary of State

I, EDWARD J. FREEL, SECRETARY OF STATE OF THE STATE OF DELAWARE, DO HEREBY CERTIFY THE ATTACHED IS A TRUE AND CORRECT COPY OF THE CERTIFICATE OF AMENDMENT OF "INFOSPINNER, INC.", CHANGING ITS NAME FROM "INFOSPINNER, INC." TO "EPICREALM INC.", FILED IN THIS OFFICE ON THE FOURTH DAY OF FEBRUARY, A.D. 2000, AT 1 O'CLOCK P.M.

A FILED COPY OF THIS CERTIFICATE HAS BEEN FORWARDED TO THE NEW CASTLE COUNTY RECORDER OF DEEDS.

Edward J. Freel, Secretary of State

2558538 8100

001058164

AUTHENTICATION: 0239571

DATE: 02-04-00

02/04/00 FRI 15:20 [TX/RX NO 6490]

 #11

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D

CONFIRMATION NO. 6557

OC000000006192282

JAMES H SALTER
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
7TH FLOOR
LOS ANGELES, CA 90025

Date Mailed: 06/18/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 06/11/2001.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

Brenda Harrison

Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

#12

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D

CONFIRMATION NO. 6557

OC000000006192311

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980

Date Mailed: 06/18/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 06/11/2001.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

Brenda Harrison

Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

#13/Req for CFR Incl Receipt 6/25/14

Atty Docket No. 02577.[illegible]D PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Keith Lowery et al.

Application No.: 09/234,048

Filed: January 19, 1999

For: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

OIPE JC1177 MAR 0 5 2001 PATENT & TRADEMARK OFFICE

Examiner: Not yet assigned

Art Unit: 2758

RECEIVED
JUN 1 9 2001
Technology Center 2100

Assistant Commissioner for Patents
Washington, D.C. 20231

REQUEST FOR CORRECTION OF FILING RECEIPT

Dear Sir:

In the Corrected Filing Receipt for the above-referenced patent application filed on January 19, 1999, the amended title was not reflected. Please issue a Corrected Filing Receipt with amended title as follows:

SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

Please have the Filing Receipt changed to reflect the amended title. Enclosed is a copy of the Preliminary Amendment and the incorrect Filing Receipt with the change noted thereon.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Dated: 2/28/2001

James H. Salter
Reg. No. 35,668

Customer No. 008791
12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408)720-8300

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231 on

February 28, 2001
(Date of Deposit)

Claire Wallters
(Typed or printed name of person mailing correspondence)

Claire Wallters
(Signature of person mailing correspondence)

PTO-103X
(Rev. 6-99)

FILING RECEIPT
CORRECTED

OIPE JC177 MAR 05 2001 PATENT & TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY AND COMMISSIONER
OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

JHS

APPLICATION NUMBER	FILING DATE	GRP ART UNIT	FIL FEE REC'D	ATTORNEY DOCKET NO.	DRWGS	TOT CL	IND CL
09/234,048	01/19/99	2758	$994.00	02577.P001D	5	16	6

JAMES H SALTER
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
7TH FLOOR
LOS ANGELES CA 90025

RECEIVED
JUL 28 1999
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
LOS ANGELES

INFO SPINNER,

Receipt is acknowledged of this nonprovisional Patent Application. It will be considered in its order and you will be notified as to the results of the examination. Be sure to provide the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please write to the Office of Initial Patent Examination's Customer Service Center. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts of Application" ("Missing Parts Notice") in this application, please submit any corrections to this Filing Receipt with your reply to the "Missing Parts Notice." When the PTO processes the reply to the "Missing Parts Notice," the PTO will generate another Filing Receipt incorporating the requested corrections (if appropriate).**

Applicant(s) KEITH LOWERY, RICHARDSON, TX; ANDREW B. LEVINE, PLANO, TX; RONALD L. HOWELL, ROWLETT, TX.

CONTINUING DATA AS CLAIMED BY APPLICANT-
THIS APPLN IS A DIV OF 08/636,477 04/23/96 PAT 5,894,554

IF REQUIRED, FOREIGN FILING LICENSE GRANTED 02/05/99
TITLE
~~METHOD AND APPARATUS FOR CREATING AND MANAGING A CUSTOM WEB SITE~~
SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS
PRELIMINARY CLASS: 709

RECEIVED
JUN 19 2001
Technology Center 2100

ENTERED
JUL 28 1999
STATUS DB-LA

ENTERED JHS DB

W

DATA ENTRY BY: MARTIN, DIANE TEAM: 04 DATE: 07/20/99

(See reverse for new important information)

02577.P001D Copy *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of: Keith Lowery, et al.

Serial No.: 09/234,048

Filed: January 19, 1999

For: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS (AS AMENDED)

A Rule 1.53(b) Divisional Application of:
U.S. Serial No. 08/636,477
Filed April 23, 1996

Examiner: Unassigned

Art Unit: 2755

OIPE JC17 MAR 0 5 2001 PATENT & TRADEMARK OFFICE

RECEIVED
AUG 2 1 2001
Technology Center 2100

Assistant Commissioner for Patents
Washington, D.C. 20231

PRELIMINARY AMENDMENT

Sir:

Applicants respectfully request that this Preliminary Amendment be entered prior to examination of the above-identified patent application.

IN THE TITLE

Please amend the title to read --SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS--.

IN THE CLAIMS

Please cancel claims 1-16 and enter new claims 17-45 as follows.

17. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing a request from a Web server to a page server, said page server receiving said request and releasing said Web server to process other requests wherein said routing step further includes the steps of:

intercepting said request at said Web server and routing said request to said page server;

processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

18. The computer-implemented method in Claim 17 wherein said step of routing said request includes the steps of:

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

19. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

20. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

21. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

22. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of logging into said one or more data sources.

23. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

24. The computer-implemented method in Claim 17 wherein said page server includes tag-based text templates for configuring said Web page.

25. The computer-implemented method in Claim 24 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

26. The computer-implemented method in Claim 24 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one

or more data sources in response to said request.

27. The computer-implemented method in Claim 24 wherein said tag-based text templates include HTML templates.

28. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

29. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of processing an object handling extension.

30. The computer-implemented method in Claim 29 wherein said object handling extension is an OLE extension.

31. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a page server, said page server receiving said request and releasing said HTTP-compliant device to process other requests wherein said transferring step further includes the steps of:

intercepting said request at said HTTP-compliant device and transferring said request to said page server;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

32. The computer-implemented method in Claim 31 wherein said step of transferring said request includes the steps of:

transferring said request from said HTTP-compliant device to a dispatcher; and

dispatching said request to said page server.

33. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

34. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of dynamically retrieving said data from said one or more data sources.

35. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

36. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of logging into said one or more data sources.

37. The computer-implemented method in Claim 31 wherein said step of

dynamically generating said page includes the step of maintaining a page cache containing said page.

38. The computer-implemented method in Claim 31 wherein said page server includes tag-based text templates for configuring said page.

39. The computer-implemented method in Claim 38 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

40. The computer-implemented method in Claim 38 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

41. The computer-implemented method in Claim 38 wherein said tag-based text templates include HTML templates.

42. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

43. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of processing an object handling extension.

44. The computer-implemented method in Claim 43 wherein said object handling extension is an OLE extension.

45. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a dispatcher;

maintaining dynamic information regarding data sources a given page server may access;

dispatching said request to an appropriate page server based on said request and based on said dynamic information, said page server receiving said request and releasing said HTTP-compliant device to process other requests;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

REMARKS

Consideration of this application in view of the foregoing amendments is respectfully requested. Claims 1-16 have been cancelled and new claims 17-45 have been added to more distinctly claim the present invention. Claims 17-45 are presented for examination. Applicants respectfully submit that the claims are in condition for allowance.

An Information Disclosure Statement accompanies this Preliminary Amendment. Applicant respectfully submits that none of the references cited in the Information Disclosure Statement anticipate or render obvious the claims of the present application.

If there are any additional charges, please charge Deposit Account No. 02-2666. If a telephone interview would in any way expedite the prosecution of this application, the Examiner is invited to contact James H. Salter at (408) 720-8598.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 7/9, 1999

James H. Salter (Reg. No. 35,668)

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA. 90025-1026
(408) 720-8598

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231 on July 9, 1999.

Claire Wallters
Name of Person Mailing Correspondence

Claire Wallters
Signature

7/9/99
Date

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
09/234,048	01/19/99	LOWERY	K 02577.P001D

EXAMINER	
PERVEEN,R	
ART UNIT	**PAPER NUMBER**
2182	14

TM02/0827

MATTHEW B. TALPIS, ESQ.
BAKER BOTTS L.L.P.
2001 ROSS AVENUE
SUITE 600
DALLAS TX 75201-2980

DATE MAILED: 08/27/01

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)
*U.S. GPO: 2000-473-000/44602

1st File Copy

Office Action Summary	Application No.	Applicant(s)	
	09/234,048	LOWERY ET AL.	
	Examiner	Art Unit	
	Rehana Perveen	2182	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 May 2001* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *17-45* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *17,19-31 and 33-45* is/are rejected.

7)☒ Claim(s) *18 and 32* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) *8* .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

Response to Amendment

1. Applicant's Amendment (B) filed on 23 May 2001 has been received, reviewed, and considered. The following rejections now apply.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 17, 19-31, and 33-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rogers et al, patent no. 5,752,246, in view of Malcolm, patent no. 5,701,463.

4. As to claim 17, Rogers et al teach routing a request from a Web server to a page server, the page server receiving the request and releasing the Web server to process other requests, intercepting the request at the Web server and routing the request to the page server,

processing the request by the page server while the Web server concurrently processes the other requests, and dynamically generating a Web page in response to the request, the Web page including data dynamically retrieved from one or more data sources (col. 5 line 18 - col. 6 line 58 and col. 11 line 64 - col. 12 line 26).

5. However, Rogers et al do not **explicitly** teach the web server simply intercepting and routing the request. Rogers et al teach the web server itself deciding whether to send the request to the other server.

6. Malcolm teaches the usage of a special program in the server to intercept the request and decide how to fulfill the request (col. 1 lines 62-65 and col. 4 lines 57-63).

7. It would have been obvious for one of ordinary skill in the art at the time of the invention to combine the teachings of Rogers et al and Malcolm because Malcolm teaches modifying the function of a server without changing the server by using a special program, thus when Malcolm's teaching is incorporated into Rogers et al, Rogers et al do

not have to modify the original web server improving the prior existing web server environment.

8. As to claims 19 and 20, Rogers et al teach identifying said one or more data sources and dynamically retrieving data from the one or more data sources (figure 1 and figure 7).

9. As to claim 21, Rogers et al, inherently, teach maintaining a connection cache to one or more data sources.

10. As to claim 22, Rogers et al teach logging into the one or more data sources (inherent).

11. As to claim 23, Rogers et al, inherently, teach maintaining a page cache containing the Web page.

12. As to claims 24-26, Rogers et al teach tag-based text templates for configuring the Web page, inserting the dynamically retrieved data from the one or more data sources into the tag-based text templates, at least one of said tag-based text templates drives a format of the data dynamically retrieved from the one or more data sources in response to

the request (col. 5 line 18 - col. 6 line 58 and col. 11 line 64 - col. 12 line 26).

13. As to claim 27, Rogers et al teach the tag-based text templates include HTML templates (col. 5 lines 18-36 and col. 7 lines 1-18).

14. As to claim 28, Rogers et al, inherently, teach dynamically updating data at the one or more data sources.

15. As to claims 29 and 30, Rogers et al, inherently, teach processing an object handling extension, the object handling extension being an OLE extension.

16. Claims 31 and 33-45 are different variations of claims 17-30, and therefore, are rejected under the same rationale.

Allowable Subject Matter

17. Claims 18 and 32 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks
Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry)

Or:

(703) 308-5359 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).

18. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Rehana Perveen, whose telephone number is (703) 305-8476. The examiner can normally be reached Monday through Friday from 8:00 AM - 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thomas C. Lee, can be reached at (703) 305-9717. The fax phone number for this Group is (703) 308-5359.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Rehana Perveen
August 20, 2001

THOMAS LEE
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Notice of References Cited	Application/Control No. 09/234,048	Applicant(s)/Patent Under Reexamination LOWERY ET AL.	
	Examiner Rehana Perveen	Art Unit 2182	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification	
	A	US-5,752,246-A	05-1998	Rogers et al	707	10
	B	US-5,701,463-A	12-1997	Malcolm	395	610
	C	US-				
	D	US-				
	E	US-				
	F	US-				
	G	US-				
	H	US-				
	I	US-				
	J	US-				
	K	US-				
	L	US-				
	M	US-				

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification	
	N						
	O						
	P						
	Q						
	R						
	S						
	T						

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Hoffner, 'Inter-operability and distributed application platform design', Web URL:http://www.ansa.co.uk/, 1995, pages 342-356.
	V	Mourad et al, 'Scalable Web Server Architectures', IEEE, 6/1997, pages 12-16.
	W	Dias et al, 'A Scalable and Highly Available Web Server', IEEE, 1996, pages 85-92.
	X	'Single System Image and Load Balancing for Network Access to a Loosely Coupled Complex', IBM TDB, Vol. 34, 2/1992, pages 464-467.

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

12-03-01 2182

ATTORNEY DOCKET NO.: 066241.0119

PATENT APPLICATION 09/234,048

#15 fax 12-7-01



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
DEC 05 2001
Technology Center 2100

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR RESPONDING TO REQUESTS ASSOCIATED WITH DYNAMIC WEB PAGE GENERATION

U.S. Patent and Trademark Office
P. O. Box 2327
Arlington, VA 22202

CERTIFICATE OF MAILING BY EXPRESS MAIL

I hereby certify that the attached Amendment (11 pages), formal drawings (4 sheets), a Baker Botts L.L.P. return receipt postcard (1 card), and this Certificate of Mailing (1 page) is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 C.F.R. § 1.10 on this 27th day of November, 2001 and is addressed to the U.S. Patent and Trademark Office, P. O. Box 2327, Arlington, VA 22202.

Levado Hamilton

Express Mail Receipt
EL501005144US

DAL01:643354.1

ATTORNEY DOCKET NO.:
066241.0119

PATENT APPLICATION
09/234,048

OIPE JC1481 NOV 27 2001 PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	Keith Lowery, et al.
Serial No.:	09/234,048
Filing Date:	January 19, 1999
Group Art Unit:	2182
Examiner:	Rehana Perveen
Title:	SYSTEM AND METHOD FOR RESPONDING TO REQUESTS ASSOCIATED WITH DYNAMIC WEB PAGE GENERATION

RECEIVED
DEC 0 5 2001
Technology Center 2100

U.S. Patent and Trademark Office
P.O. Box 2327
Arlington, VA 22202

Dear Sir:

RESPONSE TO OFFICE ACTION

In response to the Office Action mailed August 27, 2001, Applicants respectfully request the Examiner to reconsider the rejection of the claims in view of the following comments as set forth below.

IN THE CLAIMS For the convenience of the Examiner, all pending claims of the Application are reproduced below. The Claims have not been amended.

Claims 1-16 have previously been cancelled.

17. A computer-implemented method for managing a dynamic Web page generation request to a Web server, said computer-implemented method comprising the steps of:

routing a request from a Web server to a page server, said page server receiving said request and releasing said Web server to process other requests wherein said routing step further includes the steps of:

intercepting said request at said Web server and routing said request to said page server;

processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and

dynamically generating a Web page in response to said request, said Web page including data dynamically retrieved from one or more data sources.

18. The computer-implemented method in Claim 17 wherein said step of routing said request includes the steps of:

routing said request from said Web server to a dispatcher; and

dispatching said request to said page server.

19. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

20. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of dynamically retrieving said data from said one or more data sources.

21. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

22. The computer-implemented method in Claim 17 wherein said step of processing said request includes the step of logging into said one or more data sources.

23. The computer-implemented method in Claim 17 wherein said step of dynamically generating said Web page includes the step of maintaining a page cache containing said Web page.

24. The computer-implemented method in Claim 17 wherein said page server includes tag-based text templates for configuring said Web page.

25. The computer-implemented method in Claim 24 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

26. The computer-implemented method in Claim 24 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

27. The computer-implemented method in Claim 24 wherein said tag-based text templates include HTML templates.

28. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

29. The computer-implemented method in Claim 17 wherein said step of processing said request further includes the step of processing an object handling extension.

30. The computer-implemented method in Claim 29 wherein said object handling extension is an OLE extension.

31. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a page server, said page server receiving said request and releasing said HTTP-compliant device to process other requests wherein said transferring step further includes the steps of:

intercepting said request at said HTTP-compliant device and transferring said request to said page server;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

32. The computer-implemented method in Claim 31 wherein said step of transferring said request includes the steps of:

transferring said request from said HTTP-compliant device to a dispatcher; and

dispatching said request to said page server.

33. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of identifying said one or more data sources from which to retrieve said data.

34. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of dynamically retrieving said data from said one or more data sources.

35. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of said page server maintaining a connection cache to said one or more data sources.

36. The computer-implemented method in Claim 31 wherein said step of processing said request includes the step of logging into said one or more data sources.

37. The computer-implemented method in Claim 31 wherein said step of dynamically generating said page includes the step of maintaining a page cache containing said page.

38. The computer-implemented method in Claim 31 wherein said page server includes tag-based text templates for configuring said page.

39. The computer-implemented method in Claim 38 wherein said step of processing said request further includes the step of inserting said dynamically retrieved data from said one or more data sources into said tag-based text templates.

40. The computer-implemented method in Claim 38 wherein at least one of said tag-based text templates drives a format of the data dynamically retrieved from said one or more data sources in response to said request.

41. The computer-implemented method in Claim 38 wherein said tag-based text templates include HTML templates.

42. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of dynamically updating data at said one or more data sources.

43. The computer-implemented method in Claim 31 wherein said step of processing said request further includes the step of processing an object handling extension.

44. The computer-implemented method in Claim 43 wherein said object handling extension is an OLE extension.

45. A computer-implemented method comprising the steps of:

transferring a request from an HTTP-compliant device to a dispatcher;

maintaining dynamic information regarding data sources a given page server may access;

dispatching said request to an appropriate page server based on said request and based on said dynamic information, said page server receiving said request and releasing said HTTP-compliant device to process other requests;

processing said request, said processing being performed by said page server while said HTTP-compliant device concurrently processes said other requests; and

dynamically generating a page in response to said request, said page including data dynamically retrieved from one or more data sources.

IN THE DRAWINGS

Attached herewith for the Examiner's and Draftsperson's approval are formal drawings for this application.

REMARKS

This Application has been carefully reviewed in light of the Office Action mailed August 27, 2001. At the time of the Office Action, Claims 17-45 were pending in this patent application. The Examiner objected to Claims 18 and 32 and rejected Claims 17, 19-31, 33-45. Thus, Claims 17-45 are now pending in the Application. Applicants respectfully request reconsideration and favorable action in this case.

Allowable Claims

Applicants thank the Examiner for the indication that Claims 18 and 32 would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims. Applicants respectfully defer rewriting of these Claims until the Examiner has considered the following arguments.

Section 103 Rejection

Claims 17, 19-31, and 33-45 stand rejected under 35 U.S.C. §103(a) as being unpatentable over *Rogers, et al.*, U.S. Pat. No. 5,752,246 ("Rogers"), in view of *Malcolm*, U.S. Pat. No. 5,701,463 ("Malcolm"). Applicants respectfully traverse this rejection for at least the following reasons.

Claim 17 recites, in part, "routing a request from a Web server to a page server". The Examiner admits that Rogers does not teach these elements of Claim 17. Office Action, p. 3. The Examiner relies solely on Malcolm to teach these elements of Claim 17. Office Action, p. 3. Malcolm involves intercepting a file open request at the operating system level and determining whether the identity of the file to be opened should be replaced with the identity of a substitute file to be opened instead. Malcolm, Abstract. Malcolm further notes that whether to permit execution of a file is implemented at the file server by a controlling utility. Malcolm, col. 4, lines 57-59. Malcolm's interception of a file open request and determination of whether to permit execution of a file does not teach or suggest "routing a request from a Web server to a page server" because mere interception and determination do not teach or suggest "routing" the request from one place to another.

Claim 17 further recites, in part, "processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests" and "said page server receiving said request and releasing said Web server to process other requests". Neither Rogers nor Malcolm, either alone or in combination, teach or suggest

these elements of Claim 17. Rogers involves organizing distributed sub-agents as distributed integration solution servers to support a web server. Rogers, Abstract. Rogers also discusses a means for accepting web client requests for information, obtaining data from one or more databases and presenting that information to the web client. Rogers, col. 5, lines 54-61. At no time does Rogers teach or suggest "concurrently" processing other requests or "releasing said Web server to process other requests" because merely retrieving data from multiple sources does not teach or suggest these elements.

In addition, there is no motivation to combine Rogers and Malcolm. Malcolm involves intercepting file open requests at an operating system level to determine whether a substitute file should be opened instead of the original file. Rogers involves a distributed integration solution associated with web servers. Rogers, Abstract. No motivation exists to combine the web server functionality of Rogers with the operating system level functionality of Malcolm. Also, as Rogers is completely unconcerned with dealing with substitute file names, Applicants respectfully submit that the Examiner has made use of impermissible hindsight based on Applicants' disclosure when combining Rogers and Malcolm.

Therefore, for at least these reasons, Claim 17 is patentable over the cited references, either alone or in combination. Thus, Applicants respectfully request allowance of Claim 17.

Independent Claims 31 and 45 are patentable for at least the reasons discussed above in association with independent Claim 17. Thus, Applicants respectfully request allowance of independent Claims 31 and 45.

Dependent Claims 19-30 depend from independent Claim 17, and dependent Claims 33-44 depend from independent Claim 31. Independent Claims 17 and 31 are shown above to be allowable. Therefore, Claims 19-30 and 33-44 are allowable as depending from an allowable base claim and as defining further distinctions over the cited references. Thus, Applicants respectfully request allowance of Claims 19-30 and 33-44.

In particular, dependent Claim 21 recites, in part, "maintaining a connection cache to said one or more data sources" and Claim 23 recites, in part, "maintaining a page cache containing said Web page". The Examiner states that the elements of Claims 21 and 23 are inherent in Rogers. Office Action, p. 4. Rogers does not involve either a connection cache or a page cache. Applicants respectfully submit that neither a connection cache nor a page cache are inherent in Rogers. Applicants respectfully request that the Examiner indicate some teaching or suggestion of Rogers with respect to a page cache and a connection cache. Therefore, Claims

21 and 23 are patentable over Rogers. Thus, Applicants respectfully request allowance of Claims 21 and 23.

Also, Claim 24 recites "wherein said page server includes tag-based text templates for configuring said Web page." Applicants respectfully submit that Rogers does not teach or suggest this element of Claim 24. In contrast to the Examiner's assertion, Rogers merely teaches that data flows between a web server and a decision support system tool and that formatted report results are provided to a web browser. See Rogers, col. 11, lines 64-65 and col. 12, lines 18-24. Mere retrieval and formatting of data does not teach or suggest a "tag-based text template for configuring said Web page". Therefore, Claim 24 is patentable over Rogers. Thus, Applicants respectfully request allowance of Claim 24.

Dependent Claims 35, 37 and 38 are also patentable for at least the reasons discussed above in association with Claims 21, 23 and 24. Thus, Applicants respectfully request allowance of Claims 35, 37 and 38.

CONCLUSION

Applicants have now made an earnest attempt to place this case in condition for immediate allowance. For the foregoing reasons and for other reasons clearly apparent, Applicants respectfully request reconsideration and allowance of Claims 17-45.

Although Applicants believe that no other fees are due, the Commissioner is hereby authorized to charge any fees or credit any overpayments to Deposit Account No. 02-0384 of Baker Botts L.L.P.

If there are matters that can be discussed by telephone to further the prosecution of this application, Applicants respectfully request that the Examiner call its attorney at the number listed below.

Respectfully submitted,
BAKER BOTTS L.L.P.
Attorneys for Applicants

Terry J. Stalford
Reg. No. 39,522

Correspondence Address:
Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas, Texas 75201-2980
Phone: 214-953-6984
Fax: 214-661-4984

Date: 11/27/01

Notice of Allowability	Application No. 09/234,048	Applicant(s) LOWERY ET AL.
	Examiner Rehana Perveen	Art Unit 2182

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *Reconsideration Request filed on 11/27/01*.
2. ☒ The allowed claim(s) is/are *17-45*.
3. ☒ ~~The drawings filed on *19 January 1999* are accepted by the Examiner~~.
4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
      1. ☐ Certified copies of the priority documents have been received.
      2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
      3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.
5. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
   (a) ☐ The translation of the foreign language provisional application has been received.
6. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application. **THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.**

7. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

8. ☒ CORRECTED DRAWINGS must be submitted.
   (a) ☒ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
      1) ☐ hereto or 2) ☒ to Paper No. 7.
   (b) ☐ including changes required by the proposed drawing correction filed _____, which has been approved by the Examiner.
   (c) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No. _____.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the top margin (not the back) of each sheet. The drawings should be filed as a separate paper with a transmittal letter addressed to the Official Draftsperson.

9. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1☐ Notice of References Cited (PTO-892)
2☐ Notice of Informal Patent Application (PTO-152)
3☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
4☐ Interview Summary (PTO-413), Paper No.____ .
5☐ Information Disclosure Statements (PTO-1449), Paper No. ____.
6☐ Examiner's Amendment/Comment
7☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
8☐ Examiner's Statement of Reasons for Allowance
9☐ Other

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

7590 02/08/2002

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980

EXAMINER	
PERVEEN, REHANA	
ART UNIT	CLASS-SUBCLASS
2182	710-005000

DATE MAILED: 02/08/2002

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D	6557

TITLE OF INVENTION: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

TOTAL CLAIMS	APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
29	nonprovisional	NO	$1280	$0	$1280	05/08/2002

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE REFLECTS A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE APPLIED IN THIS APPLICATION. THE PTOL-85B (OR AN EQUIVALENT) MUST BE RETURNED WITHIN THIS PERIOD EVEN IF NO FEE IS DUE OR THE APPLICATION WILL BE REGARDED AS ABANDONED.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above. If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is changed, pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above and notify the United States Patent and Trademark Office of the change in status, or

B. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

If the SMALL ENTITY is shown as NO:

A. Pay TOTAL FEE(S) DUE shown above, or

B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check the box below and enclose the PUBLICATION FEE and 1/2 the ISSUE FEE shown above.

☐ Applicant claims SMALL ENTITY status. See 37 CFR 1.27.

II. PART B - FEE(S) TRANSMITTAL should be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). Even if the fee(s) have already been paid, Part B - Fee(s) Transmittal should be completed and returned. If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PTOL-85 (REV. 07-01) Approved for use through 01/31/2004.

PART B - FEE(S) TRANSMITTAL

Complete and mail this form, together with applicable fee(s), to:

Box ISSUE FEE
Assistant Commissioner for Patents
Washington, D.C. 20231

MAILING INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections or use Block 1)

7590 02/08/2002

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980

Note: The certificate of mailing below can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing.

Certificate of Mailing

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D	6557

TITLE OF INVENTION: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

TOTAL CLAIMS	APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
29	nonprovisional	NO	$1280	$0	$1280	05/08/2002

EXAMINER	ART UNIT	CLASS-SUBCLASS
PERVEEN, REHANA	2182	710-005000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363). Use of PTO form(s) and Customer Number are recommended, but not required.

❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47) attached.

2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 _____
2 _____
3 _____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) ❑ individual ❑ corporation or other private group entity ❑ government

4a. The following fee(s) are enclosed:

❑ Issue Fee
❑ Publication Fee
❑ Advance Order - # of Copies _____

4b. Payment of Fee(s):

❑ A check in the amount of the fee(s) is enclosed.
❑ Payment by credit card. Form PTO-2038 is attached.
❑ The Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number _____ (enclose an extra copy of this form).

The COMMISSIONER OF PATENTS AND TRADEMARKS is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.

(Authorized Signature) (Date)

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending on the needs of the individual case. Any comments on the amount of time required to complete this form should be sent to the Chief Information Officer, United States Patent and Trademark Office, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND FEES AND THIS FORM TO: Box Issue Fee, Assistant Commissioner for Patents, Washington, D.C. 20231

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMIT THIS FORM WITH FEE(S)

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D	6557

7590 02/08/2002

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980
UNITED STATES

EXAMINER	
PERVEEN, REHANA	
ART UNIT	**PAPER NUMBER**
2182	

DATE MAILED: 02/08/2002

Determination of Patent Term Extension under 35 U.S.C. 154 (b)
(application filed after June 7, 1995 but prior to May 29, 2000)

The patent term extension is 0 days. Any patent to issue from the above identified application will include an indication of the 0 day extension on the front page.

If a continued prosecution application (CPA) was filed in the above-identified application, the filing date that determines patent term extension is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) system. (http://pair.uspto.gov)

PTOL-85 (REV. 07-01) Approved for use through 0[illegible] 2004.

HM

#17
CK
6-5-02

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

32

NOTICE OF ALLOWANCE AND FEE(S) DUE

DOCKETED

7590 02/08/2002

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980

Issue Fee and Little
Formal Drawings - filed
Due: May 8, 2002
Pay 4.8.02

EXAMINER	
PERVEEN, REHANA	
ART UNIT	CLASS-SUBCLASS
2182	710-005000

DATE MAILED: 02/08/2002
066241.0119

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/234,048	01/19/1999	KEITH LOWERY	~~02577.P001D~~	6557

TITLE OF INVENTION: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

TOTAL CLAIMS	APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
29	nonprovisional	NO	$1280	$0	$1280	05/08/2002

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE REFLECTS A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE APPLIED IN THIS APPLICATION. THE PTOL-85B (OR AN EQUIVALENT) MUST BE RETURNED WITHIN THIS PERIOD EVEN IF NO FEE IS DUE OR THE APPLICATION WILL BE REGARDED AS ABANDONED.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above. If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is changed, pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above and notify the United States Patent and Trademark Office of the change in status, or

B. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

If the SMALL ENTITY is shown as NO:

A. Pay TOTAL FEE(S) DUE shown above, or

B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check the box below and enclose the PUBLICATION FEE and 1/2 the ISSUE FEE shown above.

☒ Applicant claims SMALL ENTITY status. See 37 CFR 1.27.

II. PART B - FEE(S) TRANSMITTAL should be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). Even if the fee(s) have already been paid, Part B - Fee(s) Transmittal should be completed and returned. If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

Wrapper ✓

RVF Docketed N/A

Reference(s) _____

OIPE
MAY 20 2002
PATENT & TRADEMARK OFFICE

Applicant or Patentee: Keith Lowery, Andrew S. Levine and Ronald L. Howell
Serial or Patent No. 09/234,041
Filed or Issued. January 19, 1999
Title: System and Method for Managing Dynamic Web Page Generation Requests
Attorney's Docket 06824-0119

VERIFIED STATEMENT (DECLARATION) CLAIMING SMALL ENTITY STATUS (37 CFR 1.9(f) & 1.27(c)) -- SMALL BUSINESS CONCERN

I hereby declare that I am an official of the small business concern empowered to act on behalf of the concern identified below:

Name of Small Business Concern: epicRealm Operating Inc.
Address of Small Business Concern: 2435 N. Central Expressway
Palisades Central II, Suite 300
Richardson, TX 75080

I hereby declare that the above-identified small business concern qualifies as a small business concern as defined in 13 CFR 121.12, and reproduced in 37 CFR 1.9(d), for purposes of paying reduced fees to the United States Patent and Trademark Office, in that the number of employees of the concern, including those of its affiliates, does not exceed 500 persons. For purposes of this statement, (1) the number of employees of the business concern is the average over the previous fiscal year of the concern of the persons employed on a full-time, part-time or temporary basis during each of the pay periods of the fiscal year, and (2) concerns are affiliates of each other when either one concern controls, directly or indirectly, or has the power to control the other, or a third party or parties controls or has the power to control both.

I hereby declare that rights under contract or law have been conveyed to and remain with the small business concern identified above with regard to the invention, entitled System and Method for Managing Dynamic Web Page Generation Requests by inventors, Keith Lowery, Andrew S. Levine and Ronald L. Howell described in the specification filed January 19, 1999.

If the rights held by the above-identified small business concern are not exclusive, each individual, concern or organization having rights in the invention is listed below, and no rights to the invention are held by any person, other than the inventor, who would not qualify as an independent inventor under 37 CFR 1.9(c) if that person made the invention, or by any concern which would not qualify as a small business concern under 37 CFR 1.9(d), or a nonprofit organization under 37 CFR 1.9(e):

NONE

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b)).

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, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

Name of Person Signing: Bradley A. Carl
Title of Person if other than owner: Vice President and General Counsel
Address of Person Signing: 2435 N Central Expressway
Palisades Central II Suite 400
Richardson, TX 75080

Signature:

Date: 5/7/02

DAL01:471737.1

TOTAL P.02

OK to Enter

PART B - FEE(S) TRANSMITTAL

Complete and mail this form, together with applicable fee(s), to: 5-20-02

Box ISSUE FEE
Assistant Commissioner for Patents
Washington, D.C. 20231

MAILING INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections or use Block 1)

7590 02/08/2002

Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue
Suite 600
Dallas, TX 75201-2980

Note: The certificate of mailing below can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing.

Certificate of Mailing

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above on the date indicated below.

Carol A. Donahue (Depositor's name)
Carol A. Donahue (Signature)
May 7, 2002 (Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/234,048	01/19/1999	KEITH LOWERY	02577.P001D	6557

TITLE OF INVENTION: SYSTEM AND METHOD FOR MANAGING DYNAMIC WEB PAGE GENERATION REQUESTS

TOTAL CLAIMS	APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
29	nonprovisional	NO	$1280	$0	$1280	05/08/2002

EXAMINER	ART UNIT	CLASS-SUBCLASS
PERVEEN, REHANA	2182	710-005000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363). Use of PTO form(s) and Customer Number are recommended, but not required.

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47) attached.

2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Baker Botts L.L.P.
2
3

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE: epicRealm Operating Inc.

(B) RESIDENCE: (CITY and STATE OR COUNTRY): Richardson, Texas

Please check the appropriate assignee category or categories (will not be printed on the patent) ☐ individual ☒ corporation or other private group entity ☐ government

4a. The following fee(s) are enclosed:

☒ Issue Fee

☐ Publication Fee

☐ Advance Order - # of Copies 10

4b. Payment of Fee(s):

☒ A check in the amount of the fee(s) is enclosed.

☐ Payment by credit card. Form PTO-2038 is attached.

☒ The Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number 02-0384 (enclose an extra copy of this form).

The COMMISSIONER OF PATENTS AND TRADEMARKS is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.

(Authorized Signature) (Date) 5-7-02

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending on the needs of the individual case. Any comments on the amount of time required to complete this form should be sent to the Chief Information Officer, United States Patent and Trademark Office, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND FEES AND THIS FORM TO: Box Issue Fee, Assistant Commissioner for Patents, Washington, D.C. 20231

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

05/21/2002 TBESHAH2 00000022 09234048

01 FC:242 640.00 OP
02 FC:561 30.00 OP

TRANSMIT THIS FORM WITH FEE(S)

PTOL-85 (REV. 07-01) Approved for use through 01/31/2004. OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

#18 B TO
J.G.



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	Keith Lowery et al.
Serial No.:	09/234,048
Filed:	January 19, 1999
Group No.:	2182
Examiner:	Rehana Perveen
Notice of Allowance Mailed:	February 8, 2002
Confirmation No.:	6557
Title:	*System and Method for Managing Dynamic Web Page Generation Requests*

ATTENTION: OFFICIAL DRAFTSMAN

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Box Issue Fee, Assistant Commissioner for Patents, Washington, D.C. 20231, on the date shown below.

Name

3/22/02
Date of Signature

Box Issue Fee
Assistant Commissioner for Patents
Washington, DC 20231

Dear Sir:

TRANSMITTAL OF FORMAL DRAWINGS

Pursuant to the Notice of Allowability mailed February 8, 2002, transmitted herewith for filing in the above-identified patent application are four (4) sheets of formal drawings.

Respectfully submitted,
BAKER BOTTS L.L.P.
Attorneys for Applicants

Matthew B. Talpis
Registration No. 45,152

Date: March 22, 2002
2001 Ross Avenue, Suite 600
Dallas, TX 75201-2980
(214) 953-6984
Attorney Docket No.066241.0119
MBT:cad

DAL01:664045.1

02.02

FIG. 1

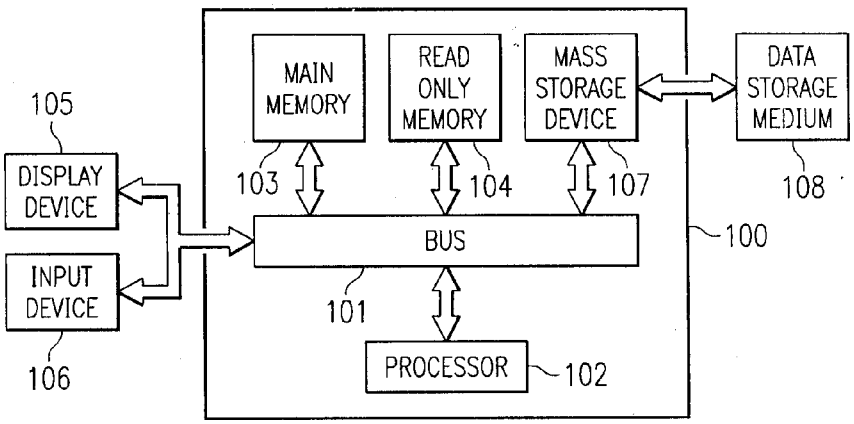


FIG. 2
(PRIOR ART)

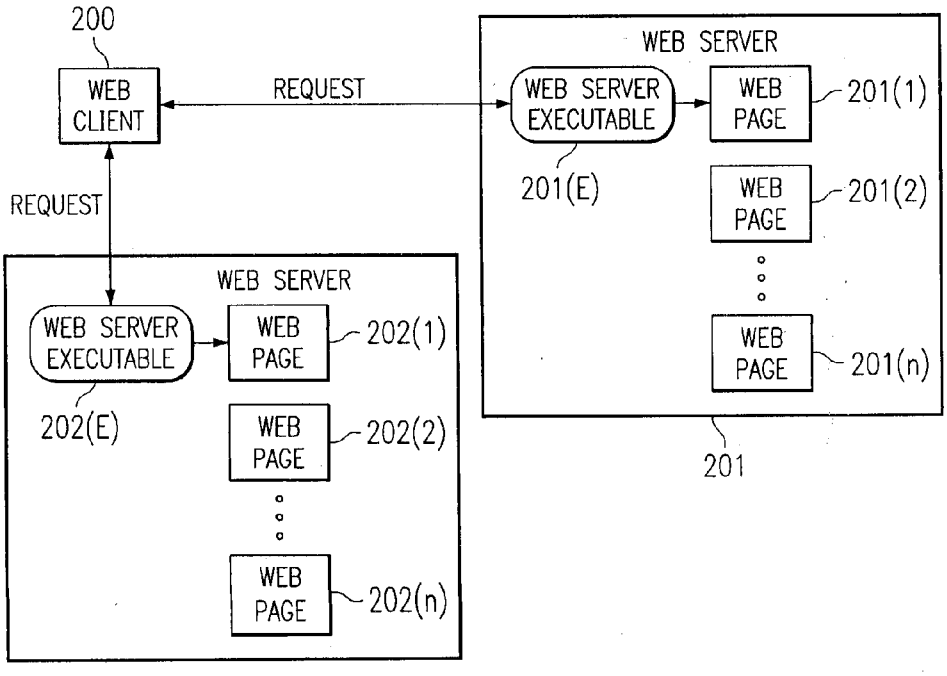
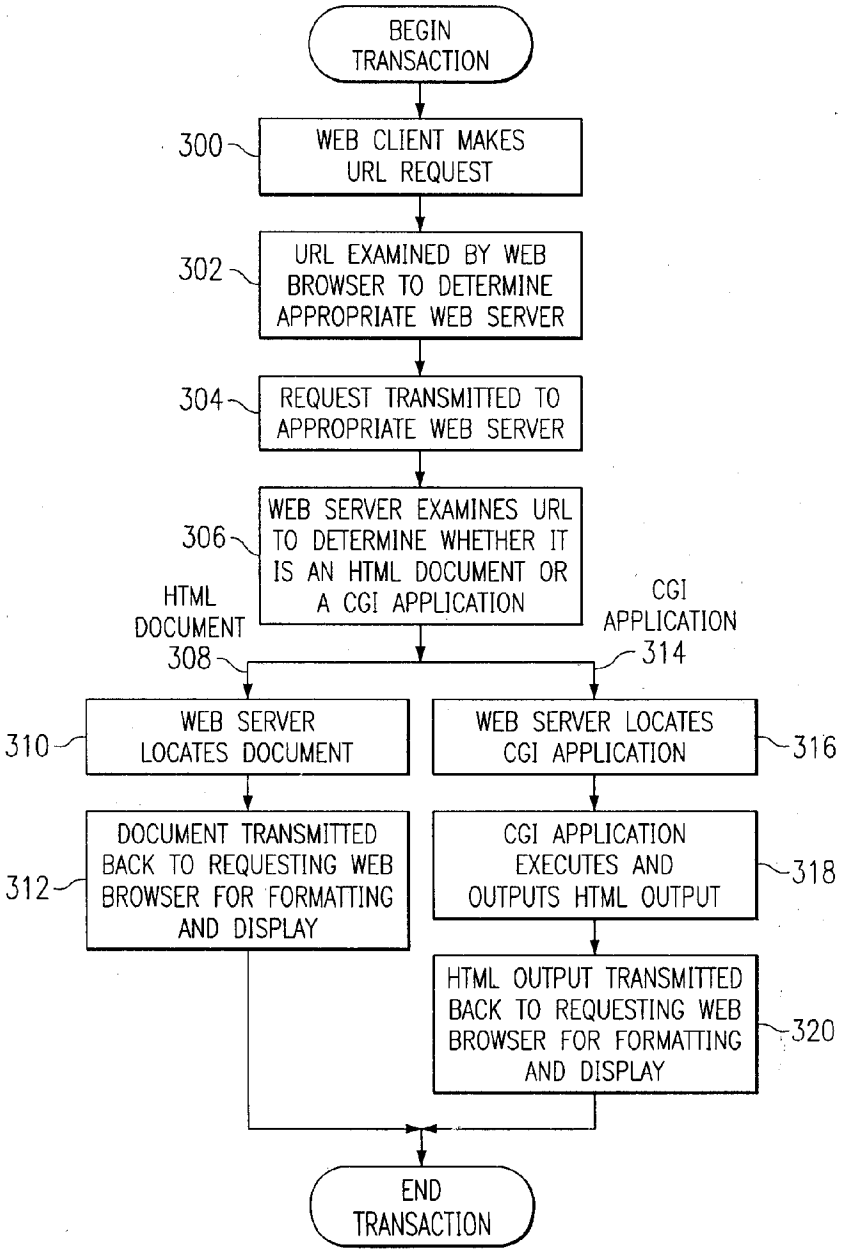


FIG. 3
(PRIOR ART)



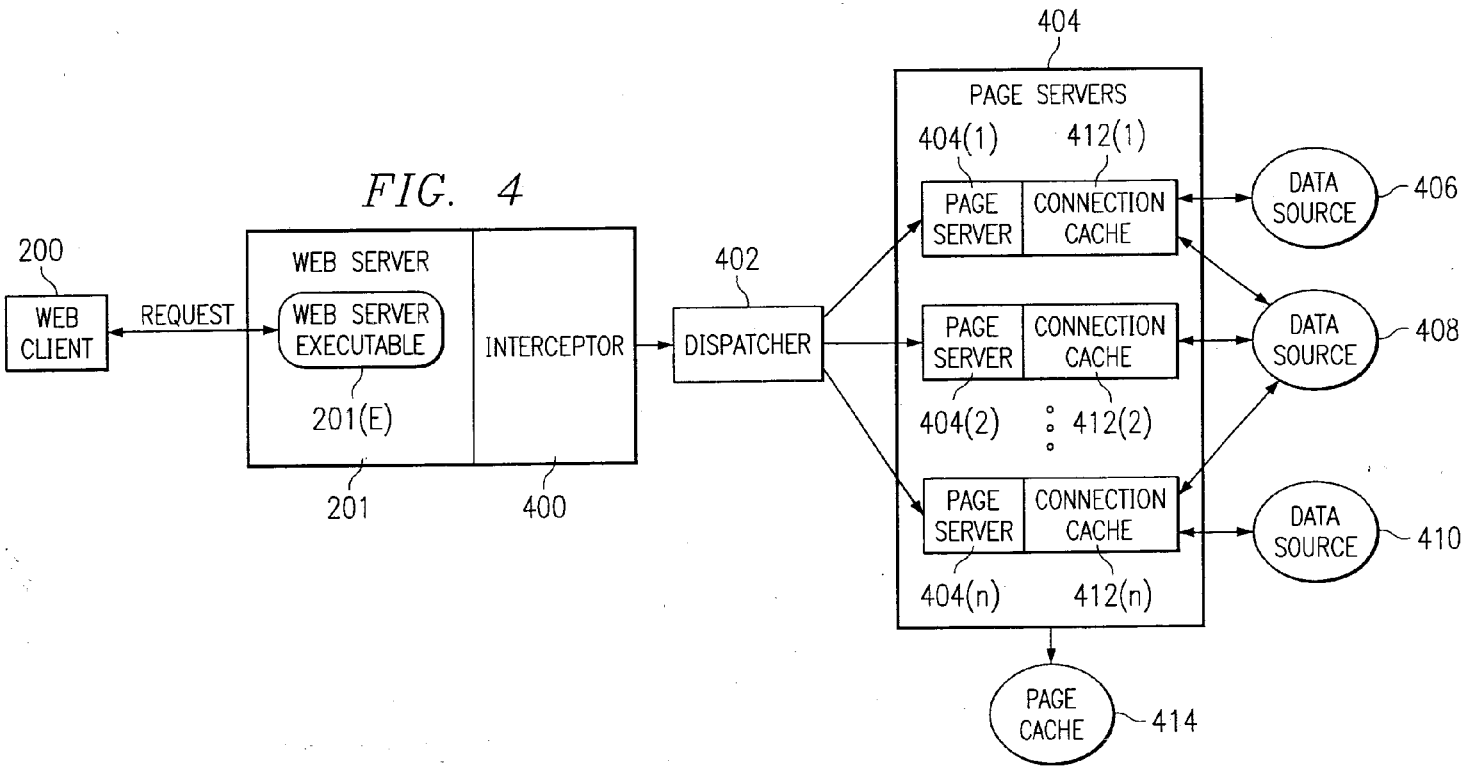

200
WEB CLIENT
REQUEST
WEB SERVER
WEB SERVER EXECUTABLE
201(E)
201
INTERCEPTOR
400
402
DISPATCHER
404
PAGE SERVERS
404(1)
412(1)
PAGE SERVER
CONNECTION CACHE
404(2)
412(2)
404(n)
412(n)
DATA SOURCE
406
408
410
PAGE CACHE
414

FIG. 4

FIG. 5

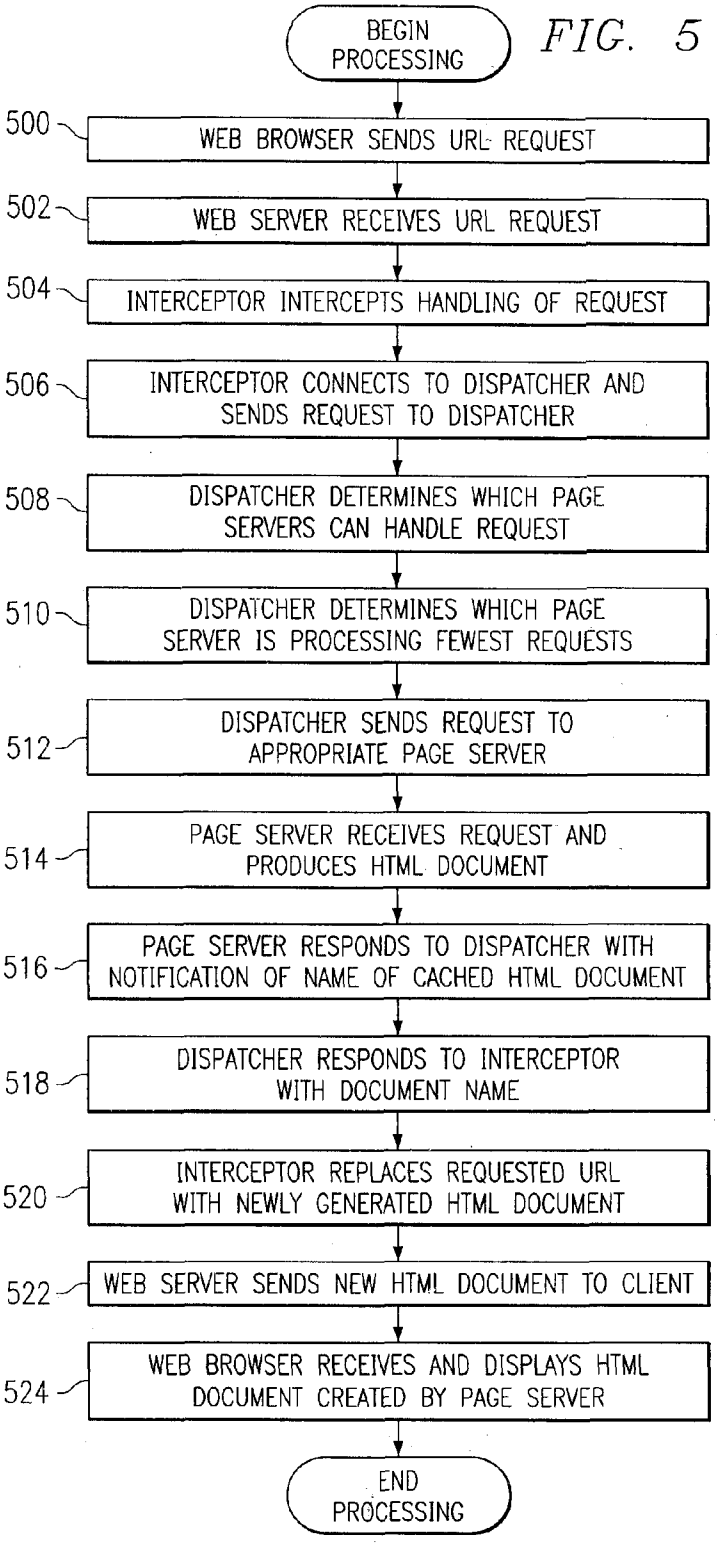

BEGIN PROCESSING
500
WEB BROWSER SENDS URL REQUEST
502
WEB SERVER RECEIVES URL REQUEST
504
INTERCEPTOR INTERCEPTS HANDLING OF REQUEST
506
INTERCEPTOR CONNECTS TO DISPATCHER AND SENDS REQUEST TO DISPATCHER
508
DISPATCHER DETERMINES WHICH PAGE SERVERS CAN HANDLE REQUEST
510
DISPATCHER DETERMINES WHICH PAGE SERVER IS PROCESSING FEWEST REQUESTS
512
DISPATCHER SENDS REQUEST TO APPROPRIATE PAGE SERVER
514
PAGE SERVER RECEIVES REQUEST AND PRODUCES HTML DOCUMENT
516
PAGE SERVER RESPONDS TO DISPATCHER WITH NOTIFICATION OF NAME OF CACHED HTML DOCUMENT
518
DISPATCHER RESPONDS TO INTERCEPTOR WITH DOCUMENT NAME
520
INTERCEPTOR REPLACES REQUESTED URL WITH NEWLY GENERATED HTML DOCUMENT
522
WEB SERVER SENDS NEW HTML DOCUMENT TO CLIENT
524
WEB BROWSER RECEIVES AND DISPLAYS HTML DOCUMENT CREATED BY PAGE SERVER
END PROCESSING

Exhibit 8

IN THE UNITED STATES DISTRICT COURT
OF THE EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION

EPICREALM, LICENSING, LLC	§	
	§	
v	§	**No. 2:05CV163**
	§	
AUTOFLEX LEASING, INC., et al.	§	

EPICREALM, LICENSING, LLC	§	
	§	
v	§	**No. 2:05CV356**
	§	
FRANKLIN COVEY CO., et al.	§	

REPORT AND RECOMMENDATION
REGARDING CLAIM CONSTRUCTION

Pursuant to the provisions of 28 U.S.C. § 636(b)(1) and (3) and the Amended Order for the Adoption of Local Rules for Assignment of Duties to United States Magistrate Judges, the above-entitled and numbered cause of action was referred to the undersigned for pretrial purposes. Claim construction arguments in cause numbers 2:05-CV-163 and 2:05-CV-356 were combined, and Defendants submitted joint briefing. The Court conducted a claim construction hearing on July 13, 2006. This Report and Recommendation construes certain terms in United States Patent Nos. 5,894,554 ("the '554 Patent) and 6,415,335 (the '335 Patent).

I. BACKGROUND

The '554 Patent issued on April 13, 1999. The '335 Patent issued on July 2, 2002 and is a divisional application of the '554 Patent. The '554 Patent and the '335 Patent share a common specification.[1] The patents generally relate to managing Web sites. More particularly, the patents relate to managing dynamic Web page generation. Col. 2:15-23. The patents distinguish some Web pages as having a static nature that remains static until manually modified and other Web pages as being dynamic Web pages which contain content that is generated dynamically by retrieving the necessary requested data and generating the requested Web page dynamically. Col. 1:38-55. The patents describe prior art Web servers as handling both static and dynamic Web page requests. Col. 3:64 - Col. 4:37; Figures 2-3. The techniques described in the patents include routing a Web request from the Web server to a Page server. The Page server may than process the request, and the Web server is released to process other requests. Col. 2:20-35; Col. 4:54 - Col. 6:32. In this manner, dynamic Web pages may be generated by the Page servers.

Some of the claim construction disagreements involve common themes. For example, in general the Plaintiff construes various terms so that Web servers and Page servers do not have to be separate machines while the Defendants seek constructions that would include a separate machine concept. The Plaintiff also seeks constructions that do not include the concept of Uniform Resource Locators (URLs) while the Defendants add the term URL to some of the constructions they seek. Other conflicting claim construction positions are more specific to individual terms that are in dispute.

[1] References to the specification will refer to the column and line numbers of the '554 Patent.

II. APPLICABLE LAW

"It is a 'bedrock principle' of patent law that 'the claims of a patent define the invention to which the patentee is entitled the right to exclude.'" *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (en banc) (*quoting Innova/Pure Water Inc. v. Safari Water Filtration Sys., Inc.*, 381 F.3d 1111, 1115 (Fed. Cir. 2004)). In claim construction, courts examine the patent's intrinsic evidence to define the patented invention's scope. *See id.*; *C.R. Bard, Inc. v. U.S. Surgical Corp.*, 388 F.3d 858, 861 (Fed. Cir. 2004); *Bell Atl. Network Servs., Inc. v. Covad Commc'ns Group, Inc.*, 262 F.3d 1258, 1267 (Fed. Cir. 2001). This intrinsic evidence includes the claims themselves, the specification, and the prosecution history. *See Phillips*, 415 F.3d at 1314; *C.R. Bard, Inc.*, 388 F.3d at 861. Courts give claim terms their ordinary and accustomed meaning as understood by one of ordinary skill in the art at the time of the invention in the context of the entire patent. *Phillips*, 415 F.3d at 1312-13; *Alloc, Inc. v. Int'l Trade Comm'n*, 342 F.3d 1361, 1368 (Fed. Cir. 2003).

The claims themselves provide substantial guidance in determining the meaning of particular claim terms. *Phillips*, 415 F.3d at 1314. First, a term's context in the asserted claim can be very instructive. *Id.* Other asserted or unasserted claims can also aid in determining the claim's meaning because claim terms are typically used consistently throughout the patent. *Id.* Differences among the claim terms can also assist in understanding a term's meaning. *Id.* For example, when a dependent claim adds a limitation to an independent claim, it is presumed that the independent claim does not include the limitation. *Id.* at 1314–15.

Claims "must be read in view of the specification, of which they are a part." *Id.* (quoting *Markman v. Westview Instruments, Inc.*, 52 F.3d 967, 978 (Fed. Cir. 1995)). "[T]he specification 'is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single

best guide to the meaning of a disputed term.'" *Id.* (*quoting Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)); *Teleflex, Inc. v. Ficosa N. Am. Corp.*, 299 F.3d 1313, 1325 (Fed. Cir. 2002). This is true because a patentee may define his own terms, give a claim term a different meaning than the term would otherwise possess, or disclaim or disavow the claim scope. *Phillips*, 415 F.3d at 1316. In these situations, the inventor's lexicography governs. *Id.* Also, the specification may resolve ambiguous claim terms "where the ordinary and accustomed meaning of the words used in the claims lack sufficient clarity to permit the scope of the claim to be ascertained from the words alone." *Teleflex, Inc.*, 299 F.3d at 1325. But, "although the specification may aid the court in interpreting the meaning of disputed claim language, particular embodiments and examples appearing in the specification will not generally be read into the claims." *Comark Commc'ns, Inc. v. Harris Corp.*, 156 F.3d 1182, 1187 (Fed. Cir. 1998); *see also Phillips*, 415 F.3d at 1323. The prosecution history is another tool to supply the proper context for claim construction because a patent applicant may also define a term in prosecuting the patent. *Home Diagnostics, Inc., v. Lifescan, Inc.*, 381 F.3d 1352, 1356 (Fed. Cir. 2004) ("As in the case of the specification, a patent applicant may define a term in prosecuting a patent.").

Although extrinsic evidence can be useful, it is "less significant than the intrinsic record in determining 'the legally operative meaning of claim language.'" *Phillips*, 415 F.3d at 1317 (*quoting C.R. Bard, Inc.*, 388 F.3d at 862). Technical dictionaries and treatises may help a court understand the underlying technology and the manner in which one skilled in the art might use claim terms, but technical dictionaries and treatises may provide definitions that are too broad or may not be indicative of how the term is used in the patent. *Id.* at 1318. Similarly, expert testimony may aid a court in understanding the underlying technology and determining the particular meaning of a term

in the pertinent field, but an expert's conclusory, unsupported assertions as to a term's definition is entirely unhelpful to a court. *Id.* Generally, extrinsic evidence is "less reliable than the patent and its prosecution history in determining how to read claim terms." *Id.*

The patents in suit also contain means-plus-function limitations that require construction. Where a claim limitation is expressed in "means plus function" language and does not recite definite structure in support of its function, the limitation is subject to 35 U.S.C. § 112, ¶ 6. *Braun Med., Inc. v. Abbott Labs.*, 124 F.3d 1419, 1424 (Fed. Cir. 1997). In relevant part, 35 U.S.C. § 112, ¶ 6 mandates that "such a claim limitation 'be construed to cover the corresponding structure . . . described in the specification and equivalents thereof.'" *Id.* (citing 35 U.S.C. § 112, ¶ 6). Accordingly, when faced with means-plus-function limitations, courts "must turn to the written description of the patent to find the structure that corresponds to the means recited in the [limitations]." *Id.*

Construing a means-plus-function limitation involves multiple inquiries. "The first step in construing [a means-plus-function] limitation is a determination of the function of the means-plus-function limitation." *Medtronic, Inc. v. Advanced Cardiovascular Sys., Inc.*, 248 F.3d 1303, 1311 (Fed. Cir. 2001). Once a court has determined the limitation's function, "the next step is to determine the corresponding structure disclosed in the specification and equivalents thereof." *Id.* A "structure disclosed in the specification is 'corresponding' structure only if the specification or prosecution history clearly links or associates that structure to the function recited in the claim." *Id.* Moreover, the focus of the "corresponding structure" inquiry is not merely whether a structure is capable of performing the recited function, but rather whether the corresponding structure is "clearly linked or associated with the [recited] function." *Id.*

III. DISCUSSION

A. Disputed Claim Terms

1. "Web Page"

"Web page" is utilized in asserted claims 1, 3, 6, 7, 9, and 11 of the '554 Patent and 1, 4, 7, and 8 of the '335 Patent. The Plaintiff asserts that "Web page" does not need construction. Alternatively, if construed, the Plaintiff asserts that the proper construction of the term is "content displayable through a Web browser." The Defendants assert that the terms should be construed as "an HTML document accessible through a URL."

The Court first notes that the Plaintiff's original briefing expressed concern that the Defendants construction implies that dynamically generated Web pages are not included within the term "Web page." The Court finds this concern somewhat unfounded, and it is noted that the Defendants clearly referred to "Web page" in their briefing and oral argument as encompassing both static and dynamic Web pages.

The other assertions by the parties primarily revolve around two issues, the inclusion of terms "HTML" and "URL" in the construction. The Plaintiff argues that the term "content" is utilized in the specification at least twice to describe what is displayed on a Web page. Col. 1:47-51; Col. 7:23-26. Further, the Plaintiff points out that in the Defendants' own briefing Web pages are referred to as containing content. Defendants' Brief at 11-12. The Plaintiff further asserts that the Defendants are also attempting to read in limitations from the specification and that such a construction would exclude documents formatted in other formats such as SGML, XHTML, XML, and JPG.

The Defendants state that HTML is a software language and argue that as described within the specification HTML documents are what are sent back as Web pages. Col. 1:18-22; Figures 3

and 5. The Defendants also cite a Microsoft Press Computer Dictionary definition which states that "A Web page consists of an HTML file…" Defendants' Brief at 9. The Defendants further assert that "content" in the specification refers to information included in a Web page and that such content itself does not form a Web page.

The Court notes that the specification does appear to consistently refer to the HTML language and does not mention other software languages. However, the Defendant does not identify persuasive support within the specification that the invention must be limited to only one type of software language. Moreover, Defendants have not persuaded the Court that in light of the specification one skilled in the art would assume a Web page as referred to in the patents could only be generated with the HTML language. Further, upon review of the whole specification and claims, with respect to Web pages the described concepts are not related to the intricacies of what particular programming languages are used to display a Web page but rather merely the higher level differentiation of static pre-existing Web pages verse dynamically generated Web pages. In these circumstances, the Court finds it improper to incorporate the limitation of HTML within the more general term Web page that is utilized in the claims themselves.

With regard to the URL concept, the Plaintiff asserts that the Defendants' definition adds additional complexity to the claim construction as the meaning and scope of "accessible through a URL" could itself require construction. Further, citing the extrinsic evidence that the Defendants themselves put before the Court, the Plaintiff argues that it is known that when a Web browser sends a request what is actually sent does not match what is commonly known as a full URL. In oral argument the Plaintiff asserted that the Defendants' extrinsic evidence shows a URL as "http_URL ="http:" "//" host [":" port] [abs_path] thus requiring four components: a protocol (HTTP), a host,

a port and an absolute path. Further the Plaintiffs argue that this same extrinsic evidence shows that a request most commonly is structured for example "GET / pub / WWW / TheProject.html HTTP / 1.0" The Plaintiff asserts that this also emphasizes the concern over the ambiguity of the meaning of "accessible through a URL."

The Defendants turn to the specification, which includes the statement "[a] URL is a Web address that identifies the Web page and its location on the Web." Col. 1:30-33. The Defendants also note the language that states "[w]hen the appropriate Web site receives the URL, the Web page corresponding to the requested URL is located…." Col. 1:33-34. Further, the Defendants point to other examples in the specification, such as Figure 3, which refer to the Web browser sending the URL request. At oral argument, the Defendants also stated that what is sent by a Web browser does include the URL information.

The arguments of the parties highlight some of the concerns the Court has with the inclusion of the term URL. A definition of the meaning of URL within the usage of the patent would further be necessitated as URL is alternatively referred to as "what is examined by the Web browser," a URL request is received by the Web server, and a URL request can be sent to a page server. Col. 4:13-14; Col. 8:28-32; Col. 8:38-39. The specification does not make clear what is the particular structure and content that is meant by the use of the term "URL" at each of these stages of the process. Again, as with the HTML term, this is not surprising as the specification and claims as a whole do not focus on the particular type of request that is made or the particular structure/content of a request as it processed through the system beyond the static and dynamic distinction discussed above. Further, to add the term "accessible" to the construction would necessitate further claim construction as to what "accessible" means. As described within the specification, a Web page is

a mechanism through which static and dynamic content may be displayed. The particular addressing mechanism at each step of the processing of a dynamic Web page is not noted in the specification to be a requirement or of particular importance to the claimed invention. An inclusion of the term URL would improperly incorporate limitations from the specification for the term Web page which has a meaning that is adequately described within the full context of the specification.[2] **Thus, the Court construes "Web page" to mean "Web content displayable through a Web browser."**

2. "Request"

"Request" is utilized in asserted claims 1, 9, 11 of the '554 Patent and 1, 15, and 29 of the '335 Patent. The Plaintiff asserts that the proper construction of "request" is "a message that asks for content." The Defendants assert that the term should be construed as "a message containing a URL that asks for a Web page specified by the URL."

The Court first notes that the term "request" generally appears in the claims in different circumstances. In some claims (for example claim 1 of the '554 Patent), request is first utilized with relation to "a dynamic Web page generation request" and multiple references are then made to "said request." Such claims also use the term "other requests" which imply a request different from "said request." Other claims (for example claim 15 of the '335 Patent) begin with the general use of "a request" but later refer to "dynamically generating a page in response to said request." These claims also refer to "other requests." As noted in the specification, a request can refer to static Web

[2] Additional discussion regarding the inclusion of URL limitations is also provided below with regard to the term "request."

pages (such as a static document) or dynamic Web pages (such as a Web page dynamically generated by an application). Col. 1:38-56, Col. 4:16-32 Thus, in its general use, request is not limited to dynamic or static requests. It is also noted that both types of claims consistently refer to either "Web page" generation or "page" generation. The specification uniformly refers to pages in the context of Web pages. Thus, it is not unreasonable that in light of the consistent specification one would interpret "requests" as relating to requests for Web pages.

The primary point of distinction between the proposed constructions relates to inclusion of "URL" in the construction of request. The specification notes that "a Web browser allows a Web client to request a particular Web page from a Web site by specifying a Uniform Resource Locator (URL). A URL is a Web address that identifies the Web page and its location on the Web." Col. 1:29-32

The Defendants cites numerous places in the specification text in which "URL request" is utilized to refer to what is requested. The Plaintiff asserts that a request may be made at multiple points, such as shown in Figure 4 between the Web client and Web server, between the Web server and the Dispatcher and between the Dispatcher and the Page servers. The Plaintiff further argues that once a message is received by a Web server a full URL is not utilized subsequently, such as by Page servers.

The Defendants counter by noting numerous specification citations to the term "URL request" including "route the URL request to a Page server" and "the dispatcher sends the URL request to an appropriate Page server." Col. 6:9-10, Col. 8:38-39, *See* Defendants' Brief at 13-14. The Plaintiff responds that there is no teaching in the specification that a page server utilizes a URL.

Moreover, the Plaintiffs assert that it would be illogical for a request provided to the page server to utilize a URL as such address would refer to the Web server.

The Court notes that within the claims "request" is used in context of multiple steps of the page generation process. For example, within claim 1 a request may be provided to a Web server while "said request" is also received by a Page server. The specification provides varied and not always consistent uses of the terms "request" and "URL request." As noted above, URL request is often utilized. However, the more general term "request" is also often utilized. Col. 2:1-12; Col. 2:18-35; Col. 4:33-53; Col. 5:8-59; Col. 6:20-32; Col. 7:5-6. In at least two of these instances, language stating "requests or 'hits'" is utilized. Col. 4:38-39; Col. 7:5-6. Further, the Defendants have not shown within the specification that a Page server utilizes a URL. Thus, even when "URL request" is provided in the specification it is not clear that such request is required to contain a URL or is merely a request generated from an initial URL provided at a Web client. To require "request" to include a URL would thus include limitations that the specification does not clearly support and clearly require.

The remaining dispute between the parties relates to the use of "content" verse "Web page." This dispute relates to the underlying meaning of the term Web page as discussed above and thus does not need to be re-addressed. As the Court has noted, the context of the patent is uniformly directed towards Web page requests. Under the guidance provided in *Phillips,* it is appropriate when viewing the specification and the language of the claims themselves to limit "request" to Web page applications. **Thus, the Court construes "request" to mean "a message that asks for a Web page" (with the term Web page having the construction provided herein)**

3. "Page Server"

"Page server" is utilized in asserted claims 1, 4, 7, and 9-11 of the '554 Patent and 1, 2, 5, 8, 15-16, 19, 22 and 29 of the '335 Patent. The Plaintiff asserts that the proper construction of "page server" is "a processing system operable to receive a request and dynamically generate content in response to the request." The Defendants assert that the term should be construed as "page-generating software that generates a dynamic Web page on a machine separate from the Web server machine." In the claim construction Oral Argument, the Plaintiff agreed to the use of "page-generating software" in place of the term "a processing system" as previous proposed by the Plaintiff. The differences between the parties with regard to the use of "content" verse "Web page" are rooted in the basic dispute over the meaning of Web page as discussed above. Both parties include the concept of dynamic generation in their proposed constructions.

In the post oral hearing briefing, the parties each acknowledged that the primary dispute regarding the construction of "page server" is whether the Page server has to be on a machine separate from the Web server. As discussed below, the Court agrees with the Plaintiff with regard to this point of dispute.

Each party points to the specification to support their asserted position. The Plaintiff asserts that the specification includes statements that indicate that the Page server could operate on the same machine as the Web server. In particular, the Plaintiffs have pointed to passages which state:

> FIG. 1 illustrates a typical computer system 100 in which the present invention operates. Col. 2:66-67.
>
> The preferred embodiment of the present invention is implemented as a software module, which may be executed on a computer system such as computer system 100 in a conventional manner. Col. 3:55-58.

Figure 1 illustrates a computer system having a processor, bus, memory and mass storage. Further, it is stated that "the preferred embodiment of the present invention" may be implemented on a personal computer or alternatively a workstation. Col. 2:67-Col. 3:5. The Plaintiff asserts that this language is consistent with the specification as a whole by asserting that the specification describes a partitioned software architecture in which in some embodiments the software modules may all reside on the same machine and in other embodiments the software modules may reside on different computers.

The Plaintiff also points to a passage that describes an embodiment that does not have the advantage of "off-loading the processing of Web requests from the Web server machine" to a separate machine. Col. 5:26-36. However, the Court notes that this passage makes specific reference to the division between a Web server and a Dispatcher, and it is not clear in this passage alone that the Page server is also included in this use of a single machine.

The Defendants argue that the specification describes a distinction over the prior art that amounts to an explicit characterization of the invention that disclaims the prior art. *See SciMed Life Sys. V. Advanced Cardiovascular Sys.*, 242 F.2d 1337, 1343 (Fed. Cir. 2001). In particular, the Defendants point to a passage of the specification that describes the multi-threading techniques of prior art Web servers. Col. 4:32-53. This passage concludes with "[t]he claimed invention addresses this need by utilizing a partitioned architecture to facilitate the creation and management of custom Web sites and servers." The Defendants assert that this clearly demonstrates that the purpose of the invention was to partition the various modules on separate machines. As to the passages cited by the Plaintiff, the Defendants assert those passages do not describe the entirety of the claimed invention. The Plaintiff asserts that the passage cited by the Defendants is directed toward the Web

site management "need" recited in the passage, and this need is addressed by a partitioned architecture.

The parties have thus each pointed to somewhat conflicting passages of the specification to support their positions. The passages cited by the Plaintiff establish that there is not a clear disavowal within the specification of the use of a partitioned software architecture on a single machine. The Defendants do correctly point to cases which stand for the proposition that when the specification makes clear that the invention does not include a particular feature than that feature is deemed to be outside of the reach of the claims of the patent. Defendants' Joint Sur-Reply, p. 8. However, in the specification before this Court, the specification does not make clear that the invention must only be operated on separate machines.

The Court construes "page server" to be "page-generating software that generates a dynamic Web page."

4. "Web Server"

"Web Server" is utilized in asserted claims 1 and 11 of the '554 Patent and 1-2 of the '335 Patent. The Plaintiff asserts that the proper construction of "Web server" is "a processing system capable of processing an HTTP request and producing a response to such a request." The Defendants assert that the terms should be construed as "a machine running a Web server executable capable of storing, locating, and returning Web pages in response to Web client requests." In the claim construction Oral Argument, the Plaintiff stated that it would agree to language including "software" in place of the Plaintiff's originally proposed "system" language similar to the Plaintiff's agreement with regard to "page server."

The focus point of the dispute between the parties is whether the term "Web server" requires a machine or whether the term may merely represent software or a combination of the two. Both the Plaintiff and Defendants cite conflicting extrinsic evidence to support their positions in the form of dictionaries, industry guides, and protocols. Some of the Plaintiff's extrinsic evidence includes citations to extrinsic evidence first brought before the Court by the Defendants. The conflicting extrinsic evidence presented by the parties fits the rationale presented in *Phillips* regarding the cautions that should be considered relating to such evidence.

Looking to the specification, the Plaintiff points to passages in which "Web server" is not used to describe a machine. In particular, the Plaintiff points to the statement in the specification that:

> The preferred embodiment of the present invention is implemented as a software module, which may be executed on a computer system such as computer system 100 in a conventional manner. Col. 3:55-58.

The Plaintiff also highlights the following passage:

> This embodiment is appropriate for Web servers such as Netsite™ from Netscape, that support such extensions. A number of public domain Web servers, such as NCSA™ from the National Center for Supercomputing Applications at the University of Illinois, Urbana-Champaign, however, do not provide support for this type of extension. Thus, in an alternate embodiment, Interceptor 400 is an independent module, connected via an 'intermediate program' to Web server 201. This intermediate program can be a simple CGI application program that connects Interceptor 400 to Web server 201. Alternate intermediate programs the perform the same functionality can also be implemented. Col. 4:63-Col. 5:7.

The Defendants counter that the specification uses the terms "Web server," "Web server executable" and "Web server machine" and that the proper interpretation is that the machine is referred to as a Web server machine, the software is referred to as the "Web server executable" and

the combination is referred to as a "Web server." To support this argument, the Defendants cite to various passages and figures in the specification. Figures 2-4; Col. 4:39-41; Col. 4:59-62; Col. 5:7-36.

While the Defendants may be correct that "Web server" may be utilized at times as indicating a combination of a machine and software, the specification clearly does not require the term "Web server" to include a machine. The passages of the specification noted above by the Plaintiff make clear that the Web server is contemplated to be at least in one embodiment, software. It is also noted that in general in other passages of the specification the term "Web server machine" is more often used when describing the machine component and "Web server" to describe merely the software component. Thus, for example, it is noted that the "Web servers process each of these requests on a single machine, namely the Web server machine," "Interceptor 400 resides on the Web server machine as an extension to Web server 201," and the Dispatcher "can, however, also reside on the same machine as the Web server." Col. 4:39-42; Col. 4:61-62; Col. 5:20-21. Passages such as these imply a utilization of the term "Web server" as the software module as opposed to the combination of both the machine and software. Thus, some usage of "Web server" implies just software and at other times implies a combination of software and hardware.

The Court construes "Web server" to be "software, or a machine having software, that receives Web page requests and returns Web pages in response to the requests.

5. "HTTP-complaint device"

"HTTP-complaint device" is utilized in asserted claims 15-16 and 19 of the '335 Patent. The Plaintiff asserts that "HTTP-complaint device" does not need construction. Alternatively, if

construed, the Plaintiff asserts that the proper construction is "a device that understands HTTP and whose behavior is affected by an HTTP request." The Defendants assert that the terms should be construed as "a machine running an executable capable of storing, locating and returning Web pages in response to Web client requests."

The Defendants assert the same construction for the terms "Web server" and "HTTP-complaint device." The Defendants' construction is based upon their assertion that the term does not appear in the specification and that the specification only discloses a Web server for performing the function described in the language surrounding the use of "HTTP-compliant device." As such, the Defendants assert that "HTTP-compliant device" should be construed the same as "Web server" as that is the only corresponding device described and enabled in the specification. To hold otherwise, assert the Defendants, would result in a claim that is overbroad and invalid for not being described and enabled.

Regarding the maxim that claims should be construed to be valid, in *Phillips* the Federal Circuit guidance states that this maxim is limited "to cases in which 'the court concludes, after applying all the available tools of claim construction, that the claim is still ambiguous.'" *Phillips,* 415 F.3d at 1327. The Court does not find that in light of the specification the term in question is ambiguous to one skilled in the art. Thus, the Defendants' validity concerns should be more properly addressed with regard to validity motions.

The Defendants also assert that the more general construction proposed by the Plaintiff would be so broad as to even encompass a Web client. However, the claim language itself makes clear that this concern is not valid as there is substantial functional language regarding what happens at the HTTP-compliant device including the transferring of a request from the HTTP-compliant device to

a page server, intercepting the request at the HTTP-compliant device, and concurrently processing other requests at the HTTP-compliant device.

The Defendants do however raise valid concerns over the Plaintiff's definition raising additional interpretation questions with regard to the meaning of "understands HTTP" and "behavior affected by an HTTP request." The Court agrees with the Defendants in this regard. The specification defines HTTP as "a communications protocol known as HyperText Transport Protocol (HTTP)." Col. 1:25-26. Mindful that not all terms in a claim need construction, **the Court adopts a construction of "HTTP-compliant device" to mean "a device that is compliant with the communication protocol known as HyperText Transport Protocol (HTTP)."**

6. "Said processing being performed by said page server while said Web server concurrently processes said other requests"

"Said processing being performed by said page server while said Web server concurrently processes said other requests" is utilized in asserted claims 1 and 11 of the '554 Patent and 1, 15 and 29 of the '335 Patent. The Plaintiff asserts that the "said processing…" phrase does not need construction. Alternatively, if construed, the Plaintiff asserts that the proper construction is "said processing being performed by said page server while said Web server processes said other requests at the same time." The Defendants assert that the phrase should be construed as "said processing being performed by said page server while said Web server executable processes other requests literally at the same time on a different machine."

Three main points of distinction exist between the parties: the inclusion of "executable" with regard to the use of Web server, the inclusion of "literally" with regard to the "same time" language,

and the inclusion of the concept of the Page server and Web servers being on different machines. With regard to the term "Web server" as utilized within the "said processing…" phrase, the Court finds that the construction the Court provided above for "Web server" is applicable, and thus the inclusion of the term executable does not need to be re-addressed with relation to the "said processing…" phrase. Similarly, with regard to the concept of different machines, the Court has addressed that concept above and does not need to re-address that concept here.

With regard to "concurrently," both parties agree that this term includes the concept of something occurring "at the same time;" however, there is still a dispute as to whether it must be "literally at the same time." The Defendants argue that "concurrently" should be analyzed in the context of the discussion in the patents of the prior art time-interleaved multi-threading techniques. Accordingly, the Defendants argue that if concurrently is not read to be "literally at the same time" the claims would read on time-interleaved multi-threading techniques. Once again, such arguments are more suited for invalidity assertions. Moreover, it is noted that in the passage in which the patents utilize the term "concurrently" to describe the Web server and the Page server operations, the patents also describe this concept as "to simultaneously process." Col. 6:21-27. With regard to the prior art time-interleaved multi-threading discussion (which the Defendants assert is not literally at the same time) the patent also uses the term "simultaneously." Col. 4:48-51. The patents do not distinguish one use of the term simultaneously from the other by the inclusion of the literal concept. The Court does not find support in the intrinsic evidence to support a requirement that "at the same time" must be "literally at the same time." **Thus, the court construes "said processing being performed by said page server while said Web server concurrently processes said other**

requests" to mean "said processing being performed by said page server while said Web server processes said other requests at the same time."

7. "Intercepting"

"Intercepting" is utilized in asserted claims 1, 9, 10 and 11 of the '554 Patent and 1 and 15 of the '335 Patent. The Plaintiff asserts that the proper construction of "intercepting" is "stopping, deflecting, or interrupting the processing of a request." The Defendants assert that the term should be construed as "diverting a request received at the Web server machine instead of the Web server executable processing it." Both parties argue that the other party's interpretations carry implicit meanings beyond the mere constructions asserted above. The Plaintiff asserts that the Defendants' construction implicitly requires a request to go around or bypass a Web server without any processing by the Web server. The Defendants assert that the Plaintiff's construction allows for the Web server executable to begin processing a request, even if only to recognize that is should not complete the processing. The parties also disagree as to whether the terms "stopping, deflecting, or interrupting" verses "diverting" are more appropriate. A portion of the disagreement between the parties is based upon the fundamental dispute as to whether a Web server is a machine, software, or combination thereof. As noted above, the Court construes a Web server as software, or a machine having software.

The Defendants assert that the prosecution histories of the '554 Patent and of the '335 Patent add clarity to the meaning of the term "intercepting." The Defendants are correct that prosecution history may be used to limit the claims so as to exclude interpretations that may have been disclaimed or disavowed. However as discussed below, the cited portions of the prosecution

histories of the '554 Patent and the '335 Patent do not amount to a clear disclaimer as suggested by the Defendants. *See Middleton, Inc. v. Minnesota Mining & Mfg. Co.*, 311 F.3d 1384, 1388 (Fed. Cir. 2002).

With regard to the '554 Patent file history, the Defendants point to language added via an Examiner's Amendment and to the Bookman reference for support of a disclaimer of claim scope. The amendment in question was made in a Notice of Allowability that was issued along with an Interview Summary of a December 17, 1998 examiner interview and a form PTO-892 Notice of References Cited. The PTO-892 Notice of References Cited listed the Bookman reference and an additional reference. At that time, other art was also included in the prosecution history including, for example, references that were the basis of previous rejections. The Interview Summary merely states that the prior art discussed was the "prior art of record." The description and other comments of the interview do not provide any other details as to why the Examiner Amendment was made. The Amendment in question added, to claim 1 for example, the language "wherein said routing step further includes the steps of intercepting said request at said Web server, routing said request from said Web server to a dispatcher, and dispatching said request to said page server." As the Plaintiff points out, the record does not show that Bookman was even particularly discussed in the interview let alone the art requiring the amendment to be made. On this basis alone, the Court may reject the Defendants' assertion that because of Bookman the claim language must be interpreted in the manner that the Defendants allege. Further, even if Bookman had been the art requiring such amendment, the record still does not provide any insight into the meaning of "intercepting" as the record is silent as to any further meaning of "intercepting" or the other additional terms recited in the amendment (routing from a Web server to a dispatcher and dispatching said request to a page server).

With regard to the '335 Patent prosecution history, the Defendants assert that statements made by the Applicants regarding the Leaf reference support the Defendants' proposition that partial processing does not equate to intercepting. In particular, the Defendants cite to a quote on pages 9-10 of a Response To Office Action dated May 23, 2001. The Defendants focus on a statement that Leaf did not suggest intercepting because "merely routing a request from a web server to the transaction gateway does not involve interception." The Defendants state that Leaf has Web server executable that partially processes a request before routing it on to the dispatcher. From this, the Defendants argue that the Applicants disclaimed partially processing a request and then routing it to the dispatcher. It is noted that the Applicants' Response in question, however, does not characterize Leaf in the manner suggested by the Defendants or make any references to partially processing. Further, the distinction between a Web server and Web server executable made by the Defendants is not clear in Leaf. In addition, it is noted that the full context of the Applicants' remarks regarding Leaf includes the statement that "Leaf does not teach or suggest 'intercepting said request.' Instead, Leaf teaches that the web server routes the request directly to the transaction gateway client." '335 Patent File History, Response to Office Action Dated May 23, 2001, p. 9-10. This statement merely suggests that directly routing a request from a web server to a transaction gateway is not intercepting and does not provide clear guidance as to the question of partial processing or the difference between a Web server and Web server executable. Thus, the prosecution history cited by the Defendants does not provide the clear guidance asserted by the Defendants.

It is noted that the Defendants' proposed construction adds Web server machine to the term of "intercepting." This language is somewhat redundant with the language surrounding the term

"intercepting" in most claims and does not conform to claim 15 of the '335 Patent which uses the term HTTP-complaint device. The context of the term as used in the claims themselves provides some guidance as to the proper construction. The claims themselves note that the intercepting of the request is at the Web server or the HTTP-compliant device.[3] For example, Claim 1 uses language such as "routing said request from said Web server to a page server," and "wherein said routing step further includes the steps of intercepting said request at said Web server, routing said request from said Web server to a dispatcher, and dispatching said request to said page server." This conforms with the specification, which states that the request is initially routed from the Web Client 200 to the Web Server 201. Col. 4:54-60. Also, Figure 4 appears to show the request going to the Web server executable 201(E). A construction that has the request bypassing the Web server is therefore not appropriate.

As to whether the beginning phrase of the construction should include "stopping, deflecting, or interrupting" as proposed by the Plaintiff, the Plaintiff provides little support for such language other than a general purpose dictionary. Although the Plaintiff asserts that the Defendants' construction requires bypassing the Web server, the Court does not interpret the phrase "diverting" to require such bypassing. However, the term "diverting" seems to carry an additional connotation that the request is sent somewhere else. When looking at the claims, however, the concept of the request being sent elsewhere is included in the "routing said request" (claims 1, 9, and 11 of the '554 Patent) or "transferring said request" (claim 15 of the '335 Patent) limitation that immediately follows the intercepting phrase. Thus, the Court does not feel that either construction adds clarity

[3] Claim 15 of the '335 Patent uses the term HTTP-compliant device while the other claims recite a Web server.

to the meaning of the concept of "intercepting" as used in the claims. Moreover, the parties have not pointed to anything in the intrinsic record that suggests which is more accurate: "diverting" or "stopping, deflecting or interrupting." These terms are not used in the specification and each in turn may need their own construction. The Court is not convinced that the term "intercepting" needs construction itself or that the constructions proposed by the parties add any needed clarity.

More helpful would be to construe the entire intercepting phrase: "intercepting said request at said Web server" (claims 1, 9, and 11 of the '554 Patent) and "intercepting said request at said HTTP-compliant device" (claim 15 of the '554 Patent). The specification describes the Interceptor as intercepting "the handling of a request." Col. 8:31-32. To conform with the description provided within the specification, the phrase "intercepting said request at said Web server" (claims 1, 9, and 11 of the '554 Patent) means at least "intercepting the handling of a request at a Web server" and the phrase "intercepting said request at said HTTP-compliant device" means at least "intercepting the handling of a request at a said HTTP-compliant device."

What is left for the Court to determine is whether the phrase "instead of the Web Server executable processing it" should be added to the end of the definition as proposed by the Defendants. The Defendants' primary support for their position is the language of the specification that states "instead of Web server executable 201(E) processing the URL request, however, interceptor 400 intercepts the request and routes it to Dispatcher 402." Col. 4:58-60. The Defendants argue that this language provides no room for partial processing of a request by Web server executable. The Court, however, finds such language ambiguous. The Defendants would like to interpret this cited quote ("instead of the Web server executable 201(E) processing the URL request") to mean "instead of the Web server executable 201(E) processing **any of** the URL request" wherein the Plaintiff would like

to interpret this citation to mean "instead of the Web server executable 201(E) **completely** processing the URL request" The specification, however, provides little guidance.

As discussed above, however, the specification does establish that a Web server may be software. Further, the specification establishes that the Web server does perform at least some action with relation to a request, namely receiving a request. Col. 4:55-58; Col. 8:28-31; Figure 4; Figure 5. Further, it is noted that in at least one embodiment the interceptor 400 is "an extension of the Web server 201" and the interceptor 400 also performs actions on a request. Col. 4:59-62. As Defendants have asserted that their proposed claim langue would exclude the Web server from any processing of the request, their proposed claim language would impermissibly exclude the embodiments disclosed within the specification. Thus, the Court declines to add the additional limitation sought by the Defendants.

For these reasons, **the Court finds that "intercepting said request at said Web server" means "intercepting the handling of a request at a Web server" and the phrase "intercepting said request at said HTTP-compliant device" means at least "intercepting the handling of a request at a said HTTP-compliant device."**

8. "Transferring"

"Transferring" is utilized in asserted claims 15-16 and 29 of the '335 Patent. The Plaintiff asserts that the "transferring" does not need construction. Alternatively, if construed, the Plaintiff asserts that the proper construction is "sending." The Defendants assert that the term should be construed as "sending toward a destination and relinquishing control of."

The parties agree to the use of the term of "sending" but disagree as to whether any additional

language is necessary. With regard to the Defendants' use of "toward a destination," it is noted that the surrounding claim language of each claim already includes this concept. For example, claim 15 states that the transferring is of "a request from an HTTP-complaint device to a page server." Similarly, claim 29 states that the transferring is of "a request from an HTTP-complaint device to a dispatcher." The inclusion of "toward a destination" within the term "transferring" itself is therefore unnecessary based upon the context of the claims themselves.

With regard to the "relinquishing control of" language sought by the Defendants, the Defendants point to a Microsoft Press Computer Dictionary definition that includes in transferring the concept of passing program control. Further, the Defendants assert that the purpose of the patent is to reduce processing burden. The Plaintiff argues that two other dictionary definitions noted by the Defendants do not include the relinquishing concept.

Once again, it is more instructive to look to the claims themselves. More particularly, in claim 15 of the '335 Patent immediately after the "transferring…to a page server" the claim continues with "said page server receiving said request and releasing said HTTP-compliant device to process other requests…." Likewise, claim 29 of the '335 Patent includes later in the claim "said page server receiving said request and releasing said HTTP-compliant device to process other requests…" To include relinquishing within the definition of transferring would possibly conflict with the latter claim language or alternatively be redundant. Further, to the extent that the Defendants assert that the purpose of the patent is to reduce processing burden, the recited releasing language is more related to that concept then the transferring.

As both parties have proposed definitions using the term "sending," the Court includes that term in its construction. **The Court construes "transferring" to mean "sending."**

9. "Dispatching"

The parties originally proposed different constructions for the term "dispatching." As the parties have subsequently submitted an agreed construction, a Court construction is no longer required.

10. "Releasing"

"Releasing" is utilized in asserted claims 1, 9 and 11 of the '554 Patent and 1, 15, and 29 of the '335 Patent. The Plaintiff asserts that the "releasing" does not need construction. Alternatively, if construed, the Plaintiff asserts that the proper construction is "freeing." The Defendants assert that the term should be construed as "communicating to said Web server that it may now process other requests."[4] A central point in the dispute between the parties is the Plaintiff's arguments that releasing can implicitly occur as a result of routing without communication to the Web server. The Defendants assert that there must be communication to the Web server.

The Plaintiff asserts that the term "freeing" is supported by the specification and notes that the specification states that the result of routing is that the Web server is free to continue servicing client requests. In particular, the Plaintiff points out that the specification states "Web server executable 201(E) is thus free to continue servicing client requests on Web serer 201 while the request is processed 'off-line.'" Col. 5:16-18. Thus, the Plaintiff asserts that the specification implies that releasing is an automatic consequence of routing the request to another processing element and that nothing in the specification requires communication to the Web server to effectuate

[4]For claim 9 of the '554 Patent, the Defendants substitute "second computer system" in place of "Web server" and similarly in claims 15 and 29 of '335 Patent the Defendants substitute "HTTP-compliant device" in place of "Web server."

the release.

The Defendants assert that the '335 Patent prosecution history shows that merely routing a request from a Web server to a Page server and thereby implicitly releasing the Web server was disclaimed by the Applicants during the prosecution history as being different from releasing. The Plaintiff counters by asserting that the full context of the prosecution history quote in question does not stand for the proposition asserted by the Defendants. The portion of the prosecution history in question includes:

> At no time does Rogers teach or suggest 'concurrently' processing other requests or 'releasing said Web server to process other requests' because merely retrieving data from multiple sources does not teach or suggest these elements. Response to Office Action, Nov. 27, 2001 at 9.

The Court is persuaded that by reviewing the full context of the portion of the prosecution history in question a clear disavowal was not made by the Applicants. Thus, the prosecution history does not mandate that releasing cannot implicitly occur due to routing from a web server to a page server.

However, the Defendants raise a more relevant argument with regard to the claim language itself. "Releasing" is used in each claim as part of the phrase "said page server receiving said request and releasing said Web server to process other requests."[5] It is therefore the Page server that does the releasing. The larger context of the use of releasing in the claims themselves indicates that the Page server has a role in the releasing as in the claims themselves it is the Page server that releases the Web server. This language also conforms to the Summary of Invention and Abstract. Col. 2:25-26; Abstract, line 8.

[5] "Second computer system" is used rather than "Web server" in claim 9 of the '554 Patent. "HTTP-compliant device" is used rather than "Web server" in claims 15 and 29 of '335 Patent.

The Court is thus persuaded that the Defendants are partly correct that as required by the claim language itself the Page server takes some action to releasing the Web server. However, the Defendants do not provide adequate support in the specification or elsewhere to mandate that such action is limited to communication from the Page server to the Web server. As there could be other actions that the Page server may take to affirmatively release the Web server, it would be inappropriate to limit the claim to one type of action, particularly when support is lacking in the record for that particular type of action. The specification does not necessarily limit the claims to a particular technique by the Page server as to how the claimed release by the Page server is accomplished. For example, the limitations proposed by Defendants could be argued to not include Page server actions that are communications from the Page server to other intermediate elements (such as the Dispatcher) or Page server actions that involve affirmatively not sending some expected regular communication. **The Court therefore interprets "said page server receiving said request and releasing said Web server to process other requests" to mean "said page server receiving said request and said page server performing an act (separate from merely receiving the request) to free the Web server to process other requests." Claims 9 of the '554 Patent and claims 15 and 29 of the '335 Patent are likewise construed with the substitution of "second computer system" and "HTTP-compliant device" respectively for "Web server."**

B. Means Plus Function Elements

The parties propose different constructions of three means plus function elements. In each case, the parties agree to the function but disagree as to what is the corresponding structure that is

disclosed in the specification pursuant to 35 U.S.C. §112, ¶6. The following means plus function elements contained in claim 9 of the '554 Patent are disputed:

(a) "means for generating said request,"
(b) "means for receiving said request from said first computer" and
(c)"page server processing means processing said requests and dynamically generating a web page in response to said request."

Throughout the pre-hearing and post-hearing briefing, the parties have substantially changed their positions as to the appropriate legal standards and the appropriate structure that should be applied to each element.

As to the function, the parties have agreed to functions that match the claimed language recited above: "generating said request," "receiving said request from said first computer," and "processing said requests and dynamically generating a web page in response to said request" respectively.

The Plaintiffs propose that the corresponding structure for each means plus function element is "any software, processor or equivalent therefore that performs the function of _____" (where the blank is the agreed function for that means plus function element). The Defendants in contrast point to particular structure disclosed within the specification. For the "means for generating," the Defendants propose "a processor, such as Box 102 of Figure 1, on a Web client 200 of Figure 4, running a Web browser." For the "means for receiving," the Defendants propose "Web server 201 of Figure 4 running a web server executable." For the "page server processing means," the Defendants propose "a processor, such as Box 102 of Figure 1, on a page server machine, such as Box 404(1)-(n) of Figure 4, running undisclosed page-generating software."

First, the Court notes that means plus function elements shall be construed to cover the corresponding structure described in the specification and equivalents thereof. 35 U.S.C. §112, ¶6. To avoid confusion, the Plaintiff indicated that it did not object to removing the word "equivalents" internal to its proposed construction. Plaintiff's Reply Brief at 58. The Court agrees that it would be clearer for "equivalents" to modify the entire construed structure.

As proposed by the Plaintiff, the corresponding structure could be any software or could be any processor that performs the agreed function. As proposed by the Plaintiff, software would not even be required as the Plaintiff uses the "or" conjunction. Plaintiff's construction of any software or processor is essentially unbounded and contradicts the guidance that has been established for interpreting means plus function elements. *Medical Instrumentation and Diagnostics Corp. v. Elekta AB*, 344 F.3d, 1205, 1211 (Fed. Cir. 2003)("We cannot allow a patentee to claim in function terms essentially unbounded by any reference to what one skilled in the art would understand from the public record.") Having chosen to utilize the means plus function claiming structure, the Plaintiff cannot now avoid its implications. Moreover, for computer implemented implementations the corresponding structure for means elements is limited to the specific structure and specific algorithms disclosed in the specification as opposed to the generic "software" or "processor" as asserted by the Plaintiff. *See Harris Corp. v. Ericsson Inc.*, 417 F.3d 1241, 1253-54 (Fed. Cir. 2005); *WMS Gaming Inc. v. International Game Technology*, 184 F.3d 1339, 1348-49 (Fed. Cir. 1999). Thus, the court must look to the specific structures and algorithms disclosed with the specification.

As noted above, the specification states:

> FIG. 1 illustrates a typical computer system 100 in which the present invention operates. Col. 2:66-67.
>
> The preferred embodiment of the present invention is implemented as a software module, which may be executed on a computer system such as computer system 100 in a conventional manner. Col. 3:55-58.

As described within the specification, the various software components operate on computer systems such as computer system 100 of Figure 1 that includes a processor 102.[6] Col. 2:66-67; Col. 3:8-11; Col. 3:55-63.

1. "means for generating said requests,"

For example, the "means for generating a request" is described as a Web client machine running a Web browser. Col. 1:24-26; Col. 2 :4-7; Col. 4:12-15; Col. 4:55-57; Col. 6:27-31; *See* Col. 8:25-29. This particular structure is noted in the specification for accomplishing the claimed function. One skilled in the art would understand from the specification that a Web client computer having a processor which operates a Web browser is the corresponding structure that accomplishes the function of generating a request.

The Court construes the corresponding structure of the "means for generating" to be "a processor of a computer that is, or has, a Web client running a Web browser" or equivalents thereof.

[6] It is noted that as to the component of the computer system, both parties agree that reference to the processor is appropriate.

2. "means for receiving said request from said first computer"

With regard to the "means for receiving said request from said first computer" element, the specification shows the Web server 201 receiving requests from the Web client 200. Figure 4; Figure 5; Col. 4:54-59; *See* Figure 2; Col. 3:64-Col. 4:10. The Web server 201 is also repeatedly described as including Web server executable. *Id.* As described above, the specification also establishes that that even if a Web server is software, the software module operates on a computer as described within the specification.

The Court construes the corresponding structure of the "means for receiving" to be "a processor of a computer that is, or has, a Web server running Web server executable" or equivalents thereof.

3. "page server processing means processing said requests and dynamically generating a web page in response to said request."

With regard to the function of the "page server processing means processing said requests and dynamically generating a web page in response to said request" element, the specification shows that such function is accomplished by Page server 404(1)-(n). Figure 4; Figure 5; Col. 5:37-Col. 6:31; Col. 8:39-43. As noted above, the parties agree that a Page server is page generation software. Further, as also noted above with the other means plus function elements, the specification establishes that software is operated on a computer system as described with the specification. Moreover, with regard to Page servers, it is noted that in one embodiment Page servers reside on a separate machine to accomplish the claimed function. Col. 5 line 49-50. Thus, the Court finds that

the Page server processing means is Page server software (as construed above) operating on a computer processor.

The Court construes the corresponding structure of the "page server processing means" to be "a processor of a computer that runs Page server software (wherein Page server software is page-generating software that generates a dynamic Web page)."

IT IS SO RECOMMENDED.

Within ten (10) days after receipt of the magistrate judge's report, any party may serve and file written objections to the findings and recommendations of the magistrate judge. 28 U.S.C.A. 636(b)(1)(C).

Failure to file written objections to the proposed findings and recommendations contained in this report within ten days after service shall bar an aggrieved party from *de novo* review by the district court of the proposed findings and recommendations and from appellate review of factual findings accepted or adopted by the district court except on grounds of plain error or manifest injustice. *Thomas v. Arn*, 474 U.S. 140, 148 (1985); *Rodriguez v. Bowen*, 857 F.2d 275, 276-77 (5th Cir. 1988).

SIGNED this 15th day of August, 2006.

CAROLINE M. CRAVEN
UNITED STATES MAGISTRATE JUDGE

Exhibit 9

IN THE UNITED STATES DISTRICT COURT
OF THE EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION

EPICREALM, LICENSING, LLC	§	
	§	
v	§	**No. 2:05CV163**
	§	
AUTOFLEX LEASING, INC., et al.	§	

EPICREALM, LICENSING, LLC	§	
	§	
v	§	**No. 2:05CV356**
	§	
FRANKLIN COVEY CO., et al.	§	

ORDER ADOPTING REPORT AND RECOMMENDATION REGARDING CLAIM CONSTRUCTION

The above-entitled and numbered civil action was heretofore referred to United States Magistrate Judge Caroline M. Craven pursuant to 28 U.S.C. § 636. The Report and Recommendation Regarding Claim Construction of the Magistrate Judge which contains her proposed findings of fact and recommendations for the disposition of such action has been presented for consideration. Plaintiff and Defendants both filed objections to the Report and Recommendation of the Magistrate Judge. The Court conducted a *de novo* review.

The Court, having reviewed the relevant briefing, finds the parties' objections are without merit. In their briefing, the parties address, among other things, the word "mechanism" as used in the Magistrate Judge's discussion of the construction of "Web page." *See* Report and Recommendation at pgs. 8-9 ("As described within the

specification, a Web page is a mechanism through which static and dynamic content may be displayed.")(emphasis added). A Web page may include static or dynamic content. However, the Magistrate Judge's construction of "Web page" as "Web content displayable through a Web browser" does not include the word "mechanism." The Court adopts the construction of "web page" as proposed by the Magistrate Judge, and with the exception of the use of the word "mechanism," the Court also adopts the reasoning of the Magistrate Judge with respect to the construction of "Web page."

The Court is of the opinion that the findings and conclusions of the Magistrate Judge are correct. Therefore, the Court hereby adopts the Report of the United States Magistrate Judge as the findings and conclusions of this Court.

IT IS SO ORDERED.

SIGNED this 30th day of October, 2006.

DAVID FOLSOM
UNITED STATES DISTRICT JUDGE

EPIC393288