985 PH Ex. 1

WHAT IS CLAIMED IS:

1. A computer program product for use in a system having at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation utilizes a browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein the portion of said first hypermedia document is displayed within a first browser-controlled window on said client workstation, wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies, either directly or indirectly, the location of at least a portion of said object, wherein said portion is external to said first distributed hypermedia document, wherein said object has type information associated with it utilized to identify and locate computer readable program code external to the first distributed hypermedia document, and wherein said embed text format is parsed by said browser to automatically invoke said computer readable program code, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising:

computer readable program code, identified by said type information, for being automatically invoked by the browser application to cause the client workstation to display an object and enable interactive processing of said object within the display area created at said first location within the portion of the first distributed hypermedia document being displayed in the first browser controlled window.

2. A computer program product for use in a system having at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising:

computer readable program code for causing said client workstation to execute a browser application to parse a first distributed hypermedia document to identify text formats included in said distributed hypermedia document and to respond to predetermined text formats to initiate processes specified by said text formats; and computer readable program code for causing said client workstation to utilize said browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein the portion of said first hypermedia document is displayed within a first browser-controlled window on said client workstation, wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies the location of at least a portion of an object external to the first distributed hypermedia document, wherein said object has type information associated with it utilized to identify and locate an executable application external to the first distributed hypermedia document, and wherein said embed text format is parsed by said browser to automatically invoke said executable application to execute on said client workstation in order to display said object and enable interactive processing of said object within a display area created at said first location within the portion of said first distributed hypermedia document being displayed in said first browser-controlled window.

3. A computer program product for use in a system having at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising:

computer readable program code for causing said client workstation to execute a browser application to parse a first distributed hypermedia document to identify text formats included in said distributed hypermedia document and to respond to predetermined text formats to initiate computer instruction sequences specified by said text formats;

computer readable program code for causing said client workstation to utilize said browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein the portion of said first hypermedia document is displayed within a first browser-controlled window on said client workstation, wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies, either directly or indirectly, the location of at least a portion of an object external to the first distributed hypermedia document, wherein said object has type information associated with it utilized by said browser, or by some other program, to identify and locate a sequence of computer instructions external to the first distributed hypermedia document, and wherein said embed text format is parsed by said browser to automatically invoke said sequence of computer instructions to execute on said client workstation in order to display said object and enable interactive processing of said object within a display area created at said first location within the portion of said first distributed hypermedia document being displayed in said first browser-controlled window.

985 PH Ex. 2

[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 07/20/2004

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	
ART UNIT	PAPER NUMBER
2154	7

DATE MAILED: 07/20/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

[illegible]

PH_001_0000784201

Office Action Summary	**Application No.** 10/217,955	**Applicant(s)** DOYLE ET AL.
	Examiner Andrew Caldwell	**Art Unit** 2151

-- 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 *09 June 2002*.

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) *1-3* 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) *1-3* is/are rejected.

7)☐ Claim(s) _____ 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).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *6*.	6) ☐ Other: _____.

Remarks

Claims 1-3.

This application is a continuation of U.S. Patent App. 08/324,443, now U.S. Patent 5,838,906, which is currently being reexamined (90/006,831). All references in the parent and reexamination files have been considered. Any references considered and any searches made in the reexamination application should be considered to have been made in this application.

Copies of the references cited on the Form 892 accompanying this Office action have not been provided since the applicants were already mailed copies with the most recent Office action in the reexamination.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary.

The Prior Art as Applied to Claims 1-10:

Berners-Lee, T., et al., Hypertext Markup Language (HTML), Internet Draft, IETF, pages 1-40, (June 1993).

Raggett, D., HTML+ (Hypertext Markup Language), (July 23, 1993). Hereinafter referred to as "Raggett I."

Raggett, D., Posting of Dave Raggett, dsr@hplb.hpl.hp.com towww-talk@nxocOl.cern.ch (WWW-TALK public mailing list), (Posted June 14, 1993). Hereinafter referred to as "Raggett II."

Toye, G., et al., SHARE : A Methodology and Environment for Collaborative Product Development, Proceedings, Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1993, IEEE, pp. 33-47, April 22, 1993.

Claims 1-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over the admitted prior art in the `906 patent and the newly cited teachings of Berners-Lee, Raggett I, Raggett II, and Toye.

Regarding claim 2, the admitted prior art teaches a portion of the claimed invention of claim 2, namely a computer program product comprising:

"at least one client workstation" (**See USP `906: Figure 2, element 130; Col. 4, Lines 32-40 which indicate that "small computer" 130 can be a client)** "and one network server" (**See USP `906: Figure 2, element 132**) "coupled to a network environment" (**See USP `906: Figure 2, element 100 Internet**), "wherein the network environment is a distributed hypermedia environment" (**See USP `906: Col. 5 lines 24-25**);

"computer readable program code for causing said client workstation to execute a browser application" (**See USP `906: Col. 3 lines 9-13**), "that parses a first distributed hypermedia document to identify text formats included in the distributed hypermedia document and for responding to predetermined text formats to initiate processing specified by the text formats" (**See USP `906: Col. 1, lines 1-Col. 3, line 51, with particular emphasis on Col. 2, line 63-Col. 3, line 25 showing a browser executing on client that parses and then displays a hypermedia document; where the user clicks on a link/image icon causing the**

browser to invoke a viewer application displaying the image in a separate window); and

"computer readable program code for causing said client workstation to utilize the browser to display, on the client workstation, at least a portion of a first hypermedia document received over the network from the server, wherein the portion of the first hypermedia document is displayed within a first browser-controlled window on the client workstation." (**See USP `906: Figure 1, element 10 as hypermedia document displayed on client; Col. 2 lines 28-36**).

While the admitted prior art describes a computer program product in which a hypermedia page (**See USP `906: Figure 1, element 10**) is displayed in a browser (**See USP `906: Col. 1, lines 1-Col. 3, line 51, particularly Col. 2, line 63-Col. 3, line 25**), the admitted prior art does not teach, as in claim 2, the particular steps used by the browser in order to process and display the hypermedia page. To summarize, the admitted prior art does not teach a computer program product wherein the browser application parses a first distributed hypermedia document to identify text formats included in the distributed hypermedia document and for responding to predetermined text formats to initiate processing specified by the text formats.

Nevertheless, Bemers-Lee teaches that HTML browsers parse HTML. (**See Berners-Lee at p. 2 as printed - paragraph starting; "Implentations of ..."**) The parsing is used to identify characters interpreted as markup elements, such as the various tags (**see Berners-Lee at page 5**) in the structured text example, and to associate text with various tags. These tags correspond to the claimed "text formats." Bemers-Lee also teaches that the browser processes the HTML by rendering it into a displayable form. (**See Berners-Lee at p. 3, definition of rendering**). Berners-Lee also discusses how specific markup elements are to be rendered. (**See for example, Berners-Lee at p. 14, typical rendering of address tag; p.15 typical rendering of block quote**). Berners-Lee therefore teaches a computer program product in which a browser application parses a first distributed hypermedia document to identify text formats included in the distributed hypermedia document and for responding to predetermined text formats to initiate processing specified by the text formats.

It would have been obvious to a skilled artisan to combine (1) the teachings of Berners-Lee regarding the processing of HTML documents performed by a browser, with (2) the HTML browser of the admitted prior art in light of the statement made by the admitted prior art that its hypermedia system is designed to handle hypermedia documents according the HTML markup standard. (**See USP `906: Col. 5, lines 28-31**).

Regarding the processing of the claimed "text formats," the admitted prior art teaches a computer program product wherein a browser invokes an external viewer program to process various file formats not handled directly by the browser. (**See USP `906: Col. 3, lines 13-20**). Specifically, the prior art describes an example wherein the file format not handled by the browser is an image file in ".TIF" or ".GIF" format and the browser invokes an image viewer program to display the full image in a separate window. (**See USP `906: Col. 3 lines 13-20**). While the prior art teaches that certain tags may cause the browser to invoke external applications to process particular file formats, these applications do not display their data in the browser window. Therefore, the admitted prior art does not teach the portion of the computer program product of claim 2 wherein:

> "Said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies the location of at least a portion of an object external to the first distributed hypermedia document;
>
> Said object has type information associated with it utilized by said browser to identify and locate an executable application external to the first distributed hypermedia document, and
>
> Said embed text format is parsed by said browser to automatically invoke said executable application to execute on said client workstation in order to display said object within a display area created at said first location within the portion of said first distributed hypermedia document being displayed in said first browser-controlled window."

However, Raggett I teaches various extensions to the HTML specification including an EMBED tag that provides a simple form of object level embedding. (**See Raggett I: p. 6 "Embedded data in an external format" and p. 26 embedded.**) For example, Ragget I teaches an HTML document including an EMBED tag that identifies embedded data in a foreign format. (**See Raggett I: p. 6 <embed ...> and <embed> tags.**) This embedded data is an object that cannot be directly processed by the browser. The foreign format data, or object, is embedded in the HTML document by placing it between the <embed ...> and </embed> tags. (**See Raggett 1: p. 6 "2 pi int sin (omega t)dt" as an example of embedded foreign data.**) Raggett I describes the example of an embedded equation, where the browser calls a program for rendering an equation by providing ascii character information to an external program and receives a pixmap image of the equation from the external program that is then displayed in the browser window. (**See Raggett I: p. 6, particularly the last ten lines.**) Raggett I therefore teaches "a first distributed hypermedia document that includes an embed text: format, located at a first location in said first distributed hypermedia document," that is used to identify embedded foreign data. Raggett I also teaches that the embed tags include a type attribute specifying a registered MIME content type that is used by the browser to identify the appropriate external filter to use to render the embedded foreign data. (**See Raggett I: p. 6 type="application/eqn".**) Raggett I thus teaches a computer program product wherein "the object has type information associated with it utilized by said browser to identify and locate an executable application external to the first distributed hypermedia document and wherein said embed text format is parsed by said browser to automatically invoke said executable application to execute on said client workstation in order to display said object."

It would have been obvious to a skilled artisan to combine (1) Raggett I's teachings regarding extensions to the HTML standard (i.e., the proposed HTML+ Specification) allowing the embedding of data in foreign formats within web pages with (2) the method as taught by the admitted prior art. This combination would have been obvious based on Raggett I's acknowledgment that this particular extension to HTML is advantageous and it represents a "substantial improvement." (**See Raggett I: p. 1 2nd paragraph of abstract**).

The combination of the admitted prior art in view of Berners-Lee and Raggett I does not explicitly teach a computer program product wherein "the embed text format specifies the location of at least a portion of an object external to the first distributed hypermedia document." Raggett I describes a method in which the object itself is embedded in the HTML document. (**See Raggett I: p. 6 embedded data in an external format - see example on the last two lines of the page where the object, the text representation of the equation, is within the embed tags**).

Raggett II, though, teaches putting the foreign data in a separate file and then referencing that file by a URL in the HTML+ embed tag. (**See Raggett II: last line.**) It is thus argued that Raggett II describes a system wherein "the embed text format specifies the location of at least a portion of an object external to the first distributed hypermedia document."

It would have been readily apparent to a skilled artisan to modify the computer program product discussed above, combining the teachings of the admitted prior art in view of Berners-Lee and Raggett I, by further substituting a URL which references a separate file containing foreign data for the embedded foreign data within the hypermedia document of the combination. Such a further modification would have been apparent based on Raggett II's explicit suggestion to make such a substitution. (**See Raggett II: last line**).

The combination of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II does not explicitly teach a computer program product that "enables interactive processing of said object." The combination teaches a computer program product that embeds static objects, as opposed to dynamic objects, within distributed hypermedia documents.

Toye on the other hand discloses a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document (**See Toye: p. 40 description of NoteMail, particularly p. 40, col. 2, first complete paragraph**).

It would have been readily apparent to a skilled artisan to modify the computer program product discussed above, combining the teachings of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II, by further modifying the combination's static embedded object to be a dynamic embedded object as taught by Toye. Such a further modification would have been apparent based on Toye's teaching that its architecture provides openness and flexibility (**See Toye: p. 40 col. 2 second complete paragraph**).

As to claim 1, it is directed to computer readable code implementing the controllable application of claim 2. Since the reasoning used to reject claim 2 discusses the controllable application, it should be readily apparent why the combination of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II, and Toye renders claim 1 obvious.

Regarding claim 3, it is essentially the same as claim 2 of this application except for the following limitations where italicized type is used to show the differences: (a) the embed text format specify, *either directly or indirectly*, the location of at least a portion of an external object; and (b) the object has type information associated with it utilized by said browser, *or some other program*, to identify a sequence of instructions external to the first distributed hypermedia document. As to the feature (a), the italicized text merely enumerates the set of possible choices for specifying an object's location, which are direct and indirect. In essence, the Applicants have merely restated a generic limitation by listing all of its species. As to the "or some other program" language, the Examiner would merely note that it is alternative language. Since the combination of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II, and Toye teaches one alternative (i.e., a browser uses type information to identify a controllable program), it is irrelevant that the claim fails to teach the other alternative. For these reasons, the reasons given for the rejection of claim 2 apply equally to claim 3.

Double Patenting

A rejection based on double patenting of the "same invention" type finds its support in the language of 35 U.S.C. 101 which states that "whoever invents or discovers any new and useful process ... may obtain a patent therefor ..."

(Emphasis added). Thus, the term "same invention," in this context, means an invention drawn to identical subject matter. See *Miller v. Eagle Mfg. Co.*, 151 U.S. 186 (1894); *In re Ockert*, 245 F.2d 467, 114 USPQ 330 (CCPA 1957); and *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970).

A statutory type (35 U.S.C. 101) double patenting rejection can be overcome by canceling or amending the conflicting claims so they are no longer coextensive in scope. The filing of a terminal disclaimer cannot overcome a double patenting rejection based upon 35 U.S.C. 101.

Claim 2 is rejected under 35 U.S.C. 101 as claiming the same invention as that of claim 6 of prior U.S. Patent No. 5,838,906. This is a double patenting rejection. As to claim 2 of this application it includes all of the limitations of claim 6 of the '906 patent except for the text at col. 8 lines 10-12: "wherein the portion of said first hyper-media document is displayed within a first browser-controlled window on said client workstation." Although claim 2 of this application does not have this exact language and, therefore, cannot be a verbatim copy, it does at lines 20-24 include language that requires the features of the limitation omitted from claim 6 of the '906 patent. The Examiner therefore fails to see how the claims can differ in scope.

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(c) 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).

Claims 1 and 3 are rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claim 6 of U.S. Patent No. 5,838,906. Although the conflicting claims are not identical, they are not patentably distinct from each other.

As to claim 1 of this application, claim 6 of the '906 patent is drawn to computer readable program code implementing a browser that calls a controllable application. Claim 1 of this application is directed to computer readable code implementing the controllable application of claim 6 of the '906 patent. Claim 6 of the '906 patent's description of the operations of the controllable application render claim 1 of this application obvious.

As to claim 3 of this application, it includes all of the limitations of claim 6 of the '906 patent except for the text at col. 8 lines 10-12: "wherein the portion of said first hyper-media document is displayed within a first browser-controlled window on said client workstation." It also introduces the new limitations as shown by the italicized text: (a) the embed text format specify, *either directly or indirectly*, the location of at least a portion of an external object; and (b) the object has type information associated with it utilized by said browser, *or some other program*, to identify a sequence of instructions external to the first distributed hypermedia document. As to the text from claim 6 of the '906 patent that is missing from claim 3 of this application, the omitted language does not, for purposes of obviousness-type double patenting, distinguish claim 3 of this application from claim 6 of the '906 patent. As to the feature (a) discussed above, claim 6 of the '906 patent requires the embed text format to specify the location of the external object. Claim 3 of this application merely enumerates the set of possible choices for specifying an object's location, which are direct and indirect. In essence, the Applicants have merely restated a generic limitation by listing all of its species. As to the "or some other program" language, the Examiner would merely note that it is alternative language. Since claim 6 of the '906 patent teaches one alternative (i.e., a browser uses type information to identify a controllable program), it is irrelevant that the claim fails to teach the other alternative.

Conclusion

A shortened statutory period for response to this action is set to expire **three months** from the mail date of this letter. Failure to respond within the period for response will result in **ABANDONMENT** of the application (see 35 U.S.C. 133, M.P.E.P. 710.02, 710.02(b)).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andrew Caldwell, whose telephone number is (703) 306-3036. The examiner can normally be reached on M-F from 9:00 a.m. to 5:30 p.m. EST.

If attempts to reach the examiner by phone fail, the examiner's supervisor, Glenton Burgess, can be reached at (703) 305-4792. Additionally, the fax numbers for Group 2100 are as follows:

Fax Responses: (703) 872-9306

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist at (703) 305-9600.

Andrew Caldwell

Andrew Caldwell
703-306-3036
July 11, 2004

985 PH Ex. 3

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Examiner: Caldwell, A. T.
DOYLE et al.	Art Unit: 2154
Application No.: 10/217,955	Response
Filed: August 9, 2002	
For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT	

Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

In response to the Office Action mailed 09/09/2004, please consider the following remarks:

Amendments to the Claims begin on page 2 of this paper.

Remarks/Conclusion begin on page 5 of this paper.

AMENDMENTS TO THE CLAIMS:

Please cancel claim 2.

This listing of the claims replaces all prior versions, and listings, of claims in the application.

1. (Original) A computer program product for use in a system having at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation utilizes a browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein the portion of said first hypermedia document is displayed within a first browser-controlled window on said client workstation, wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies, either directly or indirectly, the location of at least a portion of said object, wherein said portion is external to said first distributed hypermedia document, wherein said object has type information associated with it utilized to identify and locate computer readable program code external to the first distributed hypermedia document, and wherein said embed text format is parsed by said browser to automatically invoke said computer readable program code, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising:

computer readable program code, identified by said type information, for being automatically invoked by the browser application to cause the client workstation to display an object and enable interactive processing of said object within the display area created at said first location within the portion of the first distributed hypermedia document being displayed in the first browser controlled window.

2. (Cancelled)

3. (Original) A computer program product for use in a system having at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment, the computer program product comprising:

a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising:

computer readable program code for causing said client workstation to execute a browser application to parse a first distributed hypermedia document to identify text formats included in said distributed hypermedia document and to respond to predetermined text formats to initiate computer instruction sequences specified by said text formats;

computer readable program code for causing said client workstation to utilize said browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server, wherein the portion of said first hypermedia document is displayed within a first browser-controlled window on said client workstation, wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies, either directly or indirectly, the location of at least a portion of an object external to the first distributed hypermedia document, wherein said object has type information associated with it utilized by said browser, or by some other program, to identify and locate a sequence of computer instructions external to the first distributed hypermedia document, and wherein said embed text format is parsed by said browser to automatically invoke said sequence of computer instructions to execute on said client workstation in order to display said object and enable interactive processing of said object within a display area created at said first location within the portion of said first distributed hypermedia document being displayed in said first browser-controlled window.

REMARKS

Claims 1-3 have been reexamined, claim 2 is canceled, and claims 1 and 3 are now pending in the application. Reexamination and reconsideration of all outstanding rejections and objections is requested.

Claims 1 through 3 are rejected under 35 U.S.C. §103(a) as being unpatentable over the admitted prior art in the U.S. Patent No. 5,838,906 ('906 patent), the teachings of Berners-Lee, Raggett I, and Raggett II, and the newly cited teaching of Toye.

Claim 2 is rejected under 35 U.S.C. §101 for same invention double patenting. Claims 1 and 3 are rejected under the judicially created doctrine of obviousness-type double patenting. Claim 2 has been canceled to obviate the double patenting rejection of claim 2 and a terminal disclaimer is attached hereto to obviate the same invention obviousness-type double patenting rejection of claims 1 and 3.

Introduction

Included with this response are a Rule 132 Declaration by Professor Edward W. Felten, Professor of Computer Science at Princeton University ("Felten II, signed October 6, 2004"), traversing the rejections of claims 1 and 6 of U.S. Patent No. 5,838,906 ("the '906 patent) based on the same references cited in this Office Action and the Rule 132 Declaration by Professor Felten submitted with the response filed May 10, 2004 ("Felten I, signed May 7, 2004). Although these declarations were prepared in response to Office Actions mailed in connection with the reexamination of the parent patent application, A/N 08/324,443 (now the 906 patent), the obviousness issues raised in those Office Actions are identical to the obviousness issues raised in the present Office Action. References to these declarations relevant to identical issues raised in the present office action will be made in the following arguments.

It is Applicants' position that the reference referred to below as Raggett II is not a publication according to 35 U.S.C. §102. However, for the purposes of the following arguments this reference is being treated as if it is prior art.

Outline of the Non-Obviousness Argument for Claims 1 and 3

A. The Claimed Invention

B. Description of the References
   1. Applicants' Admitted Prior Art (Mosaic), Berners-Lee, Raggett I, and Raggett II
   2. Toye

C. The Examiner's Reasoning

D. Traverse

PART I. The establishment of a *prima facie* case of obviousness requires that all the claim limitations must be taught or suggested by the prior art. MPEP §2143.03
None of the references of the proposed combination, when considered either individually or collectively, teach or suggest the claimed features of the Applicants' invention. Accordingly, a *prima facie* case of obviousness has not been established.

a. There is no suggestion or teaching in either Toye, the admitted prior art (Mosaic), Berners-Lee, Raggett I or Raggett II of automatically invoking an external application to execute on a client computer, when an embed text format is parsed, to display and interactively control an object in a display window in a hypermedia document, received over a network from a network server, being displayed in a browser-controlled window on the client computer.

b. There is no suggestion or teaching in either Toye, the admitted prior art (Mosaic), Berners-Lee, Raggett I or Raggett II of parsing an embed text format at a first location in the hypermedia document and displaying the object and enabling interactive processing of the object within a display area created at the first location within the portion of the hypermedia document being displayed.

c. Because the claim limitations are not taught or suggested by the cited references, the combination proposed in the rejection would not include the limitations of claims 1 and 3.

PART II The establishment of a *prima facie* case of obviousness requires that the claimed combination cannot change the principle of operation of the primary reference or render the reference inoperable for its intended purpose. MPEP §2143.01. The proposed combination of Toye with the combination of Mosaic, Berners-Lee, Raggett I and II would change the operation of the latter combination and render it inoperable for its intended purpose. Accordingly, a *prima facie* case of obviousness has not been established.

a. The combination proposed in the Office Action contradicts a fundamental principle of operation of the Mosaic, Berners-Lee, Raggett I and II combination requiring that the images, rendered when the Raggett embed tag is parsed, be static images.

b. The combination proposed in the Office Action would change the Mosaic, Berners-Lee, Raggett I and II combination from being a distributed system, which is a basic principle of its operation and an intended purpose.

c. The combination proposed in the Office Action would change the Mosaic, Berners-Lee, Raggett I and II combination from a system intended to give the document author control over the user's browsing experience to a system which causes the document author to lose that control.

PART III The obviousness rejection is based on a false premise and therefore reaches a false conclusion.

a. Toye does not disclose a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document.

b. There is no teaching in Toye of a dynamic object that would make obvious modifying the static image taught by the combination of the admitted prior art (Mosaic), Berners-Lee, and Raggett I and II into a dynamic image.

PART IV There is no motivation or teaching in the cited references to combine the references to make the claimed invention obvious.

a. The language in Toye regarding "openness and flexibility" cited by the examiner teaches away from a combination that would make the claims obvious.

b. The fundamental problems solved by the Mosaic, Berners-Lee, Raggett I and II systems (HTML browser) and the Toye system teach away from a combination that would make the claimed invention obvious.

c. It is required to consider the references in their entireties, i.e., including those portions that would argue against obviousness. *Panduit Corp. v. Dennison Manufacturing Company*, 227 USPQ 337, 345 (CAFC 1985).

DETAILED ARGUMENT

A. The Claimed Invention.

The invention, as recited for example in claim 1, is for use in a system having at least one client workstation and one network server coupled to a distributed hypermedia environment, where the client workstation utilizes a browser application, executed on the client workstation, that parses a hypermedia document to identify text formats in the document and responds to predetermined text formats to initiate processing specified by the text formats and where the browser displays a portion of a first distributed hypermedia document, received over the network from the network server, in a browser-controlled window. The hypermedia document includes an embed text format, located at a first location in the hypermedia document, that specifies the location of at least a portion of an object external to the hypermedia document. The object has associated type information utilized by the browser to identify and locate a sequence of computer instructions external to the hypermedia document.

When an embed text format is parsed by the browser, the sequence of computer instructions is automatically invoked, as a result of the parsing, to execute on the client workstation.

A computer readable medium has computer program code embodied therein for being automatically invoke by the browser application to cause the client workstation to display an object and enable interactive processing of the object within a display window created at the first location of the portion of the hypermedia document being displayed in the first browser controlled window.

The invention, as recited for example in claim 3, is for use in a system having at least one client workstation and one network server coupled to a distributed hypermedia environment.

The claim recites a browser application, executed on the client workstation, that parses a hypermedia document to identify text formats in the document and responds to predetermined text formats to initiate processing specified by the text formats.

The browser displays a portion of a first distributed hypermedia document, received over the network from the network server, in a browser-controlled window. The hypermedia document includes an embed text format, located at a first location in the hypermedia document, that specifies the location of at least a portion of an object external to the hypermedia document. The object has associated type information utilized to identify and locate an sequence of computer instructions external to the hypermedia document.

When an embed text format is parsed by the browser, the sequence of computer instructions is automatically invoked, as a result of the parsing, to execute on the client workstation.

When the automatically invoked application executes on the client workstation, the object is displayed and interactive processing of the object within a display window created at the first location of the portion of the hypermedia document being displayed is enabled.

B . Description of the References

1.a. Applicants' Admitted Prior Art

The specification of the '906 patent (Applicants' Admitted Prior Art) describes a browser application, e.g., Mosaic, that functions as a viewer to view HTML documents. There are several ways to retrieve an HTML document from a network server, all of which require user interaction with the browser. [Felten I, paragraph 8]. The browser then retrieves a selected published source HTML document from a network server by utilizing a uniform resource locator (URL) that locates the HTML document on the network and stores a temporary local copy of the HTML source document in a cache on the client workstation.

The browser application then parses the local copy of the HTML document, renders the temporary local copy of the HTML document into a Web page , and displays the rendered Web page in a browser-controlled window. [Felten I, at paragraph 21]. During the rendering step, the browser may retrieve information external to the local copy of the HTML document, such as source files referenced by IMG tags, render the images from the retrieved files as static graphic images, and insert the images into the Web page of the HTML document, for display to the user.

There is no further interaction with the source HTML document or the local copy of the source HTML document subsequent to its being rendered and displayed. If a user believes the source HTML document has changed (s)he can click a refresh button in the browser GUI which causes the browser application to retrieve the source HTML document from the network server again, store a local copy again, parse and render again the newly retrieved local copy of the source HTML document, and replace the display of the previous version of the retrieved source HTML document with the subsequently retrieved version in the browser-controlled window or another window. For example, if the source HTML document were a price list of goods the user might refresh the document to determine if the prices had changed.

Although the browser application passively displays links, from text or picture elements of a first hypermedia document to other external data objects, a user may browse by actively selecting links to retrieve information identified by a link. The retrieved information either replaces the first hypermedia document or is displayed in a separate window other than the window displaying the hypermedia document. Mosaic has the capability of allowing the user to invoke an external application to open a new window to display file types that cannot be displayed by Mosaic (helper applications).

Some browsers, such as Mosaic, include the capability of rendering images in certain formats, such as GIF , designated as a native format. These images may be placed inline in an HTML document using the IMG element, which specifies a source location, URL, of the source file to be rendered by the browser, and displayed in the rendered format of the document. All static images referenced by IMG or FIG tags specified in the HTML document must be retrieved by the browser prior to rendering the HTML document.

For data formats that can not be rendered by the browser application itself, i.e., data in a foreign or non-native format such as ".TIF," Mosaic launches helper applications, in response to a user's command, in a separate window to view certain types of file types. As described in the specification, the mechanism for specifying and locating a linked object is an HTML anchor "element" that includes an object address in the format of Uniform Resource Locator (URL).

Many viewers exist that handle various file formats such as TIF. When a user commands the browser program to invoke a viewer program (helper application), typically by clicking on an anchor with a mouse, the viewer is launched as a separate program. The viewer program displays the image in a separate "window" (in a windowing environment) or on a separate screen. This means that the browser program is no longer active while the viewer program is active. The viewer program is completely independent of the browser after being invoked by the browser so that there is no communication between the viewer program and the browser program after the viewer program has been launched.

As a result, the viewer program continues to run, even after the browser program execution is stopped, unless the user explicitly stops the viewer program's execution.

Mosaic was a significant advance that made the WWW easily accessible and gave Web page authors a powerful tool to provide simplified user-activated access to viewing of hypermedia documents and related external data objects anywhere on the WWW network.

There is no disclosure of automatically invoking an external application to enable interactive processing of an object in a display area of a hypermedia document being displayed by the browser .

1.B Berners-Lee (Berners-Lee, T., et al., Hypertext Markup Language (HTML), Internet Draft, IETF, pages 1-40, (June 1993)

The Berners-Lee reference is a specification for the HTML markup language. HTML is a language used by Web page authors to describe the structure and desired contents of their pages. A browser parses an HTML document to determine its structure and then displays the specified items as a rendered Web page within a browser window.

This reference describes a model in which Web pages are written by a Web page author, then distributed by a Web server to a browser, and viewed as a Web page displayed in the browser window by the browser's user. The user views a page, and then clicks a hyperlink or button, or enters some text, to select another page to view.

There is no disclosure in the reference relating to building a browser or how a browser works, nor is there disclosure in the reference of automatically invoking an external application to enable interactive processing of an object in a display area of a hypermedia document being displayed by the browser.

1.c. Raggett I (Raggett, D., HTML+(Hypertext Markup Language), (July 23, 1993))

Raggett I is a document entitled "HTML+ (Hypertext Markup Language) A proposed standard for a light weight presentation independent delivery format for browsing and querying information across the internet" [emphasis added]. In pertinent part, Raggett I generally relates to allowing Web page authors to display static images of equations and simple drawings in a Web page. At page 3, describing the HTML+ Document Format, it is stated that "HTML+ departs slightly from pure presentation independence by allowing Web page authors to specify rendering hints to give Web page authors greater control over the final appearance of documents."

At pages 4 and 5, Inlined Graphics or Icons are discussed. It is stated that these elements are treated like characters in the text and an example of the IMG tag is given:

This line has a egyptian hieroglyph at the end of the
line. <img src = "ankh.tiff">

It is further stated that the URL notation is used to name the source of the graphics data and that sophisticated HTML+ editors should allow Web page authors to modify images using an external editor. It is also stated that larger inlined images should be specified with the FIG tag.

At page 6, Raggett's proposed EMBED tag is described that provides a simple form of object level embedding that is very convenient for mathematical equations and simple drawings. Raggett's proposed EMBED tag would allow Web page authors to continue to use familiar standards, such as *TeX* and *eqn*. It is also stated that images and complex drawings are better specified by using the FIG or IMG elements.

Raggett's proposed EMBED tag would utilize a type attribute to specify a MIME content type to be used by a browser to identify a rendering application, such as a shared library or external filter, used to render embedded data. An example of rendering the embedded data is given as returning a pixmap which is a data structure holding a static image.

An example of Raggett's proposed EMBED tag is given as follows:

<embed type="application/eqn">2 pi int sin(omega t)dt</embed>

In this example the embedded data is "2 pi int sin(omega t)dt" and the type information is "application/eqn". In this example, the embedded data is processed by the *eqn* application to render a static graphic image of the embedded data in the following form:

$$2\pi \int \sin(\omega t) dt$$

The reference also states that sophisticated browsers can link to external editor applications for creating and revising embedded data.

It is also stated at page 12 that when using the FIG tag, instead of using a *src* attribute, an EMBED element can be included immediately following the <FIG> tag and that this is useful for simple graphs etc. defined in an external format.

At page 13 the *ismap* attribute of the FIG tag is described. It is stated that arbitrary areas of the figure can be designated as hypertext links.

There is no disclosure in the reference relating to building a browser or how a browser works, nor is there disclosure in the reference of automatically invoking an external application to enable interactive processing of an object in a display area of a hypermedia document being displayed by the browser.

1.d. Raggett II (Raggett, D., Posting of Dave Raggett, dsr@hplb.hpt.hp.com to www-talk@nxocOl.cern.ch (W-WWW-TALK public mailing list) (Posted June 14, 1993))

The position of the Applicants is that Raggett II is not a publication complying with 35 U.S.C. §102. However, in the following it will be assumed that Raggett II is prior art.

Raggett II is an email message from David Raggett to Torben Nielsen and Bill Janssen having the subject line "HTML+ support for eqn & Postscript".

This reference quotes an email from Nielsen stating that he has lots of documents he wants to put on the Web and that without support for equations it is quite difficult. It also quotes an email from Janssen stating he would like to send encapsulated Postscript in his documents.

The email then states that the HTML+ DTD makes both these requests possible by providing the capability to embed foreign data inline in the HTML source. The document then gives an example of Raggett's proposed EMBED tag and states that the browser identifies the format of the embedded data from the "type" attribute. It is also stated that building in support for a large number of formats has the danger of leading to very large programs for browsers and that this can be avoided by using a common API for rendering foreign formats, e.g., as <u>rendering functions</u> that take a sequence of bytes and <u>return</u> a pixmap.

It is then stated that browsers can then be upgraded to display new formats by binding MIME content types to the function names for those formats and that the functions could be implemented as separate programs <u>driven via pipes</u> and stdin/stdout or as dynamically linked libraries (DLLs). It is also stated that foreign data can be put in a separate file referenced by a URL.

There is no disclosure in the reference relating to building a browser or how a browser works, nor is there disclosure in the reference of automatically invoking an external application to enable interactive processing of an object in a display area of a hypermedia document being displayed by the browser.

<u>2. Toye, G., et al., SHARE: A methodology and Environment for Collaborative Product Development, Proceedings, Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1993, IEEE, pp. 33-47, April 22, 1993.</u>

Toye is a paper describing a SHARE project that seeks to apply information technologies in helping design teams gather, organize, re-access, and communicate both informal and formal design information to establish a "shared understanding" of the design and design process. The paper also presents research and strategies undertaken to build an infrastructure toward the realization of SHARE. Two components of the SHARE environment are NoteMail and DIS (Distributed Information Services).

Fig. 5, at page 39, depicts an application-oriented view of the SHARE architecture. The top level architecture of SHARE is a set of services communicating over the Internet. Some of these services include DIS, link managers, and constraint managers. The diagram illustrates that SHARE can communicate over the Internet, as can other information services such as the World Wide Web, Databases, Catalogs, and Libraries.

In the SHARE architecture email is the primary medium for both human communication and tool integration. For example, NoteMail messages are formatted in MIME (Multi-purpose Internet Mail Extension) that enables them to be sent as ordinary e-mail and read using any MIME-compliant mail reader.

The shaded tear drop in Fig. 5 shows that the SHARE environment consists of three classes of tools. One class is NoteMail and DIS which helps engineers capture and manage file information. NoteMail is a tool for collaborative editing (i.e., editing by several members of a team) of engineering documents within an engineering team.

NoteMail messages are formatted in MIME (Multi-purpose Internet Mail Extension), the Internet standard for multimedia mail, and can be sent as ordinary e-mail and read by using any MIME-compliant mail reader. NoteMail uses a "Format" data type that captures and preserves the spatial arrangement of information items on each NoteMail page.

It is stated that an interesting feature of NoteMail is the open architecture of its viewer. Unlike most other engineering notebooks and multimedia authoring environments, any application that displays through an X-server can insert its output (audio, video, or graphics) dynamically into a notebook page through a "dynamic window".

This is accomplished in two steps. First, after a data object or file is <u>selected</u> by a user for inclusion in the notebook the system will invoke the appropriate application for display in the notebook. <u>Subsequently selecting</u> the displayed data with a mouse will restart the original application so that data can be edited or updated without leaving the network environment. It is stated that this functionality is similar to opening a file using Macintosh finder and automatically invoking the appropriate application for processing that file.

It is then stated that other engineering notebooks lack this openness and flexibility and only allow processing of a handful of input formats.

Because NoteMail messages are to be sent by email, full copies of the messages are not sent to everyone. Instead it is more efficient to store the components of the message in one place and just transmit a set of reference pointers. NoteMail uses an object-oriented knowledge base, known as DIS, for this repository.

There is no disclosure in the reference of building a hypermedia browser, as that term is used in claims 1 and 3, of modifying applications that edit or update files or objects, nor is there disclosure in the reference of automatically invoking an external application to enable interactive processing of an object in a display area of a hypermedia document being displayed by the browser.

C. THE EXAMINER'S REASONING

The examiner states that the combination of Applicant's admitted prior art in view of Berners-Lee, Raggett I and Raggett II does not explicitly teach a method that "enables interactive processing of said object". The combination teaches a method that embeds static objects, as opposed to dynamic objects, within distributed hypermedia documents.

It is then stated that Toye, on the other hand, discloses a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document, citing Toye's description of NoteMail, particularly p. 40, col. 2, first complete paragraph.

It is then concluded that it would have been readily apparent to a skilled artisan to modify the method discussed above, combining the teachings of the admitted prior art in view of Berners-Lee, Raggett 1 and Raggett II, by further modifying the combination's static embedded object to be a dynamic embedded object as taught by Toye. It is stated that the modification would be apparent based on Toye's teaching that its architecture provides openness and flexibility.

D. TRAVERSE

This rejection is respectfully traversed for the following reasons.

The entire Felten II declaration is incorporated herein as an independent traverse of the rejection of claims 1 and 3. The following argument recapitulates parts of the traverse set forth in Felten II, with citations to relevant parts thereof, and presents additional arguments not present in Felten II. Further, the argument also includes citations to Felten I.

The basic requirements of a Prima Facie Case of Obviousness are set forth in MPEP §2143:

> To establish a prima facie case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations.
>
> The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, not in applicant's disclosure. In re Vaeck, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).]

The level of skill in the relevant art is set forth in the Felten I declaration as:

> The benchmark for a person having ordinary skill in the art (PHOSA) is a person who is just graduating from a good computer science program at a college or a university, not a star student but just a typical, average student, or a person who has gained equivalent knowledge in the industry. This person knows how to do things in conventional ways but does not exhibit an unusual level of innovative thinking. [Felten I, paragraph 15].

PART I. The establishment of a *prima facie* case of obviousness requires that all the claim limitations must be taught or suggested by the prior art. MPEP §2143.03
None of the references of the proposed combination, when considered either individually or collectively, teach or suggest the claimed features of the Applicants' invention. Accordingly, a *prima facie* case of obviousness has not been established.

a. There is no suggestion or teaching in either Toye, the admitted prior art (Mosaic), Berners-Lee, Raggett I or Raggett II of automatically invoking an external application to execute on a client computer, when an embed text format is parsed, to display and interactively control an object in a display window in a hypermedia document, received over a network from a network server, being displayed in a browser-controlled window on the client computer.

As described above, in the Mosaic, Berners-Lee, Raggett I and II system a hypermedia document retrieved by the browser is rendered into a set of ordered static presentation formats that are subsequently displayed by the browser. As described in Berners-Lee and Raggett I and II, browsers have the ability to render graphics files into static images that can be inserted inline into the set of static presentation formats to be subsequently displayed by the browser.

Raggett I and II teach the use of an external rendering application invoked by the browser to process a graphics file in a foreign format (a format not handled by the browser itself) and to return a static image that the browser inserts inline in the static presentation form of the document that is subsequently displayed by the browser.

Accordingly, as acknowledged by the examiner, the Mosaic, Berners-Lee, Raggett I and II combination teaches that the browser displays a static, non-interactive image and the claimed feature of automatically invoking an external application to execute on the client computer to interactively control an object displayed in a display window in the hypermedia document is not taught or suggested by that combination of references.

Further, as set forth in Felten I, the rendering applications invoked in the Mosaic, Berners-Lee, Raggett I and II combination return a static image and terminate. The browser inserts the static image returned by the rendering application into the set of static presentation formats comprising the presentation form of the hypermedia document, prior to the document being displayed by the browser. Thus, the types of rendering applications taught by Raggett I and II that are invoked when Raggett's EMBED tag is parsed are not capable of providing interactive processing of an object displayed within a display area created in the hypermedia document <u>being displayed</u> in the browser controlled window as required by claim 6. Instead, the Raggett rendering applications terminate <u>before</u> the hypermedia document is displayed by the browser.

Toye discloses a NoteMail viewer that allows a user to view a static image of a notebook page. The first full paragraph on page 40 of Toye describes an authoring environment and a viewing environment.

When authoring a NoteMail page the author may actively <u>select</u> a data file or object to be included in the NoteMail page and then a static image of the file is displayed in the NoteMail page. [Felten II, at paragraphs 33-35]. The image displayed in the page must be static because

Toye states that subsequently selecting the data with a mouse will restart the original application so that the data can be edited or updated. [Toye at page 40, first full paragraph]. The fact that the original application must be restarted to interact with the data displayed in a NoteMail page teaches that the displayed data was static and that no interaction with the data was possible prior to its selection with a mouse. [Felten II, at paragraph 35].

Toye states that when an object or file is selected by the user the system will automatically invoke the application for display in a NoteMail page. Further, Toye teaches that the application launching functionality is similar to opening a file using Macintosh Finder. [Toye at page 40, first full paragraph]. Thus, Toye teaches that automatic invoking is a result of user selection, not parsing as required by claims 1 and 3, and that the result of the user's interactive selection is similar to opening a file using Macintosh Finder, where the application launched processes the file in its own window. [Felten II, at paragraph 36].

Accordingly, Toye teaches away from automatic invocation of an external application when a document is parsed to enable interactive processing of the object but instead teaches that an object must be selected by a mouse to invoke an application to enable interactive processing.

Thus, like the Mosaic, Berners-Lee, Raggett I and II combination, a static presentation format of the NoteMail page is displayed by the viewer. Subsequently selecting a static image displayed in the NoteMail page launches an application that allows a user to edit or update the data.

b. There is no suggestion or teaching in either Toye, the admitted prior art (Mosaic), Berners-Lee, Raggett I or Raggett II of parsing an embed text format at a first location in the hypermedia document and displaying the object and enabling interactive processing of the object within a display area created at the first location within the portion of the hypermedia document being displayed.

In the admitted prior art (Mosaic) and Berners-Lee combination a hypermedia document selected by the user is located on the Internet, retrieved by the browser, rendered into an ordered set of static presentation formats by the browser, and subsequently displayed in a browser controlled window.

The modification to the browser suggested by Raggett I and II does not change this fundamental viewing paradigm. As described above, the static images returned by external applications invoked in the Raggett system are inserted in line by the browser into the ordered set of static presentation formats comprising the displayable form of the hypermedia document. In Raggett I and II, the Raggett EMBED tag located at a first location in the hypermedia document is parsed, a rendering application is invoked that returns a static image and terminates, the static image is inserted at the first location in the set of static presentation formats, and the presentation form of the document is then displayed by the browser. Since the rendering application has terminated before the set of static presentation formats is displayed by the browser, it is fundamentally incapable of providing interactive processing of an object being displayed in the display area of a hypermedia document being displayed in the browser controlled window.

Turning next to the Toye reference, NoteMail messages are formatted in MIME (Multi-purpose Internet Mail Extension) and a new "Format" MIME data type is defined, for NoteMail to capture and preserve the spatial arrangement of information on a NoteMail page. The MIME "Format" data is stored separate from the text portions of the document [Felten II, at paragraph 31]. There is no teaching in NoteMail of using text formats, within the document text, intended to initiate processes specified by those text formats. Further, there is no teaching in NoteMail of parsing an embed text format at a first location and displaying and enabling interactive processing within the first location because, in NoteMail, the location of information is specified elsewhere, by the "Format" data type.

Additionally, the Toye reference teaches that any application that displays through an X-server can be restarted by subsequently selecting the displayed data in a NoteMail page with a mouse, so that the data can be updated or edited. There is no teaching of modifying an application to allow interactive processing within a display area of a hypermedia document being displayed in a browser controlled window. Toye teaches that any application able to display through an X-server would allow editing and updating of a file in a window controlled by the editing application. Toye provides no teaching that new applications should be created to provide this editing capability. Rather, he teaches that any existing application which is capable of being displayed through an X-server is suitable for this purpose. As Professor Felten points out [Felten II, at paragraph 38], this teaches away from the proposed combination, since existing editor applications at the time of the claimed invention were designed to be run in their own windows, under their own control, and contained menu bars and other graphical interface elements which would interfere with the editors being useable if run so as to provide interaction in a display area in a first location of the document being displayed.

Further, Toye teaches that the application launching functionality is similar to opening a file using Macintosh Finder. [Toye at page 40, first full paragraph]. In Macintosh Finder opening a file launches an application in a separate window. [Felten II, at paragraph 34]. Thus, Toye teaches away from enabling interaction in a display area in a first location of a document being displayed.

c. Because the claim limitations are not taught or suggested by the cited references, the combination proposed in the rejection would not include the limitations of claims 1 through 3.

As set forth below, the references provide no motivation for the combination proposed by the rejection, and such a combination would change the basic operating principles of the Mosaic, Berners-Lee, Raggett I and II Web browser technology. However, even if the combination were possible it would not include the limitations of the pending claims. [Felten II, at paragraphs 46-51].

Such a combination would not automatically invoke an external application to enable interactive processing within a display area of a hypermedia document being displayed by the browser because the Mosaic, Berners-Lee, Raggett I and II combination teaches that external data is rendered to a static bit map and then displayed by the browser, and Toye teaches that external data is displayed as a static bit map that must be selected by a mouse to launch an editor application in a separate window. [Felten II, at paragraph 47].

Instead, the combination, if it could be constructed, would include the Raggett method of creating a static bitmap within a browser window in such a way that a user clicking on that static bitmap would launch an editor program in an external window, as in Toye. [Felten II, at paragraph 50].

This combination would not show automatic invocation of the editor program when the hypermedia document is parsed or enable interactive processing within a portion of the first hypermedia document being displayed in the browser window, as required by claims 1 and 3. Instead, the external editor application of Toye would be invoked only if the user took the additional manual action of selecting the static image by clicking on it, causing interactive processing to be enabled in an external window when the external application was restarted. [Felten II, at paragraphs 48-50].

Even if the Toye combination were to show interactive processing within a portion of the first hypermedia document being displayed in the browser window, the combination would still not show automatic invocation of the editor program when the hypermedia document is parsed, as required by claims 1 and 3.

Thus at least two elements of claims 1 and 3 would be missing from the proposed combination. [Felten II, at paragraph 51].

PART II The establishment of a *prima facie* case of obviousness requires that the claimed combination cannot change the principle of operation of the primary reference or render the reference inoperable for its intended purpose. MPEP §2143.01. The proposed combination of Toye with the combination of Mosaic, Berners-Lee, Raggett I and II would change the operation of the latter combination and render it inoperable for its intended purpose. Accordingly, a *prima facie* case of obviousness has not been established.

a. The combination proposed in the Office Action contradicts a fundamental principle of operation of the Mosaic, Berners-Lee, Raggett I and II combination, requiring that the images, rendered when the Raggett embed tag is parsed, be static images.

Raggett I teaches uses of Raggett's proposed EMBED tag that require the returned image to be static. At page 12 of Raggett I, it is stated that, instead of using the *src* element, Raggett's proposed EMBED element can be used as an element of the FIG tag. It is known in the art that the FIG element is utilized to display static images in the displayed version of the HTML document. Since the use of Raggett's proposed EMBED tag, as a substitute for a *src*-defined static image file in this context is not qualified, Raggett's proposed EMBED tag is required to return only a static image, or it would cause the FIG tag to function incorrectly. [Felten I, at paragraph 44].

The requirement that Raggett's proposed EMBED tag return only a static image is further reinforced by the discussion in Raggett I of active areas at page 13. The *ismap* attribute described with respect to the FIG tag causes the browser to send mouse clicks on a figure back to the server using a selected coordinate scheme. Arbitrary areas of the figure can be designated as hypertext links. The Web page author thus creates a semantic correspondence between areas of the figure and Web pages that can be retrieved by clicking over these various areas. If the figure displayed were to be interactively changed then this semantic correspondence would be destroyed. Further, a mouse click can have only a single function. Since the *ismap* feature causes the browser to send mouse clicks to the server, the mouse click can not be utilized to interact with the image and the image must be static. Thus, an explicitly stated intended purpose of the Mosaic, Berners-Lee, Raggett I and II system is to allow the image returned by the Raggett EMBED-tag rendering application to be compatible with the *ismap* attribute of the FIG tag. [Felten II, at paragraph 19].

Thus, the ability to use Raggett's proposed EMBED tag, instead of the *src* attribute, within the FIG tag requires that a static and non-interactive image be returned. If this were not the case then Raggett I would require special discussion on the use of Raggett's proposed EMBED tag as an attribute within the FIG tag. No such discussion is included and thus Raggett I teaches that the image returned by Raggett's proposed EMBED tag must be static and non-interactive.

Accordingly, the reasoning of the rejection, that it would have been obvious to modify the static image taught by the Mosaic, Berners-Lee, Raggett I and II combination to be a dynamic object as taught by Toye, is a direct contradiction of the teaching of Raggett I and would change the principle of operation of the Mosaic, Berners-Lee, Raggett 1 and II combination, and render it inoperable for one of its intended purposes. If the displayed static image of the Mosaic, Berners-Lee, Raggett I and II combination were modified to be dynamic as suggested by the rejection, then the intended purpose of allowing the image returned by the Raggett rendering function to be compatible with the *ismap* attribute of the FIG tag would be rendered inoperable.

b. The combination proposed in the Office Action would change the Mosaic, Berners-Lee, Raggett I and II combination from being a distributed system, which is a basic principle of its operation and an intended purpose.

The admitted prior art describes a hypertext document as "a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document. In a hypertext document, the user may "click on," or select, certain words or phrases in the text that specify a link to other documents, or data objects." [Application at page 2, line 6-7] "A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents." [Application at page 2, lines 22-25]. When the hypermedia document is displayed on a browser program the browser responds to the selection of a link to retrieve and display the hypermedia document or data object referenced by the link.

A distributed hypermedia system is a "distributed" system because data objects that are imbcdded within a document may be located on many of the computer systems connected to the Internet." [Application at page 6, line 27-29].

In the Mosaic, Berners-Lee, Raggett I and II combination the Web-page author specifies the location of a linked-to object in tags such as A (anchor), IMG, or FIG defined by the HTML mark-up standard. Thus the author is responsible for and has control of the location of referenced objects. Since the Mosaic and Berners-Lee combination teaches a distributed system, the objects may be located on any computer connected to the Internet.

Further, the browser retrieves a copy of a source document from its server location, renders the presentation form of the document, and displays the document. The original source document cannot be edited or updated by a browser user.

Thus, the Mosaic, Berners-Lee, Raggett I and II combination teaches a model in which static pages can be published by anyone, on a server anywhere in the world, and read by anyone. The pages are connected by simple, unidirectional links that are used only to navigate from one page to another. A page is created and edited by its author, using a separate editing application, and is viewed, but not modified, by its readers using a separate browser application. [Felten II, at paragraph 12].

Accordingly, the Mosaic, Berners-Lee, Raggett I and II combination was designed to operate as a distributed system where objects may be stored anywhere on the Internet and retrieved by utilizing a browser application, by simply clicking on a link in a document displayed by the browser, to access another document located anywhere on the Internet.

In contrast, Toye teaches a system for collaborative editing of engineering documents within an engineering team, using a single object-oriented database (DIS) to store documents.

Toye teaches the use of a centralized, object-oriented database for storage of the workgroup's documents.

> Multimedia engineering documents containing raw text, encoded images, audio clips, video clips, etc. can get quite large. Sending such documents via email to everyone on a large design team can be costly in terms of both time and storage. Instead of transferring full copies to everyone, it is more efficient to store the components of the message in one place and just transmit a set of reference pointers. NoteMail uses an object-oriented knowledge base, known as DIS, for this repository function.
>
> Conceptually, DIS provides a <u>centralized information storage and management service for all the data associated with a design</u>: CAD files, e-mail messages, specifications, simulation results, and so forth. In practice, most data remains physically under the control of the application that created it; a persistent object is created in DIS to serve as a reference pointer or "handle."
> [Toye at p. 40-41, emphasis added].

The use of a centralized, object-oriented database makes sense given the goal of Toye to support collaboration within an engineering workgroup. [Felten II, at paragraphs 21-24]. Further, links between objects are created in the centralized database and not in the NoteMail page. [Toye, page 41 at the first partial paragraph].

The rejection states that it would have been obvious to modify the static combination taught by the Mosaic, Berners-Lee, Raggett I and II combination to be a dynamic object as taught by Toye.

However, any attempt to combine the centralized storage of referenced objects taught by Toye with the Mosaic, Berners-Lee, Raggett I and II combination would change the basic principle of operation of the combination being modified. A fundamental principle of operation and an intended purpose of the Mosaic, Berners-Lee, Raggett I and II combination is to provide a distributed system that allows objects to be stored anywhere on the Internet. A combination with Toye would turn that distributed system into a centralized database system, thereby destroying its distributed nature. Such a fundamental change teaches away from any combination of the Mosaic, Berners-Lee, Raggett I and II distributed system and the Toye centralized system.

Thus, the Mosaic, Berners-Lee, Raggett I and II and Toye references would not make the combination of claims 1 and/or 3 obvious to the PHOSA, because the differences in the basic principles under which the Mosaic, Berners-Lee, Raggett I and II combination and the Toye system operate, with regard to the storage and referencing of objects from a displayed page, are fundamentally different and incompatible.

c. The combination proposed in the Office Action would change the Mosaic, Berners-Lee, Raggett I and II combination from a system intended to give the document author control over the user's browsing experience to a system which causes the document author to lose that control.

The Web model of the Mosaic, Berners-Lee, Raggett I and II combination teaches a system which is based upon a publish-once/view-many paradigm. In that model, a document author is able to create an HTML document file which specifies precise locations for the various data objects that the browser will render for display in the page that the user sees. [Felten II, at paragraphs 13-14]. The combination proposed in the Office Action would be contrary to this basic principle.

The Web model insures document integrity. The document author can be assured that the fully-rendered document that (s)he originally created is going to appear the same for every user who subsequently retrieves that document for viewing. The end user, on the other hand, can be assured that the document being viewed has not been changed since it was last edited by the document author. This is extremely important in any document publishing system, since publishing systems are, by nature, intended to allow end users to rely on the published form of documents as accurate representations of the author's intended vision.

This notion of assuring data integrity is a fundamental principle of the Web model. That principle of data integrity assurance is destroyed in the proposed combination with the teachings of Toye. The Toye reference teaches a collaborative editing environment where any user can modify the data objects which are then rendered for display in the document. [Felten II, at paragraph 21]. An example of this is seen where Toye states: "for example, recipients can redo analyses and simulations with their own parameters" [Toye at page 40, first column, second full paragraph under the Notemail heading]. Once a recipient redoes such an analysis with different parameters than the original author specified, the resultant data object to be displayed in the document changes. When a subsequent user views the document, (s)he sees not the original document in the form specified by the original creator of the document, but rather the rendered document reflecting the sum total of changes made to both the document and the rendered data objects by any and all users who have accessed and modified that document since its creation.

Since any user can change the data objects in the Toye system, no user can rely on the document as a reflection of the original author's vision. An unavoidable consequence of the combination of Mosaic, Berners-Lee, Raggett I and II with Toye, therefore, would be a publishing system where the information communicated by the published document could be modified by users over time to the point where it would bear no resemblance to the document which the author intended to publish. This would render the Web unsuitable for its intended purpose.

Another important principle of the Web model taught by the Mosaic, Berners-Lee, Raggett I and II combination is that of referential integrity. In the Web model, the HTML document author can specify the specific locations, contained in "hypertext links," from which the browser will retrieve new HTML documents when users click upon those links. These links are easily specified by the document author, since they are directly specified through embed text formats in the document text. In the Web model, these are simple unidirectional links which are used only to navigate from document to document. The document author explicitly defines these links, and they are resolved and acted upon directly by the browser application. [Felten II, at paragraph 15].

This simple and lightweight linking model allows for the design of efficient distributed hypermedia browser applications which are optimized for the viewing of documents comprising both text and distributed data objects. It also allows for rapid navigation by the user from document to document, without limitations imposed by the physical location of either the document text or the data objects to be displayed. [Felten II, at paragraph 23].

Since it is primarily a publishing and information retrieval system, the Web system taught by the Mosaic, Berners-Lee, Raggett I and II combination employs unidirectional links, which can only be defined by the author, to insure that only the author's vision of the navigational paths out of the document is reflected in the final document that users can retrieve. This provides referential integrity to the system, which is a basic principle of the Web's fundamental design.

Since HTML authors can rely on the referential integrity of the documents they create, large information systems can be created via collections of multitudes of inter-linked distributed hypermedia documents. Without the enforcement of this referential integrity, the Web model would become unsuitable for the creation of such information systems.

Toye teaches that links should be bi-directional, and that they should be managed by separate applications. Toye teaches that links within the Share system can communicate changes in both directions. A consequence of this is that, in the Toye system, the definition of a link can be changed by agents out of the control of the document author. As Professor Felten explains: "The bi-directional links of Toye can, for example, represent formal constraints that connect two documents, so that a change in either of the two documents causes a corresponding change to happen automatically in the other document. This model is appropriate within an engineering workgroup, but it doesn't make sense on the Web, where hyperlinks often link documents written by different people who may not know or trust each other. For example, on the Web, I can create a page that links to the CNN home page; but it would not be appropriate for me to create a Toye-style link that would allow me, by changing my page, to cause changes on CNN's home page. Instead, Web hyperlinks follow a more appropriate (for the Web's goals) model in which only I can modify my own page, and only CNN can modify their page. This difference teaches away from the use of a Web browser with Toye." [Felten II, at paragraph 30]

In the Toye system, therefore, the document author can no longer be assured that the functionality of any link within an authored document will always be what the document's creator intended. This makes sense in a system designed for team-based collaborative editing of inter-linked documents, where one would naturally desire to have changes made by any collaborator instantly reflected for all to see, and for those changes to propagate through series of linked documents. In the proposed combination of Toye with Mosaic, Berners-Lee, Raggett I and II, however, the assurance of referential integrity that is so vital to the usefulness of the Web model would be unavoidably destroyed.

Furthermore, as has been discussed above, the combination with Toye would result in an embedded graphic presentation format of data that, while it would be static at the time of viewing, could change over time as various users would modify the corresponding data object, as enabled by Toye's collaborative editing environment. As a result the *ismap* functionality of the FIG tag of Raggett II would be rendered unusable, since the various intra-image links defined by the FIG tag would lose their semantic correspondence, and therefore their referential integrity, as originally defined by the HTML document's author.

So it is clear, therefore, that such a combination with Toye would destroy both the data integrity and the referential integrity that are fundamental principles behind the design of the prior art Web system, and that the proposed combination would therefore render the Web model unsuitable for its intended purposes.

PART III The obviousness rejection is based on a false premise and therefore reaches a false conclusion.

a. Toye does not disclose a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document.

The Office Action, at page 7, lines 4-7, states that Toye discloses a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document. However, this statement is incorrect in view of the precise meaning of the various terms defined in the Mosaic, Berners-Lee, Raggett I and II combination.

The admitted prior art describes a hypertext document as "a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document. In a hypertext document, the user may "click on," or select, certain words or phrases in the text that specify a link to other documents, or data objects." [Application at page 2, line 3-6] "A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents." [Application at page 2, line 22-26]. When the hypermedia document is displayed on a browser program the browser responds to the selection of a link to retrieve and display the hypermedia document or data object referenced by the link.

A distributed hypermedia system "is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet." [Application at page 6, lines 27-29].

The use of HTML allows the Internet to be an open system where a standard protocol is implemented by each computer connected to the internet. The structure of the document is defined by the author utilizing particular sets of characters that have a universal meaning.

In contrast, Toye teaches a system that is not a distributed system but requires that all referenced objects be stored in a single data base called DIS. [Toye, page 40, column 2, first and second paragraphs below the heading "Distributed Information Service (DIS)].

The NoteMail pages described in Toye use DIS as the central repository for referenced objects in contrast to the ability of a distributed hypermedia document to reference objects located in computers at different geographic locations. Thus, the Toye system does not teach or suggest using distributed hypermedia documents and its principle of operation is incompatible with the use of distributed hypermedia documents. [Felten II, at paragraph 24].

Also, for the same reasons Toye does not teach the use of a "distributed hypermedia environment" as that term is defined in the admitted prior art and used in claims 1 and 3. The use of the centralized storage of referenced objects is crucial to the intended purpose of the Toye system and contradicts the basic requirements of a distributed hypermedia environment. [Felten II, at paragraph 25].

Toye does not teach a hypermedia browser application, as that term is defined in the admitted prior art, Berners-Lee, and Raggett I and II, understood by the PHOSA at the time the application was filed, and as used in claims 1 and 3. Toye teaches no software application that parses distributed hypermedia documents or that uses text formats, and it does not teach other browser-related elements of the pending claims, such as parsing of distributed hypermedia documents by a browser, identifying text formats in distributed hypermedia documents and responding to predetermined text formats to initiate processing specified by those formats, utilizing a browser to display at least a portion of a distributed hypermedia document in a browser-controlled window, and parsing an embed text format in such a document. [Felten II, at paragraphs 26-27].

Further, the Toye reference teaches that information can be organized by adding links between objects where the links themselves are objects stored in the DIS database. [Toye, page 41, col. 1, first partial paragraph]. Thus, Toye is not a hypermedia system because, in the admitted prior art, Berners-Lee, and Raggett I and II combination, links are defined by the author as text formats in the hypermedia document and resolved by the browser application.

The Mosaic, Berners-Lee, Raggett I and II combination teaches the use of a hypermedia document that is a text document where some characters within the text are interpreted as mark-up tags specified by the HTML standard. The mark-up "tags" give structure to the document. [Berners-Lee, page 5, Felten II, at paragraph 14].

In contrast, Toye teaches that the structure, i.e., spatial arrangement of information in a NoteMail page, is preserved by a non-standard MIME "Format" data type defined by the Toye authors for the specific NoteMail system being described. [Toye, page 40, first column, last partial paragraph, Felten II, at paragraph 31]. Accordingly, Toye does not teach the use of a hypermedia document, in the sense of the Mosaic, Berners-Lee, Raggett I and II combination, or the embedding of an object in such a hypermedia document. NoteMail pages are therefore not analogous to Web-style hypermedia documents.

Also, there is no teaching in Toye of interactively processing an object embedded in a hypermedia document. Toye teaches that data displayed in a NoteMail page must be selected via a mouse click by the user to restart an application in order to update and edit data. The type of application described in Toye is any application that displays through an X-server. [Toye page 40, second column, first full paragraph]. There is no teaching of modifying such an application to process an object embedded in a hypermedia document. Further, Toye teaches that most data remains physically under the control of the application that created it, suggesting that the data must be processed using the normal interface for the application. [Felten II, at paragraphs 36-37].

b. There is no teaching in Toye of a dynamic object that would make obvious modifying the static image taught by the combination of the admitted prior art (Mosaic), Berners-Lee, and Raggett I and II into a dynamic image.

In view of the above, there is no teaching in Toye that would make the modification proposed in the rejection apparent to the skilled artisan. The failure of Toye to suggest or teach a distributed hypermedia system or the use of an analogous hypermedia document, as well as the other fundamental incompatibilities in architecture described above, would teach away from attempting to combine any features of the Mosaic, Berners-Lee, Raggett I and II combination with the Toye system.

The rejection implies that Toye teaches a dynamic object that meets the limitations set forth in claims 1 and 3. The term "dynamic object" is not used in the pending claims. However, as set forth above, the "dynamic object" described in Toye is an object that can only be activated by clicking on a static image displayed in a NoteMail page. The link between the "dynamic object" and an application to process the "dynamic object" is stored in an external database, not the NoteMail page itself. Thus, the external application for processing the "dynamic object" is not automatically invoked when an embed text format within the document is parsed nor is interactive processing of an object displayed in a display window of a hypermedia document enabled.

Accordingly, there is no teaching or suggestion in Toye of modifying the Mosaic, Berners-Lee, Raggett I and II system to make claims 1 and 3 obvious.

PART IV There is no motivation or teaching in the cited references to combine the references to make the claimed invention obvious.

a. The language in Toye regarding "openness and flexibility" cited by the examiner teaches away from a combination that would make the claims obvious.

The rejection states that the modification of the static object in the Mosaic, Berners-Lee, Raggett I and II system would have been apparent based on Toye's teaching that its architecture provides openness and flexibility.

However, the quoted language in Toye is doing nothing more than describing the benefits of the NoteMail system editor compared to other engineering notebook projects, specifically referring to the NoteMail editor's ability to insert data in different formats into a NoteMail page. As described above, Toye teaches that any application that displays using an X-server can insert a static image of a file into a NoteMail page when the file is selected by the NoteMail author. [Felten II, at paragraphs 40-41]. There is no suggestion there that NoteMail could or should be combined with any other system. Thus, the quoted language teaches away from modifying the NoteMail editor since it is already superior to the other known engineering notebook projects.

Additionally, the level of skill in the art cannot be relied upon to provide the suggestion to combine references. MPEP §2143.01 (quoting *Al-Site Corp. v. VSI Int'l Inc.*, 50 USPQ2d 1161 (Fed.Cir. 1999). In the rejection, the general and nebulous Toye language regarding "openness and flexibility" is not related to any possible motivation to combine the references. [Felten II, at paragraph 41]. It is merely highlighting advantages of the NoteMail system over other editors commonly used for engineering collaboration systems. In fact, even if it were proper to rely on the skill in the art to provide a motivation to combine, a PHOSA would only find among these references the strong suggestion that they are not combinable.

b. The fundamentally different problems solved by the Mosaic, Berners-Lee, Raggett I and II systems (HTML browser) and the Toye system teach away from a combination that would make the claimed invention obvious.

A possible source for a motivation to combine references is the nature of the problem to be solved. MPEP 2143.01. Here, the Mosaic, Berners-Lee, Raggett I and II combination and the Toye system solve problems of a completely different nature and have structures and implementations that are fundamentally incompatible.

A list of some of the fundamental differences between the teachings of the Mosaic, Berners-Lee, Raggett I and II and the Toye reference is given in Felten II:

> Toye teaches collaborative editing of documents; Berners-Lee teaches that documents are created by an author and read (without editing) by a set of readers. Toye teaches storage of documents in a centralized object-oriented database; Berners-Lee teaches that documents can be retrieved from anywhere and everywhere on the

> Internet. Toye teaches that display structure is specified using a separate "Format" data type, outside a text document; Berners-Lee teaches that display structure is specified by markup commands within a text document. Toye teaches rich, bi-directional links implemented by separate applications; Berners-Lee teaches simple unidirectional links, providing only navigation and implemented by a browser. Toye teaches that users need not know where documents are located; Berners-Lee teaches that users know URLs, which contain location information. [Felten II, at paragraph 42].

The nature of the problem solved by the Mosaic, Berners-Lee, Raggett I and II system is the need to allow authors to publish and distribute widely on the Internet documents that can be retrieved and easily viewed by end users, without regard to the types of hardware or operating systems utilized by the computers connected to the Internet. [Application at page 1, line 16-30].

To solve this problem, the Mosaic, Berners-Lee, Raggett I and II references teach a model in which static pages can be published by anyone, on a server anywhere in the world, and read by anyone with a connection to the Internet. The pages are connected by simple, unidirectional links that are used only to navigate from one page to another. A page is edited by its author using a separate editor application, and is viewed, but not modified, by its readers using a separate browser application. [Felten II, at paragraph 12].

The nature of the problem solved by the Toye reference is the need to create a system for collaborative editing of engineering documents within an engineering team. [Toye at page 36, topic 5].

To solve this problem, Toye teaches using a single object-oriented database to store the documents needed by an engineering workgroup, [Felten II, at paragraph 24] where the data base includes bi-directional links between objects. [Felten II, at paragraph 29-30].

Because of the fundamentally different problems solved by the references, the disparate techniques and structures utilized in one system are not relevant or useful in the other. For example, the use of the collaborative editing techniques of Toye would be contrary to the publish-and-view philosophy of the Internet, as embodied in the Mosaic, Berners-Lee, Raggett I and II combination. Further, the centralized storage technique of Toye works well for highly structured engineering design, but is contrary to the distributed nature of the Mosaic, Berners-Lee, Raggett I and II combination.

c. It is required to consider the references in their entireties, i.e., including those portions that would argue against obviousness. *Panduit Corp. v. Dennison Manufacturing Company*, 227 USPQ 337, 345 (CAFC 1985).

The Toye reference, when considered in its entirety, teaches a Method and Environment for Collaborative Product Management [Toye Title] to apply information technologies to help design teams gather, organize, re-access, and communicate both informal and formal design information to establish a "shared understanding" of the design process. [Toye Abstract]. Toye teaches email as the primary medium for both human communication and tool integration. [Toye at page 39]. The SHARE environment is depicted in Fig. 4 on page 38 and depicts a number of Powerbook computers connected by email to a File Server that provides shared access to files. [Toye page 38]. The information to be shared in the collaborative group is stored in a central data base called DIS (Distributed Information Services) that helps engineers work as a team to capture, organize, retrieve, modify and share design knowledge without their having to know details such as file formats and locations.

Significantly, web browsers of the type taught by the Mosaic, Berners-Lee, Raggett I and II combination existed at the time of publication of Toye but the designers of the SHARE project chose not to use them. [Felten II, at paragraph 26-28]. Such a decision made sense because the goal of the SHARE project, to allow collaborative editing and centralized, highly-structured data management, is inconsistent with the goals of the open, distributed hypermedia model taught by the Mosaic, Berners-Lee, Raggett I and II combination.

Thus, the designers of the SHARE project, innovative engineers who recognized that realizing their vision, even in the relatively circumscribed world of engineering, would be a massive undertaking [Toye, at page 46, first column, last paragraph] did not attempt to modify or redesign the web browser taught by the Mosaic, Berners-Lee, Raggett I and II combination. Instead they designed the NoteMail system which is centralized, not distributed, and which does not use hypermedia documents as that term is used in claims 1 and 3.

The level of skill in the art is:

> The benchmark for a person having ordinary skill in the art (PHOSA) is a person who is just graduating from a good computer science program at a college or a university, not a star student but just a typical, average student, or a person who has gained equivalent knowledge in the industry. This person knows how to do things in conventional ways but does not exhibit an unusual level of innovative thinking. [Felten I, at paragraph 15].

The PHOSA does things in a conventional way. The entire Toye reference teaches a collaborative environment that is the result of a massive undertaking by innovative engineers and professors. [Toye at page 46, first column, last paragraph]. As is discussed extensively above, NoteMail teaches a model of information sharing and organization that does not use the Web browser paradigm of the Mosaic, Berners-Lee, Raggett I and II combination, but instead uses an architecture fundamentally incompatible with the Web. Thus the Toye reference teaches away from modifying the Mosaic, Berners-Lee, Raggett I and II combination as proposed by the rejection. [Felten II, at paragraph 32].

CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this Application, please telephone the undersigned at (925) 944-3320.

Respectfully submitted,

[illegible]

Charles E. Krueger
Reg. No. 30,077

LAW OFFICE OF CHARLES E. KRUEGER
P.O.Box 5607
Walnut Creek, CA 94596
Tel: (925) 944-3320 / Fax: (925) 944-3363

985 PH Ex. 4

Application of: Doyle et al.

Application Num.: 90/006,831

Filed: October 30, 2003

For: Distributed Hypermedia Method for Automatically Invoking External Application Providing Interaction and Display of Embedded Objects Within a Hypermedia Document

Examiner: Caldwell, A.T.

Art unit: 2157

Declaration of Edward W. Felten

I, Edward W. Felten, declare as follows:

1. I have been retained by Eolas and the Regents of the University of California to serve as an expert in the field of computer science and Internet software. My Curriculum Vitae, which recites my technical expertise, is attached hereto to as Exhibit A.

I. Qualifications

2. I graduated with Honors from the California Institute of Technology in 1985, with a B.S. degree in Physics. I received an M.S. in Computer Science in 1991, and a Ph.D. in Computer Science in 1993, both from the University of Washington.
3. I am currently a Professor of Computer Science at Princeton University, where I have taught since 1993. I was originally hired at Princeton as an Assistant Professor, in 1993. I was promoted to Associate Professor in 1999, and to Professor in 2003.
4. I am the author or co-author of numerous publications relating to computer science and Internet software. These publications are listed in my CV.
5. I have been asked to address the arguments presented in the Office Action mailed March 12, 2004 ("the Office Action") in connection with the reexamination of United States Patent No. 5,838,906 ("the '906 patent") that the claims of the '906 patent are unpatentable as being "obvious". For the reasons described in this declaration, I disagree with the arguments presented in the Office Action and, instead, believe that the claims of the

'906 patent fully meet the requirements for patentability over the cited references, as those patentability arguments have been described to me.

6. To familiarize myself with the issues involved in the rejection of the claims, I have reviewed numerous documents, including the following: the '906 Patent and its file history, the documents cited in the Office Action, and all other documents referenced or cited in this declaration.

7. Before specifically addressing the cited references and unpatentablity arguments raised in the Office Action, I believe that it is important to discuss the relevant state of the browser art as it existed in 1994. My discussion is based on my experience as a computer science researcher and teacher, and as a Web user and network software developer. From this experience, I have gained an independent understanding of how the browser art developed.

II. Relevant State of the Art in 1994

8. In 1994, the Web was young, and browsers were a relatively new technology. Browsers offered only a very limited form of interactivity. A page could contain hyperlinks, on which the user could click to view another page. A page could be a form to be filled out by the user, with a "submit" button which, when clicked, caused the user to see another page.

9. Another technology, known as "helper applications," was implemented in the Mosaic browser. This technology allowed the browser to link to an external program, in cases where the browser encountered a file whose format the browser did not understand. For example, if the user clicked on a hyperlink that pointed to a file in .mpeg format (i.e., a movie in MPEG format), then the browser would launch an external MPEG-viewer program and pass the .mpeg file to that program. The result would be that the MPEG program ran, in a separate window from the browser.

10. Helper applications allowed the browser to link to an external program, but that program could not provide interactivity within the browser window. The helper application was just an external program that ran on the same computer, in a separate window.

11. None of these methods allowed a Web page author to place fully interactive objects within the confines of a Web page's display.

12. These methods are all implemented in today's browsers, and they are all in use on the Web today.

III. Response to the Unpatentability Arguments Raised in the Office Action

13. I have been told by patent counsel for Eolas and the Regents that a patent may not be obtained, even though the invention is not anticipated, 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 the subject matter pertains. I have further been told that I need to make a four step inquiry to evaluate "obviousness" in which the scope and content of the prior art are to be determined; the level of ordinary skill in the pertinent art resolved; and against this background, the obviousness or nonobviousness of the subject matter is determined. I have also been told that such secondary considerations as commercial success, long felt but unresolved needs, failure of others etc. might be utilized to give light to the circumstances surrounding the origin of the subject matter sough to be patented.

14. As a "useful general rule" I have been told that references that "teach away" cannot serve to create a meritorious case of obviousness. Also, I have been told that proceeding contrary to the accepted wisdom is strong evidence of nonobviousness. In addition, I have been told that the prior art must "suggest" or "motivate" one of ordinary skill in the art to combine the prior art to make the claimed invention and must further have taught that such a combination would have a "reasonable expectation of success".

A. The Level of Skill In the Art

15. My benchmark for what ordinary skill in the art means is a person who is just graduating from a good computer science program at a college or a university – not a star student but just an average student – or a person who has gained an equivalent level of knowledge through experience in the industry. This person knows how to do things in conventional ways but does not exhibit an unusual level of innovative thinking.

16. In 1994, those of ordinary skill in the art were just becoming familiar with the Web and Web browsers. One of ordinary skill would have had a general idea of how the Mosaic browser worked, and would have been familiar with hyperlinks, forms, and helper applications.

B. The Grounds of Rejection

17. Claims 1 and 6 of the'906 patent have been rejected by the United States Patent Office as being obvious under 35 U.S.C. Sec. 103(a); as being unpatentable over the admitted prior art in the '906 patent and teaching of Berners-Lee, Raggett I, and Raggett II. While I understand that the patent attorneys for Eolas and the Regents are challenging whether Raggett I and Raggett II are really "prior art" to the '906 patent, I have been asked to

assume for the purposes of my analysis that both the Raggett I and Ragett II references would have been "prior art".

B. The '906 Patent

18. The claims of the '906 Patent describe a technology that allows web page authors to include, within the boundaries of a web page, interactive objects. This is done (briefly stated) by including in the web page's HTML text an embed text format, that provides information about where to get the object's data, along with information to identify and locate an executable application that will be invoked on the client computer to display the data and to provide interactivity with it, and by providing a web browser that knows how to parse the HTML to extract the embed text format, how to use type information to identify and locate the executable application, how to invoke the executable application, to execute on the client computer, and how to interface to the executable application so as to allow the user to interact with it within the boundaries of the browser window.

C. Prior Art Browsers

19. The Office Action cites the applicants' admitted prior art. I have reviewed all prior art references referenced in the '906 Patent's file history. It appears that the Office Action's discussion of this prior art focuses on the Mosaic browser, which was the most advanced prior art browser.

20. Mosaic, and other prior art browsers, executed on a client computer, and operated by downloading copies of web pages (and other files, such as embedded static images) over a network from web servers. After downloading a copy of a file, Mosaic would sometimes keep a copy of that file in a local cache, on the user's client computer. Caching allowed the file to be referenced more quickly if it was needed again later.

21. After downloading a file, Mosaic would parse that file (i.e., analyze its structure) to determine how the file should be displayed on the screen. Mosaic would then paint the contents of the file into a browser window.

22. When Mosaic, or another prior art browser, was used to view web pages, several steps stood between the author of the web page and the user who was viewing it. First, the file would be copied, at least once and perhaps more times, while in transit between the web server and the user's browser. Second, the file would be written in one format (typically, HTML) but displayed in another form, by rendering the HTML into a visual representation that would actually be presented to the user.

23. Because these steps stood between the author and the user, there was no realistic way for the user to edit the web page on the client workstation. The user did not have access to the version of the page that was distributed – that version lived on the server, and it wouldn't make sense to let an arbitrary user edit the contents of somebody else's web page.

24. In addition, because web pages were written in one format (HTML) and viewed in another (visual representation), it did not make sense to talk about editing and viewing a document in the same window. Web page authors would typically work with two separate windows open, one (a browser) to see what the visual representation looked like, and another (an external editor) to actually modify the page's HTML representation. An author would fiddle with the HTML, then click the save button in the editor and the refresh button in the browser to see what the visual representation of the page looked like, then fiddle with the HTML some more, and so on until he was satisfied with the page's appearance.

D. The Berners-Lee Reference

25. The Berners-Lee reference is a specification for the HTML markup language. HTML is a language used by Web page authors to describe the structure and desired contents of their pages. A browser parses an HTML document to determine its structure and then displays the visual representation of the specified items within a browser window.

26. The Berners-Lee reference teaches a model in which Web pages are written by an author, then distributed by a Web server to a browser, and viewed as a static item by the browser's user. The user views a page, and then clicks a hyperlink or a button, or enters some text, to select another page to view.

27. In the model taught by Berners-Lee, a user interacts with the Web by moving from one static page to another. Thus Berners-Lee teaches away from the provision of rich interactivity within a page.

28. Berners-Lee teaches a language for authoring web pages, but it does not teach how to build a browser or how a browser works.

D. The Raggett I Reference

29. Raggett I suggests some modifications to the HTML system taught in Berners-Lee. The overall teaching of Raggett I is very similar to that of Berners-Lee.

30. Like Berners-Lee, Raggett I does not teach how to build a browser or how a browser works.

31. Raggett I teaches the use of the same model as Berners-Lee, in which Web pages are essentially static, and the user interacts with the Web by moving from page to page. Accordingly, Raggett I teaches away from the provision of rich interactivity within a page.

32. Raggett I is motivated by the problems of Web page authors. Authors want to be able to include in their pages information in a wide variety of formats. For preexisting content, an author wants to be able to use the content in the format in which it was originally created. For new content, an author wants to be able to choose a format well suited to a particular type of content. For example, if the content consists of mathematical

equations, the author wants to be able to be able to use a format designed for describing equations.

33. At the time of Raggett I, browsers such as Mosaic could handle only a limited set of data formats. Web page authors had noted a need for the display of static pages in more, and more varied, data formats.

34. One known method for displaying more formats was to do server-side translation. In this method, a web page author would take a document in some format, and generate a static image file from it. For example, an author might take a file describing a diagram, and generate from that file a static image, in GIF format, depicting the diagram. The web server could then deliver the GIF file to the browser, which would know how to render it within a web page.

35. Another known method to enable the display of more formats was to build support for displaying additional formats into the browser itself. Among the disadvantages of this approach were that it made the browser larger and more complicated, and that it required a new version of the entire browser to be distributed to a user before that user could view the new format.

36. Raggett I proposed a slight extension of this method, in which, rather than receiving an image, the browser receives information in some foreign format, and then uses an external program to render that information into an image, which the browser displays within the web page. This is a simple and natural extension of the browser's ability to display static images.

37. This extension is described in the following paragraph, which is also cited in the Office Action:

> The EMBED tag provides a simple form of object level embedding. This is very convenient for mathematical equations and simple drawings. It allows authors to continue to use familiar standards, such as TeX and eqn. Images and complex drawings are better specified using the FIG or IMG elements. The type attribute specifies a registered MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g. by returning a pixmap. It should be possible to add support for new formats without having to change the browser's code, e.g. through using a common calling mechanism and name binding scheme. Sophisticated browsers can link to external editors for creating or revising embedded data. Arbitrary 8-bit data is allowed, but &, < and > must be replaced by their SGML entity definitions. For example <embed type="application/eqn">2 pi int sin (omega t) dt</embed> gives [image of equation appears here].

(Raggett I at p. 6)

38. This paragraph teaches a method for displaying new types of *static* information within a Web page. The teaching of the use of static information is evident for several reasons.

39. First, the use of static information is consistent with the teaching of the remainder of Raggett I and with the teaching of Berners-Lee that preceded it.

40. Second, Raggett I motivates its proposed embed tag by referring to two types of data that one might want to display: "mathematical equations and simple drawings". These are types of data that one would want to display statically.

41. Third, Raggett I says that Raggett's proposed embed tag "allows authors to continue to use familiar standards, such as *TeX* and *eqn*." (italics in original). These are well-known formats for describing the display of static data. TeX is used to specify the typesetting of textual documents; it is still widely used to format scientific publications. Eqn is used to specify the typesetting of mathematical equations. The TeX format is conventionally used with a program called "tex" or "latex" that produces as output a static document. The eqn format is conventionally used with a program called "eqn" that produces as output a static image or description of an equation. (For information on TeX, see Donald E. Knuth, *The TeXbook*, Addison-Wesley, 1986. For information on eqn, see Brian W. Kernighan and Lorinda L. Cherry, "A System for Typesetting Mathematics," *Communications of the ACM* 18:3, March 1975; attached as Exhibit B.)

42. Fourth, Raggett I refers to the invocation of a "shared library or external filter to render the embedded data, e.g. by returning a pixmap". This passage uses several terms of art (in the art of computer science) in ways that teach non-interactivity. "Filter" is a term of art that refers to a type of non-interactive program that translates data from one format to another. "Render" as used by Raggett I is a term of art that refers to the generation of a static image that is to be displayed. "Pixmap" as used by Raggett I is a term of art for a data structure describing an image. "Return" is a term of art that refers to the information produced by a program when that program terminates. A program that has returned something cannot do anything else; for example it cannot provide interactive processing. The use of these four terms of art further teaches the use of static images.

43. Fifth, the only specific example of the use of Raggett's proposed embed tag that is given in Raggett I involves the use of a non-interactive filter which renders static data and then returns. The example depicts the use of the "eqn" program to translate the description of an equation into a static image.

44. Sixth, the discussion of the FIG and ISMAP features in Raggett I is inconsistent with the proposition that Raggett's proposed embed tag

allowed interaction with an embedded object. In Raggett I, an instance of Raggett's proposed embed tag can be placed within a FIG element:

> Instead of the *src* attribute, you can include an EMBED element immediately following the <fig> tag. This is useful for simple graphs, etc. defined in an external format.

(Raggett I at p. 12, emphasis in original) When the FIG element is used in conjunction with the ISMAP parameter (as described in the "Active areas" section of Raggett I, p. 13), the FIG element's display area becomes an image map: any mouse clicks made by the user within the visual depiction of the embedded data will be interpreted by the browser as pertaining to the image-map feature, and will therefore be intercepted by the browser and sent by the browser to the web server. This section of Raggett I teaches that the browser may intercept mouse clicks within the depiction of the embedded data, thereby contradicting the proposition that the embedded data itself can react to mouse clicks.

45. To my knowledge Raggett's proposed embed tag was never implemented. This is confirmed, for example, by Mr. Raggett's trial testimony:

> Q. Sure. I'm sorry. I think you mentioned on direct exam that Mr. Martin's work and Mr. Ang's work and Dr. Doyle's work weren't part of the HTML Plus specification [i.e., of Raggett I].
>
> A. Their work was not part of the specification.
>
> Q. Okay. Now, you understand that they wrote to you in 1994 to describe their use of the embed tag and in fact suggested that you use their version of the embed tag in your upcoming HTML specification, correct?
>
> A. They wrote to me saying that they'd obviously been looking at the HTML Plus specifications, and they were proposing something similar, and I responded to them that at that time there'd been a discussion in the summer of 1993, and at that time the consensus was that the group felt that there were higher priorities and so recommended that we drop the embed mechanism for that moment.

(Eolas v. Microsoft trial transcript at 1884:9-24; see Krueger declaration, Exhibit A)

46. However, if one of ordinary skill in the art (at the time) were asked to implement the Raggett I feature, he would do so by to starting with the existing code for handling IMG tags, and modifying that code. The existing IMG code was able to paint static images into the body of a page, based on an input file that described the image. This code would be modified to invoke an external program, which would return a static image that would then be pasted into the web page in the same manner as in an IMG tag. Such an implementation would not support interactivity within a web browser window.

47. The sentence about "linking to external editors for creating or revising embedded data" refers to the use of external programs by a Web page's author to edit or revise the external data before it is published on the author's Web server.

48. There is nothing in Raggett I to suggest that the "external editors" would provide any display within a web browser window. The editors that were (and still are) conventionally used to create or revise data all run in their own windows; nothing in Raggett I suggests that they would be modified to run within a browser window, or that a browser would be modified to allow the editors to operate in that way. The reference to "linking" to an "external" program refers to the use of a hyperlink or button that the user can click to launch a separate program, as is done with helper applications. (Having the browser automatically invoke an editor wouldn't make sense anyway, since only the page's author would be in a position to edit a copy of the page that anybody else would see, and it wouldn't make sense to invoke an editor automatically when ordinary users had no reason to want to invoke it.)

49. There is nothing in Raggett I that suggests how to provide an interactive program within a browser window – nothing about how to modify a browser to provide such a feature, and nothing about how to modify an editor to work with such a modified browser. No method for doing these things would have been obvious to one of ordinary skill.

D. The Raggett II Reference

50. Raggett II is a brief email message, written in response to requests for "equation support," "eqn support," and support for "embedded Postscript" in browsers. Equations, eqn data, and embedded postscript are all formats for specifying static data. The requesters ask for support for two rendering programs, eqn and ghostscript, both of which produce static images as output.

51. Raggett II responds by referring to the same functionality described in Raggett I.

52. Raggett II reiterates the teaching of Raggett I about the embedding of static images into Web pages. Raggett II refers to the use of external programs that "render[] foreign formats, e.g. as functions that take a sequence of bytes and return a pixmap." Here again the term of art "render" is used, referring to the creation of a static image.

53. Additionally, the programs are said to "return a pixmap." "Return" is a term of art that refers to the provision of information by a program when that program completes its execution. Therefore, once one of these programs has "return[ed] a pixmap", the program is no longer running and cannot do anything more. In particular, the program cannot provide any interactive functionality, since the program would have stopped running

before the browser even painted the returned static pixmap onto the screen.

54. Raggett II mentions the possibility of implementing the external program as a DLL or dynamically linked library. A DLL is just another way of packaging an executable software application.

55. Raggett II teaches that the programs could be "driven via pipes and stdin/stdout". This refers to a method by which one program invokes another, in such a way that the invoking program can provide input to the invoked program, and can receive any output produced by the invoked program. In this instance, the browser would invoke the external program, would provide the foreign data to the external program, and would receive the external program's output, as a static image.

E. The Unpatentability Arguments in the Office Action Are Unpersuasive.

56. From my knowledge of the field, my own personal experience, and the state of the art in 1994, to the extent that a person of ordinary skill in the art was familiar with the teachings of the art cited in the Office Action, I find that the rejection of claims 1 and 6 as obvious is incorrect.

57. For example, the Office Action concludes, incorrectly, that Raggett I teaches interactive processing within a browser window. As described above, Raggett I teaches the use of static content within a browser window, coupled with the use of external editors that appear in separate windows.

58. The core of the Office Action's argument on this point appears in this passage:

> Although Raggett I describes an example where the browser calls a program for rendering an equation in ASCII character format into a pixmap image of the equation, Raggett I does also recognize that more sophisticated browsers can link to external editors for creating or revising embedded data. These external editors that create or revise the embedded data would work in the same way as the simple example of providing equation support. (See Raggett 1: p. 6) However, the ability to create and revise the embedded data allows the user to interactively process the data within the browser window.

(Office Action at 5:42-6:5)

59. The Office Action is incorrect to say that an editor could work "in the same way" that the external rendering programs of Raggett I work. Raggett I's external rendering programs operate by rendering external data to a static image, such as a pixmap, and then returning. Having produced a static image, and having returned (that is, having completed their work), they could not provide interactivity. A program that worked "in the same

way" could not provide editing functionality, or any other form of interactivity.

60. In any case, the editor programs available at the time were incapable of operating in the manner suggested by the Office Action. (They were, of course, capable of being invoked in a separate window.) Raggett I does not suggest the possibility of modifying any editor program. I am not aware of any such description in the prior art of how such modifications might be done; nor does the Office Action point to such a description.

61. If anything the Raggett I and II references teach away from the combinations recited in claims 1 and 6 of the '906 patent. These references teach the use of static web pages, with which the user interacts by moving from page to page, as opposed to the model of the '906 patent where a page can contain a fully interactive object. The two Raggett references teach the inclusion of static images, in various formats, into web pages, but they do not teach interactive processing within a browser window.

62. Finally, I have been told by the patent attorney for Eolas and the Regents that I should consider as part of my obviousness analysis "secondary considerations" such as copying, long felt but unresolved need, properties of the claimed invention, licenses showing industry acceptance of the invention and skepticism of skilled artisans before the invention.

63. I believe there is exceptionally strong "secondary consideration" evidence demonstrating non-obviousness in the case. This evidence includes the failure of others to duplicate the invention. I know of no evidence that either Mr. Raggett or anyone else tried to implement the purportedly obvious combination. In fact, I understand that the "HTML+" syntax described in Raggett I was never implemented.

64. For these reasons, I conclude that the rejection of claims 1 and 6 as being unpatentable is incorrect. The claims of the '906 patent would not have been obvious in view of the references cited in the Office Action.

I declare that all statements made herein of my 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 18 U.S.C. Section 1001, and that such willful false statements may jeopardize the validity of the patent.

Dated: May 7, 2004

[illegible]

Edward W. Felten

Edward W. Felten

Dept. of Computer Science
Princeton University
35 Olden Street
Princeton NJ 08540
(609) 258-5906
(609) 258-1771 fax
felten@cs.princeton.edu

Education

Ph.D. in Computer Science and Engineering, University of Washington, 1993.
Dissertation title: "Protocol Compilation: High-Performance Communication for Parallel Programs." Advisors: Edward D. Lazowska and John Zahorjan.
M.S. in Computer Science and Engineering, University of Washington, 1991.
B.S. in Physics, with Honors, California Institute of Technology, 1985.

Employment

Professor of Computer Science, Princeton University, 2003-present.
Associate Professor of Computer Science, Princeton University, 1999-2003.
Assistant Professor of Computer Science, Princeton University, 1993-99.
Senior Computing Analyst, Caltech Concurrent Computing Project, California Institute of Technology, 1986-1989.

Director, Secure Internet Programming Laboratory, Dept. of Computer Science, Princeton University, 1996-present.

U.S. Dept. of Justice, Antitrust Division: consulting and testimony in Microsoft antitrust case, 1998-2002.
Robins Kaplan, Miller & Ciresi. Consulting and testimony in patent lawsuit, 1998-2003.
Keker & Van Nest. Consulting in intellectual property / free speech lawsuit, 2002.
Electronic Frontier Foundation. Consulting in intellectual property / free speech lawsuits, 2001-present.
Certus Ltd.: consultant in product design and analysis, 2000-2002.
Cigital Inc.: Technical Advisory Board member, 2000-present.
Cloakware Ltd.: Technical Advisory Board member, 2000-present.
Propel.com: Technical Advisory Board member, 2000-2002.
NetCertainty.com: Technical Advisory Board member, 1999-2002.
FullComm LLC: Scientific Advisory Board member, 1999-2001.
Sun Microsystems: Java Security Advisory Board member, 1997-present.
Finjan Software: Technical Advisory Board member, 1997-2002.

EXHIBIT A

International Creative Technologies: consultant in product design and analysis, 1997-98.
Bell Communications Research: consultant in computer security research, 1996-97.

Honors and Awards

Scientific America 50 Award, 2003.
Alfred P. Sloan Fellowship, 1997.
Emerson Electric, E. Lawrence Keyes Faculty Advancement Award, Princeton University School of Engineering, 1996.
NSF National Young Investigator award, 1994.
Outstanding Paper award, 1997 Symposium on Operating Systems Principles.
Best Paper award, 1995 ACM SIGMETRICS Conference.
AT&T Ph.D. Fellowship, 1991-93.
Mercury Seven Foundation Fellowship, 1991-93.

Research Interests

Computer security, especially relating to consumer products. Technology law and policy. Internet software. Operating systems. Interaction of security with programming languages and operating systems. Distributed computing. Parallel computing architecture and software.

Professional Service

Professional Societies and Advisory Groups

ACM Advisory Committee on Security and Privacy, 2002-2003.
DARPA Information Science and Technology (ISAT) advisory group, 2002-present.
Co-chair, ISAT study committee on "Reconciling Security with Privacy," 2001-2002.
National Academy study committee on Foundations of Computer Science, 2001-present.

Program Committees

USENIX General Conference, 2004.
Workshop on Foundations of Computer Security, 2003.
ACM Workshop on Digital Rights Management, 2001.
ACM Conference on Computer and Communications Security, 2001.
ACM Conference on Electronic Commerce, 2001.
Workshop on Security and Privacy in Digital Rights Management, 2001.
Internet Society Symposium on Network and Distributed System Security, 2001.
IEEE Symposium on Security and Privacy, 2000.
USENIX Technical Conference, 2000.
USENIX Windows Systems Conference, 2000.
Internet Society Symposium on Network and Distributed System Security, 2000.

IEEE Symposium on Security and Privacy, 1998.
ACM Conference on Computer and Communications Security, 1998.
USENIX Security Symposium, 1998.
USENIX Technical Conference, 1998.
Symposium on Operating Systems Design and Implementation, 1996.

Corporate Advisory Boards

Sun Microsystems, Java Security Advisory Council.
Cigital Inc.: Technical Advisory Board.
Cloakware Ltd.: Technical Advisory Board.
Propel.com: Technical Advisory Board.
Finjan Software: Technical Advisory Board.
Netcertainty: Technical Advisory Board.
FullComm LLC: Scientific Advisory Board.

University and Departmental Service

Faculty Advisory Committee on Policy, 2002-present.
Council of the Princeton University Community, 2002-present (Executive Committee)
Faculty Advisory Committee on Athletics, 1998-2000.
Computer Science Academic Advisor, B.S.E. program, class of 1998 (approx. 25 students)
Faculty-Student Committee on Discipline, 1996-98.
Faculty-Student Committee on Discipline, Subcommittee on Sexual Assault and Harrassment, 1996-98.

Students Advised

Ph.D. Advisees:

Minwen Ji (Ph.D. 2001). Dissertation: Data Distribution for Dynamic Web Content. Researcher at Compaq Systems Research Center.
Dirk Balfanz (Ph.D. 2000). Dissertation: Access Control for Ad Hoc Collaboration. Researcher at Xerox Palo Alto Research Center.
Dan S. Wallach (Ph.D. 1998). Dissertation: A New Approach to Mobile Code Security. Assistant Professor of Computer Science, Rice University.
Robert A. Shillner (Ph.D. expected 2004). Tentative dissertation title: Improving Distributed File Systems using a Shared Logical Disk. Technical staff member at Google.
Michael Schneider (Ph.D. expected 2003). Dissertation topic: Network Defenses against Denial of Service Attacks.

Significant Advisory Role:

Drew Dean (Ph.D. 1998). Advisor: Andrew Appel. Researcher at SRI International.
Stefanos Damianakis, Ph.D. 1998. Advisor: Kai Li. President, Netrics, Inc.
Pei Cao, Ph.D. 1996. Advisor: Kai Li. Assistant Professor of Computer Sciences, University of Wisconsin. On leave at Cisco Systems.

Lujo Bauer, Ph.D. 2003. Advisor: Andrew Appel. Postdoctoral researcher at Carnegie-Mellon University.

Publications

Books and Book Chapters

[1] Freedom to Tinker. Edward W. Felten. Publication expected, 2004.

[2] Securing Java: Getting Down to Business with Mobile Code. Gary McGraw and Edward W. Felten. John Wiley and Sons, New York 1999.

[3] Java Security: Web Browsers and Beyond. Drew Dean, Edward W. Felten, Dan S. Wallach, and Dirk Balfanz. In "Internet Besieged: Countering Cyberspace Scofflaws," Dorothy E. Denning and Peter J. Denning, eds. ACM Press, New York, 1997.

[4] Java Security: Hostile Applets, Holes and Antidotes. Gary McGraw and Edward Felten. John Wiley and Sons, New York, 1996.

[5] Dynamic Tree Searching. Steve W. Otto and Edward W. Felten. In "High Performance Computing", Gary W. Sabot, ed., Addison Wesley, 1995.

Journal Articles

[6] Mechanisms for Secure Modular Programming in Java. Software – Practice and Experience, 33:461-480, 2003.

[7] The Digital Millennium Copyright Act and its Legacy: A View from the Trenches. Illinois Journal of Law, Technology and Policy, Fall 2002.

[8] DRM and Fair Use: A Skeptical View. Edward W. Felten. Communications of the ACM. April, 2003.

[9] The Security Architecture Formerly Known as Stack Inspection: A Security Mechanism for Language-based Systems. Dan S. Wallach, Edward W. Felten, and Andrew W. Appel. ACM Transactions on Software Engineering and Methodology, 9:4, October 2000.

[10] Statically Scanning Java Code: Finding Security Vulnerabilities. John Viega, Tom Mutdosch, Gary McGraw, and Edward W. Felten. IEEE Software, 17(5), Sept./Oct. 2000.

[11] Client-Server Computing on the SHRIMP Multicomputer. Stefanos N. Damianakis, Angelos Bilas, Cezary Dubnicki, and Edward W. Felten. IEEE Micro 17(1):8-18, February 1997.

[12] Fast RPC on the SHRIMP Virtual Memory Mapped Network Interface. Angelos Bilas and Edward W. Felten. IEEE Transactions on Parallel and Distributed Computing, February 1997.

[13] Implementation and Performance of Integrated Application-Controlled File Caching, Prefetching and Disk Scheduling. Pei Cao, Edward W. Felten, Anna R. Karlin, and Kai Li. ACM Transactions on Computer Systems, Nov 1996.

[14] Virtual Memory Mapped Network Interface Designs. Matthias A. Blumrich, Cezary Dubnicki, Edward W. Felten, Kai Li, and Malena Mesarina. IEEE Micro, 15(1):21-28, February 1995.

Symposium Articles

[15] Receiver Anonymity via Incomparable Public Keys. Brent R. Waters and Edward W. Felten. ACM Conference on Computer and Communications Security. November 2003.

[16] Attacking an Obfuscated Cipher by Injecting Faults. Matthias Jacob, Dan Boneh, and Edward W. Felten. ACM Workshop on Digital Rights Management, November 2002.

[17] A General and Flexible Access-Control System for the Web. Lujo Bauer, Michael A. Schneider, and Edward W. Felten. 11th USENIX Security Symposium, August 2002.

[18] Informed Consent in the Mozilla Browser: Implementing Value-Sensitive Design. Batya Friedman, Daniel C. Howe, and Edward W. Felten. Hawaii International Conference on System Sciences, January 2002. (Best Paper award, organizational systems track.)

[19] Reading Between the Lines: Lessons from the SDMI Challenge. Scott A. Craver, John P. McGregor, Min Wu, Bede Liu, Adam Stubblefield, Ben Swartzlander, Dan S. Wallach, Drew Dean, and Edward W. Felten. USENIX Security Symposium, August 2001.

[20] Cookies and Web Browser Design: Toward Realizing Informed Consent Online. Lynette I. Millett, Batya Friedman, and Edward W. Felten. Proc. of CHI 2001 Conference on Human Factors in Computing Systems, April 2001.

[21] Timing Attacks on Web Privacy. Edward W. Felten and Michael A. Schneider. Proc. of 7th ACM Conference on Computer and Communications Security, Nov. 2000.

[22] Archipelago: An Island-Based File System for Highly Available and Scalable Internet Services. USENIX Windows Systems Symposium, August 2000.

[23] Proof-Carrying Authentication. Andrew W. Appel and Edward W. Felten. Proc. of 6th ACM Conference on Computer and Communications Security, Nov. 1999.

[24] An Empirical Study of the SHRIMP System. Matthias A. Blumrich, Richard D. Alpert, Yuqun Chen, Douglas W. Clark, Stefanos, N. Damianakis, Cezary Dubnicki, Edward W. Felten, Liviu Iftode, Margaret Martonosi, Robert A. Shillner, and Kai Li. Proc. of 25th International Symposium on Computer Architecture, June 1998.

[25] Performance Measurements for Multithreaded Programs. Minwen Ji, Edward W. Felten, and Kai Li, Proc. of 1998 SIGMETRICS Conference, June 1998.

[26] Understanding Java Stack Inspection. Dan S. Wallach and Edward W. Felten. Proc. of 1998 IEEE Symposium on Security and Privacy, May 1998.

[27] Extensible Security Architectures for Java. Dan S. Wallach, Dirk Balfanz, Drew Dean, and Edward W. Felten. Proc. of 16th ACM Symposium on Operating Systems Principles, Oct. 1997. Outstanding Paper Award.

[28] Web Spoofing: An Internet Con Game. Edward W. Felten, Dirk Balfanz, Drew Dean, and Dan S. Wallach. Proc. of 20th National Information Systems Security Conference, Oct. 1997.

[29] Reducing Waiting Costs in User-Level Communication. Stefanos N. Damianakis, Yuqun Chen, and Edward W. Felten. Proc. of 11th Intl. Parallel Processing Symposium, April 1997.

[30] Stream Sockets on SHRIMP. Stefanos N. Damianakis, Cezary Dubnicki, and Edward W. Felten. Proc. of 1st Intl. Workshop on Communication and Architectural Support for Network-Based Parallel Computing, February 1997. (Proceedings available as Lecture Notes in Computer Science #1199.)

[31] Early Experience with Message-Passing on the SHRIMP Multicomputer. Richard D. Alpert, Angelos Bilas, Matthias A. Blumrich, Douglas W. Clark, Stefanos Damianakis, Cezary Dubnicki, Edward W. Felten, Liviu Iftode, and Kai Li. Proc. of 23rd Intl. Symposium on Computer Architecture, 1996.

[32] A Trace-Driven Comparison of Algorithms for Parallel Prefetching and Caching. Tracy Kimbrel, Andrew Tomkins, R. Hugo Patterson, Brian N. Bershad, Pei Cao, Edward W. Felten, Garth A. Gibson, Anna R. Karlin, and Kai Li. Proc. of 1996 Symposium on Operating Systems Design and Implementation.

[33] Java Security: From HotJava to Netscape and Beyond. Drew Dean, Edward W. Felten, and Dan S. Wallach. Proc. of 1996 IEEE Symposium on Security and Privacy.

[34] Integrated Parallel Prefetching and Caching. Tracy Kimbrel, Pei Cao, Edward W. Felten, Anna R. Karlin, and Kai Li. Proc. of 1996 SIGMETRICS Conference.

[35] Software Support for Virtual Memory-Mapped Communication. Cezary Dubnicki, Liviu Iftode, Edward W. Felten, and Kai Li. Proc. of Intl. Parallel Processing Symposium, April 1996.

[36] Protected, User-Level DMA for the SHRIMP Network Interface. Matthias A. Blumrich, Cezary Dubnicki, Edward W. Felten, and Kai Li. Proc. of 2nd Intl. Symposium on High-Performance Computer Architecture, Feb. 1996

[37] Improving Release-Consistent Shared Virtual Memory using Automatic Update . Liviu Iftode, Cezary Dubnicki, Edward W. Felten, and Kai Li. Proc. of 2nd Intl. Symposium on High-Performance Computer Architecture, Feb. 1996

[38] Synchronization for a Multi-Port Frame Buffer on a Mesh-Connected Multicomputer. Bin Wei, Gordon Stoll, Douglas W. Clark, Edward W. Felten, and Kai Li. Parallel Rendering Symposium, Oct. 1995.

[39] A Study of Integrated Prefetching and Caching Strategies. Pei Cao, Edward W. Felten, Anna R. Karlin, and Kai Li. Proc. of 1995 ACM SIGMETRICS Conference. Best Paper award.

[40] Evaluating Multi-Port Frame Buffer Designs for a Mesh-Connected Multicomputer. Gordon Stoll, Bin Wei, Douglas W. Clark, Edward W. Felten, Kai Li, and Patrick Hanrahan. Proc. of 22nd Intl. Symposium on Computer Architecture.

[41] Implementation and Performance of Application-Controlled File Caching. Pei Cao, Edward W. Felten, and Kai Li. Proc. of 1st Symposium on Operating Systems Design and Implementation, pages 165-178, November 1994.

[42] Application-Controlled File Caching Policies. Pei Cao, Edward W. Felten, and Kai Li. Proc. of USENIX Summer 1994 Technical Conference, pages 171-182, 1994.

[43] Virtual Memory Mapped Network Interface for the SHRIMP Multicomputer. Matthias A. Blumrich, Kai Li, Richard D. Alpert, Cezary Dubnicki, Edward W. Felten, and Jonathan S. Sandberg. Proc. of Intl. Symposium on Computer Architecture, 1994.

[44] Performance Issues in Non-Blocking Synchronization on Shared-Memory Multiprocessors. Juan Alemany and Edward W. Felten. Proceedings of Symposium on Principles of Distributed Computing, 1992.

[45] Improving the Performance of Message-Passing Applications by Multithreading. Edward W. Felten and Dylan McNamee. Proceedings of Scalable High-Performance Computing Conference (SHPCC), 1992.

[46] A Highly Parallel Chess Program. Edward W. Felten and Steve W. Otto. 1988 Conference on Fifth Generation Computer Systems.

Other Publications

[47] Freedom to Tinker weblog, at http://www.freedom-to-tinker.com. Commentary on technology law and policy. Approximately 4000 readers per day.

[48] Secure, Private Proofs of Location. Brent Waters and Edward W. Felten. Submitted for publication, January 2003.

[49] An Efficient Heuristic for Defense Against Distributed Denial of Service Attacks using Route-Based Distributed Packet Filtering. Michael A. Schneider and Edward W. Felten. Submitted for publication, January 2003.

[50] Written testimony to House Commerce Committee, Subcommittee on Courts, the Internet, and Intellectual Property, oversight hearing on "Piracy of Intellectual Property on Peer to Peer Networks." September 2002.

[51] Written testimony to Senate Judiciary Committee hearings on "Competition, Innovation, and Public Policy in the Digital Age: Is the Marketplace Working to Protect Digital Creativity?" March 2002.

[52] Informed Consent Online: A Conceptual Model and Design Principles. Batya Friedman, Edward W. Felten, and Lynette I. Millett. Technical Report 2000-12-2, Dept. of Computer Science and Engineering, University of Washington, Dec. 2000.

[53] Mechanisms for Secure Modular Programming in Java. Lujo Bauer, Andrew W. Appel, and Edward W. Felten. Technical Report CS-TR-603-99, Department of Computer Science, Princeton University, July 1999.

[54] A Java Filter. Dirk Balfanz and Edward W. Felten. Technical Report 567-97, Dept. of Computer Science, Princeton University, October 1997.

[55] Inside RISKS: Webware Security. Edward W. Felten. Communications of the ACM, 40(4):130, 1997.

[56] Simplifying Distributed File Systems Using a Shared Logical Disk.Robert A. Shillner and Edward W. Felten. Princeton University technical report TR-524-96.

[57] Contention and Queueing in an Experimental Multicomputer: Analytical and Simulation-based Results. Wenjia Fang, Edward W. Felten, and Margaret Martonosi. Princeton University technical report TR-508-96.

[58] Design and Implementation of NX Message Passing Using SHRIMP Virtual Memory Mapped Communication. Richard D. Alpert, Cezary Dubnicki, Edward W. Felten, and Kai Li. Princeton University technical report TR-507-96.

[59] Protocol Compilation: High-Performance Communication for Parallel Programs. Edward W. Felten. Ph.D. dissertation, Dept. of Computer Science and Engineering, University of Washington, August 1993.

[60] Building Counting Networks from Larger Balancers. Edward W. Felten, Anthony LaMarca, and Richard Ladner. Univ. of Washington technical report UW-CSE-93-04-09.

[61] The Case for Application-Specific Communication Protocols. Edward W. Felten. Univ. of Washington technical report TR-92-03-11.

[62] A Centralized Token-Based Algorithm for Distributed Mutual Exclusion. Edward W. Felten and Michael Rabinovich. Univ. of Washington technical report TR-92-02-02.

[63] Issues in the Implementation of a Remote Memory Paging System. Edward W. Felten and John Zahorjan. Univ. of Washington technical report TR-91-03-09.

985 PH Ex. 5

Application of: Doyle et al.

Application Num.: 90/006,831

Filed: October 30, 2003

For: Distributed Hypermedia Method for Automatically Invoking External Application Providing Interaction and Display of Embedded Objects Within a Hypermedia Document

Examiner: Caldwell, A.T.

Art unit: 2157

Declaration of Edward W. Felten

I, Edward W. Felten, declare as follows:

1. I have been retained by Eolas and the Regents of the University of California to serve as an expert in the field of computer science and Internet software.
2. I filed a previous declaration in this matter. That declaration recites my technical expertise and attaches my Curriculum Vitae. I hereby incorporate my previous declaration into this declaration by reference.

I. Introduction

3. I have been asked to address the arguments presented in the Office Action mailed August 17, 2004 ("the Office Action") in connection with the reexamination of United States Patent No. 5,838,906 ("the '906 patent") that the claims of the '906 patent are unpatentable as being "obvious". For the reasons described in this declaration, I disagree with the arguments presented in the Office Action and, instead, believe that the claims of the '906 patent fully meet the requirements for patentability over the cited references, as those patentability arguments have been described to me.
4. To familiarize myself with the issues involved in the rejection of the claims, I have reviewed numerous documents, including the following: the '906 Patent and its file history, the documents sited in the previous office action mailed on March 12, 2004, the documents cited in the present Office Action, and all other documents referenced or cited in this declaration.
5. My previous declaration presented background material on the early history of Web technology and the prior art to the '906 Patent.

II. Response to the Unpatentability Arguments Raised in the Office Action

6. My previous declaration described my understanding of the legal standard for obviousness, the level of ordinary skill in the art at the time of the '906 Patent's invention, and the nature and purpose of the invention disclosed in the '906 Patent. I hereby incorporate that discussion into this declaration by reference.

A. The Grounds of Rejection

7. Claims 1 and 6 of the'906 patent have been rejected by the United States Patent Office as being obvious under 35 U.S.C. Sec. 103(a); as being unpatentable over the admitted prior art in the '906 patent and teaching of Berners-Lee, Raggett I, Raggett II, and Toye.

8. The Office Action asserts that a combination of the Berners-Lee, Raggett I, Raggett II, and Toye references would embody the relevant claims of the '906 Patent. For the reasons described below, I find this assertion to be incorrect.

9. From my knowledge of the field, my own personal experience, and the state of the art in 1994, to the extent that a Person Having Ordinary Skill in the Art ("PHOSA") was familiar with the teachings of the art cited in the Office Action, I find that the rejection of claims 1 and 6 as obvious is incorrect.

B. What Berners-Lee and the Raggett References Teach a PHOSA

10. In my previous declaration, I discussed the teachings of these references. I incorporate that discussion here by reference.

11. The previous Office Action proposed a combination of these three references. However, that proposed combination would lack the claim element of automatically invoking an executable application to enable interactive processing, as acknowledged in the latest Office Action.

> The combination of patentee's admitted prior art in view of Berners-Lee, Raggett I, and Raggett II does not explicitly teach a method that "enables interactive processing of said object." The combination teaches a method that embeds static objects, as opposed to dynamic objects, within distributed hypermedia documents.

(Office Action at p. 6)

12. The Berners-Lee reference teaches a model in which static pages can be published by anyone, on a server anywhere in the world, and read by anyone. The pages are connected by simple, unidirectional links that are used only to navigate from one page to another. A page is edited by its author using a separate editor application, and is viewed, but not modified, by its readers using a separate browser application.

13. Berners-Lee teaches that the browser renders a page, translating it into a set of fixed static images to be displayed, before the page is displayed to the user.

14. Berners-Lee teaches that the structure of a document is specified by markup commands that are interspersed within the text of the document. (See, e.g., Berners-Lee at p. 5.)

15. Berners-Lee teaches that the browser parses the text of a document in order to render that document, and that the browser handles the detection and resolution of hyperlinks.

16. The model taught by Berners-Lee is well suited for the purpose of Berners-Lee, which is to create a worldwide system for viewing and navigating static, published documents on a wide variety of client computers.

17. The Raggett references are directed to the problem of increasing the number of static data formats that can be viewed by users of the Berners-Lee system. Accordingly, Raggett teaches, consistently with Berners-Lee, that data is to be rendered into a static image before it is displayed.

18. Raggett teaches the use of an external "filter" program that is used to render data that is encoded in a format the browser cannot understand. This filter program does what the browser would do: it takes a description of what to display, and generates from it a fixed image to be painted onto the screen. Raggett teaches that this filter program finishes executing before the image it generates is painted onto the screen.

19. Raggett teaches that the rendered image produced by the external filter should not be interactive. This teaching can be seen, for example, in the description of the FIG and ISMAP features in Raggett I, as discussed in paragraph 44 of my previous declaration. Raggett I teaches that its EMBED tag can be placed within a FIG element:

> Instead of the *src* attribute, you can include an EMBED element immediately following the <fig> tag. This is useful for simple graphs, etc. defined in an external format.

(Raggett I at p. 12, emphasis in original) When the FIG element is used in conjunction with the ISMAP parameter (as described in the "Active areas" section of Raggett I, p. 13), the FIG element's display area becomes an image map: any mouse clicks made by the user within the visual depiction of the embedded data will be interpreted by the browser as pertaining to the image-map feature, and will therefore be intercepted by the browser and sent by the browser to the web server. In order to do this, the browser must intercept mouse clicks within the depiction of the embedded data, and this can only happen if that depiction does not itself provide interactivity.

20. Berners-Lee, Raggett I and Raggett II, alone or in combination, do not teach the claim element of enabling interactive processing of an object. Indeed, they teach away from the provision of interactive processing within the boundaries of a web page.

C. What Toye Teaches a PHOSA

21. Toye is directed to the creation of a system for collaborative editing (a term that would be understood by a PHOSA to refer to editing of one or more documents by multiple people) of engineering documents within an engineering team, using a single object-oriented database to store the documents needed by an engineering workgroup.

22. The Office Action characterizes Toye as follows:

> Toye on the other hand discloses a distributed hypermedia system in which a hypermedia browser allows a user to interactively process an object embedded within a distributed hypermedia document. **(See Toye: p. 40 description of NoteMail, particularly p. 40, col. 2, first complete paragraph).**

(Office Action at 6:23-26, emphasis in original)

23. Contrary to the Office Action's assertion, Toye does not teach the use of a "distributed hypermedia document," as that term is used in the '906 claims. The term's meaning, as understood by a PHOSA, was reiterated in the '906 Patent's specification. For example:

> A distributed hypertext or hypermedia document typically has many links within it that specify many different data objects located in computers at different geographic locations connected by a network.

('906 Patent at 2:59-62)

24. Rather than teaching that the documents accessed by a user could be "located at computers at different geographic locations," Toye teaches the use of a single centralized, object-oriented database for storage of a workgroup's documents:

> Multimedia engineering documents containing raw text, encoded images, audio clips, video clips, etc. can get quite large. Sending such documents via email to everyone on a large design team can be costly in terms of both time and storage. Instead of transferring full copies to everyone, it is more efficient to store the components of the message *in one place* and just transmit a set of reference pointers. NoteMail uses an object-oriented knowledge base, known as DIS, for this repository function.
>
> Conceptually, DIS provides a *centralized information storage and management service for all the data associated with a design*: CAD files, e-mail messages, specifications, simulation results, and so forth. In practice, most data remains physically under the control of the application that created it; a persistent object is created in DIS to serve as a reference pointer or "handle."

(Toye at p. 40-41, emphasis added) The use of a centralized, object-oriented database makes sense given the goal of Toye to support collaboration within an engineering workgroup. However, it contradicts the Office Action's assertion that Toye uses the distributed hypermedia documents of the '906 claims. Indeed, by teaching centralized storage, Toye teaches away from the use of distributed hypermedia documents.

25. For the same reason, Toye does not teach the use of a "distributed hypermedia environment," as that term is used in the '906 claims. The environment provided by Toye is not "distributed" in the sense of the '906 claims, since it relies on the centralization of a user's document storage in one place. Toye teaches away from the use of a distributed hypermedia environment.

26. Likewise, Toye does not teach the use of a hypermedia browser, as that term is used in the '906 claims. Toye teaches no software application that parses distributed hypermedia documents, and it does not teach other browser-related elements of the '906 claims, such as parsing of distributed hypermedia documents by a browser, identifying text formats in distributed hypermedia documents and responding to predetermined text formats to initiate processing specified by those formats, utilizing a browser to display at least a portion of a distributed hypermedia document in a browser-controlled window, and parsing an embed text format in such a document.

27. Toye does use the term "hypermedia browser" but with a different meaning. For example, the "hypermedia browser" of the '906 claims must parse hyperlinks from within a text document, but Toye does not provide that feature. See also the other deficiencies of Toye described in the previous paragraph.

28. At the time of Toye, a PHOSA would have known about web browser technology, and would have known that Web browser applications were available to run on widely used platforms. Yet Toye teaches that a new document viewing application (NoteMail) should be developed, rather than using existingWeb browser technology. This clearly teaches away from the use of Web browser technology with Toye.

29. Rather than teaching the use of standard hyperlinks (as described, e.g., in the '906 Patent at 2:37-47), Toye teaches the use of rich, bi-directional links (i.e., links traversable in both directions) between objects.

> The information can also be organized by adding links between objects. The links are themselves first-class objects that can be annotated with semantic labels and constraints characterizing the nature of their dependency. For example, some links may simply be hypertext pointers, used to organize e-mail message into discussion threads and link them to related documents and data. Others may be used to represent a [sic] formal constraints in a behavioral model. In either case, maintaining the links is the job of external applications that provide navigation, constraint management, change notification and other services. These applications can attach daemons to the links, which are run automatically when either side changes.

(Toye at p. 41, first partial paragraph) These links provide functionality far beyond the simple hyperlinks used on the web, and they are implemented by separate applications. Again, this teaches away from the use of the hypermedia browser of Berners-Lee.

30. The bi-directional links of Toye can, for example, represent formal constraints that connect two documents, so that a change in either of the two documents causes a corresponding change to happen automatically in the other document. This model is appropriate within an engineering workgroup, but it doesn't make sense on the Web, where hyperlinks often link documents written by different people who may not know or trust each other. For example, on the Web, I can create a page that links to the CNN home page; but it would not be appropriate for me to create a Toye-style link that would allow me, by changing my page, to cause changes on CNN's home page. Instead, Web hyperlinks follow a more appropriate (for the Web's goals) model in which only I can modify my own page, and only CNN can modify their page. This difference teaches away from the use of a Web browser with Toye.

31. Unlike Berners-Lee, Toye teaches that the structure of multimedia content is not specified within the text of the main or enclosing text document, but is specified elsewhere, for example in a separate MIME-part that uses Toye's "Format" data type. (See, e.g., Toye at p. 40, bottom of first column.)

32. The Toye reference teaches an entire system, of which the NoteMail module cited in the Office Action is just one part. The system taught by Toye has different aims, and teaches a different model, than Berners-Lee and Raggett.

D. What Toye Teaches About Interaction with External Programs

33. Toye teaches that NoteMail interacts with an external program by first displaying a static snapshot of the external content. If the user clicks on that static snapshot, the external editor application is restarted in a separate window.

34. As noted in the Office Action (at 7:3-6), the key to understanding Toye's interaction with external programs can be found in Toye's discussion of restarting the external editor when the user clicks on the snapshot of the external content:

> When a data object or file is selected for inclusion in the notebook, the system will automatically invoke the appropriate application for displaying that item in the notebook.... Subsequently selecting the displayed data with a mouse will restart the original application, so that the data can be edited or updated without leaving the notebook environment. The functionality is similar to opening a file using the Macintosh Finder and automatically invoking the appropriate application for processing that file.

(Toye at p. 40, col. 2, first full paragraph) It is clear from this discussion that before the data can be edited, the user must select the displayed data with the mouse and the application must be restarted. Since the user must take specific action to select the data before editing is enabled, the editor is not "automatically invoke[d] ... in order to display said object and enable interactive processing" as required by the '906 claims.

35. The fact that the application must be restarted in order to enable editing tells us that the application could not have been running already in a way that enabled editing. (If it were, no restarting would be necessary.) Thus Toye teaches that the data, as originally displayed (i.e., before it is selected by the user) cannot be edited. It follows that all that is displayed initially is a static, non-editable snapshot of the data.

36. Toye does teach the launching of a separate application, but that application is launched in a separate window from the enclosing document. Toye teaches that its application launching "functionality is similar to opening a file using the Macintosh Finder and automatically invoking the appropriate application for processing that file." (Toye at p. 40, col. 2, first full paragraph) In the Macintosh Finder, I know by personal experience, and a PHOSA would have known, that opening a file launched an application in a separate window area. I note also that the Finder did not invoke an application automatically, but did so only in response to a mouse click selection by the user.

37. Toye does talk about displaying a document's data within the notebook environment. Even accepting, for the sake of argument, that the notebook environment of Toye is a browser, this still does not meet the requirements of the '906 claims. The '906 claims require not only that the interactivity be provided within the browser window, but that it be provided "within a display area … within the portion of said first distributed hypermedia document being displayed …." ('906 Patent at 17:23-28). This element is not taught by Toye. (Nor is it taught by Berners-Lee or either of the Raggett references.)

38. Indeed, Toye teaches the use of external editor programs that have not been modified from their standard versions. (See, e.g., Toye at p. 40, col. 2, first full paragraph: "any application that displays through an X-server") Such unmodified programs are not suitable for use within an enclosing document display, because the unmodified programs conventionally display menus and button bars at the top, and other graphical elements around their edges. External application windows with these elements on their borders cannot naturally be displayed within a document display; at most they could be displayed in a window area elsewhere in a windowing environment, as discussed in the previous paragraph. To enable a reasonable editing experience within a document display, the applications would have to be modified; but Toye teaches that they are not modified.

E. No Teaching or Suggestion to Combine

39. Neither Toye nor any other reference suggests a combination of Toye with Berners-Lee, Raggett I and Raggett II.

40. Regarding a teaching or suggestion to combine, the Office Action says only this:

> It would have been readily apparent to a skilled artisan to modify the method discussed above, combining the teachings of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II, by further modifying

> the combination's static embedded object to be a dynamic embedded object as taught by Toye. Such a further modification would have been apparent based on Toye's teaching that its architecture provides openness and flexibility (See Toye: p. 40 col. 2 second complete paragraph).

(Office Action at 6:28-34) (I note that the term "dynamic embedded object" does not appear in the '906 claims, and that what the Office Action calls the "dynamic embedded object as taught by Toye" is not the "object" of the '906 claims, because it is not displayed in the manner required by the '906 claims, as explained above.)

41. The Office Action is incorrect when it implies that the cited paragraph of Toye suggests a combination of Toye with a web browser. The cited paragraph of Toye reads as follows:

> We are aware of only one other multimedia editor with such an architecture, MediaMosaic [citation]. Other engineering notebook projects, by contrast, lack this openness and flexibility. For example, the Virtual Notebook System [citation] can display only static bitmaps; GE's Electronic Design Notebook [citation], which is built on FrameMaker, can run only those applications whose output formats are compatible with the handful of input formats that FrameMaker accepts.

(Toye at p. 40 col. 2 second complete paragraph) In this paragraph, Toye is simply asserting that its system has advantages over other engineering collaboration systems. Toye offers more than static bitmaps; it offers also the ability to click on those bitmaps and launch an external application (in a separate window, as discussed above). Toye offers more than just FrameMaker-compatible formats. "Openness and flexibility" are little more than buzzwords here. Nothing in this paragraph would teach a PHOSA that Toye could or should be combined with a web browser.

F. The References Teach Away From the Suggested Combination

42. As discussed above, the cited references teach away from a combination. Toye teaches collaborative editing of documents; Berners-Lee teaches that documents are created by an author and read (without editing) by a set of readers. Toye teaches storage of documents in a centralized object-oriented database; Berners-Lee teaches that documents can be retrieved from anywhere and everywhere on the Internet. Toye teaches that display structure is specified using a separate "Format" data type, outside a text document; Berners-Lee teaches that display structure is specified by markup commands within a text document. Toye teaches rich, bi-directional links implemented by separate applications; Berners-Lee teaches simple unidirectional links, providing only navigation and implemented by a browser. Toye teaches that users need not know where documents are located; Berners-Lee teaches that users know URLs, which contain location information.

43. Importantly, Toye teaches away from the use of distributed hypermedia documents, which is the central idea of Berners-Lee.

G. The Suggested Combination Would Not Embody the '906 Claims

44. The Office Action suggests a combination of Berners-Lee, Raggett I, Raggett II, and Toye:

> It would have been readily apparent to a skilled artisan to modify the method discussed above, combining the teachings of the admitted prior art in view of Berners-Lee, Raggett I, and Raggett II, by further modifying the combination's static embedded object to be a dynamic embedded object as taught by Toye.

(Office Action at 6:28-31)

45. Even granting, for the sake of argument, that such a combination would be possible, the result would not embody the '906 claims.

46. The resulting system, contrary to the language of the '906 claims, would not automatically invoke an external program to enable interactive processing within a browser window.

47. The Berners-Lee/Raggett combination teaches that external data is rendered to a static bitmap that is then displayed within a browser window. Toye teaches that external data is displayed as a static image, and if the user clicks that image, an editor application is launched in a separate window.

48. The combination, assuming it could be constructed, would therefore involve the use of the Raggett method to create a static bitmap within a browser window, in such a way that a user clicking on that static bitmap would launch an editor program in an external window as in Toye.

49. This combination would not provide automatic invocation of the editor program. The editor program would not be invoked immediately when the user visited the enclosing web page. Instead, the invocation would happen only after the user took the additional manual action of selecting the static image by clicking on it.

50. This combination would not provide interactive processing within the portion of the first hypermedia document displayed within the browser window. Interactive processing would occur only within the external editor window that was launched in response to the user's mouse click.

51. The fact that these two claim elements are missing is consistent with the fact that they are missing in all four of the references being combined.

III. <u>Conclusion</u>

52. For the reasons explained above, I conclude that the rejection of claims 1 and 6 as being unpatentable is incorrect. The claims of the '906 patent would not have been obvious in view of the references cited in the Office Action.

I declare that all statements made herein of my 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 18 U.S.C. Section 1001, and that such willful false statements may jeopardize the validity of the patent.

Dated: October 6, 2004

[illegible]

Edward W. Felten

985 PH Ex. 6

MAR 1 1 2005 PATENT & TRADEMARK OFFICE

PTO/SB/26 (09-04)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TERMINAL DISCLAIMER TO OBVIATE A DOUBLE PATENTING REJECTION OVER A "PRIOR" PATENT	Docket Number (Optional) 006-1-4

In re Application of: Doyle, et al.

Application No.: 10/217,955

Filed: 08/09j/2002

For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT

The owner*, Regents of the University of California, of 100 percent interest in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term **prior patent** No. 5,838,906 as the term of said prior patent is defined in 35 U.S.C. 154 and 173, and as the term of said **prior patent** is presently shortened by any terminal disclaimer. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and the **prior patent** are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of the term of any patent granted on the instant application that would extend to the expiration date of the full statutory term as defined in 35 U.S.C. 154 and 173 of the **prior patent**, "as the term of said **prior patent** is presently shortened by any terminal disclaimer," in the event that said **prior patent** later:
expires for failure to pay a maintenance fee;
is held unenforceable;
is found invalid by a court of competent jurisdiction;
is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321;
has all claims canceled by a reexamination certificate;
is reissued; or
is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

Check either box 1 or 2 below, if appropriate.

1. ☐ For submissions on behalf of a business/organization (e.g., corporation, partnership, university, government agency, etc.), the undersigned is empowered to act on behalf of the business/organization.

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.

2. ☑ The undersigned is an attorney or agent of record. Reg. No. 30,077

[illegible]
Signature

March ~~8~~ 7, 2005
Date

Charles E. Krueger
Typed or printed name

(925) 944-3320
Telephone Number

☑ Terminal disclaimer fee under 37 CFR 1.20(d) included.

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner). Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

03/14/2005 EAREGAY1 00000040 10217955
02 FC:2814 65.00 OP

Application Number	Application No.	Applicant(s)	
	10/217,955	DOYLE ET AL.	

TERMINAL DISCLAIMER	☒ APPROVED	☐ DISAPPROVED
Document Code - DISQ	This patent is subject to a Terminal Disclaimer	
INTERNAL DOCUMENT – DO NOT MAIL		

U.S. Patent and Trademark Office

985 PH Ex. 7

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

[illegible]

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 05/05/2005

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
CALDWELL, ANDREW T	
ART UNIT	PAPER NUMBER
2137	

DATE MAILED: 05/05/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Suspension of Action

The outcome of reexamination number 90/006,831 has a material bearing on the patentability of the claims in this application. Prosecution in this application is SUSPENDED FOR A PERIOD OF **SIX (6)** MONTHS from the mailing date of this letter. Upon suspension or termination of the reexamination, whichever is earlier, applicant should make an inquiry as to the status of the application.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andrew Caldwell, whose telephone number is (571) 272-3868. The examiner can normally be reached on M-Th from 9:00 a.m. to 5:30 p.m. EST.

Any general inquiry relating to the status of this application can be answered using Patent Application Information Retrieval (PAIR) system, which is available at the USPTO web site. Any questions on using the PAIR system should be directed to the Patent Electronic Business Center toll free at (866) 217-9197.

Andrew Caldwell

Andrew Caldwell
571-272-3868
April 28, 2005

985 PH Ex. 8

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 01/18/2006

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER
DONAGHUE, LARRY D

ART UNIT	PAPER NUMBER
2154	

DATE MAILED: 01/18/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

1. The outcome of reexamination number 90/007,858 has a material bearing on the patentability of the claims in this application. *Ex parte* prosecution is SUSPENDED FOR A PERIOD OF 6 MONTHS from the date of this letter. Upon expiration of the period of suspension or termination of the rexamination, applicant should make an inquiry as to the status of the application.

2. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Larry D. Donaghue whose telephone number is 571-272-3962. The examiner can normally be reached on M-F 8:00-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Follansbee can be reached on 571-272-3964. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]
LARRY D. DONAGHUE
PRIMARY EXAMINER

APPROVED
[illegible]

PAUL SEWELL
ACTING DIRECTOR

985 PH Ex. 9

11

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 10/18/2006

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	
ART UNIT	**PAPER NUMBER**
2154	

DATE MAILED: 10/18/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

0C (Rev. 10/03)

1. The outcome of reexamination number 90/007,858 has a material bearing on the patentability of the claims in this application. *Ex parte* prosecution is SUSPENDED FOR A PERIOD OF 6 MONTHS from the date of this letter. Upon expiration of the period of suspension or termination of the rexamination, applicant should make an inquiry as to the status of the application.

2. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Larry D. Donaghue whose telephone number is 571-272-3962. The examiner can normally be reached on M-F 8:00-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Follansbee can be reached on 571-272-3964. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]
LARRY D. DONAGHUE
PRIMARY EXAMINER

APPROVED
[illegible]

PAUL SEWELL
ACTING DIRECTOR

985 PH Ex. 10

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 08/13/2007

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER
DONAGHUE, LARRY D

ART UNIT	PAPER NUMBER
2154	

DATE MAILED: 08/13/2007

Please find below and/or attached an Office communication concerning this application or proceeding.

The request for deferral/suspension of action under 37 CFR 1.103 has been approved.

PTO-90C (Rev. 10/03)

1. The outcome of reexamination number 90/007,858 has a material bearing on the patentability of the claims in this application. *Ex parte* prosecution is SUSPENDED FOR A PERIOD OF 6 MONTHS from the date of this letter. Upon expiration of the period of suspension or termination of the rexamination, applicant should make an inquiry as to the status of the application.

2. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Larry D. Donaghue whose telephone number is 571-272-3962. The examiner can normally be reached on M-F 8:00-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Follansbee can be reached on 571-272-3964. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Approved
[illegible]

PAUL SEWELL
ACTING DIRECTOR

[illegible]
LARRY D. DONAGHUE
PRIMARY EXAMINER

985 PH Ex. 11

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Doyle et al. Application No.: 10/217,955 Filed: 08/09/2002 For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT	Examiner: DONAGHUE, LARRY D Art Unit: 2154 SUPPLEMENTAL AMENDMENT AFTER NON-FINAL REJECTION

Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

In response to the Office Action mailed 09/09/2004, please amend the application as follows:

Amendments to the Claims begin on page 2 of this paper.

Remarks/Conclusion begins on page 14 of this paper.

AMENDMENTS TO THE CLAIMS:

This listing of the claims replaces all prior versions, and listings, of claims in the application.

LISTING OF CLAIMS:

1-3 (canceled)

4. (new) A method for running an application program in a distributed hypermedia network environment, wherein the network environment comprises at least one client workstation and one network server coupled to the network environment, the method comprising:

receiving, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

executing the browser application on the client workstation, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of at least a portion of an object external to the file, where the object has type information associated with it;

utilizing the type information to identify and locate an executable application external to the file; and

automatically invoking the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

5. (new) The method of claim 4 where:

the information to enable comprises text formats.

6. (new) The method of claim 5 where the text formats are HTML tags.

7. (new) The method of claim 4 where the information contained in the file received comprises at least one embed text format.

8. (new) The method of claim 4 where the step of identifying an embed text format comprises:

parsing the received file to identify text formats included in the received file.

9. (new) The method of claim 8 where the parsing is by a parser in the browser.

10. (new) The method of claim 4 where the processing specified by the text formats is specified directly.

11. (new) The method of claim 4 where the correspondence is implied by the order of the text format in a set of all of the text formats.

12. (new) The method of claim 4 where the embed text format specifies the location of at least a portion of an object directly.

13. (new) The method of claim 4 where having type information associated is by including type information in the embed text format.

14. (new) The method of claim 4 where automatically invoking does not require interactive action by the user.

15. (new) The method of claim 4, wherein said executable application is a controllable application and further comprising the step of:

interactively controlling said controllable application on said client workstation via inter-process communications between said browser and said controllable application

16. (new) The method of claim 15, wherein the communications to interactively control said controllable application continue to be exchanged between the

controllable application and the browser even after the controllable application program has been launched.

17. (new) The method of claim 16, wherein additional instructions for controlling said controllable application reside on said network server, wherein said step of interactively controlling said controllable application includes the following substeps:

issuing, from the client workstation, one or more commands to the network server;

executing, on the network server, one or more instructions in response to said commands;

sending information from said network server to said client workstation in response to said executed instructions; and processing said information at the client workstation to interactively control said controllable application.

18. (new) The method of claim 17, wherein said additional instructions for controlling said controllable application reside on said client workstation.

19. (new) One or more computer readable media encoded with software comprising computer executable instructions and when the software is executed operable to:

receive, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

cause the client workstation to utilize the browser to:

respond to text formats to initiate processing specified by the text formats;

display at least a portion of the document within the browser-controlled window;

identify an embed text format corresponding to a first location in the document, the embed text format specifying the location of at least a portion of an object external to the file, with the object having type information associated with it;

utilize the type information to identify and locate an executable application external to the file; and

automatically invoke the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

20. (new) The computer readable media of claim 19 where:
the information to enable comprises text formats.

21. (new) The computer readable media of claim 20 where:
the text formats are HTML tags.

22. (new) The computer readable media of claim 19 where:
the information contained in the file received comprises at least one embed text format.

23. (new) A method of serving digital information in a computer network environment having a network server coupled the network environment, and where the network environment is a distributed hypermedia environment, the method comprising:

communicating via the network server with at least one client workstation over said network in order to cause said client workstation to:

receive, over said network environment from said server, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

invoke, at said client workstation, a browser application, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying, on said client workstation, at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of at least a portion of an object external to the file, where the object has type information associated with it;

utilizing the type information to identify and locate an executable application external to the file; and

automatically invoking the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

24. (new) The method of claim 23 where:
the information to enable comprises text formats.

25. (new) The method of claim 24 where:
the text formats are HTML tags.

26. (new) The method of claim 24 where:
the information contained in the file received comprises at least one embed text format.

27. (new) A method for running an application program in a computer network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an

embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file, wherein the object has type information associated with it,; wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format, the method comprising:

utilizing said executable application external to said file to interactively process said object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

28. (new) The method of claim 27 where:
the information to enable comprises text formats.

29. (new) The method of claim 28 where:
the text formats are HTML tags.

30. (new) The method of claim 27 where:
the information contained in the file received comprises at least one embed text format.

31. (new) One or more computer readable media encoded with software comprising computer executable instructions for use in a system having at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file,

wherein the object has type information associated with it, wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format, with software encoded on said computer readable media, identified by said type information and when automatically invoked, operable to:

cause the client workstation to display said object and enable interactive processing of said object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

32. (new) The method of claim 31 where:
the information to enable comprises text formats.

33. (new) The method of claim 32 where:
the text formats are HTML tags.

34. (new) The method of claim 31 where:
the information contained in the file received comprises at least one embed text format.

35. (new) A method for serving digital information in a computer network environment, with a network server coupled to said network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file, wherein the object has type information associated

with it, wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format; said method comprising:

communicating via said server with at least one client workstation over said network in order to cause said client workstation to:

utilize said executable application external to said file to enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

36. (new) The method of claim 35 where:

the information to enable comprises text formats.

37. (new) The method of claim 36 where:

the text formats are HTML tags.

38. (new) The method of claim 35 where:

the information contained in the file received comprises at least one embed text format.

39. (new) A method for running an application program in a distributed hypermedia network environment, wherein the network environment comprises at least one client workstation and one network server coupled to the network environment, the method comprising:

receiving, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

executing the browser application on the client workstation, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of an object;

identifying and locating program code associated with the object; and

automatically invoking the program code, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object, while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the program code is part of a distributed application, and wherein at least a portion of the distributed application is for execution on a network server coupled to the network environment.

40. (new) The method of claim 39 where:

the information to enable comprises text formats.

41. (new) The method of claim 40 where:

the text formats are HTML tags.

42. (new) The method of claim 39 where:

the information contained in the file received comprises at least one embed text format.

43. (new) A method of serving digital information in a computer network environment having a network server coupled to said network environment, and where the network environment is a distributed network environment, the method comprising:

communicating via the network server with at least one client workstation over said network in order to cause said client workstation to:

receive, over said network environment from the network server, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

invoke, at said client workstation, a browser application, with the browser application, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying, on said client workstation, at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of an object ;

identifying and locating program code associated with the object; and

automatically invoking the program code, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the program code is part of a distributed application, and wherein at least a portion of the distributed application is for execution on the network server.

44. (new) The method of claim 43 where:
the information to enable comprises text formats.

45. (new) The method of claim 44 where:
the text formats are HTML tags.

46. (new) The method of claim 43 where:
the information contained in the file received comprises at least one embed text format.

47. (new) A method for serving digital information in a computer network environment, with a network server coupled to said network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, wherein the network environment is a distributed hypermedia environment, wherein the client workstation receives, over the network environment from the server, at least one file containing information to enable a browser application to display, on the client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein the client workstation executes a browser application,

with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of an object, wherein program code associated with the object is identified and located, wherein the executable application is automatically invoked, in response to the identifying of the embed text format, to enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the program code is part of a distributed application, and wherein at least a portion of the distributed application is for execution on the network server; said method comprising:

communicating via the network server with at least the client workstation over the network in order to receive commands from the client workstation;

executing one or more instructions in response to the commands;

sending information to the client workstation in response to the executed instructions, to allow processing of the information at the client workstation for interactively controlling the controllable application.

48. (new) The method of claim 47 where:
the information to enable comprises text formats.

49. (new) The method of claim 48 where:
the text formats are HTML tags.

50. (new) The method of claim 47 where:

the information contained in the file received comprises at least one embed text format.

REMARKS

Claims 1-3 have been examined, claims 1-3 are cancelled, and claims 4-50 are added. Accordingly, claims 4-50 are now pending in the application. Reexamination and reconsideration of all outstanding rejections and objections is requested.

Claims 1 through 3 are rejected under 35 U.S.C. §103(a) as being unpatentable over the admitted prior art in U.S. Patent No. 5,838,906, the teachings of Berners-Lee, Raggett I, and Raggett II (collectively the "four-way combination") and the newly cited teaching of Toye.

Claim 2 is rejected under 35 U.S.C. §101 for same invention double patenting. Claims 1 and 3 are rejected under the judicially created doctrine of obviousness-type double patenting.

PROCEDURAL HISTORY

This application is a continuation and claims the benefit of U.S. Application No. 09/075,359, filed May 8, 1998, which is a continuation of U.S. Application No. 08/324,443, filed October 17, 1994, which issued as U.S. Patent No. 5,838,906 (hereinafter "the '906 patent").

There have been two reexaminations of the '906 patent. The first reexamination was a Director Ordered Reexamination, Control No. 90/006,831 ("the first reexamination") which resulted in issuance of a Reexamination Certificate on 5/17/2006 without amending the claims. Shortly after the NIRC for the first reexamination was posted on the PAIR page the second reexamination, Control No. 90/007,838 ("the second reexamination") was requested.

The office action relating to the currently pending application was mailed on 09/09/2004 which non-finally rejected claims 1-3 as described above. This rejection is identical to the rejection then pending in the first reexamination. A response to the first office action was filed 03/11/2005.

Subsequent to the filing of the response, prosecution of the application has been suspended by the patent office. Letters of suspension were mailed 05/05/2005, 01/18/2006, 10/18/2006 and 08/13/2007.

The first letter of suspension stated that the outcome of the first reexamination has a material bearing on the patentability of the claims in the present application. The first reexamination concluded with issues of patentability resolved in favor of the patentee.

The subsequent letters of suspension state that the outcome of the second reexamination has a material bearing on the patentability of the claims in the present application. The second reexamination has not been resolved.

Due to the delay caused by the suspension of prosecution and the favorable outcome of the first reexamination a new set of claims is submitted herewith.

TRAVERSE

The current rejection with regard to the pending claims is traversed for the following reasons.

Claim 19 recites one or more computer readable media encoded with software comprising computer executable instructions and when the software is executed operable to receive, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window.

The software when executed is operable to cause the client workstation to utilize the browser to respond to text formats to initiate processing specified by the text formats, display at least a portion of the document within the browser-controlled window, identify an embed text format corresponding to a first location in the document, the embed text format specifying the location of at least a portion of an object external to the file, with the object having type information associated with it, utilize the type information to identify and locate an executable application external to the file and automatically invoke the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

The REEXAMINATION REASONS FOR PATENTABILITY/CONFIRMATION (hereinafter "the Confirmation"), included with the Notice of Intent to Issue Ex Parte Reexamination Certificate (mailed 09/27/2005), which concluded the first reexamination includes four parts analyzing whether the claims in reexamination are unpatenable over the admitted prior art in the U.S. Patent No. 5,838,906 ('906 patent), the teachings of Berners-Lee, Raggett I, and Raggett II (the "four-way combination") and the newly cited teaching of Toye.

Part I of the Confirmation, at page 3, determines that the cited references do not fairly suggest or teach all the elements recited in the reexamination claims.

In Part I of the Confirmation the examiner states that:

> There is no suggestion or teaching in either Toye, the admitted prior art (Mosaic), Berners-Lee, Raggett I or Raggett II of automatically invoking an external application to execute on a client computer, when an embed text format is parsed to display and interactively control an object in a display window in a hypermedia document received over a network from a network server, being displayed in a browser-controlled window on the client computer. (page 3).

It is also stated that:

> As acknowledged by the previous Examiner, the cited four-way combination of the patent owner's admitted prior art (APA), Berners-Lee, Raggett I and Raggett II, "does not explicitly teach a method that **enables interactive** processing of said object". The combination teaches a method that embeds static objects, as opposed to dynamic objects, with distributed hypermedia documents. (page 3: emphasis in original).

Further, in Part I it is stated that Toye teaches the use of an image or icon that consists of a "static snapshot" of external content and that interactive processing is enabled only after a user manually clicks on the "static snapshot". (Confirmation page 10).

The current rejection is respectfully traversed for the following reasons.

The establishment of a *prima facie* case of obviousness requires that all the claim limitations must be taught or suggested by the prior art. MPEP §2143.03

Claim 19 of the present application recites "automatically invoke the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable interactive processing of the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window".

As determined in Part I of the Confirmation, neither the four-way combination of the patent owner's admitted prior art (APA), Berners-Lee, Raggett I and Raggett II nor Toye, singularly or in combination, fairly teach or suggest automatically invoking an external application to enable interactive processing of an object being displayed within the display area.

The four-way combination was determined to teach displaying a static object. Toye was found to display a "static snapshot" of external content where interactive processing was not automatically invoked but required manual selection by the user.

Accordingly, since the recited automatically invoking to enable interactive processing is not taught or suggested by the cited references a *prima facie* case of obviousness has not been established.

Independent claims 4, 23, 27, 31, 35, 39, 43 and 47 recite similar limitations as claim 19 and are thus allowable for the same reasons. The remaining claims are dependent claims which are allowable for the same reasons as the claims on which they depend.

The other parts of the Confirmation recite other findings supporting the determination that the reexamination claims are not unpatentable over the cited references and these findings also support a determination that the pending claims of the present application are not unpatentable over the cited references. A complete copy of the Confirmation is appended to this response.

CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at (925) 944-3320.

Respectfully submitted,

/Charles E. Krueger/

Charles E. Krueger
Reg. No. 30,077

LAW OFFICE OF CHARLES E. KRUEGER
P.O. Box 5607
Walnut Creek, CA 94596
Tel: (925) 944-3320 / Fax: (925) 944-3363

985 PH Ex. 12

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 07/01/2008
LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	

ART UNIT	PAPER NUMBER
2154	

MAIL DATE	DELIVERY MODE
07/01/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES DEPARTMENT OF COMMERCE
U.S. Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450

APPLICATION NO./ CONTROL NO.	FILING DATE	FIRST NAMED INVENTOR / PATENT IN REEXAMINATION	ATTORNEY DOCKET NO.
10217955	8/9/02	DOYLE ET AL.	006-1-4

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
Larry D. Donaghue	
ART UNIT	PAPER
2154	20080409

DATE MAILED:

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner for Patents

See attached.

The outcome of reexamination number 90/007,858 has a material bearing on the patentability of the claims in this application. Ex parte prosecution is SUSPENDED FOR A PERIOD OF 6 MONTHS from the date of this letter. Upon expiration of the period of suspension or termination of the re examination, applicant should make an inquiry as to the status of the application.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Larry D. Donaghue whose telephone number is 571-272-3962. The examiner can normally be reached on M-F 8:00-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Nathan Flynn can be reached on 571-272-1915. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Larry D Donaghue/
Primary Examiner, Art Unit 2154

[illegible]
Andrew Hirshfeld
Director, TC2100

985 PH Ex. 13

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Doyle et al. Application No.: 10/217,955 Filed: 08/09/2002 For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT	Examiner: DONAGHUE, LARRY D Art Unit: 2154 Request for Removal of Suspension

Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

Prosecution of the referenced application was suspended by the Patent Office for 6 months on 07/01/2008 for the reason that "the outcome of reexamination number 90/007,858 has a material bearing on the patentability of the claims in this application". It is stated that upon expiration of the period of suspension or termination of the reexamination, applicant should make an inquiry as to the status of the application.

Attached hereto is the Notice of Intent to Issue Ex Parte Reexamination Certificate (NIRC) was mailed 09/10/2008 which terminates reexamination 90/007,858.

It is respectfully requested that the suspension of prosecution be removed and that prosecution of the application is continued at the earliest possible date.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at (925) 944-3320.

Respectfully submitted,

/Charles E. Krueger/

Charles E. Krueger
Reg. No. 30,077

LAW OFFICE OF CHARLES E. KRUEGER
P.O. Box 5607
Walnut Creek, CA 94596
Tel: (925) 944-3320 / Fax: (925) 944-3363

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
90/007,858	12/22/2005	5838906	6620-66570-01	4371

30080 7590 09/10/2008

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER

ART UNIT	PAPER NUMBER

DATE MAILED: 09/10/2008

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

DO NOT USE IN PALM PRINTER

(THIRD PARTY REQUESTER'S CORRESPONDENCE ADDRESS)

STEPHEN A. WRIGHT
KLARQUIST SPARKMAN LLP
121 SW SALMON STREET, SUITE 1600
PORTLAND, OR 97204

EX PARTE REEXAMINATION COMMUNICATION TRANSMITTAL FORM

REEXAMINATION CONTROL NO. 90/007,858.

PATENT NO. 5838906.

ART UNIT 3992.

Enclosed is a copy of the latest communication from the United States Patent and Trademark Office in the above identified *ex parte* reexamination proceeding (37 CFR 1.550(f)).

Where this copy is supplied after the reply by requester, 37 CFR 1.535, or the time for filing a reply has passed, no submission on behalf of the *ex parte* reexamination requester will be acknowledged or considered (37 CFR 1.550(g)).

PTOL-465 (Rev.07-04)

Notice of Intent to Issue Ex Parte Reexamination Certificate	Control No.	Patent Under Reexamination
	90/007,858	5838906
	Examiner	Art Unit
	JOSEPH R. POKRZYWA	3992

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

1. ☒ Prosecution on the merits is (or remains) closed in this *ex parte* reexamination proceeding. This proceeding is subject to reopening at the initiative of the Office or upon petition. *Cf.* 37 CFR 1.313(a). A Certificate will be issued in view of
   (a) ☒ Patent owner's communication(s) filed: *23 June 2008*.
   (b) ☐ Patent owner's late response filed: _____.
   (c) ☐ Patent owner's failure to file an appropriate response to the Office action mailed: _____.
   (d) ☐ Patent owner's failure to timely file an Appeal Brief (37 CFR 41.31).
   (e) ☐ Other: _____.

   Status of *Ex Parte* Reexamination:
   (f) Change in the Specification: ☐ Yes ☒ No
   (g) Change in the Drawing(s): ☐ Yes ☒ No
   (h) Status of the Claim(s):
   (1) Patent claim(s) confirmed: _____.
   (2) Patent claim(s) amended (including dependent on amended claim(s)): *1-10*
   (3) Patent claim(s) cancelled: _____.
   (4) Newly presented claim(s) patentable: 11-14.
   (5) Newly presented cancelled claims: _____.

2. ☒ Note the attached statement of reasons for patentability and/or confirmation. Any comments considered necessary by patent owner regarding reasons for patentability and/or confirmation must be submitted promptly to avoid processing delays. Such submission(s) should be labeled: "Comments On Statement of Reasons for Patentability and/or Confirmation."

3. ☐ Note attached NOTICE OF REFERENCES CITED (PTO-892).

4. ☐ Note attached LIST OF REFERENCES CITED (PTO/SB/08).

5. ☐ The drawing correction request filed on _____ is: ☐ approved ☐ disapproved.

6. ☐ Acknowledgment is made of the priority claim under 35 U.S.C. § 119(a)-(d) or (f).
   a)☐ All b)☐ Some* c)☐ None of the certified copies have
   ☐ been received.
   ☐ not been received.
   ☐ been filed in Application No. _____.
   ☐ been filed in reexamination Control No. _____.
   ☐ been received by the International Bureau in PCT Application No. _____.

   * Certified copies not received: _____.

7. ☐ Note attached Examiner's Amendment.

8. ☐ Note attached Interview Summary (PTO-474).

9. ☐ Other: _____.

cc: Requester (if third party requester)

DETAILED ACTION

Response to Amendment

1. Patent Owner's amendment was received on 6/23/08, and has been entered and made of record. The examiner notes that claims 1-10 originally issued in U.S. Patent Number 5,838,906 (hereafter "the '906 Patent"). With the current amendment dated 6/23/08, claims 1, 4, 5, 6, 9, and 10 were amended and claims 11-14 were newly added. Thus, currently, claims 1-14 are pending, and are the subject of the current reexamination proceeding.

Brief Summary of the Instant Proceedings

2. Within the current reexamination proceeding, an Office action dated 7/30/07 rejected claims 1-10 with the references of "A Brief Overview of the VIOLA Engine, and it's applications", written by Pei Wei, noted as "Viola", and rejected claims 1-3 and 6-8 with the reference of Cohen *et al.* (U.S. Patent Number 5,367,621, noted as "Cohen"), when viewed with "Introducing NCSA Mosaic" (noted as "NCSA Mosaic").

3. Subsequently, the Patent Owner submitted a Declaration under 37 CFR 1.131 on 10/1/07, which establishes the invention prior to August 16, 1994, being the date utilized as the publication date of the previously cited Viola reference. With that, in the Office action dated 4/18/08, the examiner withdrew the rejection of claims 1-10 as being anticipated by the Viola reference, but maintained the rejection of claims 1-3 and 6-8 as being unpatentable over Cohen in view of NCSA Mosaic. Finally, the Patent Owner submitted the current amendment dated 6/23/08, which amends claims 1 and 6, and places the noted patentable claims 4, 5, 9, and 10 in independent form.

STATEMENT OF REASONS FOR PATENTABILITY AND/OR CONFIRMATION

The following is an examiner's statement of reasons for patentability and/or confirmation of the claims found patentable in this reexamination proceeding:

Claims 1-14 are deemed as patentable, as amended.

With the amendment dated 6/23/08, ***claims 1, 4, 5, 6, 9, and 10*** are independent.

With respect to independent ***claims 1 and 6***, in the examiner's opinion, based on the prior art of record, it would not have been obvious to have the system, as claimed, include the features of an embed text format being parsed by the browser to automatically invoke the executable application to execute on the client workstation in order to display the object and enable an end-user to directly interact with the object within a display area created at the first location within the portion of the hypermedia document within the browser controlled window. The examiner notes that the closest prior art, Cohen (U.S. Patent Number 5,367,621), utilizes the IBM BookManager system, whereby Cohen teaches of the AUTOLAUNCH function, which automatically launches an object, whereby the system can automatically invoke multimedia objects, such as "photographic quality graphics, motion video, or sound", as read in col. 2, lines 50-66.

However, Cohen does not specifically disclose the feature of allowing an end-user to directly interact with the object within the display area of the browser window after the object is automatically invoked. Cohen shows that the graphic 190', as seen in Fig. 4b is automatically invoked. However, there is no indication that an end-user can directly interact with this graphic. Further, the specification of Cohen discusses inserting an audio object "eleph_sound.Audio 1 – Elephant's trumpet" and a movie object "eleph_movie.Motion Picture of African Elephant family", as seen in Fig. 1b. But these examples are not automatically invoked using the AUTOLAUNCH function, and if they would be set to AUTOLAUNCH, there is no indication in Cohen that would provide the function allowing the end-user to directly interact with the automatically invoked object.

As noted in the specification of the '906 Patent in col. 7, lines 12-15 "Also, the user is able to rotate, scale and otherwise reposition the viewpoint with respect to these images without exiting the hypermedia browser software." There is no indication in Cohen that the BookManager READ program allows the end-user to perform this direct interaction of the object once the multimedia is launched automatically. Further, the examiner can find no other teaching in the prior art of record that would motivate one of ordinary skill in the art to modify the Cohen teachings so as to allow the end-user to directly interact with the automatically invoked object. Therefore, because of this feature that was added in the amendment dated 6/23/08, the invention defined in claims 1 and 6 is rendered as patentable.

With respect to independent ***claims 4, 5, 9, and 10***, the examiner believes that it would not have been obvious to one of ordinary skill in the art at the time of the invention to have the method and computer program product, as claimed, further include the features of issuing one or more commands to the network server from the client workstation, executing the one or more instructions on the network server, and sending the information from the network server to the client workstation in response to the executed instructions, and processing the information at the client workstation to interactively control the application.

As discussed above, the prior art of Cohen can be interpreted as teaching of a system that includes an embed text format that specifies a location of at least a portion of the object external to a hypermedia document, which is further utilized to identify and locate an executable application that is external to the hypermedia document. However, Cohen does not explicitly teach if the external application is located at a server, whereby the instructions would be executed at the server, with the client workstation and server performing the process defined in claims 4 and 9, respectively. Further, the examiner can find no other teaching that would motivate one of ordinary skill in the art to modify the Cohen teachings so perform these features. Therefore, because of these features, the invention defined in now independent claims 4, 5, 9, and 10 is rendered as patentable.

Any comments considered necessary by PATENT OWNER regarding the above statement must be submitted promptly to avoid processing delays. Such submission by the patent owner should be labeled: "Comments on Statement of Reasons for Patentability and/or Confirmation" and will be placed in the reexamination file.

Conclusion

4. ALL correspondence relating to this ex parte reexamination proceeding should be directed as follows:

Please mail any communications to:

Attn: Mail Stop "Ex Parte Reexam"
Central Reexamination Unit
Commissioner for Patents
P. O. Box 1450
Alexandria VA 22313-1450

Please FAX any communications to:

(571) 273-9900
Central Reexamination Unit

Please hand-deliver any communications to:

Customer Service Window
Attn: Central Reexamination Unit
Randolph Building, Lobby Level
401 Dulany Street
Alexandria, VA 22314

Any inquiry concerning this communication or earlier communications from the Reexamination Legal Advisor or Examiner, or as to the status of this proceeding, should be directed to the Central Reexamination Unit at telephone number (571) 272-7705.

Signed:

[illegible]

Joseph R. Pokrzywa
Primary Patent Examiner
Central Reexamination Unit 3992
(571) 272-7410

Conferees : ESK
RGF

985 PH Ex. 14

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Doyle et al.

Application No.: 10/217,955

Filed: 08/09/2002

For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT

Examiner: DONAGHUE, LARRY D

Art Unit: 2154

SUPPLEMENTAL AMENDMENT

Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

In response to the Office Action mailed 09/09/2004, please amend the application as follows:

Amendments to the Claims begin on page 2 of this paper.

Remarks/Conclusion begins on page 17 of this paper.

AMENDMENTS TO THE CLAIMS:

This listing of the claims replaces all prior versions, and listings, of claims in the application.

LISTING OF CLAIMS:

1-3 (canceled)

4. (currently amended) A method for running an application program in a distributed hypermedia network environment, wherein the network environment comprises at least one client workstation and one network server coupled to the network environment, the method comprising:

receiving, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

executing the browser application on the client workstation, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of at least a portion of an object external to the file, where the object has type information associated with it;

utilizing the type information to identify and locate an executable application external to the file; and

automatically invoking the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable an end-user to directly interact with ~~interactive processing of~~ the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

5. (previously presented) The method of claim 4 where:

the information to enable comprises text formats.

6. (previously presented) The method of claim 5 where the text formats are HTML tags.

7. (previously presented) The method of claim 4 where the information contained in the file received comprises at least one embed text format.

8. (previously presented) The method of claim 4 where the step of identifying an embed text format comprises:

parsing the received file to identify text formats included in the received file.

9. (previously presented) The method of claim 8 where the parsing is by a parser in the browser.

10. (previously presented) The method of claim 4 where the processing specified by the text formats is specified directly.

11. (previously presented) The method of claim 4 where the correspondence is implied by the order of the text format in a set of all of the text formats.

12. (previously presented) The method of claim 4 where the embed text format specifies the location of at least a portion of an object directly.

13. (previously presented) The method of claim 4 where having type information associated is by including type information in the embed text format.

14. (previously presented) The method of claim 4 where automatically invoking does not require interactive action by the user.

15. (previously presented) The method of claim 4, wherein said executable application is a controllable application and further comprising the step of:

interactively controlling said controllable application on said client workstation via inter-process communications between said browser and said controllable application

16. (previously presented) The method of claim 15, wherein the communications to interactively control said controllable application continue to be exchanged between the controllable application and the browser even after the controllable application program has been launched.

17. (previously presented) The method of claim 16, wherein additional instructions for controlling said controllable application reside on said network server, wherein said step of interactively controlling said controllable application includes the following substeps:

issuing, from the client workstation, one or more commands to the network server;

executing, on the network server, one or more instructions in response to said commands;

sending information from said network server to said client workstation in response to said executed instructions; and processing said information at the client workstation to interactively control said controllable application.

18. (previously presented) The method of claim 17, wherein said additional instructions for controlling said controllable application reside on said client workstation.

19. (currently amended) One or more computer readable media encoded with software comprising computer executable instructions, for use in a distributed hypermedia network environment, wherein the network environment comprises at least one client workstation and one network server coupled to the network environment, and when the software is executed operable to:

receive, at the client workstation from the network server over the network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

cause the client workstation to utilize the browser to:

respond to text formats to initiate processing specified by the text formats;

display at least a portion of the document within the browser-controlled window;

identify an embed text format corresponding to a first location in the document, the embed text format specifying the location of at least a portion of an object external to the file, with the object having type information associated with it;

utilize the type information to identify and locate an executable application external to the file; and

automatically invoke the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable an end-user to directly interact with ~~interactive processing of~~ the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

20. (previously presented) The computer readable media of claim 19 where: the information to enable comprises text formats.

21. (previously presented) The computer readable media of claim 20 where: the text formats are HTML tags.

22. (previously presented) The computer readable media of claim 19 where: the information contained in the file received comprises at least one embed text format.

23. (currently amended) A method of serving digital information in a computer network environment having a network server coupled the network environment, and where the network environment is a distributed hypermedia environment, the method comprising:

communicating via the network server with at least one client workstation over said network in order to cause said client workstation to:

receive, over said network environment from said server, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

execute ~~invoke~~, at said client workstation, a browser application, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying, on said client workstation, at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of at least a portion of an object external to the file, where the object has type information associated with it;

utilizing the type information to identify and locate an executable application external to the file; and

automatically invoking the executable application, in response to the identifying of the embed text format, to execute on the client workstation in order to display the object and enable an end-user to directly interact with ~~interactive processing of~~ the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.

24. (previously presented) The method of claim 23 where:
the information to enable comprises text formats.

25. (previously presented) The method of claim 24 where:
the text formats are HTML tags.

26. (currently amended) The method of claim 23 ~~24~~ where:
the information contained in the file received comprises at least one embed text format.

27. (currently amended) A method for running an executable application ~~program~~ in a computer network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, ~~wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file, wherein the object has type information associated with it,; wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format,~~ the method comprising:

enabling an end-user to directly interact with an object by utilizing said executable application ~~external to said file~~ to interactively process said object while the object is being displayed within a display area created at a ~~the~~ first location within a ~~the~~ portion of a ~~the~~ hypermedia document being displayed in a ~~the~~ browser-controlled window, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable said browser application to display, on said client workstation, at least said portion of said distributed hypermedia document within said browser-controlled window, wherein said executable application is external to said file, wherein said client workstation executes the browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least said portion of the document is displayed within the browser-controlled window, wherein an embed text format which corresponds to said first location in the document is identified by the browser, wherein the embed text format specifies the location of at least a portion of said object external to the file, wherein the object has type information associated with it, wherein the type information is utilized by the browser to identify and locate said executable application, and wherein the executable application is automatically invoked by the browser, in response to the identifying of the embed text format.

28. (previously presented) The method of claim 27 where:
the information to enable comprises text formats.

29. (previously presented) The method of claim 28 where:
the text formats are HTML tags.

30. (previously presented) The method of claim 27 where:
the information contained in the file received comprises at least one embed text format.

31. (currently amended) One or more computer readable media encoded with software comprising an executable application ~~computer executable instructions~~ for use in a system having at least one client workstation and one network server coupled to a network environment, ~~wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file, wherein the object has type information associated with it, wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format, with software encoded on said computer readable media, identified by said type information and when automatically invoked,~~ operable to:

cause the client workstation to display an ~~said~~ object and enable an end-user to directly interact with ~~interactive processing of~~ said object while the object is being displayed within a display area created at a ~~the~~ first location within a ~~the~~ portion of a ~~the~~ hypermedia document being displayed in a ~~the~~ browser-controlled window, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable said browser application to display, on said client workstation, at least said portion of said distributed hypermedia document within said browser-controlled window, wherein said executable application is external to said file, wherein said client workstation executes said browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least said portion of the document is displayed within the browser-controlled window, wherein an embed text format which corresponds to said first location in the document is identified by the browser, wherein the embed text format specifies the location of at least a portion of said object external to the file, wherein the object has type information associated with it, wherein the type information is utilized by the browser to identify and locate said executable application, and wherein the executable application is automatically invoked by the browser, in response to the identifying of the embed text format.

32. (previously presented) The method of claim 31 where:
the information to enable comprises text formats.

33. (previously presented) The method of claim 32 where:
the text formats are HTML tags.

34. (previously presented) The method of claim 31 where:
the information contained in the file received comprises at least one embed text format.

35. (currently amended) A method for serving digital information in a computer network environment, ~~with a network server coupled to said network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, wherein said network environment is a distributed hypermedia environment, wherein said client workstation receives, over said network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed~~

~~hypermedia document within a browser-controlled window, wherein said client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds to a first location in the document is identified, wherein the embed text format specifies the location of at least a portion of an object external to the file, wherein the object has type information associated with it, wherein the type information is utilized to identify and locate an executable application external to the file, and wherein the executable application is automatically invoked, in response to the identifying of the embed text format;~~ said method comprising:

communicating via a ~~said~~ network server with at least one client workstation over said computer network environment in order to cause said client workstation to:

receive at said client workstation, over said computer network environment from said server, at least one file containing information to enable a browser application to display, on said client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window;

utilize an ~~said~~ executable application external to said file to enable an end-user to directly interact with ~~interactive processing of~~ ~~the~~ an object while the object is being displayed within a display area created at a ~~the~~ first location within the portion of the distributed hypermedia document being displayed in the browser-controlled window, with said network server coupled to said computer network environment, wherein said computer network environment has at least said client workstation and said network server coupled to the computer network environment, wherein said computer network environment is a distributed hypermedia environment, wherein said client workstation executes the browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least said portion of the document is displayed within the browser-controlled window, wherein an embed text format which corresponds to said first location in the document is identified by the browser, wherein the embed text format specifies the location of at least a portion of said object external to the file, wherein the object has type information associated with it, wherein the type information is utilized by the browser to identify and locate said executable application, and wherein the executable application is automatically invoked by the browser, in response to the identifying of the embed text format.

36. (previously presented) The method of claim 35 where:
the information to enable comprises text formats.

37. (previously presented) The method of claim 36 where:
the text formats are HTML tags.

38. (previously presented) The method of claim 35 where:
the information contained in the file received comprises at least one embed text format.

39. (currently amended) A method for running an application program in a distributed hypermedia network environment, wherein the distributed hypermedia network environment comprises at least one client workstation and one remote network server coupled to the distributed hypermedia network environment, the method comprising:

receiving, at the client workstation from the network server over the distributed hypermedia network environment, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

executing the browser application on the client workstation, with the browser application:

responding to text formats to initiate processing specified by the text formats;

displaying at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of [[an]] at least a portion of an object;

identifying and locating an executable application ~~program code~~ associated with the object; and

automatically invoking the executable application ~~program code~~, in response to the identifying of the embed text format, ~~to execute on the client workstation~~ in order to ~~display the object and~~ enable an end-user to directly interact with ~~interactive processing of~~ the object, while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the executable application ~~program code~~ is part of a distributed application, and wherein at least a portion of the distributed application is for execution on a remote network server coupled to the distributed hypermedia network environment.

40. (previously presented) The method of claim 39 where:
the information to enable comprises text formats.

41. (previously presented) The method of claim 40 where:
the text formats are HTML tags.

42. (previously presented) The method of claim 39 where:
the information contained in the file received comprises at least one embed text format.

43. (currently amended) A method of serving digital information in a computer network environment having a network server coupled to said computer network environment, and where the network environment is a distributed hypermedia network environment, the method comprising:

communicating via the network server with at least one remote client workstation over said computer network environment in order to cause said client workstation to:

receive, over said computer network environment from the network server, at least one file containing information to enable a browser application to display at least a portion of a distributed hypermedia document within a browser-controlled window;

execute ~~invoke~~, at said client workstation, a browser application, with the browser application~~, with the browser application~~:

responding to text formats to initiate processing specified by the text formats;

displaying, on said client workstation, at least a portion of the document within the browser-controlled window;

identifying an embed text format which corresponds to a first location in the document, where the embed text format specifies the location of [[an]] at least a portion of an object ;

identifying and locating an executable application ~~program code~~ associated with the object; and

automatically invoking the executable application ~~program code~~, in response to the identifying of the embed text format, ~~to execute on the client workstation~~ in order to ~~display the object and~~ enable an end-user to directly interact with ~~interactive processing of~~ the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the executable application ~~program code~~ is part of a distributed application, and wherein at least a portion of the distributed application is for execution on the network server.

44. (previously presented) The method of claim 43 where:
the information to enable comprises text formats.

45. (previously presented) The method of claim 44 where:
the text formats are HTML tags.

46. (previously presented) The method of claim 43 where:
the information contained in the file received comprises at least one embed text format.

47. (currently amended) A method for serving digital information in a computer network environment, ~~with a network server coupled to said network environment, wherein said network environment has at least one client workstation and one network server coupled to a network environment, wherein the network environment is a distributed hypermedia environment, wherein the client workstation receives, over the network environment from the server, at least one file containing information to enable a browser application to display, on the client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein the client workstation executes a browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least a portion of the document is displayed within the browser-controlled window, wherein an embed text format corresponds~~

~~to a first location in the document is identified, wherein the embed text format specifies the location of an object, wherein program code associated with the object is identified and located, wherein the executable application is automatically invoked, in response to the identifying of the embed text format, to enable the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the program code is part of a distributed application, and wherein at least a portion of the distributed application is for execution on the network server;~~ said method comprising:

communicating via a ~~the~~ network server with at least a ~~the~~ remote client workstation over the computer network environment in order to receive commands from the client workstation, with the network server coupled to said computer network environment, wherein said computer network environment has at least said client workstation and said network server coupled to the computer network environment, wherein the computer network environment is a distributed hypermedia environment, wherein the client workstation receives, over the computer network environment from the server, at least one file containing information to enable a browser application to display, on the client workstation, at least a portion of a distributed hypermedia document within a browser-controlled window, wherein the client workstation executes the browser application, with the browser application responding to text formats to initiate processing specified by the text formats, wherein at least said portion of the document is displayed within the browser-controlled window, wherein an embed text format which corresponds to a first location in the document is identified by the browser, wherein the embed text format specifies the location of at least a portion of an object, wherein an executable application associated with the object is identified and located by the browser, wherein the executable application is automatically invoked by the browser, in response to the identifying of the embed text format, to enable an end-user to directly interact with the object while the object is being displayed within a display area created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window, wherein the executable application is part of a distributed application, and wherein at least a portion of the distributed application is for execution on the network server;

executing one or more instructions in response to the commands;

sending information to the client workstation in response to the executed instructions, to allow processing of the information at the client workstation to enable said end-user to directly interact with said object ~~for interactively controlling the controllable application~~.

48. (previously presented) The method of claim 47 where: the information to enable comprises text formats.

49. (previously presented) The method of claim 48 where: the text formats are HTML tags.

50. (previously presented) The method of claim 47 where:

the information contained in the file received comprises at least one embed text format.

REMARKS

Claims 4-50 are pending. Claims 1, 19, 23, 26, 27, 31, 35, 39, 43 and 47 are amended herein. Reexamination and reconsideration of all outstanding rejections and objections is requested.

PROCEDURAL HISTORY

This application is a continuation of and claims the benefit of U.S. Application No. 09/075,359, filed May 8, 1998, which is a continuation of U.S. Application No. 08/324,443, filed October 17, 1994, which issued as U.S. Patent No. 5,838,906 ("the '906 patent).

There have been two reexaminations of the '906 patent. The first reexamination was a Director Ordered Reexamination, Control No. 90/006,831 ("the first reexamination"), which resulted in issuance of a Reexamination Certificate on 5/17/2006 without amending the claims. Shortly after the NIRC for the first reexamination was posted on the PAIR page, the second reexamination, Control No. 90/007,838 ("the second reexamination"), was requested.

The office action relating to the currently-pending application was mailed on 07/20/2004, and non-finally rejected claims 1-3. This rejection is identical to the rejection then pending in the first reexamination. A response to the first office action was filed 03/11/2005 and canceled claim 2. A first supplemental amendment was filed 4/11/2008, which presented new claims 4-50. This paper is a second supplemental amendment which presents new amendments to certain of the claims listed above and provides representative citations to support in the specification for the elements and limitations recited in the claims, as requested by the examiner.

Subsequent to the filing of the response, prosecution of the application was suspended by the patent office. Letters of suspension were mailed 05/05/2005, 01/18/2006, 10/18/2006 and 08/13/2006.

The first letter of suspension stated that the outcome of the first reexamination had a material bearing on the patentability of the claims in the present application. The first reexamination resolved all issues of patentablilty in favor of the patentee.

The subsequent letters of suspension stated that the outcome of the second reexamination had a material bearing on the patentability of the claims in the present application. The second reexamination resolved all issues of patentablilty in favor of the patentee.

INTERVIEW SUMMARY

A personal interview was conducted on January 8, 2009. Present at the interview were Examiner Donaghue, inventor Michael Doyle, and Charles E. Krueger, the attorney of record.

The prior art discussed was: 1) the five-way combination of Mosaic, Berners-Lee, Raggett I and II and Toye; and 2) Viola.

The examiner requested that citations to support in the specification for the elements and limitations of the pending claims be provided in the remarks section of a newly presented supplemental amendment.

CITATIONS TO THE SPECIFICATION OF REPRESENTATIVE EXAMPLES OF SUPPORT OF ALL ELEMENTS AND LIMITATIONS RECITED IN THE PENDING CLAIMS

As requested by the examiner, this section cites representative examples of support in the specification for the elements and limitations in the pending claims.

For ease of reference, citations in bold are to column and line numbers of U.S. Patent 5,838,906, which has a specification identical to the pending application and which is the grand-parent of the pending application. In the following, claim language is in italics.

The following citations are representative examples of support in the specification for each element and limitation recited in the claims. Many other parts of the specification, not specifically cited, further support the recitations of the claims and there are other examples that could be cited.

The representative citations are taken from the description of several example embodiments and are not intended to limit the invention, which is defined by the claims.

CLAIM 4. *A method for running an application program in a* ***distributed hypermedia***

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

network environment,

EXAMPLE SUPPORT:

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

wherein the network environment comprises at least one ***client workstation*** *and one* ***network server*** *coupled to the network environment,*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

the method comprising:

receiving, at the client workstation from the network server over the network environment, ***at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** a browser application to **display** at least a portion of a **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*within a **browser-controlled window;***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

***executing the browser application** on the client workstation,*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

with the browser application:

***responding to text formats** to initiate processing specified by the text formats;*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

***displaying at least a portion** of the document within the browser-controlled window;*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

***identifying an embed text format** which corresponds to a first location in the document,*

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

*where the embed text format **specifies the location of at least a portion of an object external** to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag. "

*where the object has **type information associated** with it;*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

utilizing the type information to ***identify and locate an executable application*** *external to the file; and*

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

automatically invoking the executable application, in response to the identifying *of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

to ***execute*** *on the client workstation*

EXAMPLE SUPPORT:

9:43 "and application client 210 executes instructions to perform processing in accordance with the present invention. "

in order to display the object and ***enable an end-user to directly interact*** *with the object*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created* ***at the first location*** *within the portion of the hypermedia document being displayed in the browser-controlled window.*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

CLAIM 5. (previously presented) *The method of claim 4 where:*
the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

CLAIM 6. *The method of claim 5 where the text formats are* ***HTML tags****.*

EXAMPLE SUPPORT:

14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>CLAIM 7</u>. *The method of claim 4 where the information contained in the file received comprises **at least one embed text format**.*

EXAMPLE SUPPORT:

14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

<u>CLAIM 8</u>. *The method of claim 4 where the step of identifying an embed text format comprises:*

***parsing** the received file to identify text formats included in the received file.*

EXAMPLE SUPPORT:

14:15 "a browser program executing on the client computer displays the document and calls a first routine in the HTMLparse.c file called "HTMLparse". This first routine, HTMLparse, is entered at step 252 where a pointer to the start of the document portion is passed. Steps 254, 256 and 258 represent a loop where the document is parsed or scanned for HTML tags or other symbols."

<u>CLAIM 9</u>. *The method of claim 8 where the parsing is by a parser **in the browser**.*

EXAMPLE SUPPORT:

14:16 "browser program executing on the client computer displays the document and calls a first routine in the HTMLparse.c file called "HTMLparse""

<u>CLAIM 10</u>. *The method of claim 4 where the processing specified by the text formats is **specified directly**.*

EXAMPLE SUPPORT:

9:30 "Embedded program link 214 identifies application client 212 as an application to invoke. In this present example, the application, namely, application client 210, resides on the same computer as the browser client 208 that the user is executing to view the hypermedia document. Embedded program link 214 may include additional information, such as parameters, that tell application client 210 how to proceed."

<u>CLAIM 11</u>. *The method of claim 4 where the correspondence is implied by the order of the text format in a set of all of the text formats.*

EXAMPLE SUPPORT:

14:53 "At step 272 the parameters of the structure are initialized in preparation for inserting a DrawingArea widget on an HTML page. This includes providing values for the width and height of a window on the display to contain an image, position of the window, style, URL of the image object, etc. Various codes are also added by routines in HTMLformat.c (such as TriggerMarkChanges) to insert an internal representation of the HTML statement into an object list maintained internally by the browser."

<u>CLAIM 12</u>. *The method of claim 4 where the embed text format specifies the location of at least a portion of an object* ***directly****.*

EXAMPLE SUPPORT:

14:66 "the data object specified by the URL in the EMBED tag. "

<u>CLAIM 13</u>. *The method of claim 4 where having type information associated is by including type information* ***in the embed text format****.*

EXAMPLE SUPPORT:

12:66 "As shown in Table II, the EMBED tag includes TYPE, HREF, WIDTH and HEIGHT elements."

<u>CLAIM 14</u>. *The method of claim 4 where automatically invoking does not require interactive action by the user.*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210

<u>CLAIM 15</u>. *The method of claim 4, wherein said executable application is a controllable application and further comprising the step of:*

interactively controlling said controllable application on said client workstation ***via inter-process communications*** *between said browser and said controllable application*

EXAMPLE SUPPORT:

10:10 "In order to make application client 210 integral with displays created by browser client 208, both the browser client and the application client must be in communication with each other, as shown by the arrow connecting the two within client computer 200. The manner of communication is through an application program interface (API), discussed below. "

<u>CLAIM 16</u>. *The method of claim 15, wherein the communications to interactively control said controllable application* ***continue to be exchanged*** *between the controllable application and the browser even after the controllable application program has been launched.*

EXAMPLE SUPPORT:

10:12 "both the browser client and the application client must be in communication with each other"

<u>CLAIM 17</u>. *The method of claim 16, wherein additional instructions for controlling said controllable application* ***reside on said network server,***

EXAMPLE SUPPORT:

10:33 "Another embodiment of the present invention uses an application server process executing on server computer 204 to assist in processing that may need to be performed by an external program."

wherein said step of interactively controlling said controllable application includes the following substeps:

issuing, from the client workstation, ***one or more commands*** *to the network server;*

EXAMPLE SUPPORT:

10:56 "This information is received by application client 210 and processed to generate a command sent over network 206 to application server 220."

***executing, on the network server, one or more instructions** in response to said commands;*

EXAMPLE SUPPORT:

10:59 "Once application server 220 receives the information in the form of, e.g., a coordinate transformation for a new viewing position, application server 220 performs the mathematical calculations to compute a new view for the embryo image."

***sending information from said network server to said client workstation** in response to said executed instructions;*

EXAMPLE SUPPORT:

10:63 "Once the new view has been computed, the image data for the new view is sent over network 206 to application client 210 so that application client 210 can update the viewing window currently displaying the embryo image."

*and processing said information at the client workstation **to interactively control** said controllable application.*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data."

10:52 "In a preferred embodiment, application client 210 receives signals from a user input device at the user's client computer 200. An example of such input would be to rotate the embryo image from a current position to a new position from the user's point of view."

<u>**CLAIM 18**</u>. *The method of claim 17, wherein said additional instructions for controlling said controllable application reside on said client workstation.*

EXAMPLE SUPPORT:

'906 Claim 5 "The method of claim 4, wherein said additional instructions for controlling said controllable application reside on said client workstation."

<u>**CLAIM 19**</u>. *One or more computer readable media encoded with software comprising computer executable instructions, for use in a* ***distributed hypermedia***

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

network environment, or

EXAMPLE SUPPORT:

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200."

wherein the network environment comprises at least one ***client workstation*** *and one* ***network server*** *coupled to the network environment,*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

and when the software is executed operable to:

receive, at the client workstation from the network server over the network environment, ***at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** a browser application to **display** at least a portion of a **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document.

When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*within a **browser-controlled window;***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

***cause the client workstation to utilize the browser** to:*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

respond to text formats *to initiate processing specified by the text formats;*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

display at least a portion *of the document within the browser-controlled window;*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

identify an embed text format *corresponding to a first location in the document,*

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

the embed text format ***specifying the location of at least a portion of an object external*** *to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag. "

with the object having ***type information associated*** *with it;*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

utilize the type information to ***identify and locate an executable application*** *external to the file; and*

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

automatically invoke the executable application, in response to the identifying *of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

*to **execute** on the client workstation*

EXAMPLE SUPPORT:

9:43 "and application client 210 executes instructions to perform processing in accordance with the present invention. "

*in order to display the object and **enable an end-user to directly interact**_with the object*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

***while the object is being displayed within a display area** created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window.*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

CLAIM 20. *The computer readable media of claim 19 where:*
*the information to enable comprises **text formats**.*
EXAMPLE SUPPORT:
14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

CLAIM 21. *The computer readable media of claim 20 where:*
*the text formats are **HTML tags**.*
EXAMPLE SUPPORT:
14:19 "the document is parsed or scanned for HTML tags or other symbols. "

CLAIM 22. *The computer readable media of claim 19 where:*
*the information contained in the file received comprises **at least one embed text format**.*
EXAMPLE SUPPORT:
14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

CLAIM 23. *A method of serving digital information in a computer network environment having a network server coupled the **network environment**,*
EXAMPLE SUPPORT:
9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200

and where the network environment is a ***distributed hypermedia environment****, the method comprising:*

EXAMPLE SUPPORT:

5:31 “Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet.”

9:48 “Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means.”

communicating via the network server with at least one client workstation *over said network*

EXAMPLE SUPPORT:

4:44 “In the first case, where computer 108 issues a request for information from server A, computer 108 is a "client" making a request of information from server A. Server A may have the information in a storage device that is local to Server A or server A may have to make requests of other computer systems to obtain the information. User 110 may also request information via their computer 108 from a server, such as server B located at a remote geographical location on the Internet.”

5:6 “Thus, in this example, computer 108 issues a command that includes the address of document 14. This command is routed through server A and Internet 100 and eventually is received by server B. Server B processes the command and locates document 14 on its local storage. Server 14 then transfers a copy of the document back to client 108 via Internet 100 and server A. After client computer 108 receives document 14, it is displayed so that user 110 may view it. ”

8:58 “In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206.”

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to."

in order to cause said client workstation to:
receive, over said network environment from said server,
EXAMPLE SUPPORT:
5:10 "Server 14 then transfers a copy of the document back to client 108 via Internet 100 and server A. After client computer 108 receives document 14, it is displayed so that user 110 may view it."
9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

at least one file
EXAMPLE SUPPORT:
2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

containing ***information to enable*** *a browser application to* ***display*** *at least a portion of a* ***distributed hypermedia*** *document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

within ***a browser-controlled window;***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

execute*, at said client workstation, a browser application, with the browser application:*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

responding to text formats *to initiate processing specified by the text formats;*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

***displaying, on said client workstation, at least a portion** of the document within the browser-controlled window;*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

***identifying an embed text format** which corresponds to a first location in the document,*

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

*where the embed text format **specifies the location of at least a portion of an object external** to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag. "

*where the object has **type information associated** with it;*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the

HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

utilizing the type information to ***identify and locate an executable application*** *external to the file; and*

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

automatically invoking the executable application, in response to the identifying *of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

to ***execute*** *on the client workstation*

EXAMPLE SUPPORT:

9:43 "and application client 210 executes instructions to perform processing in accordance with the present invention. "

in order to display the object and ***enable an end-user to directly interact with*** *the object*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at* ***the first location*** *within the portion of the hypermedia document being displayed in the browser-controlled window.*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

<u>**CLAIM 24**</u>. *The method of claim 23 where:*
the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

<u>**CLAIM 25**</u>. *The method of claim 24 where the text formats are* ***HTML tags****.*
EXAMPLE SUPPORT:
14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>**CLAIM 26**</u>. *The method of claim 23 where the information contained in the file received comprises* ***at least one embed text format****.*
EXAMPLE SUPPORT:
14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

<u>**CLAIM 27**</u>. *A method for running an executable application in a computer* ***network environment****,*
EXAMPLE SUPPORT:
9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

wherein said network environment has at least ***one client workstation and one network server*** *coupled to a network environment, the method comprising:*
EXAMPLE SUPPORT:
8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

enabling an end-user to directly interact with an object *by utilizing said executable application to interactively process said object*
EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at a* ***first location*** *within a portion of a hypermedia document*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

being displayed in a ***browser-controlled window,***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

*wherein said network environment is a **distributed hypermedia environment,***

EXAMPLE SUPPORT:

5:31 “Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet.”

9:48 “Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means.”

*wherein said client workstation receives, over said network environment from said server, **at least one file***

EXAMPLE SUPPORT:

2:14 “Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen.”

3:34 “As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** said browser application to **display**, on said client workstation, at least said portion of said **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*within said browser-controlled window, wherein said executable application is external to said file, wherein said **client workstation executes the browser application,***

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

with the browser application ***responding to text formats*** *to initiate processing specified by the text formats,*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

wherein at least ***said portion of the document is displayed*** *within the browser-controlled window,*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

wherein ***an embed text format which corresponds to said first location in the document is identified by the browser,***

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

wherein the embed text format ***specifies the location of at least a portion of said object external*** *to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."
14:67 "the data object specified by the URL in the EMBED tag. "

wherein the object has ***type information associated*** *with it,*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."
15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

wherein the type information is utilized by the browser to ***identify and locate said executable application,***

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

and wherein the executable application is ***automatically invoked by the browser, in response to the identifying*** *of the embed text format.*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"
15:11 "step 290 is executed to launch a predetermined application.

<u>CLAIM 28</u>. *The method of claim 27 where:*

the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

<u>CLAIM 29</u>. *The method of claim 28 where the text formats are* ***HTML tags****.*

EXAMPLE SUPPORT:

14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>CLAIM 30</u>. *The method of claim 27 where the information contained in the file received comprises* ***at least one embed text format****.*

EXAMPLE SUPPORT:

14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

<u>CLAIM 31.</u> *One or more computer readable media encoded with software comprising an executable application for use in a system having at least* ***one client workstation and one network server***

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

coupled to a ***network environment*** *operable to:*

EXAMPLE SUPPORT:

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

cause the client workstation to display an object and ***enable an end-user to directly interact with said object***

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at a* ***first location*** *within a portion of a hypermedia document*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

being displayed in a ***browser-controlled window,***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window

350. Another possibility is to have panel window 354 within Mosaic window 350."

wherein said network environment is a ***distributed hypermedia environment,***

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

wherein said client workstation receives, over said network environment from said server, ***at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** said browser application to **display**, on said client workstation, at least said portion of said **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

within said browser-controlled window, wherein said executable application is external to said file, wherein said ***client workstation executes the browser application,***

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

with the browser application ***responding to text formats*** *to initiate processing specified by the text formats,*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

wherein at least ***said portion of the document is displayed*** *within the browser-controlled window,*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

wherein ***an embed text format which corresponds to said first location in the document is identified by the browser,***

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

wherein the embed text format ***specifies the location of at least a portion of said object external*** *to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag. "

wherein the object has ***type information associated*** *with it,*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

wherein the type information is utilized by the browser to ***identify and locate said executable application,***

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

and wherein the executable application is ***automatically invoked by the browser, in response*** *to the identifying of the embed text format.*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"
15:11 "step 290 is executed to launch a predetermined application.

<u>**CLAIM 32**</u>. *The method of claim 31 where:*
*the information to enable comprises **text formats**.*
EXAMPLE SUPPORT:
14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

<u>**CLAIM 33**</u>. *The method of claim 32 where the text formats are **HTML tags**.*
EXAMPLE SUPPORT:
14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>**CLAIM 34**</u>. *The method of claim 31 where the information contained in the file received comprises **at least one embed text format**.*
EXAMPLE SUPPORT:
14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

<u>**CLAIM 35**</u>. *A method for serving digital information in a computer network environment, said method comprising:*

***communicating via a network server with at least one client workstation over said computer network environment** in order to cause said client workstation to:*
EXAMPLE SUPPORT:
4:44 "In the first case, where computer 108 issues a request for information from server A, computer 108 is a "client" making a request of information from server A. Server A may have the information in a storage device that is local to Server A or server A may have to make requests of other computer systems to obtain the information. User 110 may also request information via

their computer 108 from a server, such as server B located at a remote geographical location on the Internet."

5:6 "Thus, in this example, computer 108 issues a command that includes the address of document 14. This command is routed through server A and Internet 100 and eventually is received by server B. Server B processes the command and locates document 14 on its local storage. Server 14 then transfers a copy of the document back to client 108 via Internet 100 and server A. After client computer 108 receives document 14, it is displayed so that user 110 may view it. "

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206."

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to."

receive at said client workstations, over said computer network environment from said server, ***at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects.

Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** said browser application to **display**, on said client workstation, at least said portion a said **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

within a ***browser-controlled window,***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

utilize an executable application external to a file to ***enable an end-user to directly interact with an object***

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at a* ***first location*** *within a portion of the distributed hypermedia document*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9

shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

*being displayed in the **browser-controlled window**, with said network server coupled to said computer network environment,*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

*wherein said computer network environment has **at least said client workstation and said network server** coupled to the computer network environment,*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

*wherein said network environment is a **distributed hypermedia environment**,*

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

*wherein said executable application is external to said file, wherein said **client workstation executes the browser application**,*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*with the browser application **responding to text formats** to initiate processing specified by the text formats,*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*wherein at least **said portion of the document is displayed** within the browser-controlled window,*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document" *wherein* ***an embed text format which corresponds to said first location in the document is identified by the browser,***

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

wherein the embed text format ***specifies the location of at least a portion of said object external*** *to the file,*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag. "

wherein the object has ***type information associated*** *with it,*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

wherein the type information is utilized by the browser to ***identify and locate said executable application,***

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

and wherein the executable application is ***automatically invoked by the browser, in response*** *to the identifying of the embed text format.*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

<u>CLAIM 36</u>. *The method of claim 35 where:*

the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

<u>CLAIM 37</u>. *The method of claim 36 where:*

the text formats are ***HTML tags****.*

EXAMPLE SUPPORT:

14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>CLAIM 38</u>. *The method of claim 35 where:*

the information contained in the file received comprises ***at least one embed text format****.*

EXAMPLE SUPPORT:

14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

<u>CLAIM 39</u>. *A method for running an application program in a distributed hypermedia network environment, wherein the* ***distributed hypermedia***

EXAMPLE SUPPORT:

5:31 “Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet.”

9:48 “Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means.”

network environment

EXAMPLE SUPPORT:

9:48 “Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

comprises at least one client workstation and one ***remote network server***

EXAMPLE SUPPORT:

7:7 “In one application, high resolution three dimensional images are processed in a distributed manner by several computers located remotely from the user's client computer.”

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used. For ease of illustration, additional hardware and software layers are not shown in FIG. 5. "

10:61 “application server 220 performs the mathematical calculations to compute a new view for the embryo image. Once the new view has been computed, the image data for the new view is sent over network 206 to application client 210 so that application client 210 can update the viewing window currently displaying the embryo image.”

coupled to the distributed hypermedia network environment, the method comprising:

*receiving, at the client workstation from the network server over the distributed hypermedia network environment, **at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** a browser application to **display** at least a portion of a distributed hypermedia document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*within a **browser-controlled window**;*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

***executing the browser application** on the client workstation, with the browser application:*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

***responding to text formats** to initiate processing specified by the text formats;*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

***displaying at least a portion** of the document within the browser-controlled window;*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

***identifying an embed text format** which corresponds to a first location in the document,*

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

*where the embed text format **specifies the location of at least a portion** of an object;*

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."

14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."

14:67 "the data object specified by the URL in the EMBED tag.

identifying and locating an executable application *associated with the object; and*

EXAMPLE SUPPORT:

12:67 "The TYPE element is a Multipurpose Internet Mail Extensions (MIME) type. Examples of values for the TYPE element are "application/x-vis" or "video/mpeg". The type "application /x-vis" indicates that an application named "x-vis" is to be used to handle the object at the URL specified by the HREF. Other types are possible such as "application/x-inventor", "application/postscript" etc."

15:9 "At step 286 a check is made as to whether the type attribute of the object, i.e., the value for the TYPE element of the EMBED tag, is an application."

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

automatically invoking the executable application, in response to the identifying *of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

in order to ***enable an end-user to directly interact*** *with the object,*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at the* ***first location*** *within the portion of the hypermedia document being displayed in the browser-controlled window,*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

wherein the executable application is part of a ***distributed application,***

EXAMPLE SUPPORT:

10:33 "Another embodiment of the present invention uses an application server process executing on server computer 204 to assist in processing that may need to be performed by an external program. For example, in FIG. 5, application server 220 resides on server computer 204. Application server 220 works in communication with application client 210 residing on client computer 200."

11:18 "FIG. 6 shows yet another embodiment of the present invention. FIG. 6 is similar to FIG. 5, except that additional computers 222 and 224 are illustrated. Each additional computer includes a process labeled "Application (Distributed)." The distributed application performs a portion of the task that an application, such as application server 220 or application client 210, perform. In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206. Thus, several computers, such as server computer 204 and additional computers 222 and 224 can all work together"

and wherein at least a portion of the distributed application is for execution on a ***remote network server coupled to the distributed hypermedia network*** *environment.*

EXAMPLE SUPPORT:

11:24 "In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206."

__**CLAIM 40.**__. *The method of claim 39 where:*
the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

__**CLAIM 41**__. *The method of claim 40 where the text formats are* ***HTML tags****.*

EXAMPLE SUPPORT:

14:19 "the document is parsed or scanned for HTML tags or other symbols. "

__**CLAIM 42**__. *The method of claim 39 where the information contained in the file received comprises* ***at least one embed text format****.*

EXAMPLE SUPPORT:

14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

__CLAIM 43__. *A method of serving digital information in a* ***computer network environment***

EXAMPLE SUPPORT:

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

having ***a network server*** *coupled to said computer network environment,*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

and where the network environment is a ***distributed hypermedia*** *network environment, the method comprising:*

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

communicating via the network server with at least one ***remote client workstation*** *over said computer network environment in order to cause said client workstation to:*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used. For ease of illustration, additional hardware and software layers are not shown in FIG. 5. "

receive, over said computer network environment from the network server, ***at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** a browser application to **display** at least a portion of a distributed hypermedia document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

*within a **browser-controlled window;***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

execute, at said client workstation, a browser application*, with the browser application:*

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device.

responding to text formats *to initiate processing specified by the text formats;*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

displaying, on said client workstation, at least a portion of the document *within the browser-controlled window;*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

identifying an embed text format *which corresponds to a first location in the document,*

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

where the embed text format ***specifies the location of at least a portion of an object;***

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."
14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."
14:67 "the data object specified by the URL in the EMBED tag. "

identifying and locating an executable application *associated with the object; and*

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

automatically invoking the executable application, in response to the identifying *of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"
15:11 "step 290 is executed to launch a predetermined application.

in order to ***enable an end-user to directly interact*** *with the object*

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

while the object is being displayed within a display area *created at the first location within the portion of the hypermedia document being displayed in the browser-controlled window,*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

wherein the executable application is part of a ***distributed application****,*

EXAMPLE SUPPORT:

10:33 "Another embodiment of the present invention uses an application server process executing on server computer 204 to assist in processing that may need to be performed by an external program. For example, in FIG. 5, application server 220 resides on server computer 204. Application server 220 works in communication with application client 210 residing on client computer 200."

11:18 "FIG. 6 shows yet another embodiment of the present invention. FIG. 6 is similar to FIG. 5, except that additional computers 222 and 224 are illustrated. Each additional computer includes a process labeled "Application (Distributed)." The distributed application performs a portion of the task that an application, such as application server 220 or application client 210, perform. In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206. Thus, several computers, such as server computer 204 and additional computers 222 and 224 can all work together"

and wherein at least a portion of the distributed application is for execution on ***the network server****.*

EXAMPLE SUPPORT:

11:24 "In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206."

CLAIM 44. *The method of claim 43 where:*
the information to enable comprises ***text formats****.*

EXAMPLE SUPPORT:

14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

CLAIM 45. *The method of claim 44 where:*
the text formats are ***HTML tags****.*

EXAMPLE SUPPORT:

14:19 "the document is parsed or scanned for HTML tags or other symbols. "

CLAIM 46. *The method of claim 43 where the information contained in the file received comprises* ***at least one embed text format****.*

EXAMPLE SUPPORT:

14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

CLAIM 47. *A method for serving digital information in a* ***computer network environment****, said method comprising:*

EXAMPLE SUPPORT:

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200.

communicating via ***a network server*** *with at least* ***a remote client workstation*** *over the computer network environment*

EXAMPLE SUPPORT:

11:24 "In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206."

in order to ***receive commands*** *from the client workstation,*

EXAMPLE SUPPORT:

10:52 "In a preferred embodiment, application client 210 receives signals from a user input device at the user's client computer 200. An example of such input would be to rotate the embryo image from a current position to a new position from the user's point of view. This information is received by application client 210 and processed to generate a command sent over network 206 to application server 220.

with the network server coupled to said computer network environment, wherein said computer network environment has ***at least said client workstation*** *and said* ***network server*** *coupled to the computer network environment,*

EXAMPLE SUPPORT:

8:58 "In FIG. 5, client computer 200 communicates with server computer 204 via network 206. Both client computer 200 and server computer 204 use a network protocol layer to communicate with network 206. In a preferred embodiment, network 206 is the Internet and the network protocol layers are TCP/IP. Other networks and network protocols may be used."

*wherein the computer network environment is a **distributed hypermedia** environment,*

EXAMPLE SUPPORT:

5:31 "Further, it is a "distributed" system because data objects that are imbedded within a document may be located on many of the computer systems connected to the Internet."

9:48 "Note that application client 210 is in communication with network 206 via the network protocol layer of client computer 200. This means that application client 210 can make requests over network 206 for data objects, such as multidimensional image objects. For example, application client 210 may request an object, such as object 1 at 216, located in server computer 204. Application client 210 may make the request by any suitable means."

*wherein the client workstation receives, over the computer network environment from the server, **at least one file***

EXAMPLE SUPPORT:

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. FIG. 1 shows examples of hypertext and hypermedia documents and links associating data objects in the documents to other data objects. Hypermedia document 10 includes hypertext 20, an image icon at 22, a sound icon at 24 and more hypertext 26. FIG. 1 shows hypermedia document 10 substantially as it would appear on a user's display screen."

3:34 "As discussed above, hypermedia documents allow a user to access different data objects. The objects may be text, images, sound files, video, additional documents, etc. As used in this specification, a data object is information capable of being retrieved and presented to a user of a computer system."

9:20 "In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

*containing **information to enable** a browser application to **display**, on the client workstation, at least a portion of a **distributed hypermedia** document*

EXAMPLE SUPPORT:

1:61 "A hypertext document is a document that allows a user to view a text document displayed on a display device connected to the user's computer and to access, retrieve and view other data objects that are linked to hypertext words or phrases in the hypertext document."

2:14 "Objects may be text, images, sound files, video data, documents or other types of information that is presentable to a user of a computer system. When a document is primarily text and includes links to other data objects according to the hypertext format, the document is said to be a hypertext document. When graphics, sound, video or other media capable of being manipulated and presented in a computer system is used as the object linked to, the document is said to be a hypermedia document. A hypermedia document is similar to a hypertext document, except that the user is able to click on images, sound icons, video icons, etc., that link to other objects of various media types, such as additional graphics, sound, video, text, or hypermedia or hypertext documents. "

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

within a ***browser-controlled window,***

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

wherein the ***client workstation executes the browser application,***

EXAMPLE SUPPORT:

9:15 "Browser client 208 is a process that a user of client computer 200 invokes in order to access various data objects, such as hypermedia documents, on network 206. Hypermedia document 212 shown within client computer 200 is an example of a hypermedia document, or object, that a user has requested access to. In this example, hypermedia document 212 has been retrieved from a server connected to network 206 and has been loaded into, e.g., client computer 200's RAM or other storage device. "

with the browser application ***responding to text formats*** *to initiate processing specified by the text formats,*

EXAMPLE SUPPORT:

9:24 "Once hypermedia document 212 has been loaded into client computer 200, browser client 208 parses hypermedia document 212. In parsing hypermedia document 212, browser client 208 detects links to data objects as discussed above in the Background of the Invention section."

wherein at least ***said portion of the document is displayed*** *within the browser-controlled window,*

EXAMPLE SUPPORT:

14:12 "Returning to FIG. 7, it is assumed that a hypermedia document has been obtained at a user's client computer and that a browser program executing on the client computer displays the document"

wherein ***an embed text format which corresponds a said first location in the document is identified by the browser,***

EXAMPLE SUPPORT:

14:27 "a check is made as to whether the current tag is the EMBED tag."

wherein the embed text format ***specifies the location of at least a portion of an object,***

EXAMPLE SUPPORT:

6:63 "The present invention allows a user at a client computer connected to a network to locate, retrieve and manipulate objects in an interactive way."
14:32 "Each occurrence of a valid EMBED tag specifies an embedded object."
14:67 "the data object specified by the URL in the EMBED tag. "

wherein an executable application associated with the object is ***identified and located by the browser,***

EXAMPLE SUPPORT:

15:11 "step 290 is executed to launch a predetermined application. In a preferred embodiment an application is launched according to a user-defined list of application type/application pairs. The list is defined as a user-configurable XResource as described in "Xlib Programming Manual." An alternative embodiment could use the MIME database as the source of the list of application type/application pairs."

wherein the executable application is ***automatically invoked by the browser,*** *in response to the identifying of the embed text format,*

EXAMPLE SUPPORT:

9:41 "When browser client 208 encounters embedded program link 214, it invokes application client 210 (optionally, with parameters or other information)"

15:11 "step 290 is executed to launch a predetermined application.

*to **enable an end-user to directly interact with the object***

EXAMPLE SUPPORT:

10:2 "The user is then able to interactively operate controls to recompute different views for the image data. In a preferred embodiment, a control window is displayed within, or adjacent to, a window generated by browser client 208 that contains a display of hypermedia document 212. An example of such display is discussed below in connection with FIG. 9. Thus, the user is able to interactively manipulate a multidimensional image object by means of the present invention."

***while the object is being displayed within a display area** created at the **first location** within the portion of the hypermedia document being displayed in the browser-controlled window,*

EXAMPLE SUPPORT:

16:8 "FIG. 9 is a screen display of the invention showing an interactive application object (in this case a three dimensional image object) in a window within a browser window. In FIG. 9, the browser is NCSA Mosaic version 2.4. The processes VIS, Panel and VRServer work as discussed above. FIG. 9 shows screen display 356 Mosaic window 350 containing image window 352 and a portion of a panel window 354. Note that image window 352 is within Mosaic window 350 while panel window 354 is external to Mosaic window 350. Another possibility is to have panel window 354 within Mosaic window 350."

*wherein the executable application is part of a **distributed application**, and wherein at least a portion of the distributed application is for **execution on the network server**;*

EXAMPLE SUPPORT:

10:33 "Another embodiment of the present invention uses an application server process executing on server computer 204 to assist in processing that may need to be performed by an external program. For example, in FIG. 5, application server 220 resides on server computer 204. Application server 220 works in communication with application client 210 residing on client computer 200."

11:18 "FIG. 6 shows yet another embodiment of the present invention. FIG. 6 is similar to FIG. 5, except that additional computers 222 and 224 are illustrated. Each additional computer includes a process labeled "Application (Distributed)." The distributed application performs a portion of the task that an application, such as application server 220 or application client 210, perform. In the present. example, tasks such as volume rendering may be broken up and easily performed among two or more computers. These computers can be remote from each other on network 206. Thus, several computers, such as server computer 204 and additional computers 222 and 224 can all work together"

executing one or more instructions ***in response to the commands;***

sending information to the client workstation in response to the executed instructions, to allow processing of the information at the client workstation ***to enable said end-user to directly interact with said object****.*

EXAMPLE SUPPORT:

10:52 "In a preferred embodiment, application client 210 receives signals from a user input device at the user's client computer 200. An example of such input would be to rotate the embryo image from a current position to a new position from the user's point of view. This information is received by application client 210 and processed to generate a command sent over network 206 to application server 220. Once application server 220 receives the information in the form of, e.g., a coordinate transformation for a new viewing position, application server 220 performs the mathematical calculations to compute a new view for the embryo image. Once the new view has been computed, the image data for the new view is sent over network 206 to application client 210 so that application client 210 can update the viewing window currently displaying the embryo image."

<u>**CLAIM 48**</u>. *The method of claim 47 where:*
*the information to enable comprises **text formats**.*
EXAMPLE SUPPORT:
14:24 "Assuming there is more text to parse, execution proceeds to step 256 where routines in HTMLparse.c obtain the next item (e.g., word, tag or symbol) from the document."

<u>**CLAIM 49**</u>. *The method of claim 48 where:*
*the text formats are **HTML tags**.*
EXAMPLE SUPPORT:
14:19 "the document is parsed or scanned for HTML tags or other symbols. "

<u>**CLAIM 50**</u>. *The method of claim 47 where:*
*the information contained in the file received comprises **at least one embed text format**.*
EXAMPLE SUPPORT:
14:29 "If, at step 258, it is determined that the tag is the EMBED tag, execution proceeds to step 260 where an enumerated type is assigned for the tag. Each occurrence of a valid EMBED tag specifies an embedded object."

CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at (925) 944-3320.

Respectfully submitted,

/Charles E. Krueger/

Charles E. Krueger
Reg. No. 30,077

LAW OFFICE OF CHARLES E. KRUEGER
P.O.Box 5607
Walnut Creek, CA 94596
Tel: (925) 944-3320 / Fax: (925) 944-3363

985 PH Ex. 15

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

30080 7590 03/20/2009

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	
ART UNIT	PAPER NUMBER
2454	

DATE MAILED: 03/20/2009

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

TITLE OF INVENTION: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1510	$300	$0	$1810	06/22/2009

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

B. If the status above is to be removed, check box 5b on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above, or

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 box 5a on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and 1/2 the ISSUE FEE shown above.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail **Mail Stop ISSUE FEE**
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
or Fax (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

30080 7590 03/20/2009

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

TITLE OF INVENTION: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1510	$300	$0	$1810	06/22/2009

EXAMINER	ART UNIT	CLASS-SUBCLASS
DONAGHUE, LARRY D	2454	709-202000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
   - ☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
   - ☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
   (1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
   (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.
   1__________
   2__________
   3__________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. The following fee(s) are submitted:
- ☐ Issue Fee
- ☐ Publication Fee (No small entity discount permitted)
- ☐ Advance Order - # of Copies __________

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**
- ☐ A check is enclosed.
- ☐ Payment by credit card. Form PTO-2038 is attached.
- ☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number __________ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)
- ☐ a. Applicant claims SMALL ENTITY status. See 37 CFR 1.27.
- ☐ b. Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature ____________________ Date ____________________

Typed or printed name ____________________ Registration No. ____________________

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 03/20/2009

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	
ART UNIT	PAPER NUMBER
2454	

DATE MAILED: 03/20/2009

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment to date is 1367 day(s). If the issue fee is paid on the date that is three months after the mailing date of this notice and the patent issues on the Tuesday before the date that is 28 weeks (six and a half months) after the mailing date of this notice, the Patent Term Adjustment will be 1367 day(s).

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

Notice of Allowability	**Application No.** 10/217,955	**Applicant(s)** DOYLE ET AL.	
	Examiner Larry D. Donaghue	**Art Unit** 2454	

-- 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 *paper filed 02/09/2009*.
2. ☒ The allowed claim(s) is/are *4-49*.
3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ 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.
5. ☐ CORRECTED DRAWINGS (as "replacement sheets") 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./Mail Date _____.
   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).
6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. ☒ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 1/3/07,11/1/07,12/3/07,5/7/07, 1/26/07,2/5/09,12/11/07,1/29/07,02/22/08,6/20/07,10/15/07,02/02/07
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. ☐ Notice of Informal Patent Application
6. ☐ Interview Summary (PTO-413), Paper No./Mail Date _____ .
7. ☐ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. ☐ Other _____.

/Larry D Donaghue/
Primary Examiner, Art Unit 2454

The following is an examiner's statement of reasons for allowance: the claims are allowable as the claims contain the subject matter deemed allowable in both Re exam 90/006,831 and Re exam 90/007,838 for the same reasons as set forth in the NIRC of the two Re exams.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Larry D. Donaghue whose telephone number is 571-272-3962. The examiner can normally be reached on M-F 8:00-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Nathan Flynn can be reached on 571-272-1915. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Larry D Donaghue
Primary Examiner
Art Unit 2454

/Larry D Donaghue/
Primary Examiner, Art Unit 2454

985 PH Ex. 16

PATENT

Attorney Docket No. 006-1-4

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Doyle et al.

Application No.: 10/217,955

Filed: 08/09/2002

For: DISTRIBUTED HYPERMEDIA METHOD AND SYSTEM FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT

Examiner: DONAGHUE, LARRY D

Art Unit: 2454

Comments on Statement of Reasons for Allowance

Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

Applicant acknowledges with appreciation the Notice of Allowance and Patent Term Adjustment mailed March 20, 2009.

Two typographical errors were noted as follows.

In the examiner's statement of reasons for allowance "Re exam 90/007,838" should read "Re exam 90/007,8_5_8."

Also, in the Notice of Allowability, item number two, "The allowed claims are 4-49" should read "The allowed claims are 4-_50_." Claim 50 was presented in the amendment filed 02/05/2009 and depends on allowed claim 47.

Respectfully submitted,

[illegible]

Charles E. Krueger
Reg. No. 30,077

LAW OFFICE OF CHARLES E. KRUEGER
P.O. Box 5607
Walnut Creek, CA 94596
Tel: (925) 944-3320 / Fax: (925) 944-3363

985 PH Ex. 17

UNITED STATES DEPARTMENT OF COMMERCE
U.S. Patent and Trademark Office
Address : COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450

APPLICATION NO./ CONTROL NO.	FILING DATE	FIRST NAMED INVENTOR / PATENT IN REEXAMINATION	ATTORNEY DOCKET NO.
10217955	8/9/02	DOYLE ET AL.	006-1-4

LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
Larry D. Donaghue	
ART UNIT	PAPER
2454	20090826

DATE MAILED:

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner for Patents

Attached is a corrected Notice of Allowance with the proper listing of the claims. Examiner apologizes for the typographicl error in the claim numbering and Re exam number, the correct number is Re exam 90/007,858.

/Larry D Donaghue/
Primary Examiner, Art Unit 2454

PTO-90C (Rev.04-03)

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/217,955	08/09/2002	Michael D. Doyle	006-1-4	9596

30080 7590 08/31/2009
LAW OFFICE OF CHARLES E. KRUEGER
P.O. BOX 5607
WALNUT CREEK, CA 94596-1607

EXAMINER	
DONAGHUE, LARRY D	

ART UNIT	PAPER NUMBER
2454	

MAIL DATE	DELIVERY MODE
08/31/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of Allowability	**Application No.** 10/217,955	**Applicant(s)** DOYLE ET AL.	
	Examiner Larry D. Donaghue	**Art Unit** 2454	

-- 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 *paper filed 02/09/2009*.

2. ☒ The allowed claim(s) is/are *4-50*.

3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ 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.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") 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./Mail Date _____.
   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)	5. ☐ Notice of Informal Patent Application
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)	6. ☐ Interview Summary (PTO-413), Paper No./Mail Date _____ .
3. ☒ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 1/3/07,11/1/07,12/3/07,5/7/07, 1/26/07,2/5/09,12/11/07,1/29/07,02/22/08,6/20/07,10/15/07,02/02/07	7. ☐ Examiner's Amendment/Comment
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material	8. ☒ Examiner's Statement of Reasons for Allowance
	9. ☐ Other _____.

/Larry D Donaghue/
Primary Examiner, Art Unit 2454

Issue Classification

Application/Control No. 10217955	**Applicant(s)/Patent Under Reexamination** DOYLE ET AL.
Examiner Larry D Donaghue	**Art Unit** 2454

ORIGINAL

CLASS	SUBCLASS
709	202

INTERNATIONAL CLASSIFICATION

CLAIMED					NON-CLAIMED
G	0	6	F	9 / 46 (2006.01.01)	
G	0	6	F	17 / 30 (2006.01.01)	
G	0	6	F	9 / 50 (2006.01.01)	

CROSS REFERENCE(S)

CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)				
709	218	219			
715	205	738	760	777	804
719	310	315			
718	106				
345	419	427	619	638	649
345	653	654	655	656	
707	E17.119				

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

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

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 46	
/Larry D Donaghue/ Primary Examiner.Art Unit 2454 (Primary Examiner)	03/01/2009 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 5