

# 906 PH Ex. 7





UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
---------------	-------------	----------------------	---------------------

08/324,443	10/17/94	DOYLE	M 02307553
------------	----------	-------	------------

TOWNSEND AND TOWNSEND  
KHOURIE AND CREW  
STEUART STREET TOWER  
ONE MARKET PLAZA  
SAN FRANCISCO CA 94105

E3M1/0124

EXAMINER

ART UNIT, M	PAPER NUMBER
-------------	--------------

2317

9

DATE MAILED: 01/24/97

This is a communication from the examiner in charge of your application.  
COMMISSIONER OF PATENTS AND TRADEMARKS

This application has been examined  Responsive to communication filed on 1-8-97  This action is made final.

A shortened statutory period for response to this action is set to expire 3 month(s), 0 days from the date of this letter.  
Failure to respond within the period for response will cause the application to become abandoned. 35 U.S.C. 133

Part I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:

- |   |   |
|---|---|
| 1. <input type="checkbox"/> Notice of References Cited by Examiner, PTO-892.        | 2. <input type="checkbox"/> Notice of Draftsman's Patent Drawing Review, PTO-948. |
| 3. <input type="checkbox"/> Notice of Art Cited by Applicant, PTO-1449.             | 4. <input type="checkbox"/> Notice of Informal Patent Application, PTO-152.       |
| 5. <input type="checkbox"/> Information on How to Effect Drawing Changes, PTO-1474. | 6. <input type="checkbox"/> _____   |

Part II SUMMARY OF ACTION

1.  Claims 1-56 are pending in the application.

Of the above, claims \_\_\_\_\_ are withdrawn from consideration.

2.  Claims \_\_\_\_\_ have been cancelled.

3.  Claims \_\_\_\_\_ are allowed.

4.  Claims 1-56 are rejected.

5.  Claims \_\_\_\_\_ are objected to.

6.  Claims \_\_\_\_\_ are subject to restriction or election requirement.

7.  This application has been filed with informal drawings under 37 C.F.R. 1.85 which are acceptable for examination purposes.

8.  Formal drawings are required in response to this Office action.

9.  The corrected or substitute drawings have been received on \_\_\_\_\_. Under 37 C.F.R. 1.84 these drawings are  acceptable;  not acceptable (see explanation or Notice of Draftsman's Patent Drawing Review, PTO-948).

10.  The proposed additional or substitute sheet(s) of drawings, filed on \_\_\_\_\_, has (have) been  approved by the examiner;  disapproved by the examiner (see explanation).

11.  The proposed drawing correction, filed \_\_\_\_\_, has been  approved;  disapproved (see explanation).

12.  Acknowledgement is made of the claim for priority under 35 U.S.C. 119. The certified copy has  been received  not been received  been filed in parent application, serial no. \_\_\_\_\_; filed on \_\_\_\_\_.

13.  Since this application appears to be 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.

14.  Other

EXAMINER'S ACTION

Serial Number:08/324,443  
Art Unit: 2317

-2-

**Part III DETAILED ACTION**

The Wynne reference is withdraw in view of Applicant's Rule 131 Declaration filed 01-08-97.

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.*

**Claims 1-5, 10-14 and 44-48 are rejected under 35**

**U.S.C. 103(a) as being unpatentable over HyperNet as disclosed by Wynne et al. in "Lean Management, Group Support Systems, and Hypermedia: A combination whose time has come" and further in view of Hansen "Andrew as a Multiparadigm Environment for Visual Languages".**

As per claim 1, Wynne disclosed a method for running an application program in a computer network environment essentially as claimed, comprising:

providing at least one client workstation and one network server coupled to said network environment, wherein said network

Serial Number:08/324,443  
Art Unit: 2317

-3-

environment is a distributed hypermedia environment [

Abstract "world-wide over INTERNET"];

executing, at said client workstation, a browser application [p.113 col.1 2nd paragraph "browser"], that parses a distributed hypermedia document and initiates processes specified in the document [p.113 "Active link"];

Wynne did not specifically disclose the document having text formats. It is not clear whether the HyperNet's hypermedia documents disclosed by Wynne uses text formats. However it is well known in the art at the time of the invention to form hypermedia documents using text formats (e.g. SGML, HTML, etc.). Hansen teaches to use text because it is machine independent so the result is more portable [p.257 4th paragraph]. Hence, one of ordinary skill in the art would have been motivated to use text formats to form hypermedia document.

Wynne teaches utilizing browser to display, on said client workstation, a hypermedia document received over said network from said server, wherein said first hypermedia document is displayed within a first browser-controlled window [inherent] on said client workstation and wherein said first distributed hypermedia document includes an embed text format [inherent in the system as modified] that specifies the location of an object external to the first distributed hypermedia document and that specifies type information utilized by said browser to identify

and locate an executable application [p.113 col.1 4th paragraph - "Active links"] external to the first distributed hypermedia document;

invoking, with said browser application, said executable application [p.113 col.1 1st paragraph] to display and process said object.

Wynne does not disclose displaying and process said object within the first browser-controlled window while a portion of said first distributed hypermedia document continues to be displayed within said browser-controlled window. The external application is launched into a separate window to process the object [p. 113 col.1 lines 5-10].

Hansen teaches "it may be adequate to display each sublanguage element in a separate window, but this runs the risk of chaotic imagery among which it is difficult to discern the relationships among program segments. Instead, the author should have the power to organize the program fragments for perusal by the reader. The organization itself, together with commentary, aids the reader in comprehending the program." [p.256 col.1<sup>2</sup>]. Hence, it would have been obvious for one of ordinary skill in the art to provide external application to display and process the object within the browser-controlled window because it would have improved the system by reducing clustering of the display and aiding the reader comprehension of the hypermedia document.

Serial Number:08/324,443  
Art Unit: 2317

-5-

As per claim 2, Wynne disclose interactively controlling the executable application via communication over the network [p.113 col.2 "UPLINK", "DNLINK", "BILINK"].

As per claim 3, it is apparent that the network server [source] execute one or more instructions in response to commands from a client [DNLINK] and sending information from said network server to the client workstation in response to said executed instructions.

As per claim 4, it is inherent that the instruction for controlling the object reside on the client workstation [Wynne's p.113 col.1 lines 5-12"].

As per claim 5, it is apparent in the system as modified that communication would continue to be exchange between the controllable application and the browser in order for the controllable application to control the object within the browser's window.

As per claims 10-14, Wynne discloses that HyperNet communicates over the Internet. Hence, it is apparent that the HyperNet would use ISO TCP/IP standard and Hypertext Transfer Protocol. It would have been obvious for one of ordinary skill in the art at the time of the invention to use ISO TCP/IP standard and Hypertext Transfer Protocol because these are well

Serial Number:08/324,443  
Art Unit: 2317

-6-

defined standard for communicating hypermedia documents over the Internet.

As per claim 44, it is rejected under similar rationale as for claim 1 above.

As per claims 45-48, they are rejected under similar rationale as for claims 2-5 above.

**Claims 6-9, and 49-53 are rejected under 35 U.S.C. 103(a) as being unpatentable over Wynne et al. in "Lean Management, Group Support Systems, and Hypermedia: A combination whose time has come", Hansen "Andrew as a Multiparadigm Environment for Visual Languages", and further in view of Rizzo "What's OpenDoc?" (prior art submitted by applicant).**

As per claims 6-9 and 49-52, The applied references do not specifically disclose application being a multi-dimension viewer, a spreadsheet, a database, or word processor program. Rizzo discloses a systems that allows for embedding object of different applications (word processing, spreadsheet, database, movie) in one document and manipulation of the object within the document using functions of the corresponding application. Hence, it was well within the skill on one of ordinary skill in the art to provide controllable application for database, spreadsheet, word processing, etc. functions. The type of program provided would have been a matter of design choice.



Serial Number:08/324,443  
Art Unit: 2317

-7-

As per claim 53, It would have been obvious for one of ordinary skill in the art at the time of the invention to use Hypertext Markup Language because it is a well defined standard for forming hypermedia documents.

**Claims 15, 17-23, 24-33, 34-43, 54, 55, and 56 are rejected under 35 U.S.C. 103(a) as being unpatentable over Wynne et al. in "Lean Management, Group Support Systems, and Hypermedia: A combination whose time has come", Hansen "Andrew as a Multi-paradigm Environment for Visual Languages", and further in view of Moran "Tele-Nice-Slicer: A New Tool for the Visualization of Large Volumetric Data".**

As per claim 15, it is rejected under similar rationale as for claim 1 above. Wynne and Hansen do not specifically teach a multidimensional data visualization application.

Moran discloses a distributed system for interactive control and visualization of graphical object through communication over network. Moran teaches determining orientation and rendering of images by sending command comprising of text fields [p.3 col.1] over communication network.

It would have been obvious for one of ordinary skill in the art to combine Moran teaching with Wynne because it would have improved the system to provide powerful image visualization, presentation and control to scientists world wide.

Serial Number:08/324,443  
Art Unit: 2317

-8-

As per claims 17-22, the recited limitations - volume visualization, 2d image, image analysis, animated sequences, geometric viewer, and molecular modeling - would have been a matter of design choice because they are merely well known visualization methods.

As per claim 23, it is apparent in the system as modified that communication continue to be exchange between the multidimensional data visualization application and the browser in order for the visualization application to control the object within the browser's window.

As per claim 24, it is rejected under similar rationale as for claim 15 above. The references do not specifically disclose the step of transferring ..., accepting ..., executing ..., communicating ..., using ... The steps recited is inherent in the prior art as modified because:

It is well known in the art, at the time of the invention, that HTML documents contains links specified by URL's. It is known that HTML documents transfers involves HTTP protocol messages. The process involves:

transferring, over the network, a hypermedia document [the HTML document] with embedded objects [URL links, mapped images, fill-in forms, etc.] from a server computer to the client computer;

parsing the document by the browser to locate reference to external objects [URL's, images, etc.];

accepting first signals from the user input device [clicking on an URL link, or a mapped image, or a form's 'submit' button]

issuing commands [HTTP message with the linked URL, or coordinates where the mapped image was clicked, or the form's content] from the client computer to a first computer in response to the signal [it is known that an HTTP message in an HTML document can direct to any computer connected to the Internet that accept HTTP protocol];

Moran teaches executing instructions by a first additional computer and generate information about manipulating the embedded object; communicating the information to the client; and using the client to manipulate the object according to the communicated information [Apparent from p.3 "TNSD client functionality" and "TNSD Server Functionality"].

As per claims 25, 27, the document is a hypermedia document [Wynne p.112 col.2 3rd paragraph].

As per claim 26, It would have been obvious for one of ordinary skill in the art to have multiple computers to response

Serial Number:08/324,443  
Art Unit: 2317

-10-

to issued commands because it would have distributed the processing load.

As per claims 28, 30, and 32 the references do not specifically disclose multi-dimensional image displayable in plurality of orientations, and function to determine the new orientation and rendering of image. The type of objects and functions provided would have been a matter of design choice.

Moran discloses a distributed system for interactive control and visualization of graphical object through communication over network. Moran teaches determining orientation and rendering of images by sending command comprising of text fields [p.3 col.1] over communication network.

It would have been obvious for one of ordinary skill in the art to combine Moran teaching with Wynne because it would have improved the system to provide powerful image visualization, presentation and control to scientists world wide.

As per claims 29, 31, 33, Wynne teaches the document is a hypermedia document [Wynne p.112 col.2 3rd paragraph].

As per claims 34, it is rejected under similar rationale as for claim 24 above.

Serial Number:08/324,443  
Art Unit: 2317

-11-

Moran does not specifically disclose a second server. However, it would have been obvious for one of ordinary skill in the art to provide plurality of servers to speed up processing.

As per claims 35, 37, 39, 41, and 43, Wynne teaches the system is a distributed hypermedia environment [Wynne p.112 col.2 3rd paragraph].

As per claim 36, Moran teaches distributing the processing on various computers [client - server]. It would have been obvious for one of ordinary skill in the art to distribute the processing to the machine such that the instructions is executed faster.

As per claims 38 and 40, Moran teaches determining orientation and rendering of images [p.2 - p.3].

As per claim 42, Moran teaches dynamically manipulate the object [p.2 - zoom]. It is apparent that the system as modified would accept signal from user input to indicate a second orientation of an object.

As per claim 54, it is rejected under similar rationale as for claim 15 above.

As per claim 55, it is rejected under similar rationale as for claim 24 above.

Serial Number:08/324,443  
Art Unit: 2317

-12-

As per claim 56, it is rejected under similar rationale as for claim 34 above.

**THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).


A shortened statutory period for response to this final action is set to expire THREE MONTHS from the date of this action. In the event a first response is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event will the statutory period for response expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Dung Dinh whose telephone number is (703) 305-9655. The examiner can normally be reached on Monday-Thursday from 7:00 AM - 4:30 PM. The examiner can also be reached on alternate Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thomas Lee can be reached at (703) 305-9717. The fax phone number for this group is (703) 308-5359.

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

Dung Dinh  
Jan. 23, 1997

  
THOMAS C. LEE  
SUPERVISORY PATENT EXAMINER  
230

# 906 PH Ex. 8





I hereby certify that this correspondence is being sent by facsimile transmission to: D. Dinh  
Fax No.: 1-703-308-5359  
Assistant Commissioner for Patents,  
Washington, D.C. 20231,

PATENT

on 2/19/97

Attorney Docket No. 02307I-553

TOWNSEND and TOWNSEND and CREW LLP

FEB 19 1997

# 16

By [Signature]

(Not)  
BT  
2-20-97

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: )  
MICHAEL D. DOYLE et al. )  
Application No.: 08/324,443 )  
Filed: 10/17/94 )  
For: EMBEDDED PROGRAM OBJECTS IN )  
DISTRIBUTED HYPERMEDIA )  
SYSTEMS )

Examiner: D. Dinh

Art Unit: 2317

COMMUNICATION

**OFFICIAL**

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

The following is responsive to the Office Action mailed January 24, 1997:

REMARKS

It is believed that the Office Action contains a minor typographical error in the first paragraph in referring to applicant's Rule 131 Declaration. The reference which was the subject matter of that declaration was Vetter "Mosaic and the World-Wide Web," not Wynne as stated in the Office Action.

Accordingly, this communication is submitted to call the examiner's attention to this error so that the record can be corrected.

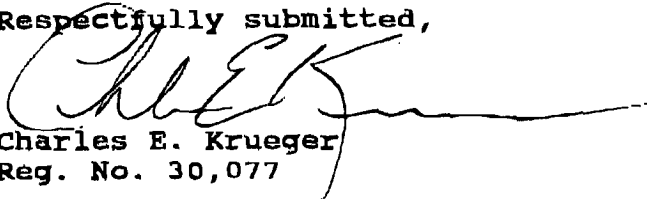
Further, this to confirm that one of the co-inventors, Michael Doyle, and his attorney, Charles Krueger, will appear at the examiner's office for an interview at 9:00 A.M. on Monday, February 24, 1997.

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 2

PATENT

If the Examiner has any comments or questions, please  
telephone the undersigned at (415) 576-0200.

Respectfully submitted,



Charles E. Krueger  
Reg. No. 30,077

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, California 94111-3834  
(415) 576-0200  
Fax (415) 576-0300  
CEK:db

s:\02307I\553\comm.1

# 906 PH Ex. 9





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

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKETT NO.
---------------	-------------	-----------------------	----------------------

08/324,413 08/324,443	10/17/94	DOYLE	M 02307553
--------------------------	----------	-------	------------

B3M1/0226

TOWNSEND AND TOWNSEND  
 KHOURIE AND CREW  
 STEUART STREET TOWER  
 ONE MARKET PLAZA  
 SAN FRANCISCO CA 94105

EXAMINER
----------

DINH, D

ART UNIT	PAPER NUMBER
----------	--------------

2317

13

DATE MAILED:

02/26/97

### EXAMINER INTERVIEW SUMMARY RECORD

All participants (applicant, applicant's representative, PTO personnel):

- (1) Michael Doyle (3) Thomas Lee  
 (2) Charles Krueger (4) Dung Dinh

Date of Interview: 2/24/97

Type:  Telephonic  Personal (copy is given to  applicant  applicant's representative).

Exhibit shown or demonstration conducted:  Yes  No. If yes, brief description: \_\_\_\_\_

How HyperNet work and different from the present invention.

Agreement  was reached with respect to some or all of the claims in question.  was not reached.

Claims discussed: All

Identification of prior art discussed: Wynee

Description of the general nature of what was agreed to if an agreement was reached, or any other comments: \_\_\_\_\_

- 1) HyperNet is a compiled system, 2) Tag in document to activate external program (delayed binding),  
3) display and process by the external application within Browser's controlled window.

Applicant's argument is persuasive to overcome the Hypernet ref. The claims are distinguished over  
the prior art of record.

(A fuller description, if necessary, and a copy of the amendments, if available, which the examiner agreed would render the claims allowable must be attached. Also, where no copy of the amendments which would render the claims allowable is available, a summary thereof must be attached.)

1. It is not necessary for applicant to provide a separate record of the substance of the interview.

Unless the paragraph below has been checked to indicate to the contrary, A FORMAL WRITTEN RESPONSE TO THE LAST OFFICE ACTION IS NOT WAIVED AND MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW (e.g., items 1-7 on the reverse side of this form). If a response to the last Office action has already been filed, then applicant is given one month from this interview date to provide a statement of the substance of the interview.

2. Since the examiner's interview summary above (including any attachments) reflects a complete response to each of the objections, rejections and requirements that may be present in the last Office action, and since the claims are now allowable, this completed form is considered to fulfill the response requirements of the last Office action. Applicant is not relieved from providing a separate record of the substance of the interview unless box 1 above is also checked.

[Signature]  
 Examiner's Signature



# 906 PH Ex. 10







UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
---------------	-------------	----------------------	---------------------

08/324,443	10/17/94	DOYLE	M 02307553
------------	----------	-------	------------

TOWNSEND AND TOWNSEND  
KHOURIE AND CREW  
STEWART STREET TOWER  
ONE MARKET PLAZA  
SAN FRANCISCO CA 94105

B3M1/0326

EXAMINER

ART UNIT, D	PAPER NUMBER
-------------	--------------

2317

DATE MAILED: 03/26/97

This is a communication from the examiner in charge of your application.  
COMMISSIONER OF PATENTS AND TRADEMARKS

This application has been examined     Responsive to communication filed on 2-19-97     This action is made final.

A shortened statutory period for response to this action is set to expire 3 month(s), 0 days from the date of this letter.  
Failure to respond within the period for response will cause the application to become abandoned. 35 U.S.C. 133

Part I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:

- |   |   |
|---|---|
| 1. <input checked="" type="checkbox"/> Notice of References Cited by Examiner, PTO-892. | 2. <input type="checkbox"/> Notice of Draftsman's Patent Drawing Review, PTO-948. |
| 3. <input type="checkbox"/> Notice of Art Cited by Applicant, PTO-1449.                 | 4. <input type="checkbox"/> Notice of Informal Patent Application, PTO-152.       |
| 5. <input type="checkbox"/> Information on How to Effect Drawing Changes, PTO-1474.     | 6. <input type="checkbox"/> _____   |

Part II SUMMARY OF ACTION

- Claims 1-56 are pending in the application.  
Of the above, claims \_\_\_\_\_ are withdrawn from consideration.
- Claims 16 have been cancelled.
- Claims \_\_\_\_\_ are allowed.
- Claims 1-15, 17-56 are rejected.
- Claims \_\_\_\_\_ are objected to.
- Claims \_\_\_\_\_ are subject to restriction or election requirement.
- This application has been filed with informal drawings under 37 C.F.R. 1.85 which are acceptable for examination purposes.
- Formal drawings are required in response to this Office action.
- The corrected or substitute drawings have been received on \_\_\_\_\_. Under 37 C.F.R. 1.84 these drawings are  acceptable;  not acceptable (see explanation or Notice of Draftsman's Patent Drawing Review, PTO-948).
- The proposed additional or substitute sheet(s) of drawings, filed on \_\_\_\_\_, has (have) been  approved by the examiner;  disapproved by the examiner (see explanation).
- The proposed drawing correction, filed \_\_\_\_\_, has been  approved;  disapproved (see explanation).
- Acknowledgement is made of the claim for priority under 35 U.S.C. 119. The certified copy has  been received  not been received  been filed in parent application, serial no. \_\_\_\_\_; filed on \_\_\_\_\_.
- Since this application appears to be 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.
- Other

EXAMINER'S ACTION

Serial Number: 08/324,443  
Art Unit: 2317

-2-

**Part III DETAILED ACTION**

The finality of the last office action is withdraw.

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.*

**Claims 1, 44 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Applicant disclosed prior art and further in view of Khoyi et al. US patent 5,206,951 and Hansen "Andrew as a Multiparadigm Environment for Visual Languages".**

As per claim 1, Applicant disclosed prior art [pages 1-10: Mosiac + HTTP + HTML + "World Wide Web"] has the limitations essentially as claimed client workstation, network server coupled in a distributed hypermedia environment [p.8 lines 15-30];

executing on the client a browser application [p.4 Mosaic] that parses distributed hypermedia document to identify text formats [HTML tags] and for responding to predetermined text

Serial Number: 08/324,443  
Art Unit: 2317

-3-

formats to initiate processes specified by the text format [p.4-5];

utilizing the browser to display, on said client workstation, portion of a first hypermedia document received over the network, wherein the hypermedia document includes an embed text format specifies the location of an object external to the hypermedia document [p.4 lines 4-12, p.5 lines 9-26].

It is apparent that specifies type of information [p.5 lines 11 - text, images, sound, video...] is utilized by the browser to identify and locate an executable application external to the hypermedia document [p.4 lines 13-22 - "viewer" software];

Furthermore, Khoyi teaches an object data processing system operating in which an extensible set of object may be embedded within one document. The system invoke a corresponding object manager (a program external to the document) in response to an invocation request to process and control the object [col.9 lines 16-30, col.12 lines 49-68, col.13 lines 40-60, col.14 lines 32-44]. Khoyi teaches links specifying the object and type [col.13 lines 50-55]. It would have been obvious for one of ordinary skill in the art to combine the teaching of Khoyi with the disclosed prior art because it would have improved the system by providing open ended for integrating new object/application and

Serial Number: 08/324,443  
Art Unit: 2317

-4-

new type of information with existing type without recompiling the browser [col.12 line 65 to col.13 line 28].

The disclosed prior art does not invoke the executable application to display and process said object within the browser controlled window while a portion of the first hypermedia document continues to be displayed within the browser controlled window. The disclosed prior art launches the external application into a separate window to process the object.

Hansen discloses a "multiparadigm" environment which combines textual and graphical elements within a document. Hansen teaches "it may be adequate to display each sublanguage element in a separate window, but this runs the risk of chaotic imagery among which it is difficult to discern the relationships among program segments. Instead, the author should have the power to organize the program fragments for perusal by the reader. The organization itself, together with commentary, aids the reader in comprehending the program." [p.256 col.1 1st paragraph]. Hence, it would have been obvious for one of ordinary skill in the art to provide external application to display and process the object within the browser-controlled window because it would have improved the system by reducing clustering of the display and aiding the reader comprehension of the hypermedia document.

Serial Number: 08/324,443  
Art Unit: 2317

-5-

As per claim 44, it is rejected under similar rationale as for claim 1 above.

**Claims 2-6, 10-14, 45-48, 15, 17-23, 24-33, 34-43, 54, 55, and 56 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Applicant disclosed prior art, Khoyi et al, Hansen "Andrew as a Multiparadigm Environment for Visual Languages" and further in view of Moran "Tele-Nicer-Dicer: A new tool for the visualization of large volumetric data".**

As per claim 2, the disclosed prior art does not disclose interactively controlling via communication sent over the distributed environment. Moran discloses a distributed application (TNSD) for interactive control and visualization of graphical object through communication over network. Moran application allow usage of remote system resources for visualization of large data set at a client station. It would have been obvious for one of ordinary skill in the art to utilize Moran application as an external application ("Viewer") in the prior art system as modified because it would have improved the system by enabling the client station access to resources on higher performance servers and to have interactive visualization of large data set capability.

Serial Number: 08/324,443  
Art Unit: 2317

-6-

As per claim 3, Moran discloses sending command to remote server, executing on the server, and sending result to the client to process and display [p.3 col.2-3 specifically col.1 3rd paragraph].

As per claim 4 and 5, the limitation recited is inherent in the system as modified.

As per claim 6, Moran teaches a multi-dimensional viewer [Abstract].

As per claims 10-14, Applicant disclosed prior art communicates over the Internet, and uses ISO TCP/IP standard and Hypertext Transfer Protocol.

As per claims 45-48, they are rejected under similar rationales as for claims 2-5 above.

As per claim 49, Moran teaches a multi-dimensional viewer [Abstract].

As per claim 15, it is rejected under similar rationale as for claim 1 above. The disclosed prior art and Hansen do not specifically teach a multidimensional data visualization application.

Moran discloses a distributed system for interactive control and visualization of graphical object through communication over network. Moran teaches determining orientation and rendering of

Serial Number: 08/324,443  
Art Unit: 2317

-7-

images by sending command comprising of text fields [p.3 col.1] over communication network.

It would have been obvious for one of ordinary skill in the art to combine Moran teaching with the disclose prior art as modified because it would have improved the system to provide powerful image visualization, presentation and control to scientists world wide.

As per claims 17-22, the recited limitations - volume visualization, 2d image, image analysis, animated sequences, geometric viewer, and molecular modeling - would have been a matter of design choice because they are merely well known visualization methods.

As per claim 23, it is apparent in the system as modified that communication continue to be exchange between the multidimensional data visualization application and the browser in order for the visualization application to control the object within the browser's window.

As per claim 24, it is rejected under similar rationale as for claim 15 above. The references do not specifically disclose the step of transferring ..., accepting ..., executing ..., communicating ..., using ... The steps recited is inherent in the prior art as modified because:

Serial Number: 08/324,443  
Art Unit: 2317

-8-

It is well known in the art, at the time of the invention, that HTML documents contains links specified by URL's. It is known that HTML documents transfers involves HTTP protocol messages. The process involves:

transferring, over the network, a hypermedia document [the HTML document] with embedded objects [URL links, mapped images, fill-in forms, etc.] from a server computer to the client computer;

parsing the document by the browser to locate reference to external objects [URL's, images, etc.];

accepting first signals from the user input device [clicking on an URL link, or a mapped image, or a form's 'submit' button]

issuing commands [HTTP message with the linked URL, or coordinates where the mapped image was clicked, or the form's content] from the client computer to a first computer in response to the signal [it is known that an HTTP message in an HTML document can direct to any computer connected to the Internet that accept HTTP protocol];

Moran teaches executing instructions by a first additional computer and generate information about manipulating the embedded object; communicating the information to the client; and using the client to



Serial Number: 08/324,443  
Art Unit: 2317

-9-

manipulate the object according to the communicated information [Apparent from p.3 "TNSD client functionality" and "TNSD Server Functionality"].

As per claims 25, 27, 29, 31, 33 the disclosed prior art document is a hypermedia document [p.5 lines 10-25].

As per claim 26, Applicant disclosed that it known in prior art to access objects over multiple computers (servers) [p.4 lines 5-12, p.7 lines 22-28]. It would have been obvious for one of ordinary skill in the art to have multiple computers to response to issued commands because it would have distributed the processing load.

As per claims 28, 30, and 32, Moran discloses a distributed system for interactive control and visualization of graphical object through communication over network. Moran teaches determining orientation and rendering of images by sending command comprising of text fields [p.3 col.1] over communication network.

As per claims 34, it is rejected under similar rationale as for claim 24 above.

Moran does not specifically disclose a second server. However, it would have been obvious for one of ordinary skill in the art to provide plurality of servers to speed up processing.

Serial Number: 08/324,443  
Art Unit: 2317

-10-

As per claims 35, 37, 39, 41, and 43, the disclosed prior art system is a distributed hypermedia environment.

As per claim 36, Moran teaches distributing the processing on various computers [client - server]. It would have been obvious for one of ordinary skill in the art to distribute the processing to the machine in such a way that the instructions is executed faster.

As per claims 38 and 40, Moran teaches determining orientation and rendering of images [p.2 - p.3].

As per claim 42, Moran teaches dynamically manipulating the object [p.2 - zoom]. It is apparent that the system as modified would accept signal from user input to indicate a second orientation of an object.

As per claim 54, it is rejected under similar rationale as for claim 15 above.

As per claim 55, it is rejected under similar rationale as for claim 24 above.

As per claim 56, it is rejected under similar rationale as for claim 34 above.

**Claims 7-9, 50-53 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Applicant disclosed prior art, Hansen "Andrew as a Multiparadigm Environment for Visual Languages"**

Serial Number: 08/324,443  
Art Unit: 2317

-11-

**,Moran "Tele-Nicer-Dicer: A new tool for the visualization of large volumetric data" and further in view of Rizzo "What's OpenDoc?"** (prior art submitted by applicant).

As per claims 7-9 and 50-52, The applied references do not specifically disclose application being a spreadsheet, a database, or word processor program. Rizzo discloses a systems that allows for embedding object of different applications (word processing, spreadsheet, database, movie) in one document and manipulation of the object within the document using functions of the corresponding application. Hence, it was well within the skill on one of ordinary skill in the art to provide controllable application for database, spreadsheet, word processing, etc. functions. The type of program provided would have been a matter of design choice.

As per claim 53, the disclosed prior art uses Hyper Text Markup language.

Serial Number: 08/324,443  
Art Unit: 2317

-12-

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Dung Dinh whose telephone number is (703) 305-9655. The examiner can normally be reached on Monday-Thursday from 7:00 AM - 4:30 PM. The examiner can also be reached on alternate Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thomas Lee can be reached at (703) 305-9717. The fax phone number for this group is (703) 308-5359.

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



Dung Dinh  
Patent Examiner  
March 18, 1997

**Notice of References Cited**

Application No. <b>08/324,443</b>	Applicant(s) <b>Doyle et al.</b>	
Examiner <b>D. Dinh</b>	Group Art Unit <b>2317</b>	Page 1 of 1

**U.S. PATENT DOCUMENTS**

	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
A	5,206,951	04/27/93	Khoyi et al.	395	683
B					
C					
D					
E					
F					
G					
H					
I					
J					
K					
L					
M					

**FOREIGN PATENT DOCUMENTS**

	DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
N						
O						
P						
Q						
R						
S						
T						

**NON-PATENT DOCUMENTS**

	DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
U		
V		
W		
X		



# 906 PH Ex. 11





70826 U.S. PTO  
06/05/97

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

on June 2, 1997

TOWNSEND and TOWNSEND and CREW LLP

By [Signature]

#14 / B  
PATENT

Attorney Docket No. 02307I-553

BK  
6-19-97

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	)	
MICHAEL D. DOYLE et al.	)	Examiner: D. Dinh.
Application No.: 08/324,443	)	Art Unit: 2317
Filed: 10/17/94	)	<u>AMENDMENT</u>
For: EMBEDDED PROGRAM OBJECTS IN)	)	
DISTRIBUTED HYPERMEDIA	)	
SYSTEMS	)	

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

Responsive to the Office Action mailed March 26, 1997, please amend the above identified application as follows:

IN THE CLAIMS: ✓

Please cancel claim 6-15, 17-43, and 49-56.

✓  
Please amend the following claims:

1	1. (Twice Amended) A method for running an application
2	program in a computer network environment, comprising:
3	providing at least one client workstation and one
4	network server coupled to said network environment, wherein said
5	network environment is a distributed hypermedia environment;
6	executing, at said client workstation, a browser
7	application, that parses a <u>first</u> distributed hypermedia document
8	to identify text formats included in [the] <u>said</u> distributed
9	hypermedia document and for responding to predetermined text
10	formats to initiate processing specified by <u>said</u> text formats;
11	utilizing said browser to display, on said client workstation, at

B1

31

12 least a portion of a first hypermedia document received over said  
13 network from said server, wherein the portion of said first  
14 hypermedia document is displayed within a first browser-  
15 controlled window on said client workstation, [and] wherein said  
16 first distributed hypermedia document includes an embed text  
17 format, located at a first location in said first distributed  
18 hypermedia document, that specifies the location of at least a  
19 portion of an object external to the first distributed hypermedia  
20 document [ and that specifies], wherein said object has type  
21 information associated with it utilized by said browser to  
22 identify and locate an executable application external to the  
23 first distributed hypermedia document [;], and wherein said embed  
24 text format is parsed by said browser to automatically invoke  
25 [invoking, with said browser application,] said executable  
26 application to execute on said client workstation in order to  
27 display said object and enable interactive processing of said  
28 object within [the] a display <sup>area</sup> ~~window~~ created at said first  
29 location within the portion of said first distributed hypermedia  
30 document being displayed in said [the] first browser-controlled  
31 window [while a portion of said first distributed hypermedia  
32 document continues to be displayed within said browser-controlled  
33 window].

1 2. (Twice Amended) The method of claim 1, wherein  
2 said executable application is a controllable application and  
3 further comprising the step of:

4 interactively controlling said controllable  
5 application (from) on said client workstation via  
6 [communications sent over  
7 said distributed hypermedia environment]  
8 inter-process communications between said browser and said  
9 controllable application.

1 4 ~~3~~. (Twice Amended) The method of claim ~~2~~ <sup>3</sup> [2], wherein  
2 additional instructions for controlling said controllable  
3 application reside on said network server, wherein said step of

4 interactively controlling said controllable application includes  
5 the following substeps:

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

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

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

1 <sup>5</sup>/<sub>4</sub>. (Twice Amended) The method of claim <sup>4</sup>/<sub>2</sub> [2], wherein  
2 said additional instructions for controlling said controllable  
3 application reside on said client workstation.

B 1 <sup>3</sup>/<sub>5</sub>. (Twice Amended) The method of claim 2, wherein the  
2 communications to interactively control said controllable  
3 application [from said client workstation] continue to be  
4 exchanged between the controllable application and the  
5 [hypermedia] browser even after the controllable application  
6 program has been launched.

1 <sup>6</sup>/<sub>4</sub>. (Amended) A computer program product for use in a  
2 system having at least one client workstation and one network  
3 server coupled to said network environment, wherein said network  
4 environment is a distributed hypermedia environment, the computer  
5 program product comprising:

6 a computer usable medium having computer readable  
7 program code physically embodied therein [for causing a client  
8 workstation to invoke an external executable application  
9 referenced by a hypermedia document to display and process an  
10 external object referenced by the hypermedia document], said  
11 computer program product further comprising:

12 computer readable program code for causing said client  
13 workstation to execute a browser application to parse a  
14 first distributed hypermedia document to identify text  
15 formats included in [the] said distributed hypermedia

16 document and to respond to predetermined text formats to  
17 initiate processes specified <sup>by</sup> ~~said by the~~ text formats;  
18 computer readable program code for causing said client  
19 workstation to utilize said browser to display, on said  
20 client workstation, at least a portion of a first hypermedia  
21 document received over said network from said server,  
22 wherein the portion of said first hypermedia document is  
23 displayed within a first browser-controlled window on said  
24 client workstation, [and] wherein said first distributed  
25 hypermedia document includes an embed text format, located  
26 at a first location in said first distributed hypermedia  
27 document, that specifies the location of at least a portion  
28 of an object external to the first distributed hypermedia  
29 document [ and that specifies], wherein said object has type  
30 information associated with it utilized by said browser to  
31 identify and locate an executable application external to  
32 the first distributed hypermedia document[;  
33 computer readable program code for causing said client  
34 workstation to invoke, with said browser application], and  
35 wherein said embed text format is parsed by said browser to  
36 automatically invoke said executable application to execute  
37 on said client workstation in order to display said object  
38 and enable interactive processing of said object within  
39 [the] a display <sup>area</sup> window created at said first location within  
40 the portion of said first distributed hypermedia document  
41 being displayed in said first browser-controlled window  
42 [while a portion of said first distributed hypermedia  
43 document continues to be displayed within said  
44 browser-controlled window].

1 44. <sup>3</sup> (Amended) The computer program product of claim  
2 44, wherein said executable application is a controllable  
3 application and further comprising:

4 computer readable program code for causing said client  
5 workstation to interactively control said controllable  
6 application [from] on said client workstation via  
7 [conununications sent over said distributed hypermedia

8 environment] inter-process communications between said browser  
9 and said controllable application.

1 <sup>9</sup>/~~46~~. (Amended) The computer program product of claim <sup>8</sup>/~~48~~  
2 [45], wherein additional instructions for controlling said  
3 controllable application reside on said network server, wherein  
4 said step of interactively controlling said controllable  
5 application includes:

6 computer readable program code for causing said client  
7 workstation to issue, from the client workstation, one or more  
8 commands to the network server;

9 computer readable program code for causing said network  
10 server to execute one or more instructions in response to said  
11 commands;

12 computer readable program code for causing said network  
13 sever to send information to said client workstation in response  
14 to said executed instructions; and

15 computer readable program code for causing said client  
16 workstation to process said information at the client workstation  
17 to interactively control said controllable application.

1 <sup>10</sup>/~~47~~. (Amended) The computer program product of claim <sup>9</sup>/~~46~~  
2 [45], wherein said additional instructions for controlling said  
3 controllable application reside on said client workstation.

1 <sup>7</sup>/~~48~~. (Amended) The computer program product of  
2 claim ~~45~~, wherein the communications to interactively control  
3 said controllable application [from said client workstation]  
4 continue to be exchanged between the controllable application and  
5 the [hypermedia] browser even after the controllable application  
6 program has been launched.

REMARKS

Claims 1-15 and 17-56 have been examined, claims 1-5  
and 44-48 are amended herein, and claims 6-15, 17 43, and 49-56  
are canceled. Accordingly, claims 1-5 and 44-48 are now pending  
in the application.

THE REJECTION OF CLAIM 1

Claim 1 is rejected under 35 U.S.C. §103 as being unpatentable over Applicants' disclosed prior art (Mosaic, HTTP, HTML, and the World-Wide Web) and further in view of Khoyi et al. and Hansen.

THE CLAIMED INVENTION

The present invention, as defined for example in amended claim 1, includes the step of executing a browser that parses a first distributed hypermedia document to identify text formats included in the distributed hypermedia document and that responds to predetermined text formats to initiate processing specified by the text formats. The browser displays a portion of a first distributed hypermedia document in a browser-controlled window.

The first distributed hypermedia document includes an embed text format located at a first location in the document. The embed text format specifies the location of an object, at least a portion of which is external to the first distributed hypermedia document, that has type information associated with it which is utilized by the browser to identify and locate an executable application external the document.

The embed text format is parsed by the browser to cause the browser to automatically invoke the external application to execute on the client workstation. The external application displays, and allows the user to interactively process, the object in a display window created within the portion of the document being displayed in the browser-controlled window, at the location within the document of the embed text format.

THE CITED REFERENCES

1. Mosaic.

The Applicants' prior art (Mosaic) launches helper applications, in response to a user's interactive 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) (pg. 3, line 30). Many viewers exist that handle various file formats such as ".TIF", ".GIF", etc. When a user commands the browser program to invoke a viewer program, typically by clicking on an anchor with a mouse, the viewer is launched as a separate process. The viewer displays the full 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 is active. The viewer program is completely independent of the browser after being invoked by the browser. This means 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.

The attached pages (attachment A) describe how helper applications are invoked in Netscape Navigator, which uses the same mechanism as Mosaic. The user creates an association in a table between a file extension, e.g., the file extension ".MPG" indicates a file formatted in MPEG video. The browser could be configured to launch the helper application VMPEG to display a video file accessed using a URL in hypermedia document. As described above, the MPEG video file would be displayed in a separate window and the browser would be inactive.

## 2. Khoyi et al.

The reference Khoyi et al. describes an object-based data processing operating system. In that operating system data from a child object may be internalized in a parent object. Objects are related to one another through a linking mechanism (col. 10, line 21). The terms "parent" and "child" refer to the direction of the link (col. 10, line 31). Programs for operating on objects are known as "object managers", sometimes referred to as applications (col. 9, line 17). Each type of object has associated with it at least one object manager designed as the primary means for operating on data stored in that type of object (col. 9, line 20). For example, the system

may support a "document type" of object for word processing and a word processing object manager will be associated with that object type. Similarly, a "data base type" object will have associated with it a data base object manager as the primary means for operating upon data stored in the data base type object.

Application Integration Services of this operating system provide a mechanism by which an individual application (object manager) can appear to a user to integrate its operation and manipulation of data with that of other applications (col. 13, line 40). To a user, the display of a page of a newsletter having both text and a picture indicates to the user that the text and picture are integrated into a single document. However, in the Khoyi system this integration effect is accomplished by the operation of two different object managers coordinated by the use of the operating system's Application Integration Services. The newsletter is stored in an object type document, which is a type of object for storing text and formatting information. This document object includes a link to a separate object of type "image". The display is accomplished by a document object manager displaying the text and the image object manager displaying the picture. The information describing the link is communicated from the document object manager to the image object manager with the assistance of the operating system's Application Integration Services (col. 12, lines 60).

For most object types each object is stored in a separate file (col. 15, line 5). If a destination object is not capable of storing data from a source object then the data is encapsulated. For example, a document object cannot directly store picture data (col. 18, line 25). In order to edit encapsulated data, the user must select the data and issue an edit command, thereby invoking an object manager capable of handling objects of the type in which the encapsulated data is stored: because of this difference encapsulated data will typically be visually marked for the user (col. 18, lines 33-40). The user will observe that editing the picture requires an extra operation that results in opening a new window (col. 18, line



44). No changes occur in the children while the viewer is viewing the parent (col. 19, line 23).

Link markers are included in the body of an object's data to indicate the presence of linked data (col. 20, line 11).

The link marker is used to identify the object type so that the child (source) object's object manager can be called to display the child (source) object's data (col. 20, line 25). A link marker need not be physically stored at the location in the parent's data where the linked data is to appear (col. 20, line 45). Furthermore, the link marker does not specify the location of encapsulated data. The source object's object manager is used to locate that object's encapsulated data (col.15, line 48).

### 3. Hansen.

Hansen describes utilizing Andrew as a programming environment for authoring and editing software programs which are made up of multiple visual sub-languages. The reference teaches the creation of hierarchically-embedded windows within a document which provide views and interfaces to sub-elements of the parent document as depicted in Fig. 1. The *Layout* command provides for scattering objects in a rectangle as depicted in Fig. 1 (page 258, second column, 4th para.). Hansen states that the author should have the power to organize a program's constituent fragments for perusal by the reader. Since Hansen's system is intended for use with visual languages, which are languages that employ various graphical symbols to represent program elements and relationships between program elements, Hansen points out that it is preferable for the author to graphically arrange the various program fragments within a single window, in order to aid reader comprehension of the program, as a whole. Hansen makes no reference to embedding of any external executable program objects within a document being edited by a program author. Indeed, Hansen makes no reference to external executable program objects at all.

THE EXAMINER'S REASONING

The Examiner states that the Applicants' prior art (Mosaic) discloses utilizing the browser to display a portion of a hypermedia document that includes an embed text format that specifies the location an object external to the hypermedia document. It is also stated that is apparent that type information is specified which is used by the browser to identify and locate an executable application external to the hypermedia document.

With regard to Khoyi, the Examiner states that Khoyi teaches an object data processing system in which an extensible set of objects may be embedded within an object. The system invokes a corresponding object manager (a program external to the document) in response to an invocation request to process and control the object and teaches links specifying the object and type.

The Examiner acknowledges that Mosaic and Khoyi disclose launching the external application into a separate window to process the object.

However, the Examiner states that Hansen teaches that displaying each sublanguage element in a separate window runs the risk of chaotic imagery among which is difficult to discern the relationships among program segments.

The Examiner concludes that it "would be obvious to combine Khoyi with the Applicants' prior art to improve the system by providing open-ended for integrating new object/application and new type of information without recompiling the browser" (col. 12, line 65 to col. 13, line 28).

The Examiner also concludes that, based on the Hansen disclosure, it would have been obvious to provide an external application to display and process the object within the browser-controlled window because it would have improved the system by reducing clustering of the display and aiding the reader comprehension of the hypermedia document.

TRAVERSE

The above rejection is respectfully traversed. The traverse is organized into two parts. Part I establishes that the features recited in claim 1 are not disclosed in the cited references. Part II establishes that: 1) there is no suggestion in the prior art that would make the claimed invention obvious; 2) the exercise of invention would be required by one skilled in the art, apprised of the teachings of the cited references, to make the claimed combination; 3) the commercial success of products, developed subsequent to filing the present application, incorporating the claimed features establishes that the combination was not obvious at the time of invention.

Part I

A. Mosaic does not disclose the recited embed text format that is parsed by the browser to initiate processing to automatically invoke an executable application external to the hypermedia document.

As described above, in Mosaic the URL is an address to an object. A URL anchor in Mosaic is not an embed text format that is parsed by the browser to initiate invocation of an executable application external to the document. Rather, when the anchor is activated, by the user interactively selecting the anchor, the browser retrieves the object and, if the object is another hypermedia document, replaces the first document with the second document. If the object has a file name associated with a helper application the application is launched and the object is viewed and/or edited in a separate window controlled by the helper application.

Accordingly, the external application is not automatically invoked as a result of the browser parsing the hypermedia document text, as required by the claim, but rather it is invoked by an interactive command given by the user, namely interactively selecting the URL anchor.

Further, a display window is not created in the first hypermedia document at the location in the document of the embed text format as required by the claim.

B. Khoyi et al.

This reference discloses an operating system based on capabilities similar to OLE, as used for example in Windows 95, as described at page 5, lines 27-33 of the present application. Source data in a form not displayable by a document manger is displayed in a window by an object manager which can display the source data. In the example described above, a picture is displayed in a document by the Khoyi system.

However, the display of the source data in the destination object is non-interactive. As stated above, the source data displayed in the destination document is delimited to be recognizable to the user. If the data is to be processed it must be interactively selected by the user and the source object manager invoked to process the data in a separate window. Depending on the link, the updated data will be displayed when the source object manager is displayed.

Thus, the presentation of the source data in the destination document is non-interactive. In the claimed invention, the external object is displayed in a window in the document and interactively processed using the external executable application. As set forth in the attached Doyle declaration, the claimed invention "lifted the glass" of the browser display to allow interactive control of document elements while being displayed in the browser controlled window. The Applicants' claimed invention allowed these elements to become "active" or "live" without requiring external programs to be first launched by the user's interactive commands. Furthermore, the Applicants' invention accomplished this functionality without requiring Khoyi-like capabilities in the operating system, making it practical for widespread use on a variety of operating systems. The claimed invention is a quantum leap over the disclosure of Khoyi or other OLE-type operating systems.

C. Hansen et al.

This reference discloses a programmer's source-code editing environment for visual languages that allows sub-elements

of the program being edited to be displayed in hierarchically-embedded subdocument windows. The Hansen system does not teach embedding, within the source-code document, of any executable applications which are external to the source-code document being edited.

Part II

1) There is no suggestion in the prior art that would make the claimed invention obvious.

Applicants' invention solves a different problem than each of the references, and such different problem is recited in the claims. In re Wright, 6 USPQ 2d 1959 (1988).

The Applicants' invention allows the hypermedia document to act as a coordinator and deployment mechanism, as well as a container, for any arbitrary number of external **interactive** data/application objects, while hiding the details of such coordination and deployment from the document's reader as the reader uses the various data/application objects. This allows the hypermedia document to act as a platform for entirely new kinds of applications that could not have been possible before the invention.

Because most of the functionality exposed to the reader is defined most directly by the hypermedia document, rather than any specific computer operating system, document-based applications using the invention tend to have the same look and feel to the reader, regardless of what type of computer or operating system is being used to run the browser application.

Additionally, because the embed text formats in the document cause the browser to automatically invoke the external application the document, the hypermedia document itself, and by implication the author of that document, directly control the extension of the functions of the browser.

Mosaic displays links, embedded in a first hypermedia document, and retrieves information identified by a link when a user activates the link. The retrieved information either replaces the first hypermedia document, or is displayed in a

separate window than the window displaying the first hypermedia document. Mosaic has the capability of invoking external applications to open a new window to display file types that cannot be displayed by Mosaic (helper apps).

Mosaic was a significant advance that made the WWW accessible and gave document authors a powerful tool to provide references external objects anywhere on the WWW.

The Khoyi system is an object oriented operating system that allows different types of objects to be integrated and object managers (applications) developed to manipulate and display the objects. The operating system coordinates interaction between the object managers, but has no knowledge of how any particular object is handled. An important advantage of the system is that new types of object-handling capability can be added without modifying the existing code. Thus the system is extensible.

OLE-style linking of objects is enabled by allowing different types of object data to be displayed in one document. The object managers for the different types of data are coordinated by the operating system so that each type of displayed data is rendered by its associated object manager. The actual linking operations are coordinated by the operating system.

Khoyi is an advance that allows applications to display different types of data and work together seamlessly. By the use of links, when an object is displayed the linked data is displayed in its latest format.

However, as described in detail above, there is no provision, suggestion, or motivation in Khoyi to provide for interactive processing of source data actually being displayed in a destination document. There is no need for such a feature in Khoyi because the different object managers can be invoked by the user at any time and source data can be processed in a window opened for that purpose.

The Examiner reasons that it would have been obvious to combine the teaching of Khoyi with Mosaic to improve Mosaic by providing an open ended mechanism for integrating new

object/applications and new types of information without recompiling Mosaic.

It is respectfully asserted that this reasoning is incorrect for the following reasons. In Khoyi the operating system is extensible in the sense that new types of data may be displayed, for example, in a destination document without recoding the destination document display application. However, the functionality of the document display application is not extended. There is still no capability for interactively processing the source data within the destination document window while the destination document is displayed within the same window.

Thus, there is no suggestion in Khoyi of modifying Mosaic so that an external application, by analogy to Khoyi the source document manager, is invoked to display and interactively process the object within the document window while the document is displayed by Mosaic in the same window.

The Examiner relies upon Hansen's teaching that, in a programmer's source code editor, a programmer should have the power to organize various fragments of the program for perusal by a reader, in order to aid the reader in comprehending the program [p256 col.1 1st paragraph]. The Examiner then states: "Hence, it would have been obvious for one of ordinary skill in the art to provide external application to display and process the object within the browser-controlled window because it would have improved the system by reducing the display and aiding the reader comprehension of the hypermedia document."

However, there is no suggestion in Hansen that an executable application external to the programming editor environment should be displayed and interactively processed within a document window. There is no discussion of external application programs at all. The fact that Hansen teaches that it is good to graphically organize the sub-elements of a document for better comprehension would not suggest to the person of skill in the art to combine parts of one reference, Khoyi's object data processing system, with another reference in order to meet Applicants' novel claimed combination.

In view of the above, it is concluded that the cited references neither explicitly nor implicitly suggest the claimed combination to a person of ordinary skill in the art.

2) The exercise of invention would be required by one skilled in the art, apprised of the teachings of the cited references, to make the claimed combination.

The combination of Mosaic and Khoyi would require either that a) Mosaic be modified in view of the teachings of the present invention or b) Khoyi be modified in view of the teachings of Mosaic to make the claimed combination. The only two possible ways to attempt to combine Mosaic and Khoyi include either adding the functionality of Khoyi to Mosaic or implementing the functionality of Mosaic within the Khoyi operating system.

**Turning first to modifying Mosaic,** to combine these references as proposed would have required novel and unobvious inventive steps. One must first consider that Mosaic is an application program which operates on any one of three operating systems: UNIX, Windows, and the Mac OS. Much of the current commercial success of the World Wide Web is due to this cross-platform compatibility of Web browsers. The system taught by Khoyi, on the other hand, is a *fully-independent and proprietary operating system*. As is stated in section 1.5 of Khoyi, "The operating system of the present invention differs from the traditional operating system in that, firstly, the actual functions and services performed by the operating system are reduced to the minimum ... functions and services which would normally be performed by an operating system, together with many functions and operations which would normally be performed by the applications programs themselves, are performed by libraries of routines [pack routines]. Examples of services and functions performed by pack routines include, but are not limited to, input/output operations, graphics/text and display operations, file access and management operations, and mathematical operations."



As is shown in the enclosed Doyle declaration, the only **presently-known** way to combine an operating system with an application program which runs on a different operating system is through the use of what is presently-known as a "**virtual machine**." This concept of a virtual machine was invented by a team of engineers working at Sun Microsystems Inc., in 1995, and was first exposed to the world in the form of a Web-browser plug-in called the "Java Virtual Machine." ***This virtual machine technology was not known at the time of the Applicants' invention.*** Therefore, any combination of the references attempted at the time of the Applicants' invention would have yielded an inoperable result.

As is further shown in the enclosed Doyle declaration, the Java Virtual Machine concept has, together with Web browser plug-ins and applets, been hailed by the Industry as inventive and innovative. Since use of such an inventive step would be required in order to combine Mosaic and Khoyi, it follows that such a combination would have required novel and nonobvious combinative steps not taught in the prior art.

Therefore, adding the functionality of Khoyi to Mosaic would have been impossible at the time of the Applicants' invention without the creation of new novel and nonobvious technology. Even if such a combination had been possible and operable at the time of the Applicants' invention, Mosaic would have had to be significantly modified in a number of additional complex and nonobvious ways to achieve the combination.

First of all, Mosaic would have had to have been modified to incorporate the elements of a "virtual machine," that is, a program, which runs on a first computer and operating system, that emulates all of the necessary operations and resources of a second computer, and is under the control of an application executing on the first operating system. Such operations and resources include, in part, the machine instructions, graphics and I/O devices, and file system of the second computer. This "virtual" second computer would have to possess all of the characteristics necessary to allow the execution of the operating system taught by Khoyi. The Khoyi

system would then have to be integrated with Mosaic so as to execute on this virtual machine under the control of Mosaic.

Next, some sort of data interchange interface would have to be constructed between Mosaic and the Khoyi virtual machine to allow Khoyi's "packs" to create data structures that could be transferred to Mosaic, and for messages created by Mosaic to be transferred to Khoyi.

Mosaic would then have to be modified to allow data object components of the document to be "linked" to Khoyi's applications which can process the data. These links would be defined by an external link table, and the linking relationship would not be affected by the text of the document.

These links would be distinguished from the HTML anchor links defined in the hypermedia document, which would require incorporating two incompatible linking systems to be maintained by the system. Mosaic teaches that a major advantage of the HTML document format is that all links should be defined by the document text. This teaches away from combining the two systems in the proposed way, since the result would be awkward, overly complex and difficult to maintain.

In order for Mosaic to display the results of any computations that may have been made on the data object during viewing of the hypermedia document, Mosaic would have to be modified to allow the Khoyi virtual machine to write data directly to the Mosaic document data structure.

Mosaic would also have to be modified to allow it to be caused to re-render the document window in response to any change in the object imposed by the external program. Such re-rendering would require synchronization messages to be continuously exchanged between the browser application and the external program. This would involve the creation of some kind of message event loop that would wait for re-rendering messages to come from Khoyi's external program.

Similarly Khoyi would have to be modified to synchronize with Mosaic so that changes to the data would cause to external program to send a message to Mosaic to cause it to re-render its display. *Of course, even after doing all of the*

*above, the external applications still could not be interacted with from within the Mosaic document, as required by the claimed invention, since Khoyi must launch any external application into a separate window before the reader can interactively control it.*

**Turning second to modifying Khoyi,** any implementation of the functionality of Mosaic in the Khoyi operating system would be inoperable, due to the incompatibilities between the two systems' linking systems described below, and due to the differences between the two systems in storing, locating and processing data objects. Furthermore, such a combination would be impractical and nonobvious since the resulting combination would not conform to popular hypermedia protocol standards and would not be operable on any of the three most popular operating systems in the industry: UNIX, Mac OS, and Microsoft Windows.

The two linking systems could not be combined into one, due to the architecture of Khoyi's object system. Since Khoyi does not represent links within the document itself, but rather uses a link table which is external to the document, some sort of mechanism would have to be created to allow such links to be fully defined by the document text itself. This modification would render Khoyi's object system inoperable, since Khoyi's entire application architecture depends upon the link tables being the source of all link definitions, and being accessible to all of the various programs that may have a need to operate on a given data object.

Furthermore, since a document in the Khoyi system does not allow the document author to explicitly define or control the definition of the link's internal details, the document itself cannot specify such details as the precise location of a data file on a remote network disk drive. The Web, on the other hand, employs a uniform resource locator (URL) construct to manage both link definition and object localization on networked systems, from within the Web document, under the precise control of the Web document author. The URL mechanism would be incompatible with the linking mechanism requirements imposed by the Khoyi operating system. Since the URL-based mechanism for linking and object management is one of the major requirements for a successful Web

browser, such an incompatibility would render the resulting system useless for its intended purpose. Furthermore, even if the above combination was operable, the external applications still could not be interacted with from within the hypermedia document, as required by the claimed invention, since both Mosaic and Khoyi must launch any external application into a separate window before the reader can interactively control it.

None of the modifications listed above are taught in the prior art. **Nor are there any suggestions in the prior art that the references should be combined.** To combine Mosaic and Khoyi in the manner suggested would require a multiplicity of separate, novel, inventive and awkward combinative steps that are too complicated to be considered obvious.

Combining Hansen with any combination of Mosaic and Khoyi, while perhaps possible, would produce features that are irrelevant to the present application. Such a combination would involve modifying the hypermedia document data structure to allow multiple hierarchical subdocument windows to be contained within a parent document. This would involve substantial modifications to the Mosaic document rendering engine, as well as the development of a new version of the HTML document definition protocol to allow definition of hierarchical relationships within subdocument elements. Such a protocol would be exceedingly complex and would likely be incompatible with existing HTML standards. Combining Hansen with Khoyi would involve novel and unobvious steps similar to those described above for combining Mosaic and Khoyi. Furthermore, the features that would result from the combination of Hansen and Mosaic are irrelevant to the Applicants' claimed invention, since, even after the combination, they would not show external executable applications being embedded within Hansen's documents.

Thus, the Applicants submit that combining Mosaic, Khoyi and Hansen would require novel and unobvious inventive steps, and that the Applicants' invention is therefore novel and unobvious.

3) The commercial success of products, developed subsequent to filing the present application, incorporating the claimed features establishes that the combination was not obvious at the time of invention

As is shown in the enclosed Doyle declaration, several major competitors have incorporated the features of Applicants' invention into products, rather than to use the techniques of the prior art. The most notable of these products include the Navigator Web browser application from Netscape corporation, the ActiveX applet system from Microsoft corporation, and the Java Web applet system from Sun Microsystems corporation. The enclosed Doyle declaration further shows that the success of these products is directly attributable to the features of the claimed invention which each of these products incorporate, including an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed text format's location within the hypermedia document being displayed in the browser-controlled window.

Some of these competitors have made laudatory statements about the elements of the Applicants' invention which are incorporated into their respective products, and have characterized those features as being a significant advance over prior art techniques.

**Products incorporating the features of the invention have attained extensive commercial success**

It is well known that, in the 1966 case of Graham v. John Deere, the U.S. Supreme Court decreed that Section 103 is to be interpreted by taking into consideration "**secondary and objective factors such as commercial success, long-felt but unsolved need, and failure of others.**"

As is shown in the enclosed Doyle declaration, there is universal acceptance within the computer software industry that the aforementioned products incorporating claimed features of the Applicants' invention have attained an extremely high degree of commercial success. Java, Navigator and ActiveX (all products incorporating features of the invention) represent among the most popular current technology in the computer industry for new application development.

The vast degree of commercial success that products incorporating features of the Applicants' invention have attained argues strongly **against** obviousness of the invention, and strongly **for** the patentability of the Applicants' claims.

**The products incorporating features of the invention by others have been given many awards and have received considerable recognition in professional publications.**

As is shown below, and in the enclosed Doyle declaration, Netscape, ActiveX and Java, all incorporating features of the Applicants' invention, have each been lauded as among the most innovative technologies to appear in the computer industry in recent years.

Some examples are:

The 1996 Discover Awards for Technical Innovation -- to Java and Navigator

PC Magazine 1995 Technology of the Year Awards -- To Java and Navigator

New Media Magazine 1996 Hyper Awards -- to Java and Navigator

New Media Magazine 1997 Hyper Awards -- to Active

As is evidenced in the enclosed Doyle declaration, this acclaim is due to the innovative nature of features of the claimed invention incorporated into those products and argues strongly against the obviousness of the Applicants' claims and argues strongly for the patentability of those claims.

**Accordingly, since the Applicants' Claim 1 defines novel and unobvious structure that provides new and unexpected results as described above, and also because of the other numerous arguments**

***against the obviousness of Applicants' invention, made above, Applicants submit that Claim 1 is clearly patentable.***

THE REJECTION OF CLAIMS 1-5

**The Dependent Claims are A Fortiori and Independently Patentable over Mosaic, Khoyi and Hansen**

Amended dependent claims 2 to 5 incorporate all the subject matter of Claim 1 and are therefore patentable for the same reasons as claim 1. Further, claims 2-5 add additional subject matter which makes them further and independently patentable over these references.

Claims 2-5 are rejected under 35 U.S.C. §103 as being unpatentable over Applicant's disclosed prior art, Khoyi, Hansen, and further in view of Moran "Tele-Nicer-Dicer: A new tool for the visualization of large volumetric data".

**The rejection of Claim 2 on Mosaic, Khoyi, Hansen and Moran is overcome**

Applicants' Claim 2 recites the additional step over Claim 1 of interactively controlling the controllable application on the client workstation via inter-process communications between the browser and said controllable application.

The disclosure of Mosaic, Khoyi, and Hansen has been described above. The reference Moran discloses a tool for interactive visualization of large, rectilinear volumetric data called Tele-Nicer-Slice-Dicer (TNSD). TNSD is based on client-server design where the client-side process is an extended version of a stand-alone visualization tool and the server process runs on a high-performance system where the data are located.

The client-side process describes data sets by text fields. Each data set description is used as a command which is sent to the server when a volume from the corresponding data set is requested. The use of a remote server is transparent.

The Examiner states that it would have been obvious to utilize the Moran application as an external application ("Viewer") in the prior art system as modified because it would

have improved the system by enabling the client station access to resources on higher performance servers to have interactive visualization of large data set capability.

Neither Mosaic, nor Khoyi, nor Hansen shows an executable application which is external to a document being displayed and interactively processed within that document's display window, nor do they show such an application where said executable application is interactively controlled on said client workstation by interprocess communications between the external application and the browser. This feature produces surprising and unexpected results over the prior art, since it allows the reader to perform all necessary interactive functions with external applications without directing his or her attention away from the hypermedia document. Additional surprising and unexpected results are yielded by the fact that the hypermedia browser application can have its functionality extended without making any changes to the hypermedia browser's object code. Further, surprising and unexpected results come from the ability of the document author to design interactive hypermedia document content that displays a similar look and feel to the reader, regardless of what the underlying operating system or computer platform the browser program is being executed upon.

The amendments to these claims have made the Moran reference irrelevant to the case, since Moran teaches a remotely-networked application being controlled via communications over a network, not an embedded (in a hypermedia document) interactive external application on the client workstation being controlled via inter-process communications between the document browser application and the external application.

Even if Moran was still in some way relevant, and even if the proposed combination was possible, was suggested by the prior art, and showed the features of the invention, all of which the above arguments for Claim 1 clearly show is not the case, the fact that a large number of references (more than 3) must be combined to meet the invention is further evidence of unobviousness



**The rejection of Claim 5 on Mosaic, Khoyi, Hansen and Moran is overcome**

Applicants' Claim 5 shows the additional steps over Claim 2 of communications to interactively control said controllable application which continue to be exchanged between said controllable application and said hypermedia browser even after said controllable application program has been launched.

None of the cited references show this feature. This feature produces the additional unexpected and surprising results over the prior art of allowing the browser application and the external application to precisely coordinate their activity, such as caching of the external application and shutting down its execution when no longer needed, entirely under the control of the browser application, in a manner that is transparent to the user. This drastically clarifies and simplifies the user's use of the hypermedia document and its related embedded applications.

**The rejection of Claim 3 on Mosaic, Khoyi, Hansen and Moran is overcome**

Applicants' Claim 3 shows the additional steps over Claim 5 of "additional instructions for controlling said controllable application reside on said network server, wherein said step of interactively controlling said controllable application includes the following sub-steps: issuing, from said client workstation, one or more commands to the network server; executing, on said 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."

None of the cited references show this feature. This feature leads to the additional surprising an unexpected results over the prior art of allowing the user to employ the hypermedia document as an interface to control and/or edit data objects which reside on the network server, remotely, from the client workstation. One of many possible uses of this feature is to

allow the user to make modifications to the original data object, which may remain in place on the network server, and which is referenced in the hypermedia document, so that others viewing the hypermedia document in the future from other client workstations will see those modifications.

**The rejection of Claim 4 on Mosaic, Khoyi, Hansen and Moran is overcome**

Applicants' Claim 4 shows the additional steps over Claim 3 "wherein said additional instructions for controlling said controllable application reside on said client workstation."

None of the claimed references show this feature. This feature produces the additional surprising and unexpected results of enabling a client & server system to be self-contained on the client workstation.

**The rejection of claims 44-48 is overcome**

Claims 44-48 are apparatus of the same scope as claims 1-5 and are thus allowable for the reasons recited above.

***Accordingly Applicants submit that the dependent claims are a fortiori and independently patentable and should also be allowed.***

**Conclusion**

For all of the above reasons, Applicants submit that the claims are now in proper form, and that the claims all define patentability over the prior art. Therefore they submit that this application is now in condition for allowance, which action they respectfully solicit.

**Conditional Request for Constructive Assistance**

Applicants have amended the claims of this application so that they are proper, definite, and define novel structure which is also unobvious. If, for any reason, this application is not believed to be in full condition for allowance, Applicants respectfully request the constructive assistance and suggestions of the Examiner pursuant to M.P.E.P Section 706.03(d) and Section

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 27

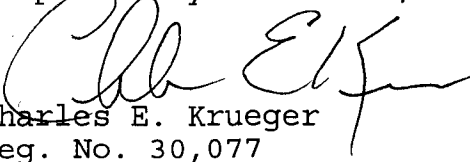
PATENT

707.07(j) in order that the undersigned can place this application in allowable condition as soon as possible and without the need for further proceedings.

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 (415) 576-0200.

Respectfully submitted,

  
Charles E. Krueger  
Reg. No. 30,077

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, California 94111-3834  
(415) 576-0200  
Fax (415) 576-0300  
CEK:db

i:\cek\share\02307i\553\june2.amd

Amendment

2347

TOWNSEND and TOWNSEND and CREW LLP

Two Embarcadero Center, 8th Floor  
San Francisco, CA 94111-3834  
(415) 576-0200

70826 U.S. PTO



06/05/97

Atty. Docket No. 023071-553

Date June 2, 1997

In re application of MICHAEL D. DOYLE et al.

I hereby certify that this is being deposited with the United States Postal Service as first class mail in an envelope addressed to:

Appl. No. 08/324,443  
Filed 10/17/94  
Group Art Unit 2317

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

For EMBEDDED PROGRAM OBJECTS IN  
DISTRIBUTED HYPERMEDIA SYSTEMS

Date: June 2, 1997

*Charles E. Krueger*

THE ASSISTANT COMMISSIONER FOR PATENTS  
Washington, D.C. 20231

RECEIVED  
JUN 17 97  
GROUP 2600

Sir:

Transmitted herewith is an amendment in the above-identified application.

- Enclosed is a petition to extend time to respond.
- Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
- A verified statement to establish small entity status under 37 CFR 1.9 and 1.27 is enclosed.
- A declaration by co-inventor Michael D. Doyle is submitted herewith.
- A change of address is submitted herewith.

If any extension of time is needed, then this response should be considered a petition therefor.

The filing fee has been calculated as shown below:

(Col. 1)		(Col. 2)		(Col. 3)	SMALL ENTITY		OTHER THAN A SMALL ENTITY		
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NO. PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE	ADDIT. FEE	OR	RATE	ADDIT. FEE
TOTAL	*	MINUS	**	=	x11 =	\$		x22 =	\$
INDEP.	*	MINUS	***	=	x40 =	\$		x80 =	\$
[ ] FIRST PRESENTATION OF MULTIPLE DEP. CLAIM					+130 =	\$		+260 =	\$
					TOTAL	\$	OR	TOTAL	\$

- \* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
- \*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
- \*\*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space. The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

No fee is due.

Please charge Deposit Account No. 20-1430 as follows:

- Claims fee \$ \_\_\_\_\_
- Any additional fees associated with this paper or during the pendency of this application.

no extra copies of this sheet are enclosed.

TOWNSEND and TOWNSEND and CREW LLP

*Charles E. Krueger*  
Charles E. Krueger / Reg. No.: 30,077  
Attorneys for Applicant

#14

ATTACHMENT A

Netscape<sup>TM</sup>

UNLEASHED

*Dick Oliver, et. al.*



201 West 103rd Street  
Indianapolis, IN 46290

Netscape is the most important and basic tool you will need to explore the Internet. However, there are other small tools, called Helper Applications, that you use in conjunction with Netscape to enhance your surfing. These Helper Applications open up a vast array of multimedia files for your viewing and/or listening pleasure. This chapter walks you through what types of files you'll encounter on the Internet, what Helper Applications you'll need in order to view those files, and how these Helper Applications work with Netscape. Many of the Helper Applications are available on the *Netscape Unleashed* CD-ROM; these Helper Applications are tagged in the text by the CD-ROM icon. For all the other Helper Applications you'll find the information necessary to download them immediately from the Net.

## Overview of Helper Applications

Netscape Navigator is an excellent tool for browsing the Internet, but it is also very limited in its functionality for viewing multimedia files. Netscape, without any assistance, has the capability to view only two major types of multimedia files: GIF and JPEG graphics files. Netscape Navigator does not have the capability to view any of the multitude of other kinds of sound, video, and graphics files available on the Internet without the assistance of Helper Applications and Plug-Ins. This chapter explains how to use Helper Applications effectively with Netscape. Chapter 15, "Netscape Navigator Plug-Ins," will explore the closely related topic of Plug-Ins for Netscape Navigator.

### The Helper Applications Concept

Helper Applications, or Helper Apps, are applications that extend your ability to view and manipulate multimedia and other types of files while browsing the Web using Netscape. A Helper App can be any application you can use to view, listen to, and/or manipulate a file you encounter on the Internet. Netscape associates Helper Apps with the file types that you set up in the program. When the browser comes across a file with a file extension that has an associated Helper App, Netscape automatically launches the Helper App after the selected file has been downloaded to your computer. From there, you can use the full functionality of the Helper Application to use and work with the file.

The World Wide Web contains a huge amount and variety of information. The types of files this information is contained in often can also be just as varied. The idea of Helper Apps is to allow you to add capabilities for viewing files that experience and usage will show you are necessary for your enjoyment of the Internet. Almost any application can be utilized as a Helper App in conjunction with Netscape. For instance, Netscape comes with a small, preconfigured Helper App for playing popular types of sound files you will run into while browsing the Web. It is a small, dedicated program with no real manipulation capabilities—it will simply play back the sound file.

On the other hand, it is possible to make a large, complex application like Microsoft Excel work as a Helper App with Netscape. An example is when you go to a Web page that has pointers

to an Excel file containing financial data about a company whose stock you are considering purchasing. When you click on that Excel file and it is downloaded, Netscape launches Excel. Then you can use the full potential of the application to work with the information—moving it around, creating charts, and so on—and make your decision.

## File Types

The first step in working with Netscape and Helper Apps is understanding file types and how to identify them. You may or may not be familiar with the types of files you will encounter while browsing the Web. Many of the graphics files, such as JPEG and GIF files, are common file types that most computer users will have had experience with, regardless of whether or not they have ever used the Internet.

However, more and more multimedia file types are appearing on the Internet for the first time and are useful only when used online. A good example of this kind of file is RealAudio files, which are used to deliver real time audio over the Internet using Progressive Network's RealAudio Player. RealAudio will be covered at greater length later in this chapter. Every Helper App has a specific file type that it is programmed to access and/or manipulate. Some Helper Apps can be used like a multimedia Swiss army knife to access a wide variety of file types.

The first step in being able to effectively set up and use Helper Apps is to identify what file types you want or need to access using Netscape. A file type can normally be readily ascertained by referring to the two- or three-digit file extension following the period in the filename. Table 14.1 provides a brief overview and quick reference of the most popular file types that you are likely to come across in your travels with Netscape.

**Table 14.1. Helper App quick reference.**

<i>File extension</i>	<i>File type</i>	<i>Popular Helper Apps</i>
WAV	Windows Sound	Windows Media Player WHAM WPLANY
AU	Sun/NeXT Sound	Netscape Audio Player WHAM WPLANY
AIF, SND	Mac/SGI Sound	Netscape Audio Player WHAM WPLANY
MP2	MPEG Audio	Xing Player
RA, RAM	RealAudio	RealAudio Player

*continues*

## Part III

Table 14.1. continued

<i>File extension</i>	<i>File type</i>	<i>Popular Helper Apps</i>
MID	Midi Sound	Windows Media Player Midi Gate
AVI	Windows Video	Windows Media Player
MPG	MPEG Video	VMPEG
MOV, QT	QuickTime Video	QuickTime Player

Often times when browsing the Web with Netscape, the filename may not appear in the hyperlink that you are selecting on a Web page. For example, there is a Web page featuring numerous clips from the popular animated series "MTV's Beavis and Butt-head." As you can see from Figure 14.1, none of the filenames of the clips are visible on the Web page. However, by placing the pointer over any of the links, Netscape indicates the URL address of the linked document in the status bar at the bottom of the browser. At the end of the URL address is the name of the linked file with the corresponding file extension. In the Beavis & Butt-head example, you can see that the file extension is MOV and thus it is a QuickTime movie file. You can then determine whether or not you have the proper Helper App or which Helper App you will need to get for viewing the file.

FIGURE 14.1.

The status bar at the bottom reveals the selection is a MOV file.



If you come across an unknown or unfamiliar file type that you are interested in accessing while using Netscape, there are plenty of places online to get help. You shouldn't be surprised if you come across a file type that is completely foreign to you. Many companies are developing new



multimedia software products every day to increase the speed and usefulness of the Web. These products often result in new, proprietary file types. A good place to start looking for assistance is in the Usenet newsgroup `comp.infosystems.www.browsers.ms-windows` or `alt.winsock`. Posting a question with the name of the file you're having trouble with and its URL will usually get you an answer within a few hours.

After you've identified the file type, the next step is to locate a Helper App that is suitable to your needs and circumstances.

## Locating Helper Applications on the Internet

Because Helper Apps work with Netscape, it shouldn't surprise you to learn that all of the most popular and useful Helper Apps are available online. The vast majority of Helper Apps are either freeware or shareware, so you shouldn't hesitate to sample as many different apps as you can to find the ones that you like the best and are most comfortable using. Some users like big, powerful Helper Apps with lots of options for manipulating multimedia files; others prefer small, simple apps that do one thing and do it well. And, of course, like everything else on the Internet, change is a constant, so you may want to keep your eyes open for the new and improved Helper Apps that may replace that old favorite on your hard drive.

There are thousands of FTP sites on the Internet that store the Helper Apps that you will be seeking to help you more fully enjoy Netscape. Some of the Helper Apps, like the RealAudio Player discussed earlier, are only available from the software manufacturer's FTP or Web site or those authorized by them to distribute the app. As you go through each of the various types of popular Helper Apps in this chapter, I will give you pointers to URL addresses for each of the Helper Apps covered. In addition, the accompanying CD-ROM contains a Helper App Web page with links to all these sites so that you will be able to access each Helper App's FTP site after you're online. When you're looking for a Helper App for a particular file type, you need to know some basics that will make your search for the potential Helper Apps much easier.

The Internet contains several FTP sites that are huge depositories for Windows software. Most of the Helper Apps you will be using with Netscape are available on public FTP sites and are either shareware or freeware. You should always remember to view the README file that accompanies most apps to determine what type it is: freeware or shareware. If it's shareware, you need to determine what your limitations are for using the program and how much it costs to register if you decide you want to keep using the app. One of the biggest FTP sites for Windows applications can be found at CICA at Indiana University: `ftp://ftp.cica.indiana.edu`. This site contains separate directories for Windows 3.1 and Windows 95 files. Each directory then has subdirectories for different types of apps, such as graphics, sound, and so on. You should know from the file type which directory will be most likely to reveal useful Helper Apps. Also, there are some good Helper Apps that have not yet been ported to Windows 95, so be sure to look in both Windows directories on the site.



# 14

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

on June 2, 1997

TOWNSEND and TOWNSEND and CREW LLP

By [Signature]

PATENT

Attorney Docket No. 02307I-553

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: )  
MICHAEL D. DOYLE et al. )  
Application No.: 08/324,443 )  
Filed: 10/17/94 )  
For: EMBEDDED PROGRAM OBJECTS IN )  
DISTRIBUTED )  
HYPERMEDIA )  
SYSTEMS )

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

I, MICHAEL D. DOYLE, hereby declare that:

1. I am a co-inventor of the subject matter disclosed and claimed in U.S. Patent Application No. 08/324,443.
2. I am an expert in the subject matter claimed in the referenced patent application as evidenced by the following curriculum vitae:

**MICHAEL DAVID DOYLE**

Born: April 27, 1959, in Chicago, Illinois

**Education**

- Ph.D. in Anatomy and Cell Biology, 1991, Department of Cell and Structural Biology, School of Life Sciences, University of Illinois at Urbana-Champaign
- Bachelor of Science, with honors, 1983, Department of Biocommunication Arts, University of Illinois at Chicago, Health Sciences Center

**Employment**

- Chairman and CEO, Eolas Technologies Inc., Chicago, IL  
(<http://www.eolas.com>, email: [mike@doyles.com](mailto:mike@doyles.com))
- Adjunct Professor, 1996-present, Department of Computer Science, DePaul University of Chicago, Chicago, IL
- Asst. Adjunct Professor, 1994-present Department of Anatomy, School of Medicine, Univ. of Calif., San Francisco
- Director, Center for Knowledge Management, 1993 - 1994, (the UCSF Academic Computing and Informatics Research Center), University of California, San Francisco
- Director, UIC Biomedical Visualization Laboratory, 1990 - 1993, Department of Biomedical Visualization, College of Associated Health Professions, University of Illinois at Chicago
- Assistant Professor, 1989 - 1993, Department of Biomedical Visualization, College of Associated Health Professions, University of Illinois at Chicago

**Selected Professional Societies**

Sigma Xi; Phi Kappa Phi; Mensa; International Society of Lymphology; Healthcare Information and Management Systems Society; Association for Computing Machinery, UIUC Chapter: Chairman and Founder, Special Interest Group for Biological

Computing, (SIGBIO), member ACM national; IEEE Engineering in Medicine & Biology Society; SPIE - The International Society for Optical Engineering

**Selected Honors and Awards**

Procter and Gamble Fellowship in Cell Biology, 1988-1989;  
National Research Service Award, Cell and Molecular Biology Training Program of the National Institutes of Health, University of Illinois at Urbana-Champaign, 1987-88; Nominated for the Charles A. Dana Award for Pioneering achievements in higher education, 1988; Francis and Harlie M. Clark Research Support Award, University of Illinois, 1987; University of Illinois Research Fellowship, 1987

**Selected Committee and Board Memberships**

- NIH Special Emphasis Panel: Computer Applications in Biology and Medicine, Division of Research Grants, National Institutes of Health, 1993-present
- Scientific Advisory Board: The Visible Human Project, National Library of Medicine, National Institutes of Health, 1991-1994
- Scientific Advisory Board: Center for Human Developmental Anatomy, National Museum of Health and Medicine, Armed Forces Institute of Pathology, 1992-1995
- Heads of Information Systems Committee, University of California System, 1993-1994
- Executive Committee: Integrated Advanced Information Management Systems, University of California, San Francisco, 1993-1994
- Organizational Committee for the First NIH Workshop on Osteoporosis Research in Dental Science, National Institute for Dental Research, 1991
- Curriculum Committee, College of Medicine, University of Illinois at Urbana-Champaign, 1986-1988

**Selected Publications (of 54)**

- Doyle, M.D.: *The Visible Embryo: Embedded Program Objects for Knowledge Access, Creation, and Management through the World Wide Web*, Computerized Medical Imaging and Graphics, 20/6(1996)
- Williams, B.S., and M.D. Doyle: *An Internet Atlas of Mouse Development*, Computerized Medical Imaging and Graphics, 20/6: 433-447 (1996)
- Doyle, M.D, C.S. Ang and D.C. Martin: *Proposing a Standard Web API*, lead article in Dr. Dobb's Journal, February (1996)
- Doyle, M.D., C.S. Ang, and D.C. Martin: *Embedding interactive external program objects within open-distributed-hypermedia documents*, High Speed Networking and Multimedia '94, SPIE Press (1995)
- Ang, C.S., D.C. Martin and M.D. Doyle: *Integrated Control of Distributed Volume Visualization Through the World Wide Web*. Proc. Visualization '94, IEEE Press (1994)
- Doyle, M.D, C. Ang, R. Raju, G. Klein, B.S. Williams, T. DeFanti, A. Goshtasby, R. Grzesczuk, and A. Noe: *Processing cross-sectional image data for reconstruction of human developmental anatomy from museum specimens*. SIGBIO Newsletter (The Jnl. of the ACM SIG for Biological Computing), 13/1 (1993)
- Carlbom, I., W.M. Hsu, G. Klinker, R. Szelski, K. Waters, M.D. Doyle, J. Gettys, K.M. Harris, T.M. Levergood, R. Palmer, L. Palmer, M. Picart, D. Terzopoulos, D. Tonnessen, M. Vannier, and G. Wallace: *Modeling and Analysis of Empirical Data in Collaborative Environments*, Communications of the ACM, 33/6, 75-84 (1992)
- Doyle, M.D.: *The MetaMAP Process: A New Approach to the Creation of Object-oriented Image Databases for Medical Education*, Proc. 13th Annual International Conference of the IEEE Eng. in Medicine and Biology Society, IEEE Press (1991)

- Doyle, M.D.: *The Use of Fractal Analysis in the Screening of Medical/Dental X-ray and Tomographic Images for Early Signs of Osteoporosis*, Proceedings of the 13th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE Press (1991)
- Doyle, M.D.: *A New Method for Identifying Features of an Image on a Digital Video Display*, Biostereometrics Technology and Applications, SPIE Press (1991)
- Doyle, M.D.: *The Interactive Digital Video Interface Process*, "Spatial Displays and Spatial Instruments", NASA Conference Publication #10032 (1989)
- Doyle, M.D., P.J. O'Morchoe, V. Navas, and C.C.C. O'Morchoe: *A Combined Enzyme Histochemical and Computer Image Analysis Technique for the Identification of Intraorgan Lymphatic Vessels* in: *Progress in Lymphology-XI*, H. Partsch, editor, Elsevier Science Publishers (1988)

#### **Patents**

- Lead inventor of "*Embedded program objects in distributed hypermedia systems*," together with two additional co-inventors. The US patent is currently pending approval (assigned to the University of California, 1994). This patent is for the original development of World Wide Web plug-in and applet technology in 1993. It covers such currently popular technologies as Java, Navigator plug-ins, and Virtual Reality Markup Language (VRML). The first public demonstration of this work was given by Dr. Doyle's group at the second meeting of the Bay Area SIGWEB (Special Interest Group for the World Wide Web), at Xerox PARC, in November, 1993.
- Sole inventor: Awarded United States, Canadian and Australian patents for the development of what may have been the earliest hypermedia imagemap technology, entitled "*Method and Apparatus for Identifying Features of an Image on a Video Display*" (US,

#4,847,604, issued on July 11, 1989). Currently has related patent pending in Japan. In addition to its use for Web imagemap servers, this technology is also currently in use throughout the computer game industry as a method for optimizing collision detection algorithms.

**Selected Invited Colloquia (of 23)**

- *"Interactive Content on the Web: The Next Wave of Computing,"* keynote address at the first Internet@Chicago conference, sponsored by the Chicago Software Association, May, 1996
- *"The Embryology Metacenter: Biological applications of supercomputing in a distributed hypermedia environment."* sponsored by the Department of Anatomy and Cell Biology, School of Medicine, Univ. of Michigan, Ann Arbor, March, 1994
- *"The Visible Embryo Project: A National Resource for Biomedical Information Technology,"* presented at the Annual Meeting of the American Association for the Advancement of Science, San Francisco, CA, February, 1994
- *"Accessing the Knowledge Base of Medicine in a Networked Environment,"* UCSF Grand Rounds in Medical Education, University of California, San Francisco, October, 1993
- *"Televisualization for the Support of Research and Education in Developmental Anatomy,"* a course in the Telemedicine Seminar and Workshop, sponsored by the Armed Forces Institute of Pathology, American Registry of Pathology, and the International Academy of Telemedicine, May, 1993
- *"The Embryology Metacenter: A distributed computational resource 'center' for developmental anatomy,"* Second Annual Conference on Human Developmental Anatomy, National Museum of Health and Medicine, Washington D.C., December, 1992

3. In the course of my work in both academic research and in the software industry I have supervised and have been associated with designers and developers of software, and I have personal knowledge of the ordinary level of skill in the art relevant to the claimed invention.

4. In my opinion the combination of the Mosaic and the Khoyi and Hansen references would not make the subject matter of claim 1 obvious to a person of ordinary skill in the art. I base this opinion on my belief that, firstly, based on the state-of-the-art in software technology prior to October of 1994, the combination of Mosaic and Khoyi would have been impossible or nonfunctional without the invention of novel and nonobvious technology, and secondly, the combination of Mosaic and Khoyi, or of Hansen with these two other references, would not have shown the features of the applicants' invention.

The only two possible ways to combine Mosaic and Khoyi would have been to either a) modify Mosaic in view of Khoyi or b) modify Khoyi in view of Mosaic. The discussion below shows that either approach would have been infeasible.

a) Modifying Mosaic in view of Khoyi:

Mosaic is an application program that was available in 1994 for the UNIX, Mac and Windows platforms. The system disclosed by Khoyi was a fully-independent operating system, developed by Wang Laboratories for use on proprietary word processing systems. In 1994, there was no published work, that I know of, that taught any method which would allow an application program, such as a Web browser, developed for one operating system, to "embed" the functionality of a different operating system, such as Khoyi, while still maintaining compatibility with the original operating system that the Web browser was developed



for. Even in light of the technologies available today, in 1997, there is no obvious solution to such a problem.

If asked to attempt to accomplish such a combination today, a person with an ordinary level of skill in the relevant art would most likely attempt to employ a Java Virtual Machine (JVM), which would allow a simulated computer platform to be embedded within the Web browser. The JVM was first employed as a mechanism to insure cross-platform compatibility of applet programs embedded within Web pages. This technology was first demonstrated in 1995 by Sun Microsystems, and the virtual machine component of it is generally acknowledged to have been innovative and nonobvious (Appendix A, Reference #1). Although the bytecode interpreter element of the Java language derives from work done on Smalltalk in the 1970s (Appendix A, Reference #2), the virtual machine concept was absolutely new at the time of Java's release.

As Reference #2 of Appendix A states, "The final requirement is what has stymied many attempts at ubiquity in the past. If you base your system on any assumptions about what is "beneath" the run-time system, you lose. If you depend in any way on the computer or operating system below, you lose. **Java solves this problem by inventing an abstract computer of its own and running on that.**"

Prior to Java, simulated computers had already been developed, but the JVM within the HotJava Web browser was the first known instance of a simulated computer system being incorporated into an application program running on another computer system. This allowed the Web browser to control the execution of applications running on the virtual machine, and to exchange information with those applications.

Even though there is no evidence that the JVM is robust enough to support a complete operating system, such as disclosed by Khoyi, there is a possibility that such an implementation could be accomplished. Alternatively, a new virtual machine could possibly be designed specifically for this purpose.

Attempts are underway at companies such as IBM to develop so-called universal virtual machine technology that can run programs written in languages other than Java. As Reference #3 of Appendix A points out, "Once developed, a universal virtual machine would, in theory, allow developers to write an application in any language and run it on any system. But at this stage, the challenges associated with developing UVMs are immense."

Assuming that the JVM would suffice for the purposes of combining the references, a person with an ordinary level of skill in the relevant art would then attempt to use the Java programming language to recreate the functionality of the Khoyi operating system on top of the JVM platform. Assuming that it would be possible to replicate the Khoyi system in the Java language, It is not at all clear that the subsequent combination with Mosaic would be useable, since there are a number of incompatibilities between the fundamental architectures of World Wide Web document systems and the type of document linking and object management system described by Khoyi. These incompatibilities are described below, in the section on modifying Khoyi in view of Mosaic.

b) Modifying Khoyi in view of Mosaic:

In order to modify Khoyi in view of Mosaic, one would attempt to implement a new document type on the Khoyi operating system through the development of an object manager to process documents formatted in the Hypertext Markup Language (HTML). Such an implementation would have been nonfunctional due to fundamental incompatibilities between the document linking and object management architectures in the two systems.

These incompatibilities relate to the way that Web documents handle object location and retrieval, and hypermedia links between documents and objects. The Khoyi system uses operating system services in order to manage the definition and resolution of links between data objects. Each link has a unique

identifier, which is referenced in a document, for example, as a "link marker." The actual definition of the link referenced by any particular link marker is located in an operating system data structure called a "link table." The document itself does not allow the document author to explicitly define or control the definition of the link's internal details, such as the precise location of a data file on a disk drive. The Web, on the other hand, employs a uniform resource locator (URL) construct to manage both link definition and object localization on networked systems, from within the Web document, under the precise control of the Web document author. It appears that the URL mechanism would be incompatible with the linking mechanism requirements imposed by the Khoyi operating system. Since the HTML-based mechanism for linking and object management is one of the major requirements for a successful Web browser, such an incompatibility would likely render the resulting system useless for its intended purpose.

An attempt to actually perform the proposed combination of Mosaic and Khoyi would likely reveal further incompatibilities, but the obvious linking and object management problems are sufficient to conclude that the combination of these references is infeasible without the development of new and unobvious technology.

This opinion is further supported by statements made in the press by Microsoft Corp. concerning the difference between their Khoyi-like Object Linking and Embedding (OLE) technology and their ActiveX technology, first released in 1996, which allows the implementation within HTML documents of the features of the claimed invention, namely an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed tag's location within the hypermedia document being displayed in the

browser-controlled window. Although many believed ActiveX to be just "OLE for the Internet," the above discussion of the difficulty of combining Khoyi with Mosaic may partly explain the delay that Microsoft took in coming up with a competing technology to Sun's competing Java product. This is supported by Reference #14 from Appendix A (Web Techniques, 10/96), which recalls a conversation between Web Techniques' editor in chief and ActiveX product managers from Microsoft:

``[Web Techniques:]--So tell me about ActiveX. Isn't it just OLE for the Internet?'

`[Microsoft:]--Wrong,... ActiveX is a new API that, like OLE, is based on Microsoft's Component Object Model (COM). While OLE supports a compound document architecture for desktops, ActiveX is designed specifically to embed rich media objects within Web-based documents.'

This was part of a scene that recently played out in the offices of Web Techniques...Microsoft marketers arrived on our doorstep not for the standard dog-and-pony show, but specifically to 'debunk the myths' surrounding ActiveX technologies."

Of course, even if the combination of Mosaic and Khoyi had been possible and functional, it still would not have shown the features of the claimed invention, including an embed text format that is parsed by a hypermedia browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed text format's location within the hypermedia document being displayed in the browser-controlled window. Furthermore, the additional combination of Hansen with these references would not have yeided the features of the claimed invention either. Hansen discloses a programmer's source-code editing environment for visual languages that allows sub-elements

of the program being edited to be displayed in hierarchically-embedded subdocument windows. The Hansen system does not teach embedding, within the source-code document, of any executable applications which are external to the source-code document being edited. Therefore the Hansen reference would not add features relevant to the claimed invention, even if combined with Mosaic and Khoyi.

The claimed invention lifted the glass, so to speak, from the Web browser, making it possible to embed fully-interactive external applications within Web pages, thereby turning the browser into a *platform* for the development of *entirely new* kinds of applications.

5. Further, in my opinion secondary considerations, including, in part, commercial success of products incorporating features of the claimed invention and industry recognition of the innovative nature of these products, demonstrate that the claimed invention is not obvious over the cited references.

The three exemplary products which incorporate the features of the claimed invention include Netscape Navigator 2.0 (or newer versions), Java, from Sun Microsystems, and ActiveX, from Microsoft. One need only open the pages of any major business publication to see that these three products have been tremendously successful in the marketplace. Appendix A of this declaration presents a collection of excerpts from prestigious Industry publications which support the contention that the success of these products is directly attributable to the claimed features of the invention.

Approximately 12 to 18 months after the applicants initially demonstrated the first Web plug-in and applet technology to the founders of Netscape and engineers employed by Sun Microsystems in November and December of 1993, as described in reference #4 from Appendix A (Dr. Dobb's Journal, 2/96), both Netscape and Sun released software products that incorporated

features of the claimed invention, including an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed text format's location within the hypermedia document being displayed in the browser-controlled window. Sun released the Java applet programming environment and the HotJava applet-capable Web browser in May of 1995, and Netscape release version 2.0 of their Navigator Web browser, which incorporated both Java technology and a plug-in API, in October of 1995.

From Marc Andreessen's comments, quoted in Reference #5 of Appendix A (Wired, 12/95), it is clear that corporate management at Netscape put considerable emphasis on the "platform" nature of the browser (enabled by plug-in and Java technology which first appeared in version 2.0 of Navigator) in their marketing communications for the product. Andreessen stresses the term "live online applications" in describing the product's features. He says, "We use the term *live online applications* for the types of applications that people build on our platform - *online* because the applications are network-centric and distributed, **live because they're highly interactive** with users and with data retrieved in real time from databases and other sources over the network. *Live Objects* is our term for things like Java applets and inlined viewers embedded in HTML documents. So, a live online application is built using HTML as a framework."

Andreessen's comments relate directly to the capabilities given to their Web browser through the use of "plug-in" applications and Java applets. These capabilities result directly from both the plug-ins' and Java's incorporation of the features of the claimed invention, including an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client

workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed tag's (or "applet" tag's) location within the hypermedia document being displayed in the browser-controlled window. Andreessen's use of the terms "live objects" and "live online applications" refer specifically to the seemingly "live" nature of these fully-interactive embedded windows within plug-in-enabled or Java-enabled Web pages, as opposed to the static nature of Web document content seen prior to the applicants' invention.

A good indicator that Sun Microsystems felt that enabling interactivity in Web pages was the key feature of Java is given in the first chapter of "Hooked on Java," which was written by members of the original Java development team. They say, "With applets written in the Java programming language, Web users can design Web pages that include animation, graphics, games, and other special effects. **Most important, Java applets can make Web pages highly interactive.**"

This statement shows that the developers of Java felt that the most important feature of the Java technology was the ability of Java to allow an embed text format (the applet tag) within a Web document to be parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the applet tag's location within the hypermedia document being displayed in the browser-controlled window. The book's authors further emphasize the novelty and nonobviousness of this technology when they say, "Quite simply, Java-powered pages are Web pages that have Java applets embedded in them. They are also the Web pages with the coolest special effects around....Remember, **you need a Java-compatible Web browser such as HotJava to view and hear these pages and to**

**interact with them; otherwise, all you'll access is static Web pages minus the special effects."**

The novel and nonobvious nature of the claimed invention is further emphasized by a quote taken from Reference #7 of Appendix A, where Marc Andreessen's early opinion of Java is reported: "In December 1994, Java and HotJava (at this stage called Oak) were posted in a secret file deep in the Net; only a select few were given pointers and invited to check it out. **Three months later, Marc Andreessen gushed to the San Jose Mercury News 'What these guys are doing is undeniably, absolutely new. It's great stuff.'**"

Others in the Industry supported this opinion as well, as Reference #8 clearly shows: "'The Java programming language radically advances the multimedia potential of the Net, enabling faster animation, games, and powerful interfaces within Web sites,' says Nova Spivack, director of marketing and co-founder, EarthWeb. **'Sun's Java technology is a stroke of genius** that will transform the Internet in a matter of months.' ... **'The Java language is a revolutionary technology** with profound implications for the Internet as well as the computer industry in general,' says Jack D. Hidary, chief executive officer, and co-founder of EarthWeb."

That the interactive nature of embedded interactive program objects was the specific reason for the commercial success of Java was forcefully illustrated in Reference #9 (Communications Week, 10/95): "The impact of Java on our [financial services] marketplace is that **static browsers have to become interactive and event-oriented,**" said Bill Adiletta, president at Market Vision, in Santa Cruz, Calif. **'That is absolutely essential for us to even consider [the Web] as an option for our marketplace.'**" The "interactive and event-oriented" capabilities which Mr. Adiletta refers to are those that are specifically enabled through the Web browsers'



incorporation of the features of the claimed invention, including an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed tag's location within the hypermedia document being displayed in the browser-controlled window.

The status of Java, and by association, Netscape Navigator, as one of the most popular development platforms for new applications is supported by Reference #10 (Forbes, 11/96), which states that **"Java is also well on its way to becoming the most important Internet software standard, catapulting Sun past Netscape and Microsoft as the leader in Internet computing."**

That the results of the features of the claimed invention are surprising and unexpected is supported by the above references, and by Reference #10, which quotes Sun's CEO, Scott McNealy as saying, "We always thought we were onto something with Java--that it was our one big chance to challenge Microsoft and to change the economics of the business... But I have to admit I never thought Java computing could unfold quite this quickly, or with such universal support from customers and competitors. **It's astounding, really."**

The novel and unobvious nature of the claimed invention is strongly supported by Reference #11 (PC WEEK, 4/96), which reports the award, by PC Week, of a technical innovation award to Java. They go on to state that **"The best example so far of why Java is more than a cool concept is Sun's Java WorkShop, a development tool for creating applets that is completely written in Java. It is basically a set of Java applets that run on any platform that has a Java virtual machine, which for now includes Solaris and 32-bit Windows operating systems. A Macintosh version is due this summer. Java WorkShop has another unique quality that is likely to affect the way all applications are**

developed in the future. **The WorkShop applets and the applications created with the development environment live within a Web browser** written in Java. The basic notion is that if you are developing for the Web, the container should be Web-centric. **The user interface is a Web browser that integrates Net technologies like Hypertext Markup Language at the core of the product.** You click on an icon to load a new tool that lives on a local Web page, and you can go to other pages to access other tools, code and documentation." This award illustrates that there was Industry consensus that the most important part of the Java technology was its ability to allow external interactive applications to be embedded within Web pages.

Sun's Java WorkShop, as described in this reference, shows many of the features of the claimed invention. It is an excellent example of the surprising and unexpected results of the claimed invention.

Despite the advantages and popularity of the Java applet platform, it is not all things to all people. Web plug-ins, at least those that don't depend on applet interpreters, are oftentimes a more practical solution for enhancing the interactivity of Web pages. Reference #12 (Boardwatch, 6/96) describes the feelings of one well-known expert in the field: "Well, even though Java is supposed to be the elixir for the woes of Internet standardization, every day seems to bring a new flavor - smoother, tastier, more powerful. How about sticking to one flavor and form so that someone can actually figure out what to do with it? On to the stuff that's real. **Plug-ins are currently the most useful browser enhancement.** The variety and quantity of plug-ins that have become available in the last few months has astounded me. They open up immense possibilities of all kinds for unique and interesting applications." Again, one should note that the term "plug-ins" used here specifically refers to programs which incorporate the features of the claimed invention, including an embed text format that is parsed by a Web

browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed tag's location within the hypermedia document being displayed in the browser-controlled window.

Of course, Microsoft did not sit idly by during this period of rapid technological advance. Marc Andreessen comments on Microsoft's recent licensing of Mosaic in his Dec. '95 Wired interview (Reference #5): "We have a wide range of competitors entering this space, competing with one part of what we do. The Microsoft browser is basically what we did with Mosaic - I'm glad to see that they've caught up to what we did two years ago."

It is clear from this comment that Microsoft's browser at the time did not support embedded interactive program objects.

It took them almost a year to come up with a competing technology, which they called ActiveX, and which was released in 1996.

Microsoft felt that embedded interactive program objects in Web pages (a feature of the claimed invention) was so fundamental a requirement for commercial success, that they even licensed Java for incorporation in their Web browser, at a time when it was apparent that doing so might be at Microsoft's own peril (from Reference #10, Forbes, 11/96): "Gates can thank little Netscape for putting the \$9 billion software behemoth in the extraordinary position of having to support a technology that could badly undermine Windows. **Once Gates decided he wanted to unseat Netscape as the Internet browser king, Microsoft had to incorporate Java into its own browser, Internet Explorer, just as Netscape had done with Navigator...**Microsoft still derides Java as merely a 'mildly interesting programming language' and is doing all it can to torpedo Java with its own Internet software component technology, ActiveX. Microsoft claims ActiveX uses PC hardware and software better than Java does."

Like Netscape's Navigator and like Sun's Java products, Microsoft's ActiveX technology has become very successful. This is evidenced by the following excerpt from New Media Magazine's 1997 "Hyper Awards" issue, where they give ActiveX the award of "Technology of the Year":

"TECHNOLOGIES OF THE YEAR: Microsoft ActiveX

Microsoft ActiveX was first maligned as an unnecessary Java competitor and part of Bill Gates' plot to dominate the world, but this standard for Internet applets has assumed the high ground as a universal OLE-based wrapper that embraces Java, VRML, Shockwave, Visual Basic, C++, and other scripting tools.

To further establish it as an industry norm, Microsoft is ensuring development of **ActiveX plug-ins for Mac and UNIX Web browsers**, and is even spinning off control of ActiveX into an independent standards body, a first for Microsoft. The strategy is working -- ActiveX acceptance among developers has been phenomenal, and nobody is comparing it to Java anymore."

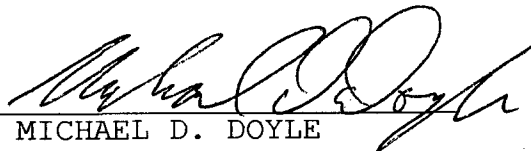
The above citations, as well as the additional details given in Appendix A, provide ample evidence of the commercial success of products incorporating features of the claimed invention, as well as evidence of the widespread acclaim that these products have garnered for the technical innovations which the features of the claimed invention allowed them to provide. They further show that the successes of these products was a direct result of the features of the claimed invention, which they incorporated *through implementation of an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed text format's location within the hypermedia document being displayed in the browser-controlled window.*

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 20

PATENT

I further 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 issuing thereon.

Dated: May 27th, 1997.

  
MICHAEL D. DOYLE

CEK:db  
s:\02307I\553\DECL.01

**Appendix A to Doyle Declaration**

List of relevant references:

- 1) "Today the Web, Tomorrow the World," by Tom Halfhill, Byte, Jan. 1997:

"Never in the history of computing has a new language attracted so much support from toolmakers, software developers, and OS vendors in such a short time....**Java transcends being a language to being a software platform because of the Java virtual machine (VM), which simulates a computer in software.** The Java VM can run on existing computers and OSes (e.g., Windows and the Mac OS), or it can run on hardware designed only for Java....Java could trigger the biggest platform shift since Windows surpassed DOS--all without forcing you to change your hardware and OS...Java is a stealth platform that propagates entirely in software and coexists peacefully with the native OS....Java carries software abstraction to the next level because it abstracts everything below the VM. It's designed for a world in which the OS and CPU are interchangeable parts that can be replaced without breaking applications."

- 2) "Teach yourself Java in 21 days," by Laura LeMay and Charles Perkins, Sams Net Publishers, 1996, pages 423-424:

"It [Java] deserves to be there [at center stage]. It is the natural outgrowth of ideas that, since the early 1970s inside the Smalltalk group at Xerox PARC, have lain relatively dormant in the mainstream. Smalltalk, in fact, invented the first object-oriented bytecode interpreter and pioneered many of the deep ideas that Java builds on today....The final requirement is what has stymied many attempts at ubiquity in the past. If you base your system on any assumptions about what is "beneath" the run-time system, you lose. If you depend in any way on the computer or operating system below, you lose. **Java solves this problem by inventing an abstract computer of its own and running on that.**"

- 3) "Virtual Machines: Vendors move past Java to look for UVMS," By Jim Balderston and Bob Trott, InfoWorld, April 7, 1997:

"IBM, in conjunction with its Taligent subsidiary, has begun a research effort intended to create a single virtual machine capable of running applications written in C++, Smalltalk, or Java....Once developed, a universal virtual machine would, in theory, allow developers to write an application in any language and run it on any system. But at this stage, the challenges associated with developing UVMS are immense, and any such mechanism would probably only be capable of running

interpreted code, which would have a big impact on performance and memory requirements."

4) "Proposing a Standard Web API," by Michael Doyle, Cheong Ang and David Martin, Dr. Dobb's Journal, February, 1996:

"We designed and implemented an API for embedded inline applets that allowed a Web page to act as a container document for a fully-interactive remote-visualization application, allowing real-time volume rendering and analysis of huge collections of 3-D biomedical volume data, where most of the computation was performed by powerful remote visualization engines. Using our enhanced version of Mosaic, later dubbed 'WebRouser,' a scientist using a low-end client workstation could exploit computational power far beyond anything that could ever be found in one location.

This work was shown to several groups in 1993, including many that were later involved in projects to add APIs and applets to Web browsers at places like NCSA, Netscape, and Sun."

5) "Why Bill Gates wants to be the next Marc Andreessen," Interview by Chip Bayers with Netscape founder, Marc Andreessen, Wired Magazine, December, 1995, page 165:

**Bayers:** Because it allows plug-ins, you're calling Netscape 2.0 a *platform*, rather than a *browser* in the company's descriptions. But Photoshop has some pretty sophisticated plug-ins, and it's only an application. Is Netscape 2.0 a platform or an application?

**Andreessen:** It's a little of both. It's an application in that it runs on top of what is traditionally thought of as an operating system - like Windows or UNIX - but **it's a platform in that people can build applications on it.** We use the term *live online applications* for the types of applications that people build on our platform - *online* because the applications are network-centric and distributed, **live because they're highly interactive** with users and with data retrieved in real time from databases and other sources over the network. *Live Objects* is our term for things like Java applets and inlined viewers embedded in HTML documents. So, a live online application is built using HTML as a framework. This gives developers great flexibility in linking together people and information over networks.

Our platform is also operating-system independent; you can run the client on Windows 3.1, Windows 95, Windows NT, UNIX - any of 12 flavors - or Mac; the server can run on Windows NT, UNIX, and in the near future, Windows 95.

We don't use the term *browser* because we think it's pretty clear that Netscape Navigator 2.0 is far more than a browser. On one hand it's a suite: it handles e-mail, threaded

discussion groups, ftp, gopher, chat, et cetera. On the other hand, **it's a platform: it allows people to build these live online applications on top of it.**"

**"Bayers:** Since this is an entrepreneurial environment with a level playing field, who do you see as your biggest competitor right now? Microsoft and its new browser?

**Andreessen:** We have a wide range of competitors entering this space, competing with one part of what we do. The Microsoft browser is basically what we did with Mosaic - I'm glad to see that they've caught up to what we did two years ago."

6) "Hooked on Java," by Arthur Van Hoff, et al., of Sun Microsystems:

Page 1: "With applets written in the Java programming language, Web users can design Web pages that include animation, graphics, games, and other special effects. **Most important, Java applets can make Web pages highly interactive.** Of course, users need a Java-compliant Web browser like Netscape Navigator or HotJava to view and use Java-powered pages....Maybe we're a little bit biased because we are part of the programming team that has been developing Java at Sun, but we think it's true. Java allows you to do exciting things with Web pages that weren't possible before....Greater interactivity is one of the hallmarks of Web pages that use Java. Users can interact with the content of a Java-powered page via the mouse, keyboard, and other user-interface elements such as buttons, slides, and text fields."

Page 2: "Small programs written in the Java programming language called Java applets make this all possible. Java applets are embedded right in Web pages. When users access these pages, the applets are downloaded to their computers and executed. Instead of the activity happening on the server side as is the case with CGI programming, it happens on the client side in a Java-compatible Web browser."

Page 3: "Quite simply, Java-powered pages are Web pages that have Java applets embedded in them. They are also the Web pages with the coolest special effects around....Remember, you need a Java-compatible Web browser such as HotJava to view and hear these pages and to interact with them; otherwise, all you'll access is static Web pages minus the special effects."

7) "The Java Saga," by David Bank, Wired Magazine, Dec., 1995:

Page 166: "While today's Web is mostly a static brew - a grand collection of electronically linked brochures - Java holds the



promise of caffeinating the Web, supercharging it with interactive games and animations and thousands of application programs nobody's ever heard of....Software developers are busy shaping Java into applications that will add new life to Web browsers, like Netscape and Mosaic, producing programs that combine **real-time interactivity** with multimedia features that have been available only on CD-ROM...What's a Java application? Point to the Ford Motor web-site, for instance, and all you'll get are words and pictures of the latest cars and trucks. Using Java, however, Ford's server could relay a small application (called an applet) to a customer's computer. From there, the client could customize options on an F-series pickup while calculating the monthly tab on various loan rates offered by a finance company or local bank."

Page 167: "Sun is giving away Java and HotJava for free for noncommercial use, in a fast-track attempt to make them the standard before Microsoft begins shipping a similar product, codenamed Blackbird, in early 1996" (note: Blackbird was later renamed to "ActiveX" by Microsoft)

Page 242: "A new business plan was drawn up early in 1994, which unceremoniously dumped the speculative markets FirstPerson had pursued and began focusing on personal computers - the technology the project was supposed to leapfrog in the first place. The new plan was to create a corps of CD-ROM developers who would write in Oak and, ideally, stick with it as their platform language while moving applications to the commercial online services....**The plan, remarkably, contained no mention of Mosaic or the Web....**FirstPerson was scrapped in the spring of 1994..."

Page 243: "Joy and Schmidt wrote yet another plan for Oak and sent Gosling and Naughton back to work adapting Oak for the Internet. Gosling, whom Joy calls 'the world's greatest programmer,' worked on the Oak code, while Naughton set out to develop a true 'killer app.'...In January 1995, Gosling's version of Oak was renamed the more marketable Java. Naughton's killer app was an interpreter for a Web browser, later named HotJava."

Page 244: "In December 1994, Java and HotJava (at this stage called Oak) were posted in a secret file deep in the Net; only a select few were given pointers and invited to check it out. **Three months later, Marc Andreessen gushed to the San Jose Mercury News 'What these guys are doing is undeniably, absolutely new. It's great stuff.'** That was how the Java team knew it was finally going to make it. 'That quote was a blessing from the god of the Internet,' Polese says"

Page 245: "Sun is racing to stay ahead of the accelerating wave. The day after that midnight deadline for sending the finished code to Netscape, Joy was already at work pushing the limits of what Java could do.... 'I've got 15 patents I could file as soon as I type them,' Joy says. 'I figure I've got five years. It's like we've got a blank sheet and it says 'Internet.' Normally the best products don't win. The Internet is an opportunity for the best products to win. Java is great technically and people want it. I'm happy to get that once in my life - or maybe twice."

8) "Earthweb and Sun unveil Gamelan on the Internet; Premier directory for the Java revolution goes online," PRNewswire, October 11, 1995:

"The Java programming language radically advances the multimedia potential of the Net, enabling faster animation, games, and powerful interfaces within Web sites," says Nova Spivack, director of marketing and co-founder, EarthWeb. **'Sun's Java technology is a stroke of genius** that will transform the Internet in a matter of months.' ... **'The Java language is a revolutionary technology** with profound implications for the Internet as well as the computer industry in general,' says Jack D. Hidary, chief executive officer, and co-founder of EarthWeb."

9) "Goal/A multiplatform operating system/Sun positions Java as universal interface," by Richard Karpinski, Communications Week, October 9, 1995:

"What Sun sees with Java is the opportunity to build a universal, multiplatform operating system - a 'ubiquitous API,' said Schmidt - that will open up a new world of distributed, networked applications....According to Bill Joy, co-founder and chief technologist at Mountain View, Calif.-based Sun, **Java not only will enable more interactive and animated Web content**, but eventually it will replace C++ as the standard building block for computer applications, paving the way for a new paradigm of distributed, network-based applications....Nearer term, however, Java mainly will be used to jazz up the Web.... 'The impact of Java on our [financial services] marketplace is that **static browsers have to become interactive and event-oriented**,' said Bill Adiletta, president at Market Vision, in Santa Cruz, Calif. **'That is absolutely essential for us to even consider [the Web] as an option for our marketplace.'**"

10) "Sun's Java: The threat to Microsoft is real," by Brent Schlender and Eryn Brown, Forbes, November 11, 1996:

"Originally known as a way to jazz up Web pages with graphic animations--stock tickers that crawl across your screen, for example, and dancing icons--Java has quickly evolved into a whole lot more. To Microsoft's dismay, it is fast becoming what is known as a computing platform--a sturdy base upon which programmers can build software applications. Java is making possible the rapid development of versatile programs for communicating and collaborating on the Internet....Java is also making possible a controversial new class of cheap machines called network computers, or NCs, which Sun, IBM, Oracle, Apple and others hope will proliferate in corporations and our homes....To graph the numbers, you'll call in a charting applet that will let you print out your report nice and pretty, **all without leaving your browser.** And you'll always get the latest, greatest version of the applets too: Since the software is stored in only one place, corporate techies can keep it up to date more easily....**Java is also well on its way to becoming the most important Internet software standard,** catapulting Sun past Netscape and Microsoft as the leader in Internet computing....**The halo effect from Java is a big reason Sun's stock has become hot."**

"For Sun CEO Scott McNealy, it's a pipe dream come true. 'We always thought we were onto something with Java--that it was our one big chance to challenge Microsoft and to change the economics of the business,' he says, 'But I have to admit I never thought Java computing could unfold quite this quickly, or with such universal support from customers and competitors. It's astounding, really.' ... Java is also one of those charmed technologies--Microsoft's original DOS operating system is another--that arrived at exactly the right place at the right time. Since Sun introduced Java in May 1995, a constellation of forces--other Internet innovations, software economics, industry politics, and customer need--aligned almost simultaneously to let Java emerge....The keys to Java's success as a platform are ubiquity and absolute compatibility throughout the industry. To achieve ubiquity, Sun hitched a ride on Netscape's popular Navigator Internet browser, the program that unleashed the whole Internet phenomenon in the first place....Gates can thank little Netscape for putting the \$9 billion software behemoth in the extraordinary position of having to support a technology that could badly undermine Windows. **Once Gates decided he wanted to unseat Netscape as the Internet browser king, Microsoft had to incorporate Java into its own browser, Internet Explorer, just as Netscape had done with Navigator....**Microsoft still derides Java as merely a 'mildly interesting programming language' and is doing all it can to torpedo Java with its own Internet software component technology, ActiveX. Microsoft claims ActiveX uses PC hardware and software better than Java does."

- 11) "The First Annual IT Excellence Awards," PCWEEK, April, 1996:

"We had planned to pick **one product that stood out from the pack in terms of innovation, advancement, and improvement in cost of ownership of information delivery technology. Java met those criteria....**[Java] may enhance Sun's role as a mainstream industry innovator and leader. Until recently, Java was famous as hypeware, exemplified by cute but shallow applets. As one of our readers said, it's 100% buzzword-compliant. Nonetheless, we give Java a PC WEEK Corporate IT Excellence award for innovation this week, and it appears that the language environment is quickly moving beyond novelty to practical application."

"**The best example so far of why Java is more than a cool concept is Sun's Java WorkShop**, a development tool for creating applets that is completely written in Java. It is basically a set of Java applets that run on any platform that has a Java virtual machine, which for now includes Solaris and 32-bit Windows operating systems. A Macintosh version is due this summer. Java WorkShop has another unique quality that is likely to affect the way all applications are developed in the future. **The WorkShop applets and the applications created with the development environment live within a Web browser** written in Java. The basic notion is that if you are developing for the Web, the container should be Web-centric. **The user interface is a Web browser that integrates Net technologies like Hypertext Markup Language at the core of the product.** You click on an icon to load a new tool that lives on a local Web page, and you can go to other pages to access other tools, code and documentation."

- 12) "Browser Plug-Ins: Good, Bad and Ugly," by Chris Babb, Boardwatch, June, 1996:

"Well, even though Java is supposed to be the elixir for the woes of Internet standardization, every day seems to bring a new flavor - smoother, tastier, more powerful. How about sticking to one flavor and form so that someone can actually figure out what to do with it? On to the stuff that's real. **Plug-ins are currently the most useful browser enhancement.** The variety and quantity of plug-ins that have become available in the last few months has astounded me. They open up immense possibilities of all kinds for unique and interesting applications."

- 13) "1997 Hyper Awards," New Media Magazine, January, 1997:

"TECHNOLOGIES OF THE YEAR:  
Microsoft ActiveX

Microsoft ActiveX was first maligned as an unnecessary Java competitor and part of Bill Gates' plot to dominate the world, but this standard for Internet applets has assumed the high ground as a universal OLE-based wrapper that embraces JavaVRML, Shockwave, Visual Basic, C++, and other scripting tools.

To further establish it as an industry norm, Microsoft is ensuring development of **ActiveX plug-ins for Mac and UNIX Web browsers**, and is even spinning off control of ActiveX into an independent standards body, a first for Microsoft. The strategy is working -- ActiveX acceptance among developers has been phenomenal, and nobody is comparing it to Java anymore."

14) "ActiveX, a Standard?," by Michael Floyd (Editor in Chief), WebTechniques, October, 1996, page 5:

"--So tell me about ActiveX. Isn't it just OLE for the Internet?"

--Wrong,... ActiveX is a new API that, like OLE, is based on Microsoft's Component Object Model (COM). While OLE supports a compound document architecture for desktops, ActiveX is designed specifically to embed rich media objects within Web-based documents.'

This was part of a scene that recently played out in the offices of Web Techniques...Microsoft marketers arrived on our doorstep not for the standard dog-and-pony show, but specifically to 'debunk the myths' surrounding ActiveX technologies."



# **906 PH Ex. 12**







**UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office**

Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

NM

SERIAL NUMBER	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
---------------	-------------	----------------------	---------------------

08/324,443 10/17/94 DOYLE

M 02307553

020350 B3M1/0825  
TOWNSEND AND TOWNSEND AND CREW  
TWO EMBARCADERO CENTER EIGHTH FLOOR  
SAN FRANCISCO CA 94111

EXAMINER

ART UNIT PAPER NUMBER

2317

5

DATE MAILED: 08/25/97

This is a communication from the examiner in charge of your application.  
COMMISSIONER OF PATENTS AND TRADEMARKS

This application has been examined  Responsive to communication filed on 6-2-99  This action is made final.

A shortened statutory period for response to this action is set to expire 3 month(s), 0 days from the date of this letter.  
Failure to respond within the period for response will cause the application to become abandoned. 35 U.S.C. 133

**Part I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:**

- 1.  Notice of References Cited by Examiner, PTO-892.
- 2.  Notice of Draftsman's Patent Drawing Review, PTO-948.
- 3.  Notice of Art Cited by Applicant, PTO-1449.
- 4.  Notice of Informal Patent Application, PTO-152.
- 5.  Information on How to Effect Drawing Changes, PTO-1474.
- 6.

**Part II SUMMARY OF ACTION**

- 1.  Claims 1-56 are pending in the application.  
Of the above, claims \_\_\_\_\_ are withdrawn from consideration.
- 2.  Claims 6-43, 49-56 have been cancelled.
- 3.  Claims \_\_\_\_\_ are allowed.
- 4.  Claims 1-5, 44-48 are rejected.
- 5.  Claims \_\_\_\_\_ are objected to.
- 6.  Claims \_\_\_\_\_ are subject to restriction or election requirement.
- 7.  This application has been filed with informal drawings under 37 C.F.R. 1.85 which are acceptable for examination purposes.
- 8.  Formal drawings are required in response to this Office action.
- 9.  The corrected or substitute drawings have been received on \_\_\_\_\_. Under 37 C.F.R. 1.84 these drawings are  acceptable;  not acceptable (see explanation or Notice of Draftsman's Patent Drawing Review, PTO-948).
- 10.  The proposed additional or substitute sheet(s) of drawings, filed on \_\_\_\_\_, has (have) been  approved by the examiner;  disapproved by the examiner (see explanation).
- 11.  The proposed drawing correction, filed \_\_\_\_\_, has been  approved;  disapproved (see explanation).
- 12.  Acknowledgement is made of the claim for priority under 35 U.S.C. 119. The certified copy has  been received  not been received  been filed in parent application, serial no. \_\_\_\_\_; filed on \_\_\_\_\_.
- 13.  Since this application appears to be 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.
- 14.  Other

EXAMINER'S ACTION

**Part III DETAILED ACTION**

Applicant's arguments filed 06-02-97 have been considered but are moot in view of the new ground(s) of rejection.

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.*

Claims 1,2,5, and 44,45,48 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Applicant disclosed prior art and further in view Koppolu et al. US patent 5,581,686.

As per claim 1, Applicant disclosed prior art [pages 1-10: Mosiac + HTTP + HTML + "World Wide Web"] has the limitations essentially as claimed client workstation, network server coupled in a distributed hypermedia environment [p.8 lines 15-30];

executing on the client a browser application [p.4 Mosaic] that parses distributed hypermedia document to identify text formats [HTML tags] and for responding to predetermined text formats to initiate processes specified by the text format [p.4-5];

utilizing the browser to display, on said client workstation, portion of a first hypermedia document received over the network, wherein the hypermedia document includes an embed text format specifies the location of an object external to the hypermedia document [p.4 lines 4-12, p.5 lines 9-26].

It is apparent that specifies type of information [p.5 lines 11 - text, images, sound, video...] is utilized by the browser to identify and locate an executable application external to the hypermedia document [p.4 lines 13-22 - "viewer" software];

The prior art does not have embed text format specifying an external object which automatically invoke an external application to execute and enable interactive processing within a portion of the browser controlled window. The prior art provide display and interaction with an external object by launching an associated program in a separate window.

Koppolu teaches a method for in-place interaction with a contained object. Koppolu method permit automatic display and interaction of a linked object in a compound document [i.e. hypermedia document] within a portion of a window controlled by a container application [i.e. the browser] (see fig.1, col.8 line 50-68, col.9 lines 10-28, claims 1-2). It would have been obvious for one of ordinary skill in the art to use the teaching of Koppolu to the processing of the hypermedia document of the disclosed prior art because it would have provided a more

Serial Number: 08/324,443  
Art Unit: 2317

-4-

integrated and expandable system for processing/viewing of linked objects in the hypermedia documents and reduces clustering of window display.

HTML is a text tag structure document encoding. It is apparent the prior art as modified would have had a text tag for indicating links to an in-place interactive object.

As per claim 2, Koppolu teaches the interactively controlling via inter-process communication between the browser [container application] and said controllable application [server application] (see col.8 lines 1-7).

As per claim 5, Koppolu teaches the applications continue to communicate after the controllable application has been launched (col.13 lines 65 to col.14 line 5).

As per claim 44, it is rejected under similar rationale as for claim 1 above.

As per claim 45, it is rejected under similar rationale as for claim 2 above.

As per claim 48, it is rejected under similar rationale as for claim 5 above.

Claims 3-4 and 46-47 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Applicant disclosed prior art and Koppolu et al. US patent 5,581,686, and further in views of Moran

Serial Number: 08/324,443  
Art Unit: 2317

-5-

**"Tele-Nicer-Dicer: A new tool for the visualization of large volumetric data".**

As per claim 3, the disclosed prior art does not disclose interactively controlling via commands sent over the distributed environment. Moran discloses a distributed application (TNSD) for interactive control and visualization of graphical object through communication over network. Moran application allow usage of remote system resources for visualization of large data set at a client station. Moran discloses sending command to remote server, executing on the server, and sending result to the client to process and display [p.3 col.2-3 specifically col.1 3rd paragraph]. It would have been obvious for one of ordinary skill in the art to utilize Moran application as an external application ("Viewer") in the prior art system as modified because it would have improved the system by enabling the client station access to resources on higher performance servers and to have interactive visualization of large data set capability.

As per claim 4, it is apparent that the system as modified would have instructions residing on the client workstation in order to provide the resulting graphic representation [NSD visualization tool - p.1 col.2 last paragraph].

As per claims 46-47, they are rejected under similar rationales as for claims 3-4 above.

Serial Number: 08/324,443  
Art Unit: 2317

-6-

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for response to this final action is set to expire THREE MONTHS from the date of this action. In the event a first response is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event will the statutory period for response expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Dung Dinh whose telephone number is (703) 305-9655. The examiner can normally be reached on Monday-Thursday from 7:00 AM - 4:30 PM. The examiner can also be reached on alternate Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thomas Lee can be reached at (703) 305-9717.

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

**Any response to this final action should be mailed to:**

**Box AF**

Commissioner of Patents and Trademarks  
Washington, DC 20231

**or faxed to:**

(703) 308-9051, (for formal communications; please mark "EXPEDITED PROCEDURE")

(703) 308-5359 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington, VA., Sixth Floor (Receptionist).



August 17, 1997

**DINH C. DUNG  
PATENT EXAMINER  
GROUP 2300**

**Notice of References Cited**

Application No.  
**08/324,443**

Applicant(s)  
**Doyle et al.**

Examiner  
**Dung Dinh**

Group Art Unit  
**2317**

Page **1** of **1**

**U.S. PATENT DOCUMENTS**

	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
A	5,581,686	12/03/96	Koppolu et al.	395	340
B					
C					
D					
E					
F					
G					
H					
I					
J					
K					
L					
M					

**FOREIGN PATENT DOCUMENTS**

	DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
N						
O						
P						
Q						
R						
S						
T						

**NON-PATENT DOCUMENTS**

	DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
U		
V		
W		
X		





# 906 PH Ex. 13





GAU 2317  
#154 1/2

Atty Docket No. 02307I-553

PTO FAX NO.: 1-703-308-5359

ATTENTION: Examiner D. Dinh  
Group Art Unit 2317

RECEIVED  
OCT 31 1997  
GROUP ART UNIT

**OFFICIAL COMMUNICATION**  
**FOR THE PERSONAL ATTENTION OF**  
**EXAMINER D. DINH**

**CERTIFICATION OF FACSIMILE TRANSMISSION**

I hereby certify that the following Communication, including the Declaration of Michael D. Doyle and Attachments A and B, in re Application of Michael D. Doyle, Serial No. 08/324,443, filed October 17, 1994, for EMBEDDED PROGRAM OBJECTS IN DISTRIBUTED HYPERMEDIA SYSTEMS, is being facsimile transmitted to the Patent and Trademark Office on the date shown below.

Number of pages being transmitted, including this page: 26

Dated: October 31, 1997

*Irene Rodas*  
\_\_\_\_\_  
Irene Rodas

**PLEASE CONFIRM RECEIPT OF THIS PAPER BY  
RETURN FACSIMILE AT (415) 576-0300**

**CONFIRMATION COPY**

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, CA 94111-3834  
Telephone: (415) 576-0200  
Fax: (415) 576-0300  
i:\cek\share\02307i\553\ptofax.com

**FAXED**  
10/31/97  
2248 French

I hereby certify that this correspondence is being sent by facsimile transmission to: D. Dinh  
Fax No.: 1-703-308-5359  
Assistant Commissioner for Patents,  
Washington, D.C. 20231,  
on

10-31-97

Attorney Docket No. 02307I-553

PATENT

TOWNSEND and TOWNSEND and CREW LLP

By *Irene Rodas*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: )

MICHAEL D. DOYLE et al. )

Examiner: D. Dinh

Application No.: 08/324,443 )

Art Unit: 2317

Filed: 10/17/94 )

COMMUNICATION

For: EMBEDDED PROGRAM OBJECTS IN )  
DISTRIBUTED HYPERMEDIA )  
SYSTEMS )

Assistant Commissioner for Patents  
Washington, D.C. 20231

REMARKS

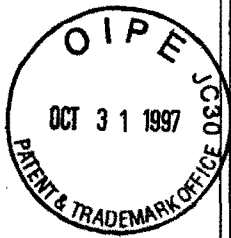
In the Office Action mailed June 25, 1997, the claims pending in the present application were rejected over Koppolu reference (U.S. Patent No. 5,581,686 "the '686 patent").

The '686 patent was filed June 6, 1996 and is a continuation of parent Appln. No. 229,264, filed April 15, 1994, which is a C-I-P of grandparent Appln. No. 984,868 filed December 1, 1992.

The file history of the grandparent application has been examined and it has been determined that Figs. 32-56 and Secs. 6.0 to 6.4.4 of the '686 patent were added as new matter in the C-I-P parent application.

Attached hereto is a declaration and evidence proving that the claimed invention was conceived and reduced to practice prior to the filing date of the C-I-P parent application. Accordingly, Figs. 32-56 and Secs. 6.0 to 6.4.4 of the '686 patent are not prior art.

Further, this to confirm that one of the co-inventors, Michael Doyle, and his attorney, Charles Krueger, will appear at



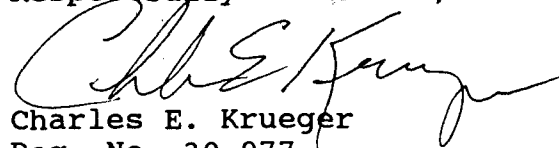
MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 2

PATENT

the examiner's office for an interview at 9:00 A.M. on Thursday,  
November 6, 1997.

If the Examiner has any comments or questions, please  
telephone the undersigned at (415) 576-0200.

Respectfully submitted,

  
Charles E. Krueger  
Reg. No. 30,077

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, California 94111-3834  
(415) 576-0200  
Fax (415) 576-0300  
CEK:db

i:\cek\share\02307I\553\com.2

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

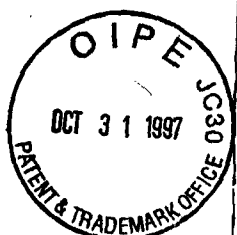
**PATENT**

on 10-31-97

Attorney Docket No. 023071-553

TOWNSEND and TOWNSEND and CREW LLP

By Jane Rodas



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	)	
MICHAEL D. DOYLE et al.	)	Examiner: D. Dinh
Application No.: 08/324,443	)	Art Unit: 2317
Filed: 10/17/94	)	<u>DECLARATION OF MICHAEL D. DOYLE</u>
For: EMBEDDED PROGRAM OBJECTS IN)	)	<u>UNDER RULE 131</u>
DISTRIBUTED HYPERMEDIA	)	
SYSTEMS	)	

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

I, MICHAEL D. DOYLE, hereby declare that:

1. I am a co-inventor of the subject matter disclosed and claimed in U.S. Patent Application No. 08/324,443.

2. The subject matter claimed in the above patent application was reduced to practice in this country prior to April 15, 1994, the filing date of the parent of the Koppolu reference cited by the examiner.

3. The reduction to practice of the claimed invention is evidenced by ATTACHMENTS A and B. ATTACHMENT A is a copy of a paper entitled "Integrated Control of Distributed Volume Visualization Through the World-Wide-Web", by Ang, Martin, and Doyle. This paper was submitted for publication prior to April 15, 1994. ATTACHMENT B is a transcript of the audio portion and still photographs of a video tape presented to an audience of scientists prior to April 14, 1994.

4. As stated in ATTACHMENT A, at page 5, paragraph 3.2, Mosaic (the browser) interprets the HTML <EMBED> tag included in a document to create a drawing area widget in a

RECEIVED  
OCT 31 1997  
PATENT & TRADEMARK OFFICE

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 2

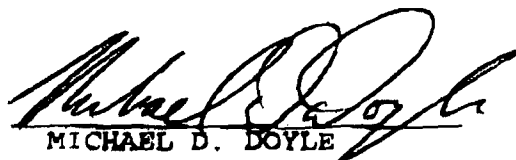
PATENT

document presentation and creates a shared window system buffer to receive visualization results. In addition, when the browser parses an <EMBED> tag in the document, the browser automatically launches the external application specifying the location of the visual object to render and identify the shared image buffer. The format and operation of an EMBED tag for 3D image data is described at paragraph 3.1.

5. As stated in ATTACHMENT B, starting at the bottom of page 2, interface and control software had been developed that allows the embedding of a visualization application within a Mosaic document. As is apparent from the photographs, the object is displayed and processed within the browser-controlled window. The visualization application is external to the hypermedia document displayed by the browser. Automatic launching of the external application when an HTML document is opened by the browser is depicted in the video.

I further 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 issuing thereon.

Dated: October 29, 1997.

  
MICHAEL D. DOYLE

CEK:db  
i:\cek\share\023071\553\decl.02

SPEAKER CONSENT FOR AUDIO/VIDEO RECORDINGS

We would like to audio/video tape your presentation at **Medicine Meets Virtual Reality II: Interactive Technology and Healthcare**, January 27-30, 1994.

Audio and video tapes will be available for immediate distribution to attendees and will be marketed and sold after the conference. Your colleagues will be able to benefit from your remarks by listening to cassettes and viewing tapes whether or not they were in attendance. The tapes will be copyrighted and marketed under Title 17 of the U.S. Code or other law as may be enacted, and all rights to royalties, if any, in conjunction with cassette sales shall hereby be assigned to Medicine Meets Virtual Reality. This agreement does not preclude the publication by you of your paper, speech or comments at any time and in any format.

Please cooperate with the audio/visual taping staff to obtain the best possible recordings by using the microphone at all times and repeating the questions that are asked by persons not near a microphone.

Sign this form below and return it to us at your earliest convenience.

Thank You.

---

I hereby give permission to **Medicine Meets Virtual Reality** to record and distribute audio and video tapes containing my presentation as outlined above.

Signature Michael D. Doyle Date 11-30-93

Please print name Michael D. Doyle

Thank you very much for your cooperation. Remember to pick up the complimentary audiotape of your presentation before leaving the meeting.

Please return this completed form to:  
MEDICINE MEETS VIRTUAL REALITY  
P.O. Box 23220, San Diego, CA 92193  
For further information call 619/751-8841,  
or Fax 619/751-8842.





## Integrated Control of Distributed Volume Visualization Through the World-Wide-Web

Cheong S. Ang, M.S.  
David C. Martin, M.S.  
Michael D. Doyle, Ph.D.

University of California, San Francisco  
Library and Center for Knowledge Management  
530 Parnassus Avenue  
San Francisco, California 94143-0840  
contact: doyle@ckm.ucsf.edu

### Abstract

The World-Wide-Web (W3) signals the entré of the digital library with the goal of providing immediate and ubiquitous access to digital information of any type from data repositories located throughout the world. The web's development enables not only effective access for the generic user, but also more efficient and timely information exchange among scientists and researchers. We have extended the repertoire of the web to include access to three-dimensional volume data sets with integrated control of a distributed client-server volume visualization system. This paper provides a brief background on the World-Wide-Web, an overview of the extensions necessary to support new data types and a description of the distributed visualization system.

### 1. Introduction

Advanced scanning devices, such as magnetic resonance imaging (MRI) and computer tomography (CT), have been widely used in the fields of medicine, quality assurance and meteorology [Pommert, Zandt, Hibbard]. The need to visualize resulting data has given rise to a wide variety of volume visualization techniques and computer graphics research groups have implemented a number of systems to provide volume visualization (e.g. AVS, ApE, Sunvision Voxel and 3D Viewnix)[Gerleg, Mercurio, Varde Wattering]. Previously these systems have depended on specialized graphics hardware for rendering and significant secondary storage for the data. The expense of these requirements has limited the ability of researchers to exchange findings. To overcome the barrier of cost and provide additional means for researchers to exchange and examine three-dimensional volume data, we have implemented a distributed volume visualization tool for general purpose hardware and integrated that visualization service with the distributed hypermedia [Flanders, Broering, Kiong, Robison, Story] system provided by the World-Wide-Web [Nickerson].

Our distributed volume visualization tool, VIS, utilizes a pool of general purpose workstations to generate three dimensional representations of volume data. The VIS tool provides integrated load-balancing across any number of heterogenous UNIX™ workstations (e.g. SGI, Sun, DEC, etc...) [Giertsen] taking advantage of the unused cycles that are generally available in academic and research environments. In addition, VIS supports specialized graphics hardware (e.g. the RealityEngine from Silicon Graphics) when available for real-time visualization.

Distributing information that includes volume data requires the integration of visualization with a document delivery mechanism. We have integrated VIS and volume data into the W3, taking advantage of the client-server architecture of W3 and its ability to access hypertext documents stored anywhere on the Internet [Obraszka, Nickersen]. We have enhanced the capabilities of the most popular W3 client, Mosaic [Andressen] from the National Center for Supercomputer Applications (NCSA), to support volume data and have defined an inter-client protocol for communication between VIS and Mosaic for volume visualization.

Attachment A

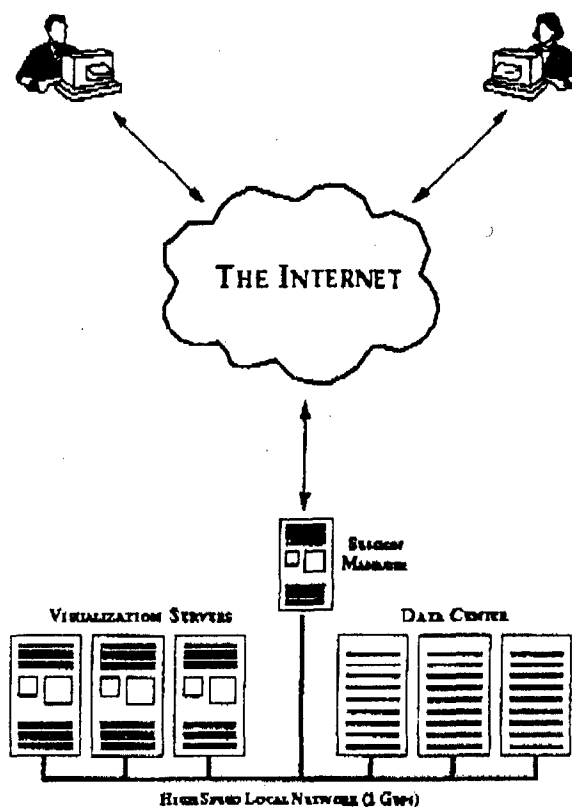


Figure 1: VIS client/server model.

### 1.1 The World-Wide-Web

The World-Wide-Web is a combination of a transfer protocol for hyper-text documents (HTTP) and a hyper-text mark-up language (HTML) [Nickersen]. The basic functionality of HTTP allows a client application to request a wide variety of data objects from a server. Objects are identified by a universal resource locator (URL)[Obraczka] that contains information sufficient to both locate and query a remote server. HTML documents are defined by a document type definition (DTD) of the Standard Generalized Mark-up Language (SGML). These documents are returned to W3 clients and are presented to the user. Users are able to interact with the document presentation, following hyper-links that lead to other HTML documents or data objects. The client application may also directly support other Internet services, such as FTP, Gopher, and WAIS, [Andreessen] or may utilize gateways that convert HTTP protocol requests and return HTML documents. In all interactions, however, the user is presented with a common resulting data format (HTML) and all links are accessible via URL's.

### 1.2 Mosaic

The National Center for Supercomputer Applications (NCSA) has developed one of the most functional and popular World-Wide-Web clients: Mosaic. This client is available via public FTP for the most popular computer interfaces (n.b. Motif, Windows and the Macintosh). Mosaic interprets a majority of the HTML DTD elements and presents the encoded information with page formatting, type-face specification, image display, fill-in forms, and graphical widgets. In addition, Mosaic provides inherent access to FTP, Gopher, WAIS and other network services [Andreessen].

## 1.3 VIS

VIS is a simple but complete volume visualizer. VIS provides arbitrary three-dimensional transformation (e.g. rotation and scaling), specification of six axial clipping planes (n.b. a cuboid), one arbitrary clipping plane, and control of opacity and intensity. VIS interactively transforms the cuboid, and texture-maps the volume data onto the transformed geometry. It supports distributed volume rendering [Argiro, Drebin, Kaufman] with run-time selection of computation servers, and isosurface generation (marching cubes)[Lorenson, Levoy] with software Gouraud shading for surface-based model extraction and rendering. It reads NCSA Hierarchical Data Format (HDF) volume data files, and has a graphical interface utility to import volume data stored in other formats.

The rest of the paper is divided into five parts. Section 2 will describe VIS and how it fulfills the role as a W3 visualization tool. Section 3 is devoted to Mosaic, and will include the implementation of its interface with VIS. Section 4 presents the project results. Section 5 is about ongoing and future development, and the last section is conclusions.

## 2. VIS: A Distributed Volume Visualization Tool

VIS is a highly modular distributed visualization tool, following the principles of client/server architecture (figure 1), and consisting of three cooperating processes: VIS, Panel, and VRServer(s). The VIS module handles the tasks of transformation, texture-mapping, isosurface extraction, Gouraud shading, and manages load distribution in volume rendering. VIS produces images that are drawn either to its own top-level window (when running stand-alone) or to a shared window system buffer (when running as a cooperative process). The Panel module provides a graphical user-interface for all VIS functionality and communicates state changes to VIS. The VRServer processes execute on a heterogeneous pool of general purpose workstations and perform volume rendering at the request of the VIS process. The three modules are integrated as shown in figure 3 when cooperating with another process. A simple output window is displayed when no cooperating process is specified.

### 2.1 Distributed Volume Rendering

Volume rendering algorithms require a significant amount of computational resources. However, these algorithms are excellent candidates for parallelization. VIS distributes the volume rendering among workstations with a "greedy" algorithm that allocates larger portions of the work to faster machines [Bloomer]. VIS segments the task of volume rendering based on scan-lines, with segments sized to balance computational effort versus network transmission time. Each of the user-selected computation servers fetches a segment for rendering via remote procedure calls (RPC), returns results and fetch another segment. The servers effectively compete for segments, with faster servers processing more segments per unit time, ensuring relatively equal load balancing across the pool. Analysis of this distribution algorithm [Giertsen, 93] shows that the performance improvement is a function of both the number of segments and the number of computational servers, with the optimal number of sections increasing directly with the number of available servers. Test results indicate that performance improvement flattens out between 10 to 20 segments distributed across an available pool of four servers. Although this algorithm may not be perfect, it achieves acceptable results.

### 2.2 Cooperative Visualizaton

The VIS client, together with its volume rendering servers, may be launched by another application as a visualization server. The two requirements of cooperation are a shared window system buffer for the rendered image and support for a limited number of inter-process messages. VIS and the initiating application communicate via the ToolTalk service, passing message specifying the data object to visualize, options for visualization, and maintaining state regarding image display. The VIS Panel application appears as a new top-level window and allows the user control of the visualization tool.

## 3. Visualization with Mosaic

Ang, Martin and Doyle: "Integrated Control of Distributed Volume Visualization Through the World Wide Web", Paper submission for Visualization '94,

We have enhanced the Mosaic W3 browser to support both a three-dimensional data object and communication with VIS as a cooperating application (figure 2). Mosaic provides the user with the ability to locate and browse information available from a wide variety of sources including FTP, WAIS, and Gopher. HTTP servers respond to requests from clients, e.g. Mosaic, and transfer hypertext documents. Those documents may contain text and images as intrinsic elements and may also contain external links to any arbitrary data object (e.g. audio, video, etc...). Mosaic may also communicate with other Internet servers, e.g. FTP, either directly - translating request results into HTML on demand - or via a gateway that provides translation services. As a W3 client, Mosaic communicates with the server(s) of interest in response to user actions (e.g. selecting a hyperlink), initiating a connection and requesting the document specified by the URL. The server delivers the file specified in the URL, which may be a HTML document or a variety of multimedia data files (for example, images, audio files, and MPEG movies) and Mosaic uses the predefined SGML DTD for HTML to parse and present the information. Data types not directly supported by Mosaic are displayed via user-specifiable external applications and we have extended that paradigm to both include three-dimensional volume data as well as to integrate the external application more completely with Mosaic.

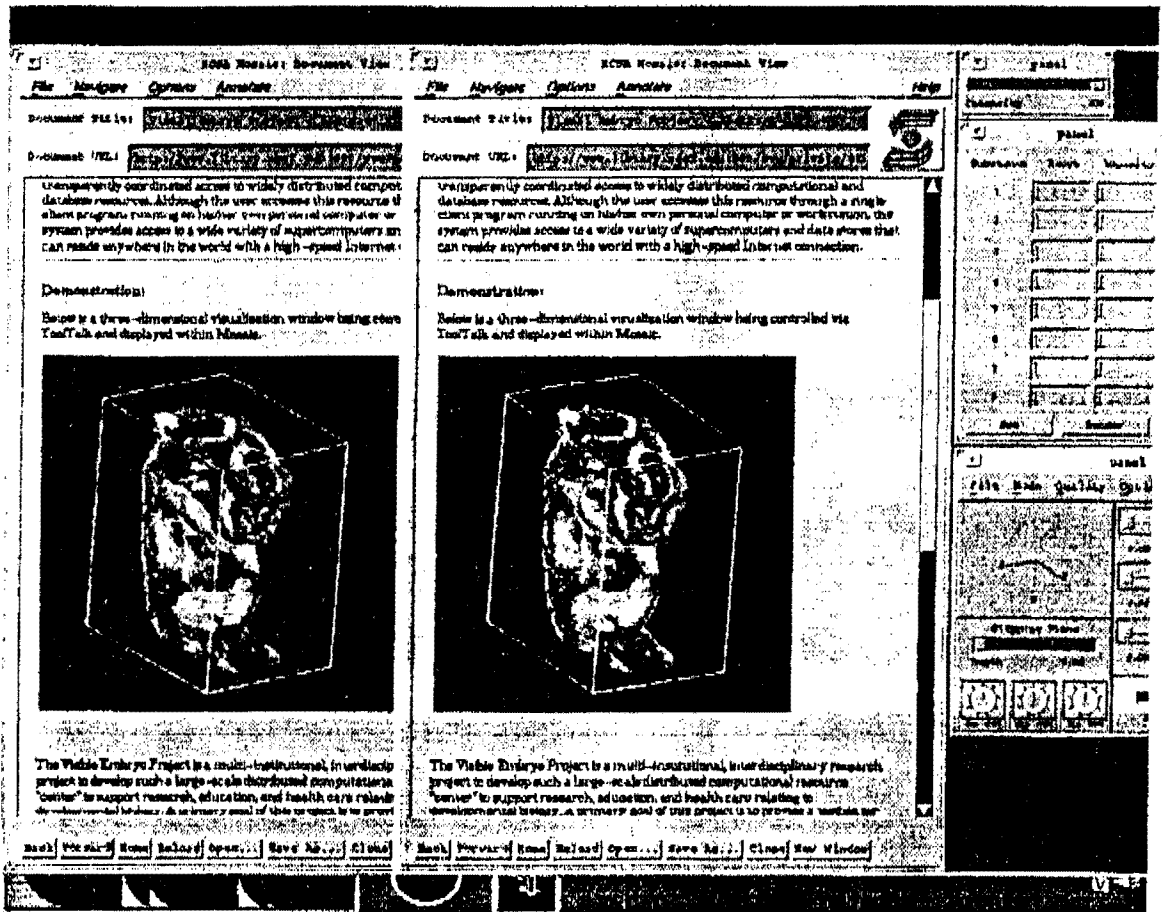


Figure 2: Mosaic utilizing VIS to embed an interactive visualization within an HTML document. Here the data has been rotated 6 degrees in a "cloned" Mosaic window to create a stereo pair.

### 3.1 Mosaic 3D Image support

We have extended the HTML DTD to support three dimensional data via the introduction of a new SGML element: IMG3D. This element provides information to the presentation system (i.e. Mosaic) about the content that is referenced in the document. The IMG3D element is defined in the HTML DTD as follows:

```
<!ELEMENT IMG3D EMPTY>
<!ATTLIST IMG3D VOLUME CDATA #REQUIRED
              WIDTH NUMBER #REQUIRED
              HEIGHT NUMBER #REQUIRED
              IMAGE CDATA #IMPLIED>
```

which is translated as "SGML document instance element tag IMG3D containing no content; three required attributes: volume data set, window width and height; and an optional image". In a HTML document, a 3D image element would be represented as:

```
<IMG3D VOLUME="http://www.library.ucsf.edu/.../data/Embryo.hdf"
        WIDTH=400
        HEIGHT=400
        IMAGE="http://www.library.ucsf.edu/.../images/Embryo.gif">
```

which may be interpreted as "create a 3D visualization window of width 400 pixels, height 400 pixels, and visualize the data embryo.hdf located at the HTTP server site www.library.ucsf.edu".

### 3.2 Interface with Mosaic

The VIS/Mosaic software system consists of three elements: VIS, Mosaic, and Panel (which provides GUI for data manipulation). Currently, VIS communicates with Mosaic via ToolTalk™, but the system will work with any inter-client communication protocol. When Mosaic interprets the HTML tag IMG3D, it creates a drawing area widget in the document page presentation and creates a shared window system buffer to receive visualization results. In addition, Mosaic launches the VIS application, specifying the location of the data object to render and identifying the shared image buffer. When the VIS process begins execution, it verifies its operating parameters and launches the Panel application that in turn presents the user with the control elements for manipulating the visualization and manages the communication between VIS and Mosaic.

In addition to coordinating communications, the Panel application attempts to balance the rendering load across the selected VRServer(s), sending rendering parameters to the selected server(s), and integrating the resulting image segments(s). Thus the scenario following a user's action on the Panel will be (1) Panel sends rendering requests to VRServer(s) with the parameters from its GUI, (2) Panel fetches the returned image data, then writes it to the pixmap, (3) Panel sends a message to notify Mosaic upon completion, and (4) Mosaic bit-blots the image in the pixmap into its DrawingArea widget. This will be addressed in more details under section 3.2. The configuration of this software system is depicted in figure 3.

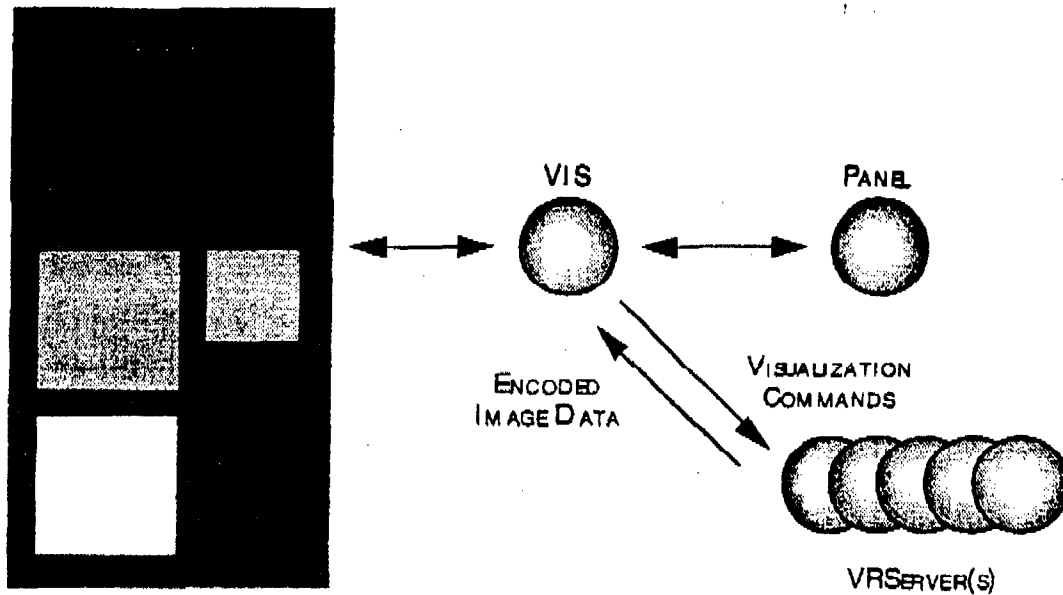


Figure 3: Interprocess communication among Mosaic, VIS and distributed rendering servers.

### 3.3 Interclient communication

We recognized the minimum set of communication protocols between Mosaic and Panel:

(a) Mosaic notifies Panel when the window id is no longer valid. This happens when the user navigates from the current page to the previous (with BACK button), or to another document through a link.

(b) Panel notifies Mosaic when the scene needs to be updated. This occurs when the user manipulates the data through the Panel's GUI, which may be any actions (rotations, scaling, changes in clipping planes, mode switching, for instance, from texture mapping to volume rendering) that result in rerendering of the picture.

The first case of (a) does not impose any problem since Mosaic can simply terminate VIS (this solution also applies when Mosaic is quitting). The second case is harder to handle because Mosaic caches the page, and in principle, we expect VIS to remain running because it should not be loading the HDF data, which may be quite a few megabytes, again. In the previous case, we simply have Mosaic kill the child process that launched Panel, which may subsequently send a terminate message to the VIS servers. In the latter case, Mosaic will send a message to Panel requesting it to unrealize/unmap its window. Then when the user comes back to the 3D object page, Mosaic may send another message to Panel ordering it to realize the GUI window, and update the window id of the DrawingArea that Panel has been sending message to (ironically Mosaic does not cache the widgets/windows on the cached page).

In (b), Panel will send an update message directly to the DrawingArea window id provided by Mosaic. Upon receiving the redrawing message, the DrawingArea widget calls its registered refreshing function, which simply bit-blots the pixmap Panel updated into the window. The protocols are summarized in Table 1.

MESSAGES	MOSAIC NOTIFIES VIS	VIS RESPONDS TO MOSAIC
----------	---------------------	------------------------

Refresh	Compute Image	Done Drawing
PageCaching	Hide Panel	Panel Hidden
BacktoCachedPage	New Window ID	ID Changed, Panel Shown
DestroyPage	Terminate	Terminating

Table 1: Mosaic/VIS IPC communication.

**4. Results**

The results of the above implementation are very encouraging. The Mosaic/VIS successfully allows users to visualize HDF volume datasets from various HTTP server sites. Fig 2 shows a snapshot of the W3 visualizer. Distributing the volume rendering loads results in a remarkable speedup in image computations. However, we did not do any detail performance analysis, because the results of such an analysis would not be reproducible for varying network traffics loads, and fluctuating server workstation loads. The server pool we tested the system consists of heterogenous workstations: a SGI Indigo2 R4400/150MHz, two SGI Indy R4000PC/100MHz, a DEC Alpha 3000/500 with a 133MHz Alpha processor, two Sun SparcStations 10, and two Sun SparcStations 2, which were located arbitrarily on a Ethernet network. To our knowledge this is the first demonstration of the embedding of interactive control of a client/server visualization application within a multimedia document in a distributed hypermedia environment, such as the World Wide Web.

**5. Ongoing/Future work**

We have begun working on several extensions and improvements on the above software system:

**5.1 MPEG Data Compression**

The data transferred between the visualization servers and the clients constitute the exact byte streams computed by the servers, packaged in the XDR machine independent format. One way to reduce network transferring time would be to compress the data before delivery. We propose to use the MPEG compression technique, which will not only perform redundancy reduction, but also a quality-adjustable entropy reduction. Furthermore, the MPEG algorithm performs interframe, beside intraframe, compression. Consequently, only the compressed difference between the current and the last frames is shipped to the client.

**5.2 Generalized External-Application-to-Mosaic-Document-Page Display Interface**

The protocols specified in Table 1 are simple, and general enough to allow most image-producing programs be modified to display in the Mosaic document page. Our undertakings will be to extend the protein database (PDB) displaying program, and the xv 2D image processing program, to be Mosaic PDB visualization server, and Mosaic 2D image processing server.

**5.3 Multiple Users**

With multiple users, the VIS/Mosaic distributed visualization system will need to manage the server resources, since multiple users utilizing the same computational servers will slow the servers down significantly. The proposed solution is depicted in Fig 4. The server resource manager will allocate servers per VIS client request only if those servers are not overloaded. Otherwise, negotiation between the resource manager and the VIS client will be necessary, and, perhaps the resource manager will allocate less busy alternatives to the client.

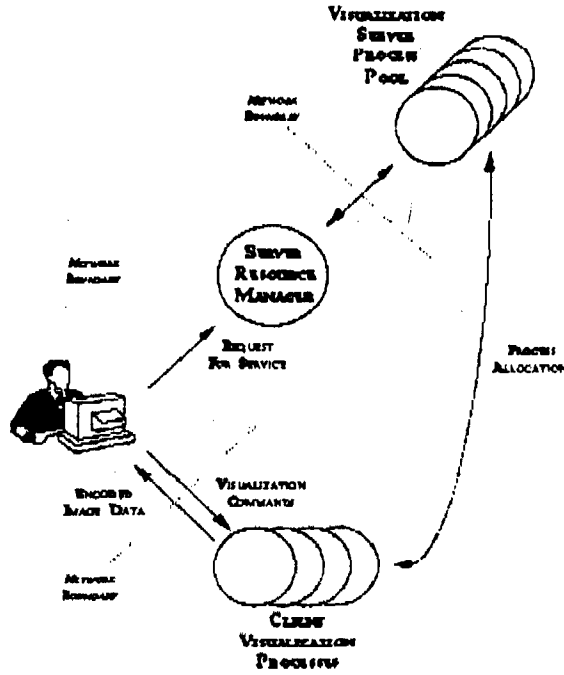


Fig 4: Server Resource Management

### 5.4 Load Distributing Algorithm

Since the load distributing algorithm in the current VIS implementation is not the most optimal load distribution solution, we expect to see some improvement in the future implementation, which will be using sender-initiated algorithms, described in [Shivaratri].

### 6. Conclusions

Our accomplishment takes the technology of networked multimedia system (especially the World Wide Web) a step further by proving the possibility of adding new interactive data types to the W<sup>3</sup> servers and clients, and coordinating the execution of the applications that handle them with the W<sup>3</sup> clients. The addition of the 3D volume data object in the form of HDF to the W<sup>3</sup> is welcomed by many medical researchers, for it is now possible for them to view volume datasets without a high-cost workstation, and accessing datasets in the W<sup>3</sup> fashion, through hypertext and hypergraphics links within an HTML page. As for the researchers who would like to share their findings with the world, they merely have to run a W<sup>3</sup> server that serves the HDF files, which in turn, can be easily created with the various import facilities available freely from NCSA.



Ang, Martin and Doyle: "Integrated Control of Distributed Volume Visualization Through the World Wide Web". Paper submission for Visualization '94.

7. References

Andreessen, Marc. "NCSA Mosaic Technical Summary", FTPed from ftp.nesa.uiuc.edu, Revision 2.1, May 8, 1993.

Argiro, V. "Seeing in Volume", Pixel, July/August 1990, 35-39.

Avila, R., Sobierajski, L. and Kaufman A., "Towards a Comprehensive Volume Visualization System", Visualization '92 Proceedings, IEEE Computer Society Press, October 1992, 13-20.

Bloomer, John, "An RPC Case Study: Networked Ray Tracing", Power Programming with RPC, O'Reilly & Associates, Inc., Sept 92, 409-451.

Brinkley, J.F., Eno, K., Sundsten, J.W., "Knowledge-based client-server approach to structural information retrieval: the Digital Anatomist Browser", Computer methods and Programs in Biomedicine, Vol. 40, No. 2, June 1993, 131-145.

Broering, N. C., "Georgetown University, The Virtual Medical Library," Computers in Libraries, Vol. 13, No. 2, February 1993, 13.

Drebin, R. A., Carpenter, L. and Hanrahan, P., "Volume Rendering", Computer Graphics, Vol. 22, No. 4, August 1988, 64-75.

Flanders, B., "Hypertext Multimedia Software: Bell Atlantic DocuSource". Computers in Libraries, Vol 13, No. 1, January 1993, 35-39.

Gelerg, L., "Volume Rendering in AVS5", AVS Network news, Vol. 1, Issue 4, 11-14.

Giertsen, C. and Petersen, J., "Parallel Volume Rendering on a Network of Workstations", IEEE Computer Graphics and Applications, November 1993, 16-23.

Jäger, M., Osterfeld, U., Ackermann, H, and Hornung, C., "Building a Multimedia ISDN PC", IEEE Computer Graphics and Applications, September 1993, 24-33.

Kaufman, A., Cohen, D., and Yagel, R., "Volume Graphics", Computer, July 1993, 51-64.

Kiong B., and Tan, T., "A hypertext-like approach to navigating through the GCG sequence analysis package", Computer Applications in the Biosciences, Vol. 9, No. 2, 1993, 211-214.

Levoy, M., "Display of Surfaces from Volume Data", IEEE Computer Graphics and Applications, Vol. 8, No. 5, May 1988, 29-37.

Lorenson, W., Cline, H.E., "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", Computer Graphics, Vol. 21, No. 4, July 1987, 163-169.

Mercurio, F., "Khoros", Pixel, March/April 1992, 28-33.

Narayan, S., Sensharma D., Santori, E.M., Lee, A.A., Sabherwal, A., Toga, A.W., "Animated visualization of a high resolution color three dimensional digital computer model of the whole human head", International Journal of Bio-Medical Computing, Vol 32, No. 1, January 1993, 7-17.

Nickerson, G., "WorldWideWeb Hypertext from CERN", Computers in Libraries, Vol. 12, No. 11, December 1992, 75-77.

*Ang, Martin and Doyle: "Integrated Control of Distributed Volume Visualization Through the World Wide Web". Paper submission for Visualization '94.*

✓ Obraczka, K., Danzig, P, and Li, S., "Internet Resource Discovery Services", Computer, Vol. 26, No. 9, September 1993, 8-22.

Pommert, A., Riemer, M., Schiemann, T., Schubert, R., Tiede, U., Hoehne, K-H, "Methods and Applications of Medical 3D-Imaging", SIGGRAPH 93 course notes for volume visualization, 68-97.

Robison, D., "The Changing States of Current Cites: The Evolution of an Electronic Journal", Computers in Libraries, Vol. 13, No. 6, June 1993, 21-26.

Shivaratri, N.G., Krueger, P., and Singhal, M., "Load Distributing for Locally Distributed Systems", Computer, December 1992, 33-44.

Singh, J, Hennessy, J. and Gupta A., "Scaling Parallel Programs for Multiprocessors: Methodology and Examples", Computer, July 1993, 42-49.

Story, G., O'Gorman, L., Fox, D, Schaper, L. and Jagadish, H.V., "The RightPages Image-Based Electronic Library for Alerting and Browsing", Computer, September 1992, 17-26.

VandeWettering, M., "apE 2.0", Pixel, November/December 1990, 30-35.

Woodward, P., "Interactive Scientific Visualization of Fluid Flow", Computer, Vol. 26, No. 10, June 1993, 13-25.

Zandt, W.V., "A New 'Inlook' On Life", UNIX Review, Vol 7, No. 3, March 1989, 52-57.



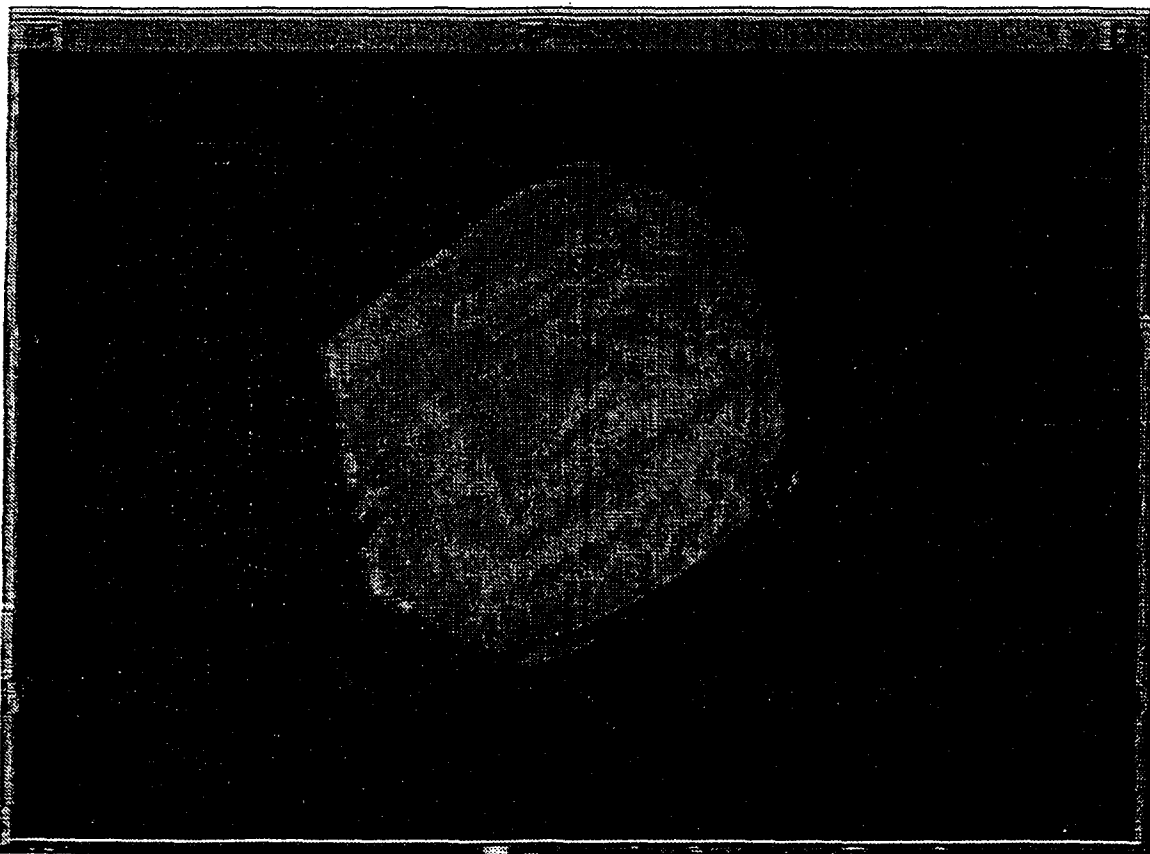
## Attachment B

M.D. Boyle, et. al., *The Virtual Embryo: VR Applications in Human Developmental Anatomy*, presented at "Medicine Meets Virtual Reality II, Interactive Technology and Healthcare: Visionary Applications for Simulation, Visualization & Robotics," sponsored by the UCSD School of Medicine and the Advanced Projects Research Agency, San Diego, CA, **January 27, 1994.**

### Video Presentation Transcript:

This is a status report of some of the work that's been accomplished during the first years of the Visible Embryo Project.

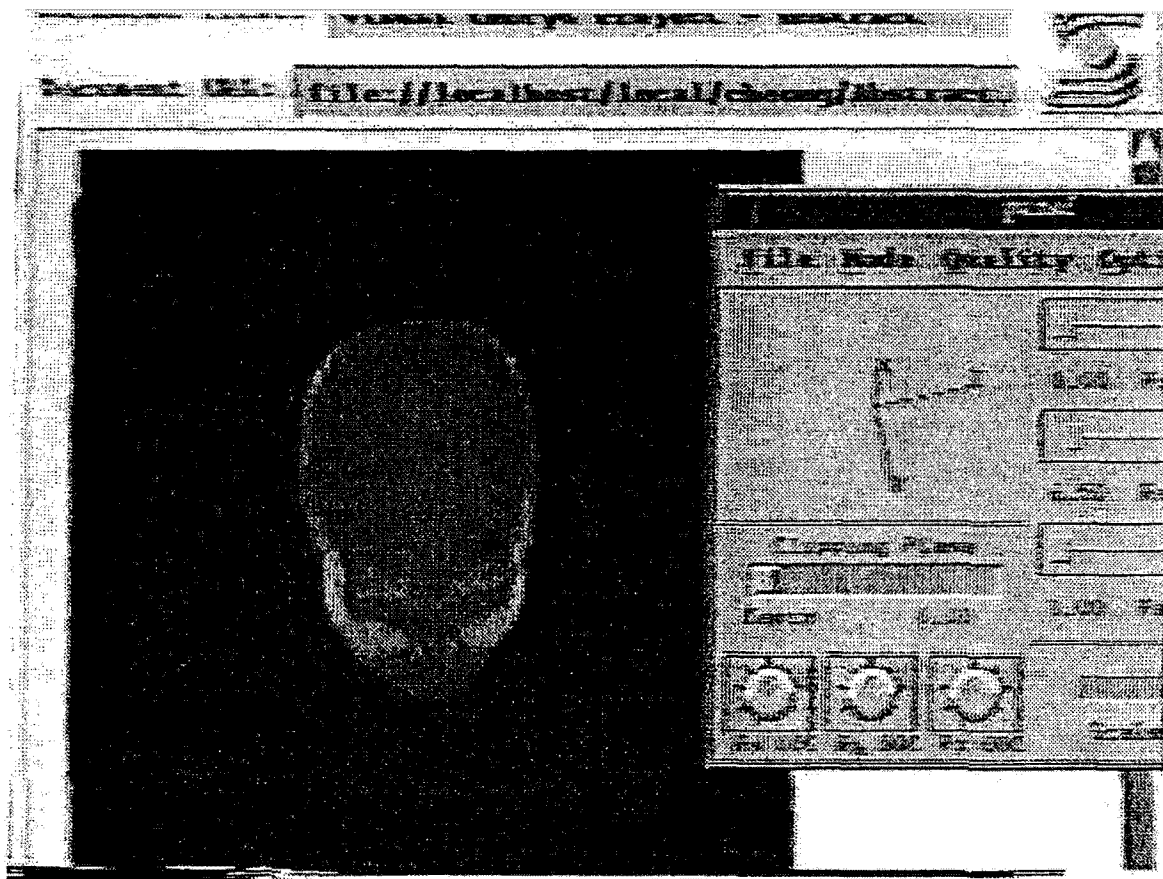
One of our first tasks was to develop some volume visualization software that we could use for imaging and analysis of the embryo reconstructions that we planned to create during the full term of the embryo project. One thing that was an absolute requirement was that this software be able to distribute its computational load across a network of graphics computers that weren't necessarily all in the same place. Basically we wanted to be able to have computers that could be all over the country connected by high-speed networking, able to contribute to a computation of three-dimensional datasets.



What you see here is a package called "VIS," which was developed in our group, for three-dimensional volume visualization. This is a very portable

package that has been generated using as generic code as possible, although this particular image that you see here is running on a Silicon Graphics Reality Engine II, which is optimized for volume visualization. We need to use that kind of high-speed optimization to accomplish the real-time interactivity that we need to accomplish for this project. And as you see, as this rotates, and it starts rotating faster and faster, only a very powerful graphics - basically a graphics supercomputer - can accomplish this much computation on a three dimensional dataset. One of our goals is to allow anybody on the Internet with a very low-level access workstation to accomplish this kind of interactivity through their network connection, and the way that we do that is through a client-server architecture where we have a very powerful server computer accessed by a very low-end client machine.

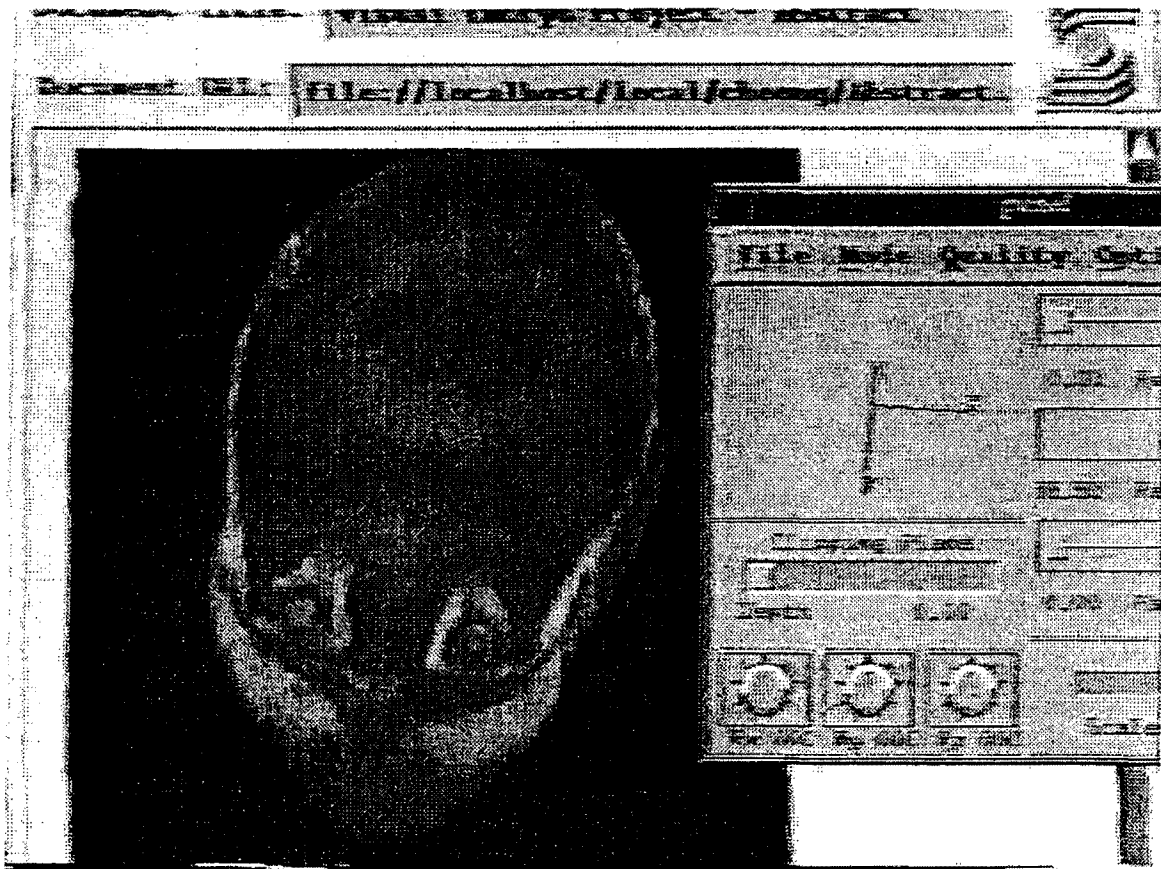
We decided early on to use NCSA's Mosaic program and the World Wide Web to integrate access to this system. One problem with Mosaic is, as it exists today, is that the images within Mosaic are typically static or passive-playback images.



What you see here is an enhancement that we've created to the Mosaic interface and control software that allows the embedding of a dynamic real-time

visualization application within a Mosaic document. You see a head, a volume MRI model of a human head, that's being rotated in real-time interactively by the viewer. You see a little panel to the right which is, it's a control panel that's popped up - it's really external to Mosaic itself but it can talk to the internal control programs that drive the Mosaic client - and allow you to interactively control the display from within Mosaic. This is actually controlling the volume visualization software that you just saw a few minutes ago.

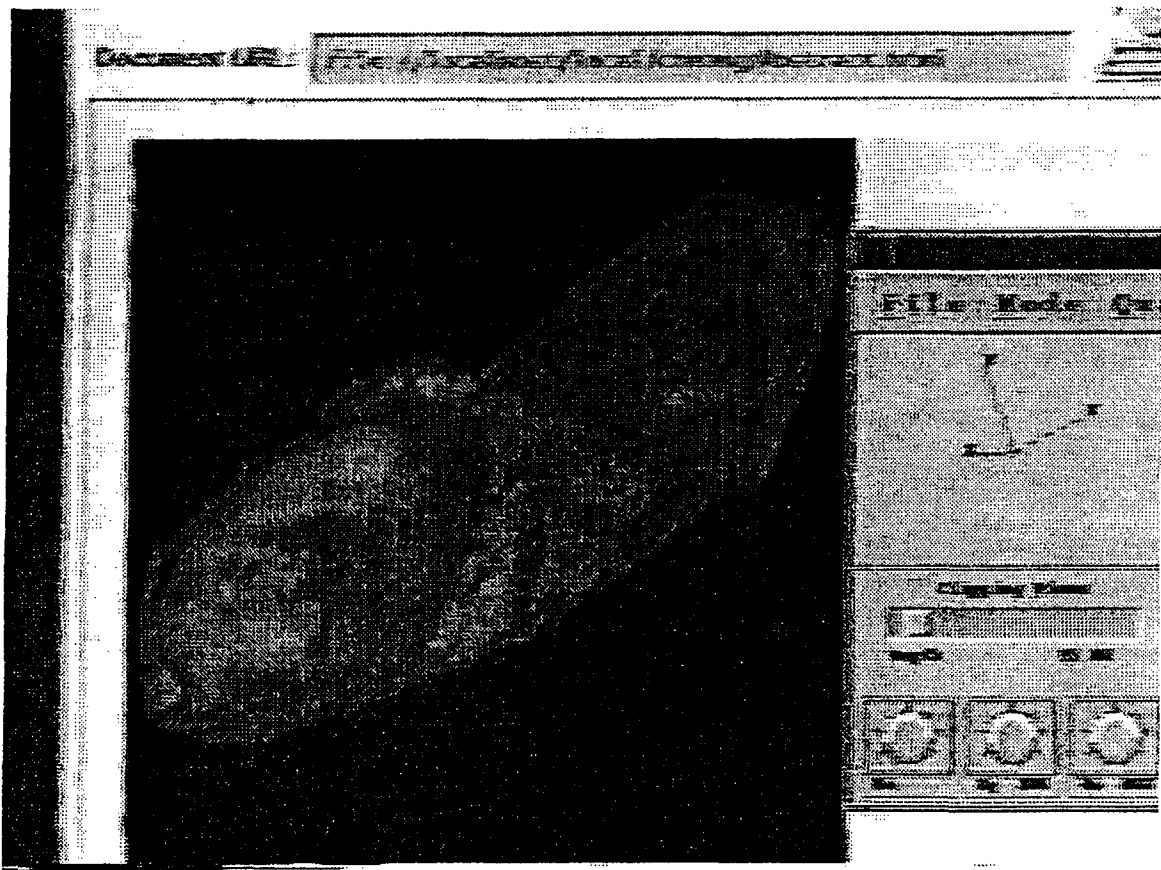
By moving around the controls in the control panel we can do things like rotate, we can control slices through the dataset, and so on. As you can see, there's rotation in x, y, and z planes. We can also compute arbitrary oblique sections through the data and look at the internal anatomy. Here we can see the brain of this individual; we can rotate and view that section of the dataset from a variety of vantage points. There are zooming capabilities that allow us to zoom up on the data or zoom back to look at things in more or less detail, and you can see here that once zoomed, we're moving our cutting plane down through the dataset and looking at more and more inferior levels.



Normally graphics in Mosaic are static, as I said, but this embedding of graphic applications within Mosaic is really going to form the basis of how we integrate information access through the Visible Embryo Project. What you are about to

see is our prototype system of a Visible Embryo Mosaic document that has embedded realtime visualizations within it.

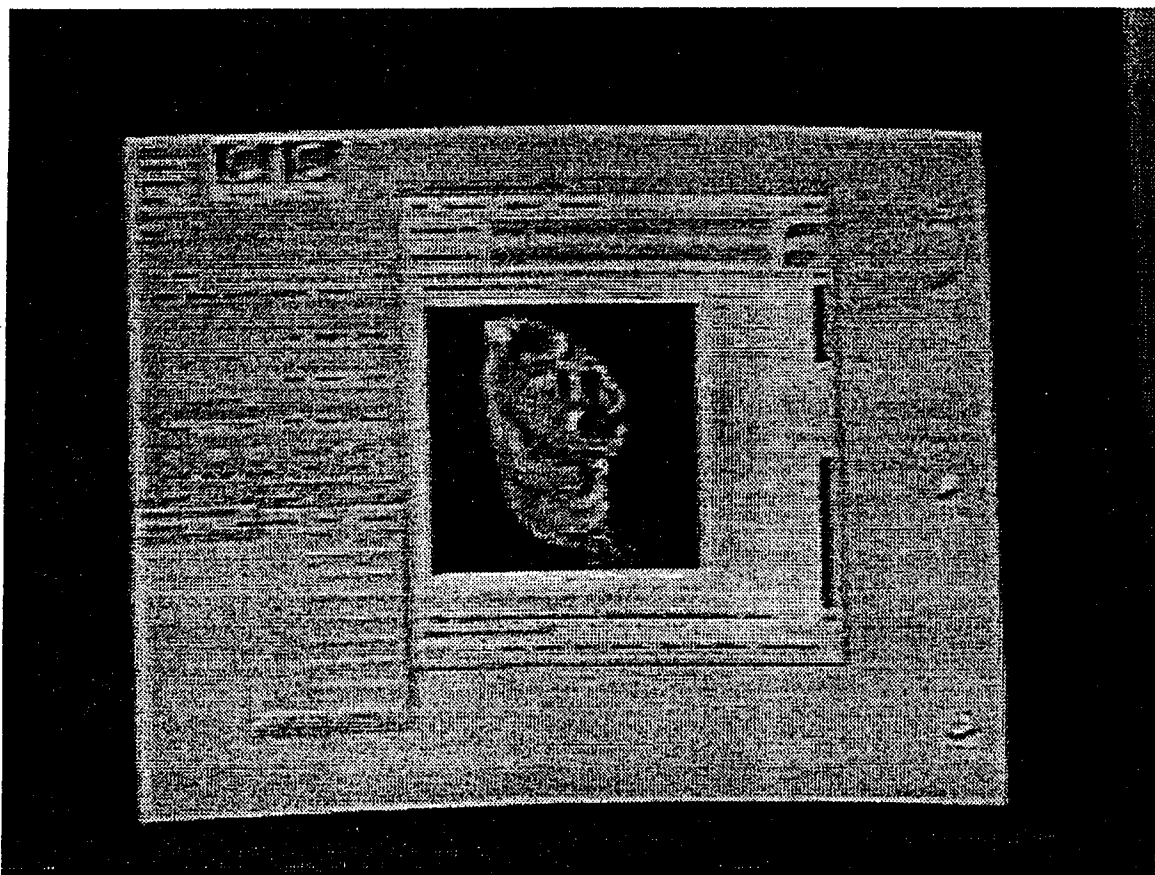
We just loaded the Visible Embryo Mosaic page, the system is about to scroll down this page - it is an abstract about the Visible Embryo Project - and then we see a window. It looks like a static image just like is normally found within the World Wide Web databases that you can access today through Mosaic, but you can see that suddenly, by moving controls on the control panel, we can zoom in and see that this is a reconstruction of a seven week old human embryo. This is a reconstruction from approximately 2900 serial cross sections of an embryo sectioned about in the 1930s. It's part of the Carnegie Collection of Human Embryology.



We're looking at this in volume visualization mode, we can rotate the embryo around, we can see internal structures, neurological structures; just in the lower abdomen area, we can see the liver, the arms are very evident - so we are actually looking through the dataset. We can also slice through this dataset obliquely, and look at the internal anatomy that way as well. We can load a volume visualization table that allows us to interactively enter tissue characteristic numbers that control the translucency, transparency, or opacity of various ranges of voxal intensity. And what we've done now is we've made the

exterior of the embryo a little more opaque so that as we rotate around with an oblique cutting plane we can see the difference between the cutting plane and the exterior of the embryo a little better. So now we're looking, slicing - we've rotated the embryo to an inferior view and we're looking up at the embryo from below. And we can see, we've just gone through the heart region, we're going through the liver, we're moving inferiorly and we start to get to an area where we can see the herniated gut.

Now the real key to all of this is that these are embedded visualizations. We're actually creating documents that are - I guess you'd call them currently compound documents where you have the traditional type of information, but you've also got, within that document, links to the raw data rather than just pictures generated from data. This allows you to tie together representations of data with the actual data themselves as well as with notes and different kinds of descriptive textual information based on that data.



Our basic objective here is really to create what we're calling a national meta-center which is going to be a computational resource for the entire nation that allows people interested in many different areas, including developmental biology, also multi-dimensional imaging and high-speed networking, and parallel computing. All these people can access this database. And the parallel

nature of the computation that's taking place is invisible to the user. They log in through a Mosaic window, and that window is giving them very high-performance control of interactive visualizations of datasets. By scrolling the window, we can see that this is actually embedded within the Mosaic document.

During a recent demonstration of this technology at the corporate briefing center at Silicon Graphics Corporation in Mountain View, California, I discussed some of the implications of this technology for researchers of human genetics.

-----

"We're also looking at using these models as a basis for creating three-dimensional maps of gene expression, which is a way to correlate the findings of the Human Genome project within a context that everyone uses. It sort of sets up a standard space within which everybody can report their findings, so that you can finally have some way of comparing studies that happen in different laboratories.



If you're studying the three-dimensional distribution of gene expression of a gene relating to heart development, what you do now is you have a little fluorescent marker that glows under an ultraviolet light, and you use confocal

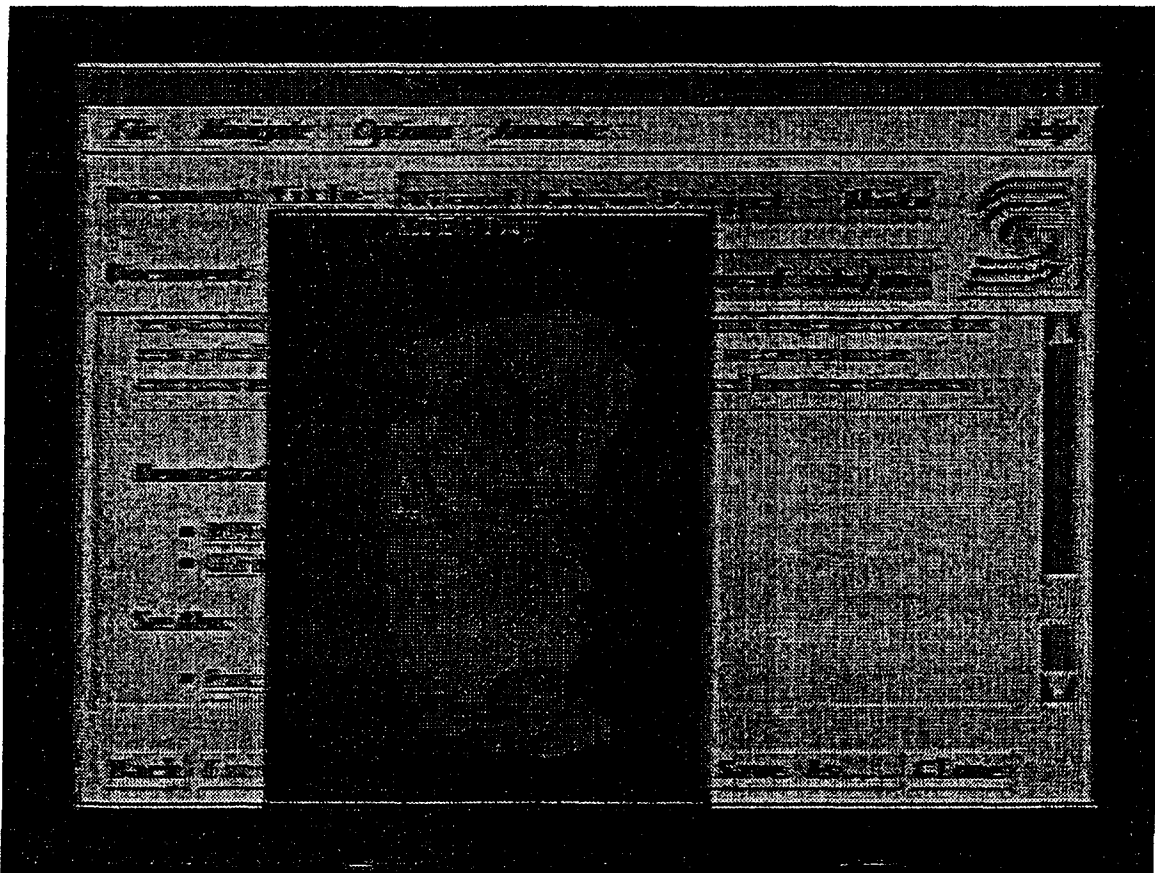


microscopy to develop a three-dimensional model of it, and then you say, well, it's on here and it's on there and it's on there and you try to describe it in anatomical terms but it is a qualitative description, right?.

But here there would be a standard anatomy space that people could use to describe their findings so that they could, rather than say, yeah, we saw five different studies that said that this was expressed at the bifurcation of the aorta with the Common Carotid artery - you don't have to do in terms of verbal descriptions, you can do it in terms of a true measurable Cartesian coordinate system.

-----

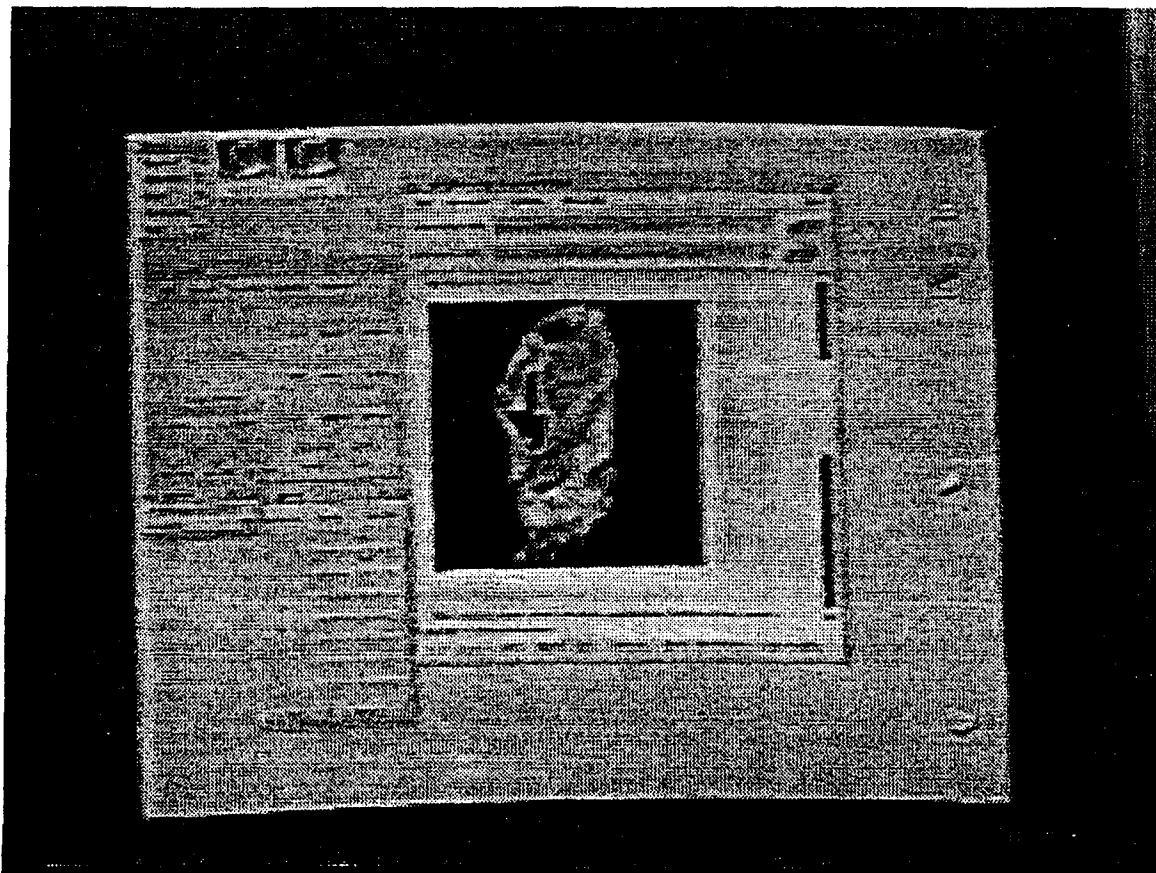
If you take your current version of Mosaic, the kind that is accessible for free through the Internet today, and log onto our home page of the Visible Embryo Project, what you'll see is a series of multi-media documents that basically give you information about the status of the Visible Embryo Project and the status of our current proposal development efforts. You'll be able to load MPEG movies of visualizations of human embryos, as you can see here.



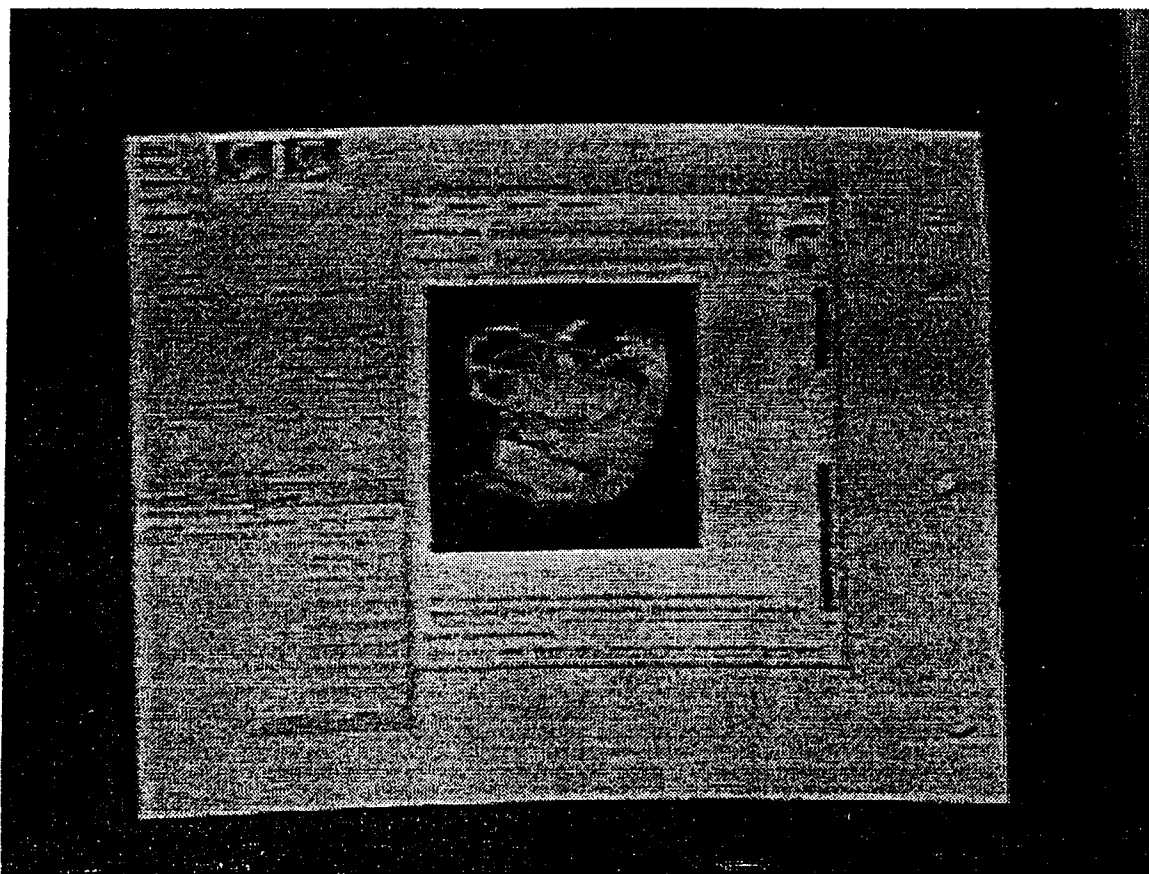
One thing you should keep in mind is that this little MPEG movie is just a canned movie, it's not interactive. Once you hit Play it just goes and it plays and then it goes away, but you can't stop it and interact with it, and rotate that embryo, for instance, to different vantage points.

Also available is an image that shows some of the early work, some screen shots representing the volume visualization tool current as of about last summer. We've come much farther and in fact a lot of the video that you saw earlier in this presentation shows you the current status of this volume visualization package. Last summer, that imaging package was separate from Mosaic, as you see it's off to the right, and Mosaic could just call it but couldn't actually embed visualizations. Now, everything is tied together into a single multi-media document.

You'll also see articles that relate to the Visible Embryo Project. This project has been going on for several years now, mostly on the coattails of other research projects, collaborators funding it wherever they could find the money. But already a significant body of literature is starting to be built up around this project.



Of course, in the very near future, you'll be able to log on through an enhanced version of Mosaic. **We're working together with the National Center for Supercomputing Applications on enhancing the standard release of Mosaic to allow these capabilities.** And you'll be able to access interactive dynamic visualizations that are being served by a network of high-end supercomputers across the country. Even if you're only accessing the system with a machine like a Macintosh or PC, you'll still be able to access the power of these supercomputers from your own location.



What I've attempted to demonstrate in the last several minutes in this presentation is that there's been a considerable amount of work already done in this project that we call the Visible Embryo Project. Many collaborators across the country have worked together to create a set of enabling technologies to allow this project to accomplish its goals. The Visible Embryo Project represents an effort to serve the needs of both the biology community as well as the information science community, in that we are attempting through current applications in information technology to break through barriers that have prevented biological researchers from asking and answering questions about the most fundamental mechanisms of human growth and development. We're also creating a technology development testbed for the information sciences that will allow researchers to push the envelope, so to speak, of information technologies to their very limits.

UNIVERSITY OF CALIFORNIA, SAN DIEGO  
OFFICE OF CONTINUING MEDICAL EDUCATION  
DISCLOSURE STATEMENT

It is the policy of the UCSD Office of Continuing Medical Education to insure balance, independence, objectivity, and scientific rigor in all its individually sponsored or jointly sponsored educational programs. All faculty participating in any UCSD sponsored programs are expected to disclose to the program audience any real or apparent conflict(s) of interest that may have a direct bearing on the subject matter of the continuing education program. This pertains to relationships with pharmaceutical companies, biomedical device manufacturers, or other corporations whose products or services are related to the subject matter of the presentation topic. The intent of this policy is not to prevent a speaker with a potential conflict of interest from making a presentation. It is merely intended that any potential conflict should be identified openly so that the listeners may form their own judgments about the presentation with the full disclosure of the facts. It remains for the audience to determine whether the speaker's outside interests may reflect a possible bias in either the exposition or the conclusions presented.

MEDICINE MEETS VIRTUAL REALITY II  
**INTERACTIVE TECHNOLOGY & HEALTHCARE**  
January 27-30, 1994

PRESENTER'S NAME: Michael D. Doyle  
(please print or type)

I have no actual or potential conflict of interest in relation to this program.

Michael D. Doyle                      11-30-93  
Signature    (date)

I have a financial interest/arrangement or affiliation with one or more organizations that could be perceived as a real or apparent conflict of interest in the context of the subject of this presentation.

<u>Affiliation/Financial Interest</u>	<u>Name of Organization(s)</u>
Grant/Research Support	_____
Consultant	_____
Speakers' Bureau	_____
Major Stock Shareholder	_____
Other Financial or Material Support	_____

\_\_\_\_\_  
Signature    (date)

Your cooperation in complying with these guidelines is appreciated.

Please return this form as soon as possible to: Medicine Meets Virtual Reality II, P.O. Box 23220, San Diego, CA 92193  
Fax: 619-751-8842; Phone: 619-751-8841

# 906 PH Ex. 14





**UNITED STATES DEPARTMENT OF COMMERCE**  
**Patent and Trademark Office**  
 Address: **COMMISSIONER OF PATENTS AND TRADEMARKS**  
 Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKETT NO.
---------------	-------------	-----------------------	----------------------

08 / 324443

EXAMINER

D. DINH

ART UNIT	PAPER NUMBER
----------	--------------

2319

16

DATE MAILED:

**EXAMINER INTERVIEW SUMMARY RECORD**

All participants (applicant, applicant's representative, PTO personnel):

- (1) DUNG DINH (3) \_\_\_\_\_  
 (2) CHARLES KRUEGER (4) MICHAEL DOYLE

Date of interview 11/6/97

Type:  Telephonic  Personal (copy is given to  applicant  applicant's representative).

Exhibit shown or demonstration conducted:  Yes  No. If yes, brief description: \_\_\_\_\_

document OLE object showing "window overlaid" in document

Agreement  was reached with respect to some or all of the claims in question.  was not reached.

Claims discussed: all

Identification of prior art discussed: all

Description of the general nature of what was agreed to if an agreement was reached, or any other comments: cause of claim invention

"automatic invoke of external application to provide interactive control". Applicant argues the prior art does not teach or suggest this feature.

(A fuller description, if necessary, and a copy of the amendments, if available, which the examiner agreed would render the claims allowable must be attached. Also, where no copy of the amendments which would render the claims allowable is available, a summary thereof must be attached.)

1. It is not necessary for applicant to provide a separate record of the substance of the interview.

Unless the paragraph below has been checked to indicate to the contrary, A FORMAL WRITTEN RESPONSE TO THE LAST OFFICE ACTION IS NOT WAIVED AND MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW (e.g., items 1-7 on the reverse side of this form). If a response to the last Office action has already been filed, then applicant is given one month from this interview date to provide a statement of the substance of the interview.

2. Since the examiner's interview summary above (including any attachments) reflects a complete response to each of the objections, rejections and requirements that may be present in the last Office action, and since the claims are now allowable, this completed form is considered to fulfill the response requirements of the last Office action. Applicant is not relieved from providing a separate record of the substance of the interview unless box 1 above is also checked.

D. Dinh  
 Examiner's Signature





# 906 PH Ex. 15





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

on 12-23-97

TOWNSEND and TOWNSEND and CREW LLP

By [Signature]

PATENT

Attorney Docket No. 02307I-553

#18  
[Handwritten notes]

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: )  
MICHAEL D. DOYLE et al. )  
Application No.: 08/324,443 )  
Filed: 10/17/94 )  
For: EMBEDDED PROGRAM OBJECTS IN )  
DISTRIBUTED HYPERMEDIA )  
SYSTEMS )

Examiner: D. Dinh  
Art Unit: 2317

DECLARATION OF MICHAEL D. DOYLE  
UNDER RULE 131

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

I, MICHAEL D. DOYLE, hereby declare that:

1. I am a co-inventor of the subject matter disclosed and claimed in U.S. Patent Application No. 08/324,443.

2. The subject matter claimed in the above patent application was reduced to practice in this country prior to April 15, 1994, the filing date of the parent of the Koppolu reference cited by the examiner.

3. The reduction to practice of the claimed invention is evidenced by ATTACHMENTS A and B. ATTACHMENT A is a copy of a paper entitled "Integrated Control of Distributed Volume Visualization Through the World-Wide-Web", by Ang, Martin, and Doyle. This paper was submitted for publication prior to April 15, 1994. ATTACHMENT B is a transcript of the audio portion and still photographs of a video tape presented to an audience of scientists prior to April 14, 1994.

4. As stated in ATTACHMENT A, at page 5, paragraph 3.2, Mosaic (the browser) interprets the HTML <EMBED> tag included in a document to create a drawing area widget in a

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 2

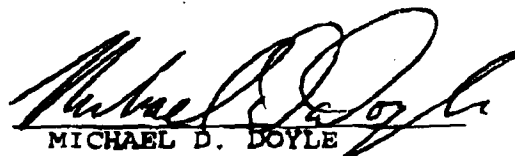
PATENT

document presentation and creates a shared window system buffer to receive visualization results. In addition, when the browser parses an <EMBED> tag in the document, the browser automatically launches the external application specifying the location of the visual object to render and identify the shared image buffer. The format and operation of an EMBED tag for 3D image data is described at paragraph 3.1.

5. As stated in ATTACHMENT B, starting at the bottom of page 2, interface and control software had been developed that allows the embedding of a visualization application within a Mosaic document. As is apparent from the photographs, the object is displayed and processed within the browser-controlled window. The visualization application is external to the hypermedia document displayed by the browser. Automatic launching of the external application when an HTML document is opened by the browser is depicted in the video.

I further 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 issuing thereon.

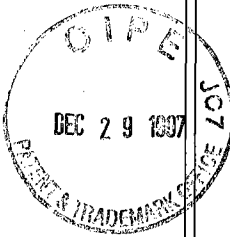
Dated: October 29, 1997.

  
MICHAEL D. DOYLE

CEX:db  
i:\cek\share\02307I\553\decl.02

# 906 PH Ex. 16





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

on 12-23-97

TOWNSEND and TOWNSEND and CREW LLP

By V. Sullivan

Attorney Docket No. 02307Y-553

*P. Gregory*  
#19 *Patent*  
1/24/98

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	)	
MICHAEL D. DOYLE et al.	)	Examiner: D. Dinh
Application No.: 08/324,443	)	Art Unit: 2317
Filed: 10/17/94	)	<u>RESPONSE</u>
For: EMBEDDED PROGRAM OBJECTS IN)	)	
DISTRIBUTED HYPERMEDIA	)	
SYSTEMS	)	

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

The following is responsive to the Office Action mailed August 25, 1997, setting a response period expiring on November 25, 1997.

REMARKS

Claims 1,2, 5, and 44, 45, 48 are rejected over the Applicants' disclosed prior art (Mosaic + HTTP + HTML + "World Wide Web), referred to herein as "Mosaic," in view of Koppolu et al. The Examiner is thanked for extending the courtesy of an interview to one of the inventors, Professor Doyle, and his attorney, Mr. Krueger.

THE EXAMINER'S REASONING REJECTING CLAIM 1

The Examiner states that Mosaic does not have embed text format specifying an external object which automatically invokes an external application to execute and enable interactive processing within a portion of the browser controlled window.

Mosaic provides display and interaction with an external object by launching an associated program in a separate window.

It is further stated that Koppolu teaches a method for in-place interaction with a contained object and permits automatic display and interaction of a linked object in a compound document (i.e., hypermedia document) within a portion of a window controlled by a container application (i.e., the browser).

The Examiner concludes that it would have been obvious to apply the teaching of Koppolu to the processing of the hypermedia document of Mosaic because it would have provided a more integrated and expandable system for processing/viewing of linked objects in the hypermedia documents and reduces clustering of the window display. It is also concluded that since HTML is a text tag structure document encoding method, it is apparent that Mosaic as modified would have had a text tag for indicating links to an in-place interactive object.

#### THE INVENTION OF CLAIM 1

A browser application 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 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 an executable application external to the hypermedia document.

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

When the automatically-invoked application executes, it displays the object and enables interactive processing of said object within a display window created within the portion of the hypermedia document being displayed.



SUMMARY OF THE CLAIM 1 ARGUMENT

The first part of the argument demonstrates that the cited art does not disclose or suggest several of the elements and limitations recited in claim 1. The second part of the argument demonstrates that the purpose, functions, and technology utilized in Mosaic and Koppolu are so different that, even if the missing features were disclosed in isolation, it would not have been obvious or even feasible for a person of skill in the art to combine the teachings of the reference to realize the claimed invention.

Turning to the first part of the argument, there is no disclosure or suggestion in Mosaic or Koppolu of automatically invoking an external application when an embed text format is parsed. Each of those references require user input, specifically clicking with a mouse pointer, to activate external applications to allow display and interaction with an external object.

As will be set forth in detail below, the object handlers in Koppolu (OLE) do not automatically invoke an external application to permit display and interaction with an object.

Turning to the second part of the argument, the functions and purposes of Mosaic and Koppolu (OLE) are fundamentally different.

On the one hand, Mosaic is directed to retrieving and displaying HTML documents received over the WWW, and to providing an efficient system for retrieving documents having links embedded in an HTML document being viewed. There is no provision within Mosaic for **editing** a document being viewed.

On the other hand, Koppolu (OLE) is directed to providing **editing** capabilities on a single user workstation for a compound document having embedded or linked objects which must be processed by applications external to a container object. When a compound document is opened, static bit maps of embedded or linked objects are displayed. An object to be edited must be selected by the user to invoke an external application to interactively process the object.

The different functions and purposes of Mosaic and Koppolu (OLE) are reflected in the different document structure. In Mosaic, since HTML documents are designed to be platform independent, the document structure is simple ASCII text. A browser parses a received document to identify HTML tags which specify various aspects of the document's appearance and links to other documents. In Koppolu, a container application creates a complex file structure which is utilized to render a document. There is no text parsing in Koppolu to render the compound document.

The complex binary data file structure of an OLE compound document file can only be created and utilized by platform-dependent OLE enabled applications. In contrast, an HTML document can be created utilizing the simplest ASCII text editor and can be viewed by any browser on any platform connected to a network. Furthermore, OLE is designed for manipulating data which is entirely resident on the user's machine, while Mosaic is designed for viewing data files which are typically remotely-networked. OLE teaches no facility for working with remotely-networked components, and, in fact, employs an architecture which would most likely have hindered Mosaic's capabilities in this regard. (For a detailed description of how Koppolu works see below, Argument at II.C.2.)

Thus, a person of skill in the art, considering the teachings of Mosaic and Koppolu (OLE) at the time of the invention, would not have conceived of the present invention. To so combine the references would require ignoring the purposes, functions, and document structures of both references.

As stated by Judge Markey:

"The third fundamental error was the refusal... to credit the real world environment surrounding the inventions disclosed in the reference patent and in the patents in suit.

It must be remembered that the Examiner is required to consider the references in their entirety, i.e. including those portions that would argue against obviousness ..."

The Court specifically stated "I don't care how things work or don't work." But "how things work" is critical to encouragement of every research and development activity, and every advancement of the arts useful to the public, the very purposes of the patent system.

It was a refusal to consider "how things work" that caused the court to cite isolated minutia from various references, while ignoring critically important structural **distinctions** that significantly affect the different achievements of which the references and claimed structures are capable. Panduit Corp. v. Dennison Manufacturing Company, 227 USPQ 337, 345 (CAFC 1985).

Turning to the real world environment, as discussed at the interview, Microsoft Corporation had the rights and technology to both Mosaic and OLE but did not conceive of the claimed combination or attempt to combine the features of Mosaic and OLE. Instead, Microsoft developed ActiveX, which does not employ a combination of OLE and the teachings of Mosaic, but utilizes the claimed features of the invention, including an embed tag text format as defined, which is parsed to cause the automatic execution of an external application to display and interactively process an external object within the browser controlled window. These claimed features have been adopted by the entire industry and lauded as a major breakthrough in Internet and computing technology.

All the above factors compel a conclusion that the claimed combination is patentable. As described above, the browser application is not analogous to an OLE container application and an HTML document is not analogous to an OLE compound document. Equating a browser and an HTML document to container application and an OLE compound document requires impermissibly selecting isolated features from Mosaic and OLE, utilizing Applicants' disclosure as a roadmap, while ignoring important structural and functional differences relating to the purposes and implementations of the systems disclosed.

DETAILED ARGUMENT

TABLE OF CONTENTS

- I. The proposed combination does not show the features of the Applicants' invention. There is no suggestion or teaching in Koppolu (OLE) of modifying Mosaic to automatically invoke an external application, when an embed text format is parsed, to display and interactively control an object in a display window in a document being displayed in a browser controlled window. (Pg. 7)
  - A. Mosaic. (Pg. 7)
  - B. Koppolu. (OLE) (Pg. 8)
    - 1. General. (Pg. 8)
    - 2. OLE Object Handlers. (Pg. 12)
  - C. Conclusion. (Pg. 14)
  
- II. The selection and combination of features of the claimed invention from references such as Mosaic and Koppolu is only possible by disregarding the striking differences in the structure and operation of the references. Such disregard is only possible by utilizing the Applicants' disclosure as a roadmap to randomly select and combine features from the references. (Pg. 15)
  - A. Introduction. (Pg. 15)
  - B. The Applicable Law. (Pg. 15)
    - 1. Impermissible Hindsight. (Pg. 15)
    - 2. Improper Combination of References. (Pg. 16)
  - C. How the References Work. (Pg. 16)
    - 1. Mosaic. (Pg. 16)
    - 2. Koppolu. (OLE) (Pg. 19)
  - D. How Mosaic and Koppolu (OLE) Work Teach Away from the Claimed Combination. (Pg. 24)
  - E. What Workers of Skill in the Art, Aware of and in Possession of Rights to Mosaic and Koppolu (OLE), Actually Did. (Pg. 25)
    - 1. Owners of Mosaic and OLE did not Combine, Adopted Applicant's Invention Instead. (Pg. 25)

- F. How the Real World Responded to Applicants' Invention. (Pg. 27)
1. Commercial Success of Products Adopting Claimed Invention. (Pg. 27).
  2. Industry Recognition of the Importance of the Claimed Invention. (Pg. 29)

CLAIM 1 ARGUMENT

PART I. The proposed combination does not show the features of the Applicants' invention. There is no suggestion or teaching in Koppolu (OLE) of modifying Mosaic to automatically invoke an external application, when an embed text format is parsed, to display and interactively control an object in a display window in a document being displayed in a browser controlled window.

A. Mosaic

Mosaic parses a received document, passively displays links from text or picture elements of a first hypermedia document to other external data objects, and retrieves information identified by a link when a user interactively selects the 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 interactively invoke an external application to open a new window to display file types that cannot be displayed by Mosaic (helper applications).

Mosaic launches helper applications in response to a user's interactive 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). (Application at pg. 3, line 30).

Many viewers exist that handle various file formats such as ".TIF," ".GIF," etc. When a user commands the browser program to invoke a viewer program, typically by clicking on an

anchor with a mouse, the viewer is launched as a separate process. The viewer program displays the full 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 document 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.

## B. Koppolu (OLE)

### 1. General

Koppolu's OLE system provides a method for interacting with a contained object within the window environment of a container application of a container document. When a user interactively selects a bitmapped image of the contained object, the method integrates a plurality of the server resources with the displayed container window environment. When the user then interactively activates the previously-selected object image, OLE invokes a server application to process the original data referenced by the contained object image. Since OLE was designed for integration of very large programs, a facility is provided whereby the server application can conserve space on the computer display by integrating the server application's menu and GUI system with that of the container application.

The designers of OLE did not attempt to include the capability to have the entire server application (data display, menus and GUI) run within the embedded window, (as is enabled by the Applicants' invention), since such a system would have been

impractical with large monolithic applications, and to incorporate such a feature would have required a total redesign of the object activation and event handling system of OLE.

Furthermore, OLE does not parse text tags in the document in order to render the document (as required by the Applicants' claims). OLE uses an internal binary pointer mechanism to provide for accessing the referenced source data structures that contain the document objects. The actual linking mechanism between the container document and the containee server application is coordinated by the operating system's registry database. Therefore, the document text is not used to determine contained the object's type. At the time of initial object selection by the user, and prior to server application launching, OLE references the operating system's global registry database in order to identify which server application is related to a particular data object and to determine what interactive operations are provided by the relevant server application.

Koppolu et al. describes a method referred to as in-place interaction, for integrating the functionality of two application programs within the context of a compound document. It allows the user to interact with OLE-embedded or OLE-linked data within the context of the compound document. This method is embodied in the Object Linking and Embedding (OLE) API from Microsoft Corporation.

Koppolu defines a **compound document** as "a document that contains information in various formats. For example, a compound document may contain data in text format, chart format, numerical format, etc." Embedded data is then defined as follows: "Data that is copied from the clipboard into a compound document is referred to as '**embedded**' data. The word processing program treats the embedded data as simple bitmaps that it displays with a BitBit operation when rendering the compound document on an output device." Koppolu goes on to define "**linked**" data by stating: "Some prior systems store links to the data to be included in the compound document rather than actually embedding the data. When a word processing system pastes the data from a clipboard into a compound document, a link is stored in the

compound document. The link points to the data (typically residing in a file) to be included. These prior systems typically provide links to data in a format that the word processing system recognizes or treats as a presentation format (such as a bitmapped image of the rendered data)."

When a document containing embedded or linked data is loaded into the application which displays the compound document, a bitmapped **picture** or a vector **drawing** (metafile) of the data (the "**object presentation format**," Col. 33, line 60) is typically displayed at a location in the document, and the user can indicate selection of the object to the word processor application by, for example, interactively single-clicking on the picture of the budget data which is shown in the document. An example is given of a spreadsheet object, containing budgeting data, that is embedded in a word processor document.

The embedded or linked data is made accessible to the user through a process called "activation in-place." As Koppolu describes, "when embedded or linked (contained) data is activated in place, the menus of the application that implements the contained data are merged with the menus of the application that implements the compound document to create a composite menu bar." Koppolu states, "when the user decides to edit the budgeting data, **the user selects the spreadsheet object** and requests the word processing application to edit the object (e.g., by double clicking on the object using the mouse). **The word processing application then launches the spreadsheet application**, requesting that it edit the spreadsheet object. The spreadsheet application negotiates with the word processing application to edit the spreadsheet object, using windows and the menu bar of the word processing application. This process of launching the server application is called "activation." In this example, the word processor is described as a "container application" and the spreadsheet application is termed a "server application." It is important to note here that the container document does not automatically launch the server application at document-rendering time, but rather, **the user must issue two separate interactive**



**commands** after the time of document rendering in order to cause the container application to invoke the server application.

OLE provides a limited and specific set of methods to allow user interaction with embedded or linked objects within compound documents. As stated in "Inside OLE 2.0" (hereafter referred to as "Brockschmidt") by Kraig Brockschmidt, published by Microsoft Press in 1993 (ISBN 1-55615-618-9):

"Creating new objects and displaying or printing them doesn't do anything more for us than a static bitmap or metafile. One of the most important features of compound documents under OLE is that you can activate the object, that is, ask it to execute one of its verbs and thus perform some action such as playing a sound or opening its data for editing. This ability for activation is the only thing that separates an embedded or linked object from a static one. But some way or another you have to allow the end-user to select the actions available for that object which varies object to object. For this reason the OLE 2.0 user interface defines two methods to allow end-users to invoke verbs.

The first method is to execute what is known as the primary or default verb when the object is double-clicked with the left mouse button. The exact meaning of this primary verb is defined by the object, not by the container, so the container blindly tells the object to execute without any knowledge of what will happen. The second method, of which the container is equally ignorant, is to provide a menu item on the container's edit menu that lists all the available verbs on the currently selected object. When the end-user selects one of these menu items, the container again blindly tells the object to execute a verb. However, the container can ask the object to perform known actions by using predefined verbs, although these options are generally not shown directly to the end user."

Brockschmidt sums up the three possible states of a containee object in the Summary section of Chapter 9 by stating:

"An object in a container document may have three states: passive, loaded, and running. A passive object exists entirely on disk and is not visible, printable, or available for any manipulation. When an end-user opens the document in which the object lives and the container application loads the object, it transitions to the loaded state where it may be seen and printed but not edited or otherwise manipulated in any way. **Only when the object is activated does it transition to the running state where the user may perform any number**

**of actions on that object, such as playing or editing the data."**

Thus, OLE was an advance that allowed a user to create and edit a single compound document while calling upon the capabilities of a plurality of large, monolithic stand-alone applications and having those applications share a single container graphical user interface environment, thereby conserving display space and allowing the user to focus more directly on the document editing process.

## 2. OLE Object Handlers

During Applicant's interview with the Examiner on November 6, 1997, the Examiner asserted that since OLE shows an object handler which is dynamic-link library (DLL) code that can automatically be invoked at document rendering time, and since this object handler DLL code may be considered to be part of the server application, then it would have been obvious to enhance Mosaic by providing such a DLL object handler that would be automatically invoked at document rendering time to provide display and interactive processing of the object within a window in the hypermedia document. Applicants respectfully assert that such reasoning is incorrect.

As Koppolu states, "the invoking of a server application can be relatively slow when the server application executes as a separate process from the container application. In certain situations, this slowness may be particularly undesirable, such as, for example, if the user wants to print a compound document that includes many containee objects." Such a problem is solved in the Koppolu system by providing special code that is loaded when the word processor application is launched, where that code provides a subset of the functionality of the full server application. An example would be code which allows printing of embedded or linked spreadsheet data, without having to start up the spreadsheet application itself. This code is called an **"object handler."**

Koppolu does not give much detail about the functioning of object handlers, which are shown to be implemented as dynamic

link libraries (DLLs). Full details on object handler implementation and functionality can be found in Brockschmidt.

Brockschmidt comments on object handlers (Chapter 11, section: "Why Use a Handler?"): "There are two main reasons for implementing an object handler to work with your local server: speed and portability. First, an object handler can generally satisfy most requests a container might make on an object such as drawing an object on a specific device or making a copy of the object in another IStorage. Object handlers may also be capable of reloading a linked file and providing an updated presentation to the container. **Object handlers do not, however, provide any sort of editing facilities** for the object itself."

If the entire server application is implemented as an in-process DLL, this is called an "in-process server." The above statement by Brockschmidt shows that the use of the term "object handler" relates specifically to a limited set of object-related facilities that can be automatically invoked by the container application at document rendering time, but which **do not include** capabilities for interactively processing object data. Interactive processing of object data can only be accomplished through interactive invoking of either an in-process server or a local server (standalone executable). In-process servers do allow editing capabilities for object native data, but these editing capabilities are invoked **only after the containee object has been interactively activated by the user**, as described below. This is further supported by the teaching of Koppolu that user interaction with containee objects is provided by OLE only after interactive activation of the containee object server by the user (Col. 7, lines 56-66).

Brockschmidt goes on to describe the problems associated with use of both object handlers and in-process servers. These include 1) limited interoperability, even across different versions of OLE and different versions of Intel processors, and 2) the lack of message loops in DLLs, drastically limiting use of interactive capabilities such as keyboard accelerators. According to Brockschmidt, "The other technical issue of an in-process server specifically (but not a handler) is that since

there is nothing that can ever run stand-alone (like a local server EXE can) **there is no possibility to provide linked objects.**" Brockschmidt explains this by pointing out that in-process DLLs cannot access files external to the compound document file. This limits the use of in-process servers to working with embedded (encapsulated) data stored **within** the container document file, rather than linked **external** data files.

To repeat, Brockschmidt clearly states that "an object handler can generally satisfy most requests a container might make on an object such as drawing an object on a specific device or making a copy of the object in another IStorage. Object handlers may also be capable of reloading a linked file and providing an updated presentation to the container. **Object handlers do not, however, provide any sort of editing facilities for the object itself.**" Brockschmidt goes on to emphasize: "When an end-user opens the document in which the object lives and the container application loads the object, it transitions to the loaded state where it may be seen and printed but not edited or otherwise manipulated in any way. **Only when the object is activated does it transition to the running state where the user may perform any number of actions on that object, such as playing or editing the data.**"

### C. Conclusion

The Examiner has stated that Mosaic does not have an embed text format specifying an external object which automatically invokes an external application to execute and enable interactive processing with a portion of the browser controlled window.

In view of the above, it is clear that the Koppolu (OLE) reference does not disclose or suggest the missing features. In OLE, when a compound document is opened, static pictures of included objects are rendered in presentation format. Invoking of an external application requires a user-activated selection of the object. The object handlers provide no interactive control of a displayed object.

PART II. The selection and combination of features of the claimed invention from references such as Mosaic and Koppolu is only possible by disregarding the striking differences in the structure and operation of the references. Such disregard is only possible by utilizing the applicant's disclosure as a roadmap to randomly select and combine features from the references.

A. Introduction

Mosaic was designed as a **network communications and information retrieval** tool for **viewing** static documents. OLE was designed to integrate the **document creation and document editing** capabilities of multiple **non-networked** monolithic applications whose primary purposes were to act as stand-alone applications, and to accomplish such integration primarily through extensive modification of the user interface of a single container application. OLE is based upon a **document-centric** paradigm, where the document creation and editing applications move around a single centered document, rather than moving the document around to each application separately. The applications shown in these two references were designed for **fundamentally different** purposes, and neither was designed to solve the problem that the claimed invention solves. There is no suggestion or motivation in either Mosaic or Koppolu's OLE system that would lead one to attempt a combination of the two to achieve the features of the claimed invention.

B. The Applicable Law

1. Impermissible Hindsight

The test for obviousness is whether the subject matter of the claimed invention would have been obvious to one skilled in the art at the time the invention was made, not what would be obvious to an Examiner after reading the Applicants' disclosure.

Panduit Corp. v. Dennison Manufacturing Company, 227 USPQ 337, 343 (CAFC 1985).

It is impermissible to ascertain factually what the applicant's did and then view the prior art in such a manner as to select from the art random facts only those which may be modified and then utilized to reconstruct the applicant's invention from the prior art. Panduit Corp. v. Dennison Manufacturing Company, 227 USPQ 337, 345 (CAFC 1985).

## 2. Combining References

A suggestion for combining the references must appear in the prior art. 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).

Additionally, the real world environment surrounding the references must be considered. The Examiner must consider "how things work" when considering whether the references would have made the claimed combination obvious at the time of the invention. It is necessary to consider what workers of ordinary skill in the art actually did. Panduit Corp. v. Dennison Manufacturing Company, 227 USPQ 337, 345 (CAFC 1985).

## C. How the Systems Described in the References Work.

### 1. How Mosaic Works.

The World Wide Web was designed as an information retrieval and hypertext document viewing system. A major priority in the design of the Web was that the documents would be viewable and similarly navigable irrespective of the type of computer on which the document was viewed. That is, the **look and feel** of the document would be similar on as many different types of computing platforms as possible. The document creation and dissemination model was **write once, publish many**. This refers to the fact that a Web document author would create a single document file and publish that file merely by making it accessible on an Internet server. An unlimited number of users could then retrieve and view that document by simply entering the

Internet address of the document (the URL) into their Internet-connected Web browsers. The browsers would use a simple request/response protocol to retrieve specified documents and related data from remotely-networked Web server programs.

In order to insure cross-platform uniformity of document appearance, the document was defined through the use of ASCII text, where specific text formats, otherwise known as "tags," would be used within the document text to specify various aspects of the document's appearance and linkages to other documents or related data. Each browser, therefore, incorporated a parser which would distinguish the formatting tags from the document's narrative text, classify those tags into pre-defined categories, break each tag into its basic components, and then invoke appropriate browser subroutines to respond appropriately to the meanings of the tag components. Although the browser subroutines were built from machine-specific native code, this text tag mechanism allowed the design of a variety of browsers for various computing platforms that could respond in similar ways to similar types of text tags, and therefore result in similar-appearing documents on dissimilar computers. A binary document data format was avoided in order to promote cross-platform compatibility, due to the variation in binary data handling methodologies on various different operating systems, and to simplify the requirements for document creation tools. All that a Web document author needs in order to create a Web document file is a simple ASCII text editor, which is a pre-existing application in all commonly-found operating system packages.

Because of the write-once-publish-many paradigm of the Web, all users would see the same basic document content that the document author originally designed. End users, the remotely-networked individuals using Web browsers, were allowed to view the document, but could not edit the source document in any way. To allow user-editing of the original document text would have been counter productive, since it would be contrary to the write-once-publish-many design philosophy of the fundamental Web architecture.

Mosaic was an advance that provided a graphical environment within which to view and navigate Web documents. Mosaic allowed Web documents to display text in a variety of fonts and type styles, as well as to display static bitmapped images on the page with the text. Hypertext links to external documents could be displayed by rendering linked text in a distinguishing color, or by outlining a linked image with the same distinguishing color (usually blue).

Unusual data formats that were not supported by the native rendering subroutines of the browser could nevertheless be made available to the user by encoding a link in a Web document directly to the data file in question. The type of the data was designated through the use of a data-type-specific filename extension, such as .TIF or .MPG. If the user had pre-installed a viewer application that was capable of viewing that unusual data type, then Mosaic could be configured to allow the user to specify downloading of the data and launching of the external viewer application by clicking on a link to the data object. This would allow viewing of the data in a window external to the Mosaic application window.

This link-based data retrieval for viewing documents and related data objects was designed specifically for viewing document-related data, not for end-user editing of that data. This applied to the external viewer applications as well. Although one could, in theory, configure a data editing program as an external viewer, to actually edit the downloaded data would not have made much sense, since those edits would have only affected the local temporary copy of the document data, and would have had no effect on the original document files on the remote Web server. This design principle is reflected in the file caching architecture of Mosaic. When document text, image data, or external object data are downloaded by the browser, these data are stored in local temporary cache files, to speed up document rendering for later requests for the same data. Since these data are only intended to be viewed, and not edited by the end-user, these cache files are only temporary, and are routinely automatically purged by the Web browser after a certain amount of



time, or after it is determined that the user is not likely to need them again. This makes more efficient use of the end-user's disk drive space over time, preventing cluttering up of the disk drive with unneeded files. To provide editing capabilities for these files would have been nonsensical, since end-user edits would automatically be lost at the subsequent cache purge.

Mosaic was therefore not designed to allow interactive manipulation of object data. On the contrary, it appears to have been specifically designed to **discourage** such object data manipulation. External viewer programs could, theoretically, allow such manipulation of object data, but the fundamental design of Mosaic shows that such manipulation was intended only for viewing of temporary local copies of object-related data.

## 2. How Koppolu (OLE) Works

The OLE system, as defined in Koppolu and as embodied in OLE 2.0 from Microsoft Corporation, was designed primarily as an environment for compound-document creation and editing. Through OLE-based applications, users create compound documents by transferring object data through the clipboard during the document creation process. That object data is then used to create either an embedded object, where the contained object data is encapsulated in the container document file, or linked data, where the object data is stored in a separate file. In both cases, the container document file stores a presentation format, or picture, of the object data. OLE Compound Files store the necessary information in the container document file for rendering the various document components and store object data for embedded objects. The file structure was designed as a simulated file system in order to facilitate incremental fast saves and incremental viewing of large documents during the document creation and editing process. These data are stored in platform-specific binary format to accelerate the task of reading and writing necessary data. Since the document is represented in an hierarchical ordered data structure, and since object-specific object handlers determine how various document components are

rendered, there is no need to parse the document in order to accomplish rendering.

The Microsoft Dictionary of Computer Terms defines the term "parsing" as follows: "To break input into smaller chunks so that a program can act upon the information. Compilers have parsers for translating the commands and structures entered by a programmer into machine language. A natural-language parser accepts text in a human language, such as English, attempts to determine its sequence structure, and translates its terms into a form the program can use. Data base management programs and expert systems often support natural-language parsing. A user could ask such a program to display the relationship between inflation and home buying in the last decade." The program might break the sentence apart and interpret it in the following way:

*display*

Present the results as a chart.

*the relationship  
between*

Do a linear regression analysis.

*inflation and home buying*

The independent and dependent variables, respectively.  
*in the last decade*

Use data from 1980-1989.

Comparing OLE binary data formats to Mosaic's ASCII text tag mechanism, from the point of view of parsing, would be similar to comparing machine-code programming to the use of a higher level programming language. Similar to OLE's document data files, machine code programs are stored in a binary data structure format that is specifically tailored to the computer processor architecture. They are especially efficient, since they do not require a parser for execution. Use of Mosaic's tag-parsing mechanism, however, is more like using a higher-level programming language. What is given up by Mosaic in terms of run-time performance is recouped through ease of document

development, simplification of browser design, and cross-platform compatibility for document viewing.

OLE uses two binary data structures to store objects in compound documents' IStorage and IStream. The IStorage data elements correspond to analogs for directories, and the IStream data elements are file analogs. These data structures store the document objects, as IStreams, in the order in which they appear in the document. Each document element, such as a paragraph or an embedded object, has an associated object handler. When the document is rendered by the container application, this rendering is merely a matter of invoking the objects in the order in which they appear. There is no need to **parse** the document file, to break it into components and to classify those components (as in Mosaic), since the objects are, by definition, already associated with the methods necessary to render them.

Since cross-platform compatibility is not a priority in the OLE system, the fundamental architecture of OLE argues **against** the idea of modifying OLE to allow such parsing for compound document rendering. This is because parsing the document would slow down the performance of the OLE system, and because of the fact that parsing in OLE would provide redundant and unneeded capabilities, since, by definition, each document object element in OLE already has an object handler associated with it to handle the rendering chores.

Another consequence of the OLE architecture is that document rendering of containee object data is closely bound to the specific application methods that were used for creating that object data. In order to manipulate OLE embedded or linked object data, the user must have a copy of the application that originally created that data. This is appropriate for a system that was designed specifically to ease the document creation and editing tasks for a single user working on a single computer, but it would drastically reduce the usefulness of a program such as Mosaic that is intended for the viewing of documents that are simultaneously being widely distributed over networks to heterogeneous populations of other client workstations.

OLE simplifies the task of the compound document author, by allowing interactive editing of containee object data from within the same windowing and graphical user interface (GUI) environment of the container application. In this way, the document author can invoke large external object-editing applications which can take over the GUI of the container application, rather than forcing him or her to pop up another set of application windows in order to edit the document. Since OLE was designed to integrate a group of already-popular large programs, such as Microsoft Excel and Microsoft Word, OLE provided the advance that allowed these applications to be interactively invoked to change the GUI environment that surrounded the compound document so that the user would not have to move his or her attention to an external windowing environment in order to edit the object. Since these external applications were intended to be invoked only for editing purposes, and since that invocation inevitably resulted in a modification of the container application's GUI, it made sense that OLE containee server applications could **not** be automatically invoked to allow interactive processing of object data. In fact OLE forced the user to make not one but **two** interactive commands prior to server invocation, thereby reducing the possibility that one of these large external applications would be inadvertently invoked.

In order to facilitate the ability of several large and complex applications to share each other's GUIs, OLE requires that all interaction with containee server application methods occur by way of the container application's internal message handling subroutines. When a given containee object image (a bitmap or metafile of linked or embedded data) is interactively selected by the user within the container document, OLE then invokes functions in order to allow the container application to determine how to modify its graphical user interface and menu structure for a particular type of embedded data. This occurs prior to invoking the server application. OLE determines from the operating system's global registry what actions are supported by the server application implementing the selected object, and then displays the actions in a menu. The user must then issue

yet another interactive command before the OLE object becomes activated and available for interactive processing. Koppolu states: "Once a user has selected a desired action (from the menu or by double clicking on the object), the container application can then invoke the server application by passing it an indication of the action to perform on behalf of the container application."

After in-place activation, the user then interacts with the server application by indicating to the container application the actions that are desired to be performed on the contained data. Koppolu states: "Once the user has activated an object in place, the user interacts with the object within the container application by selecting actions through the menu bar of the container application (which is the composite menu bar). Because some of the menus belong to the server application and others belong to the container application, the window procedure for the container application frame window must decide whether to send the menu input event to a function within the container application or within the server application. **Thus, all messages received by the container application that correspond to its frame window are thereafter routed fast to the special message handler. This special message handler then decides to which application to route the message event received.**"

Koppolu teaches that OLE-based applications provide users the capability to interact with containee objects via the menu and messaging system of the container application. OLE does not teach that the user interacts directly with an embedded application's user interface within a window *in* the document, as in the Applicants' invention. Koppolu's OLE system does not teach interactive processing of the contained object *within* the embedded object's window, as required by the claims of the Applicants' invention, since OLE teaches that the user should employ the modified menu system, and the messaging system, of the container application in order to process the contained object's data.

D. The way Both Mosaic and Koppolu work therefore teach away from a combination of the two

Mosaic teaches that document tag parsing, rather than binary data representations, should be used to specify the elements and layout of the hypermedia document. Mosaic further teaches that the content of a hypermedia document should not interfere with the functionality of the hypermedia browser application. For example, an important feature of Mosaic was the ability to interrupt the downloading of a partially-rendered long document. OLE would interfere with this, feature since an activated server would deactivate the GUI (including the Stop and Back buttons) in Mosaic.

As described in detail above, Koppolu teaches away from the method of parsing the text of a hypermedia document in order to identify text formats which specify the locations of contained object data. OLE teaches that the various component data elements of a document should be stored as binary data structures that contain objects that are invoked in storage order, to render their presentation formats on the screen or printed page in a binary representation. In fact, OLE employs an emulated file system, within the OLE compound document file format, to organize and store component data. This scheme facilitates fast incremental saves of document component data during the OLE document editing process, but would be totally unworkable in Mosaic.

Koppolu also teaches away from the use of OLE for networked data distribution. There is no provision, suggestion, or motivation in Koppolu to provide for automatic invocation of a server application to allow interactive processing of object data when a container document is viewed. Furthermore, there is no suggestion in either Mosaic or Koppolu of modifying Mosaic so that an external application, by analogy to Koppolu the server application, is automatically invoked at the time of Web document rendering to display and enable interactive processing of the object within an embedded window within the Web document.

The Applicants' invention allows the networked hypermedia document to act as a coordinator and deployment mechanism, as

well as a container, for any arbitrary number of external interactive data/application objects, irrespective of where those objects are located on a network, while hiding the details of such coordination and deployment from the documents reader (user) as the reader uses and interacts with the various data/application objects on a variety of computer platforms. This allows the networked hypermedia document to act as a platform for entirely new kinds of applications that could not have been possible before the invention.

Because most of the functionality exposed to the user is defined most directly by the hypermedia document itself, rather than any specific computer operating system or document container application, document-based applications using the claimed invention tend to have the same look and feel to the reader, regardless of what type of computer or operating system is being used to run the operating system, and regardless of what type of browser is being used to view the document.

Additionally, because the claimed embed text formats in the document cause the browser to automatically invoke the external application, the hypermedia **document** itself, and by implication the author of that document, **directly** control the extension of the functionality of the browser. As a consequence of the features of the claimed invention, the **document**, rather than the browser, **becomes the application**; that is, the document together with its embedded program objects, exposes all the functionality that the user needs to interact with and process the entire content of the compound hypermedia document.

E. What Workers of Skill in the Art, Aware of and in Possession of Rights to Mosaic and Koppolu (OLE) Actually Did

1. The owner of the commercial rights to both the technology cited in Koppolu and the applicant-cited prior art (Mosaic) chose to employ features of the claimed invention, rather than to attempt the combination proposed by the Examiner

During the 2 years after the date of the Applicants' patent application, Microsoft Corp. obtained the commercial rights to the Mosaic Web browser and, of course, developed many applications based upon the OLE technology disclosed in Koppolu. The software company then set out to implement improvements to Mosaic to allow the use of embedded interactive applications within Web pages. However, Microsoft did not combine the features of Mosaic and OLE but instead developed ActiveX. The fact that ActiveX is not a combination of Mosaic and OLE is made clear by Microsoft itself in a conversation between Web Technique's editor and the ActiveX product managers from Microsoft:

--So tell me about ActiveX. Isn't it just OLE for the Internet?

--Wrong, ... ActiveX is a new API that, like OLE is based on Microsoft's Component Object Model (COM). While OLE supports a compound document structure for desktops, ActiveX is designed specifically to embed rich media objects within Web-based documents.

From "ActiveX, a Standard?" by Michael Floyd (Editor in Chief), WebTechniques, October 1996, page 5.

As described below, ActiveX utilizes an embed text format, adopted from the present invention, and employs all the features of applicant's claim 1 to function as a universal wrapper to bridge the gap between a browser and OLE applications as well as Java, VRML, Shockwave, Visual Basic, C++, and other scripting tools.

The Microsoft Press book, ActiveX Controls Inside Out, by Adam Denning, 1997 (ISBN 1-57231-350-1), shows specifically that ActiveX employs the features of the claimed invention for the implementation of embedded program objects in distributed hypermedia documents, rather than to attempt to use the combination proposed by the Examiner.



Specifically, the Denning reference shows, in Chapter 13, page 402, that the ActiveX system uses an embed text format (<OBJECT>), at a location within the hypermedia document, that specifies the location of at least a part of an object external to the hypermedia document (using the CODEBASE parameter), wherein the object has type information associated with it (the CLASSID) that is utilized by the browser to identify and locate an executable application external to the hypermedia document, and wherein the embed text format is parsed by the browser to automatically invoke the external application to execute on the client workstation in order to display the object and enable interactive processing of the object within a display window (defined by the WIDTH and HEIGHT parameters) created at the embed text format's location within the distributed hypermedia document being displayed in the browser-controlled window, as required by Claim 1. That the ActiveX embedded program object **internally** supports a modified OLE-type API for communication between the ActiveX control and other OLE-compatible applications is **irrelevant** to the patentability of the claimed invention.

The fact that Microsoft chose to implement a system (ActiveX) that employs the features of the Applicants' invention rather than to perform a combination of OLE and Mosaic as proposed by the Examiner, even though Microsoft would have had a strong incentive to make such a combination if it were obvious, feasible and useful, provides **direct and undeniable evidence that such a combination was not obvious and would not have improved Mosaic.**

F. How the Real World Responded to Applicant's Invention

1. The commercial success of products developed subsequent to filing the present application, incorporating the claimed features, established that the combination was not obvious at the time of the invention.

As is shown in the Inventor's declaration filed 6-2-97, several major competitors have incorporated the features of

Applicants' invention, rather than to use the techniques of the prior art. The most notable of these products include the ActiveX applet system from Microsoft corporation, the Navigator Web browser application from Netscape corporation and the Java Web applet system from Sun Microsystems corporation. The Inventor's declaration further shows that the success of these products is directly attributable to the features of the claimed invention which each of these products incorporate, including an embed text format that is parsed by a Web browser to automatically invoke an external executable application to execute on the client workstation in order to display an external object and enable interactive processing of that object within a display window created at the embed text format's location within the hypermedia document being displayed in the browser-controlled window. The developers of these products chose **not** to attempt the proposed combination, **even though the OLE technology was widely available for development purposes at the time of the Applicants' invention**. Rather, they chose to implement the features of the Applicants' invention, in a manner that is **virtually identical** to the preferred embodiment disclosed in the Applicants' specification, in making improvements to Mosaic.

Some of these competitors have made laudatory statements about the elements of the Applicants' invention which are incorporated into their respective products, and have characterized those features as being a significant and innovative advance over prior art techniques.

It is well known that, in the 1966 case of Graham v. John Deere, the U.S. Supreme Court decreed that Section 103 is to be interpreted by taking into consideration secondary and objective factors such as commercial success, long-felt but unsolved need, and failure of others.

As is shown in the 6/2/97 Inventor's declaration, there is universal acceptance within the computer software industry that the aforementioned products incorporating the features of the claimed invention have attained an extremely high degree of commercial success. Java, Navigator and ActiveX (all products incorporating the features of the claimed invention)

represent among the most popular current technology in the computer industry for new application development, and are among the most successful products in the history of computer software.

2. The products by others incorporating the features of the claimed invention have been given many awards and have received considerable recognition in professional publications.

As is shown below and in the 6/2/97 Inventor's declaration, Netscape, ActiveX and Java, all incorporating features of the Applicants' invention, have each been lauded as among the most innovative (and therefore non-obvious) technologies to appear in the computer software industry in recent years.

Some examples are:

The 1996 Discover Awards for Technical Innovation -- to Java and Navigator

PC Magazine 1995 Technology of the Year Awards -- to Java and Navigator

New Media Magazine 1996 Hyper Awards -- to Java and Navigator

New Media Magazine 1997 Hyper Awards -- to Microsoft's ActiveX applet system

As is evidenced in the 6/2/97 Inventor's declaration, this acclaim is due to the innovative nature of the features of the claimed invention incorporated into those products, argues strongly against the obviousness of the pending claims.

#### DEPENDENT CLAIMS

Claims 2-5 which depend on claim 1 are allowable for the same reasons as set forth above and are further allowable because those claims recite additional limitations not disclosed in the cited references.

In particular, claim 3 recites the additional steps over Claim 1 of "interactively controlling said controllable application by interprocess communications between the browser and the controllable application" (claim 2); "continuing to exchange communications after the controllable application has

been launched" (claim 5), and "additional instructions for controlling said controllable application reside on said network server, wherein said step of interactively controlling said controllable application includes the following sub-steps: issuing, from said client workstation, one or more commands to the network server; executing, on said 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."

None of the cited references show these features. As described above, in Mosaic there is no disclosure of interacting with an external controllable application. Further, in Koppolu (OLE) the client and server applications are on the same computer. There is no provision for interacting with an application on a network server.

The reference Moran discloses a tool for interactive visualization of large, rectilinear volumetric data called Tele-Nicer-Slice-Dicer (TNSD). TNSD is based on client-server design where the client-side process is an extended version of a stand-alone visualization tool and the server process runs on a high-performance system where the data are located.

The client-side process describes data sets by text fields. Each data set description is used as a command which is sent to the server when a volume from the corresponding data set is requested. The use of a remote server is transparent.

The only relevance of Moran to the subject matter of claim 3 is the use of a network. There is no disclosure relating to HTML, the WWW, or embedding controllable objects in a document displayed in a browser-controlled window.

The teachings of Mosaic, Koppolu (OLE), and Moran would not make the combination of claim 3 obvious. Such a combination is in violation of the requirement of 35 U.S.C. §103 because the combination requires taking isolated features from the references, utilizing applicant's disclosure as a roadmap, while ignoring the operation and purposes of the references.

MICHAEL D. DOYLE et al.  
Application No.: 08/324,443  
Page 31

PATENT

Claim 4 recites the additional steps over Claim 3 "wherein said additional instructions for controlling said controllable application reside on said client workstation."


None of the claimed references show this feature. This feature produces the additional surprising and unexpected results of enabling a client and network server system to be self-contained on the client workstation.

Claims 44-48 are apparatus claims of the same scope as claims 1-5 and are thus allowable for the reasons recited above.

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 (415) 576-0200.

Respectfully submitted,



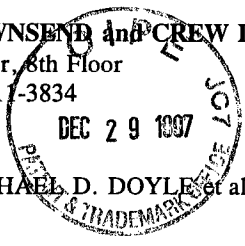
Charles E. Krueger  
Reg. No. 30,077

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, California 94111-3834  
(415) 576-0200  
Fax (415) 576-0300  
CEK:db  
i:\cek\share\02307i\553\dec23.res

GAU 27821 \$

Amendment

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, 8th Floor  
San Francisco, CA 94111-3834  
(415) 576-0200



Atty. Docket No. 023071-553

Date December 23, 1997

In re application of MICHAEL D. DOYLE et al.  
Appln. No. 08/324,443  
Filed 10/17/94  
Group Art Unit 2317

I hereby certify that this is being deposited with the United States Postal Service as first class mail in an envelope addressed to:

For EMBEDDED PROGRAM OBJECTS IN DISTRIBUTED HYPERMEDIA SYSTEMS

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

Date: 12-23-97  
D Bullena

THE ASSISTANT COMMISSIONER FOR PATENTS  
Washington, D.C. 20231

Sir:

Transmitted herewith is an amendment in the above-identified application.

- Enclosed is a petition to extend time to respond.
- Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
- A verified statement to establish small entity status under 37 CFR 1.9 and 1.27 is enclosed.
- Declaration of Michael D. Doyle, Attachments A and B.

If any extension of time is needed, then this response should be considered a petition therefor.  
The filing fee has been calculated as shown below:

(Col. 1)		(Col. 2)		(Col. 3)	SMALL ENTITY		OTHER THAN A SMALL ENTITY		
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NO. PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE	ADDIT. FEE	OR	RATE	ADDIT. FEE
TOTAL	*	MINUS	**	=	x11=	\$	OR	x22=	\$
INDEP.	*	MINUS	***	=	x41=	\$	OR	x82=	\$
[ ] FIRST PRESENTATION OF MULTIPLE DEP. CLAIM					+135=	\$	OR	+270=	\$
					TOTAL ADDIT. FEE	\$	OR	TOTAL	\$

- \* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
- \*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
- \*\*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space. The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[x] No fee is due.

Please charge Deposit Account No. 20-1430 as follows:

- Claims fee \$ \_\_\_\_\_
- Any additional fees associated with this paper or during the pendency of this application.

no extra copies of this sheet are enclosed.

TOWNSEND and TOWNSEND and CREW LLP

Charles E. Krueger / Reg. No.: 30,077  
Attorneys for Applicant

# 906 PH Ex. 17







UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET NO.
08/324,443	10/17/94	DOYLE	M 02307553

020350 LM21/0330  
TOWNSEND AND TOWNSEND AND CREW  
TWO EMBARCADERO CENTER EIGHTH FLOOR  
SAN FRANCISCO CA 94111

EXAMINER

DINH, D

ART UNIT	PAPER NUMBER
2756	2300

2756

03/30/98

DATE MAILED:

NOTICE OF ALLOWABILITY

PART I.

- This communication is responsive to paper filed 12/29/97
- All the 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 And Issue Fee Due or other appropriate communication will be sent in due course.
- The allowed claims are 1-5, 44-48
- The drawings filed on \_\_\_\_\_ are acceptable.
- Acknowledgment is made of the claim for priority under 35 U.S.C. 119. The certified copy has [ ] been received. [ ] not been received. [ ] been filed in parent application Serial No. \_\_\_\_\_, filed on \_\_\_\_\_.
- Note the attached Examiner's Amendment.
- Note the attached Examiner Interview Summary Record, PTOL-413.
- Note the attached Examiner's Statement of Reasons for Allowance.
- Note the attached NOTICE OF REFERENCES CITED, PTO-892.
- Note the attached INFORMATION DISCLOSURE CITATION, PTO-1449.

PART II.

A SHORTENED STATUTORY PERIOD FOR RESPONSE to comply with the requirements noted below is set to EXPIRE THREE MONTHS FROM THE "DATE MAILED" indicated on this form. Failure to timely comply will result in the ABANDONMENT of this application. Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

- Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL APPLICATION, PTO-152, which discloses that the oath or declaration is deficient. A SUBSTITUTE OATH OR DECLARATION IS REQUIRED.
- APPLICANT MUST MAKE THE DRAWING CHANGES INDICATED BELOW IN THE MANNER SET FORTH ON THE REVERSE SIDE OF THIS PAPER.
  - Drawing informalities are indicated on the NOTICE RE PATENT DRAWINGS, PTO-948, attached hereto or to Paper No. 4. CORRECTION IS REQUIRED.
  - The proposed drawing correction filed on \_\_\_\_\_ has been approved by the examiner. CORRECTION IS REQUIRED.
  - Approved drawing corrections are described by the examiner in the attached EXAMINER'S AMENDMENT. CORRECTION IS REQUIRED.
  - Formal drawings are now REQUIRED.

Any response to this letter should include in the upper right hand corner, the following information from the NOTICE OF ALLOWANCE AND ISSUE FEE DUE: ISSUE BATCH NUMBER, DATE OF THE NOTICE OF ALLOWANCE, AND SERIAL NUMBER.

Attachments:

- Examiner's Amendment
- Examiner Interview Summary Record, PTOL- 413
- Reasons for Allowance
- Notice of References Cited, PTO-892
- Information Disclosure Citation, PTO-1449
- Notice of Informal Application, PTO-152
- Notice re Patent Drawings, PTO-948
- Listing of Bonded Draftsmen
- Other

*Dinh*

DINH

Serial Number: 08/324,443  
Art Unit: 2756

#23 C  
S. Ford

-2-

### Part III DETAILED ACTION

An Examiner's Amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 C.F.R. § 1.312. To ensure consideration of such an amendment, it **MUST** be submitted no later than the payment of the Issue Fee.

Pursuant to MPEP 606.01, the title has been changed to read:  
--DISTRIBUTED HYPERMEDIA METHOD FOR AUTOMATICALLY INVOKING  
EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED  
OBJECTS WITHIN A HYPERMEDIA DOCUMENT--.

Authorization for this examiner's amendment was given in a telephone interview with Charles Krueger on 01/27/98.

In claim 1 line 28, replace "window" with --area--.

In claim 44 line 17, delete "by the".

In claim 44 line 17, insert -by- before "said".

In claim 44 line 39, replace "window" with --area--.

The following is an examiner's statement of reasons for allowance:

Applicant's 131 affidavit filed 01-09-97 (paper #7) is persuasive to antedate the Vetter reference and the in-part of Koppula.

The claims are allowable over the prior art of record because the prior art does not teach nor reasonably suggest the claimed combination of a browser, while parsing a hypermedia

Serial Number: 08/324,443  
Art Unit: 2756

-3-

document in a distributed hypermedia environment, automatically invokes an external executable application associated with an embedded object to provide interactive processing and to display the object within an area of the hypermedia document's display window.

The examiner agrees that the claimed external executable application is not a code library extension nor object handler (e.g. windows dll and OLE) as pointed out in applicant's argument (paper #19 pages 12-14).

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 Dung Dinh whose telephone number is (703) 305-9655. The examiner can normally be reached on Monday-Thursday from 7:00 AM - 4:30 PM. The examiner can also be reached on alternate Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Frank Asta can be reached at (703) 305-3817.

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

**Any response to this action should be mailed to:**  
Commissioner of Patents and Trademarks

Serial Number: 08/324,443  
Art Unit: 2756

-4-

Washington, DC 20231

**or faxed to:**

(703) 308-9051, (for formal communications intended for entry)

(703) 308-5359 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).



Dung Dinh  
Patent Examiner  
January 27, 1998

**Notice of References Cited**

Application No. <b>08/324,443</b>	Applicant(s) <b>Doyle et al.</b>
Examiner <b>Dung Dinh</b>	Group Art Unit <b>2756</b>

**U.S. PATENT DOCUMENTS**

*		DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
x	A	5,606,493	02/25/97	Duscher et al.	395	200.32
x	B	5,418,908	05/23/93	Keller et al.	395	200.32
x	C	5,367,635	11/22/94	Bauer et al.	395	200.32
x	D	5,544,320	08/06/96	Konrad	395	200.09
x	E	5,274,821	12/28/93	Rouquie	395	705
x	F	5,146,553	09/08/92	Noguchi et al.	707	516
x	G	4,815,029	03/21/89	Barker et al.	707	516
x	H	5,652,876	07/29/97	Ashe et al.	707	516
	I					
	J					
	K					
	L					
	M					

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
	N						
	O						
	P						
	Q						
	R						
	S						
	T						

**NON-PATENT DOCUMENTS**

	DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
U	Stephen Le Hunte, "<EEMBED> - Embedded Objects", HTML Reference Library - HTMLIB v2.1, 1995: n.pag. Online. Internet.	1995
V	"A Little History of the world Wide Web", n.pag. Online. Internet: available <a href="http://www.w3.org/History.html">http://www.w3.org/History.html</a>	no date
W	"NCSA Mosaic Version Information", n.pag. Online. Internet: available <a href="http://www.ncsa.uiuc.edu/SDG/Software">http://www.ncsa.uiuc.edu/SDG/Software</a>	no date
X	"The second phase of the revolution", WIRED, October 1994, pp. 116-152.	10/94

\* A copy of this reference is not being furnished with this Office action (See Manual of Patent Examining Procedure, Section 707.05(a).)



UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office

**NOTICE OF ALLOWANCE AND ISSUE FEE DUE**

020350 LM21/0330  
TOWNSEND AND TOWNSEND AND CREW  
TWO EMBARCADERO CENTER EIGHTH FLOOR  
SAN FRANCISCO CA 94111

APPLICATION NO.	FILING DATE	TOTAL CLAIMS	EXAMINER AND GROUP ART UNIT	DATE MAILED
08/324,443	10/17/94	010	DINH, D	2756 03/30/98
First Named Applicant	DOYLE, MICHAEL D.			

TITLE OF INVENTION  
DISTRIBUTED HYPERMEDIA METHOD FOR AUTOMATICALLY INVOKING EXTERNAL APPLICATION PROVIDING INTERACTION AND DISPLAY OF EMBEDDED OBJECTS WITHIN A HYPERMEDIA DOCUMENT (AS AMENDED)

ATTY'S DOCKET NO.	CLASS-SUBCLASS	BATCH NO.	APPLN. TYPE	SMALL ENTITY	FEE DUE	DATE DUE
2	02307553	395-200.320	C26 UTILITY	NO	\$1320.00	06/30/98

**THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED.**

**THE ISSUE FEE 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.**

**HOW TO RESPOND TO THIS NOTICE:**

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

- A. If the status is changed, pay twice the amount of the FEE DUE shown above and notify the Patent and Trademark Office of the change in status, or
- B. If the status is the same, pay the FEE DUE shown above.

If the SMALL ENTITY is shown as NO:

- A. Pay FEE DUE shown above, or
- B. File verified statement of Small Entity Status before, or with, payment of 1/2 the FEE DUE shown above.

II. Part B-Issue Fee Transmittal should be completed and returned to the Patent and Trademark Office (PTO) with your ISSUE FEE. Even if the ISSUE FEE has already been paid by charge to deposit account, Part B Issue Fee Transmittal should be completed and returned. If you are charging the ISSUE FEE to your deposit account, section "4b" of Part B-Issue Fee Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give application number and batch number. Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

**IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.**

PATENT AND TRADEMARK OFFICE COPY

# 906 PH Ex. 18





#24

<b>Interview Summary</b>	Application No. <b>08/326,443</b>	Applicant(s) <b>Doyle et al</b>
	Examiner <b>Dung Dinh</b>	Group Art Unit <b>2756</b>

All participants (applicant, applicant's representative, PTO personnel):

(1) Dung Dinh (3) \_\_\_\_\_

(2) Charles E. Krueger (4) \_\_\_\_\_

Date of interview Jan 26, 1998

Type:  Telephonic  Personal (copy is given to  applicant  applicant's representative).

Exhibit shown or demonstration conducted:  Yes  No. If yes, brief description:

Agreement  was reached.  was not reached.

Claim(s) discussed: 1 and 44

Identification of prior art discussed:

None

Description of the general nature of what was agreed to if an agreement was reached, or any other comments:

The applicant agreed to amend "display window" in line 28 of claim 1 to --display area-- to distinguish that it is an area within the hypermedia document that displays the object and not a separate window. The same amendment was made to claim 44, line 39. Further in line 17 of claim 44, "said by the text formats" was amended to read --by said text formats--.

(A fuller description, if necessary, and a copy of the amendments, if available, which the examiner agreed would render the claims allowable must be attached. Also, where no copy of the amendments which would render the claims allowable is available, a summary thereof must be attached.)

1.  It is not necessary for applicant to provide a separate record of the substance of the interview.

Unless the paragraph above has been checked to indicate to the contrary, A FORMAL WRITTEN RESPONSE TO THE LAST OFFICE ACTION IS NOT WAIVED AND MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a response to the last Office action has already been filed, APPLICANT IS GIVEN ONE MONTH FROM THIS INTERVIEW DATE TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW.

2.  Since the Examiner's interview summary above (including any attachments) reflects a complete response to each of the objections, rejections and requirements that may be present in the last Office action, and since the claims are now allowable, this completed form is considered to fulfill the response requirements of the last Office action. Applicant is not relieved from providing a separate record of the interview unless box 1 above is also checked.

*Frank J. Asta*  
for  
Dung Dinh

**FRANK J. ASTA**  
**SUPERVISORY PATENT EXAMINER**  
**GROUP 2700**

Examiner Note: You must sign and stamp this form unless it is an attachment to a signed Office action.

