IBM techexplorer: Scientific Publishing for the Internet

Robert S. SUTOR and Angel L. DÍAZ

Interactive Scientific Publishing
IBM T.J. Watson Research Center
Yorktown Heights, New York, 10598 USA
{sutor,aldiaz}@us.ibm.com

Abstract. *The IBM techexplorer Hypermedia Browser is an application for the interactive publication of scientific and technical documents. The original project started as an experiment at IBM Research to see how a from-scratch implementation of a subset of TEX, LATEX, and AMS-LATEX could be extended to support interactive viewing of documents for a computer algebra system. This interactivity is accomplished via support for hypertext, multimedia, user-defined pop-up windows and menus, and a modular architecture that allows connections with other applications and Java applets. The primary version of techexplorer operates as a Netscape Navigator plug-in and is available for several platforms, including Windows 95 and NT, IBM AIX, and Sun Solaris. In addition to being able to display full documents using the supported TEX subset, techexplorer is being extended to support the new "Mathematical Markup Language" from the HTML Math Working Group of the World Wide Web Consortium. In this paper, we provide an overview of techexplorer and detail how it can be used to deliver mathematical articles, books, and course materials via the World Wide Web. We also discuss our intended use of the OpenMath standard to allow documents to contain reusable semantically attributed math objects.*

1. Introduction

Perhaps one of the most pressing issues facing publishers today is the development of electronic or Internet-based alternatives to hard-copy textbooks and journals that are both pedagogically sound and economically viable. This task is even more challenging in the scientific and technical publishing arena where mathematical layout is complex and semantic mathematical information should be maintained for searching, indexing, or computational reuse in other applications.

Until recently, the publication of documents containing mathematical notation on the World Wide Web had been very awkward due to the absence of HTML support. Scientific and technical document publishers had no convenient way of electronically disseminating TEX and LATEX content. They often resorted to using markup conversion tools that produce text with GIF images of scientific notation. Such conversions resulted in documents that were of poor visual quality, printed badly, were difficult to author and failed to preserve mathematical semantic information. Indeed, how do you select a subexpression of a mathematical object display in a GIF image? Other solutions that essentially provide images of pages suffer from not being especially adaptable to the wide range of display hardware for maximum viewing convenience.

IBM's techexplorer Hypermedia Browser directly renders a large subset of TEX and LATEX markup. Our algorithms do not convert the original document source into a static paginated binary format such as DVI but rather into an object-oriented representation of the document. Such a representation enables high quality dynamic rendering at varying resolutions and screen sizes, rapid document reflow/redisplay, and the opportunity to preserve semantic mathematical information. In fact, this object-oriented approach to document representation facilitated the implementation of hypertext, multimedia content, inter-application communication, and interactivity.

The dynamic rendering makes it possible for other applications to generate TEX and LATEX markup that can then be incorporated into the document being displayed. In particular, if one can send a mathematical expression to another application for evaluation, and that application can generate TEX and LATEX for the result, techexplorer can be used as a kind of browser-based interactive front-end for the application.

2. Project history

techexplorer is our second-generation TEX-based hypertext system. The original application was a viewer for documents for the computer algebra system that was eventually to be called AXIOM and distributed by the Numerical Algorithms Group, Ltd. (NAG) [3]. AXIOM is a sophisticated system for performing mathematical computation. It offers two- and three-dimensional graphics, a hypertext help package, and various forms of output, including LATEX and Fortran. For UNIX platforms, *HyperDoc* was developed in the late 1980s and was AXIOM's hypertext front-end for viewing text intermixed with the results of computations. The AXIOM/HyperDoc link enabled users to open a workspace or start a graphics manager by clicking on AXIOM input or a graphic, respectively.

The first challenge when we considered expanding the applicability of HyperDoc was that it accepted a non-standard dialect of LaTeX. Part of our initial efforts were geared toward supporting standard LaTeX and HyperDoc was improved in this way. However, HyperDoc still did not render many common LaTeX/TeX control sequences. What is perhaps most surprising, HyperDoc had very weak support for displaying mathematics. Some early implementation choices made adding such support very difficult and hindered porting HyperDoc to other platforms, such as Microsoft Windows. Although HyperDoc had excellent connectivity to AXIOM and the graphics manager, there was no way to update the document in-place with the results of a computation or a modification to a graph. For these reasons, in 1994 we embarked on developing a from-scratch implementation of a large subset of standard TeX and LaTeX, with extensions to support the interactive viewing of documents.

Our decision to provide a Microsoft Windows 3.1 and later a Windows 95/NT based implementation was prompted by the sophisticated set of tools geared toward rapid C++ code development, a rich set of user interface components, a robust implementation of interprocess communication via Object Linking and Embedding (OLE), and a potentially large user community. Our aim was to leverage these technologies for quickly developing a framework for laying out and interacting with mathematics.

The project's efforts resulted in an early version of techexplorer (then codenamed "Saturn") that produced familiar output from LaTeX/TeX source. Our approach to orthogonally extending LaTeX/TeX with control sequences for hyperlinking, multimedia, and communication with math software ensured techexplorer's compatibility with LaTeX/TeX markup for printed documents. An early version of this standalone edition of techexplorer is used as the front end for NAG's AXIOM for Windows product.

It became evident in early 1996 that we could augment our core technology to serve interactive scientific and technical documents over the World Wide Web via the Netscape Navigator plug-in interface. The first version for Windows 95 was made publicly and freely available in May 1996. This "Standard Edition" techexplorer plug-in allows authors and publishers to effectively expand the reach of their articles, books, and journals by making them available on the Internet and in intranets. As the community of techexplorer users grew, we realized that a UNIX edition of the plug-in was very important for our colleagues in the scientific community. In September 1997, we released our first "preview release" of techexplorer on a UNIX platform on IBM alphaWorks [4].

Today, the Interactive Scientific Publishing Research Group at IBM Research continues to distribute the Standard Edition of the techexplorer plug-in for Window 95/NT, IBM AIX 4.1 and SUN Solaris 2.5, with more platforms planned for the future. A stand-alone version of techexplorer is used as the framework for Springer-Verlag's forthcoming interactive textbook *Linear Functions and Matrix Theory* by Bill Jacob. The techexplorer product line will continue to evolve as we create an important tool for the Internet delivery of scientific and technical journals, reports, textbooks, and courseware.

3. An overview of techexplorer

In creating techexplorer we did not set out to re-implement all of TeX, but rather we decided to implement "enough." Thus, for example, at the time of this writing we support the LaTeX `\newcommand` and `\newenvironment` commands but we do not support the forms with optional arguments. Similarly, we support `\def` but we do not yet support the form with delimited macro arguments (for example, `\def\X#1,#2{#1 and #2}`). Support for these features and other control sequences has traditionally and will continue to be user and application driven. The techexplorer user guide lists most of the control sequences that are now supported [5].

When a document is parsed, an internal tree structure of objects is created. Thus when you click on the display screen, we have enough information to identify the object in the structure hierarchy to which you are pointing. This means, in particular, that we can do things such as provide hypertext links or have certain messages show up on the status line as the cursor passes over an object. We provide links that can

— cause the display to jump elsewhere in the current or a different document, possibly in a different browser frame;
— start an application (an option allows you to state whether this should happen directly, ask you first, or not at all);
— play an audio clip from a URL;
— play an video clip from a URL;
— pop up various kinds of dialog boxes for user input;
— display fully-formatted text in a pop-up window;
— display one or the other of two expressions (e.g., you could display a question and when you click on it the answer is displayed in its place); and
— send input to another application, with the output possibly being pasted back into the current document.

Here is an example of the syntax for our most general hypertext link:

`\docLink[` *frame* `]{` *URL* `}[` *label* `]{` *displayText* `}` where

— *frame* is the name of an HTML frame as defined in an HTML document;
— *URL* is the source location URL for the document;
— *label* is the label in the source document to which we should jump;
— *displayText* is the expression that should be displayed. Clicking on this expression executes the link.

We eventually plan to support additional hypertext syntaxes, such as that used in HyperRef.

Color support is very important for high quality on-screen display and techexplorer provides the commands `\color`, `\textcolor`, `\colorbox`, `\fcolorbox`, `\rgb`, `\pagecolor`, and `\colorbuttonbox` (a techexplorer extension). We provide the `\includegraphics` command to embed GIF and JPEG images in a document. The `\backgroundimage` command is a techexplorer extension that allows the author to set the image displayed behind the text in a window.

Documents delivered over the web should not be monolithic and we encourage authors to break up their files into reasonably sized pieces with a rich collection of hyperlinks. For many documents, this creates a natural tree hierarchy. The commands `\aboveTopic`, `\previousTopic`, and `\nextTopic` allow the reader to move up, left or right, respectively, in the hierarchy. When these commands are defined in a document, the default document context menu (gotten by clicking the right mouse button) allows the reader to quickly jump to the specified sections.

techexplorer documents are not paginated on screen. To make footnotes more easily readable, they appear in their own fully-formatted pop-up windows. Printing support will be provided in the techexplorer Professional Edition that will be released in 1998.

Pop-up menus, sometimes called context menus, are very useful for making selections. Here is the menu definition for section 3.2 of the book *Linear Functions and Matrix Theory* by Bill Jacob [2]:

```
\newmenu{theorem-menu-3.2}{
    \labelLink{theorem-3}{Theorem 3}
}
%
\newmenu{section-menu-3.2}{
    \labelLink{sec-3.2}{Section 3.2}
\hrule
    \usemenu{theorem-menu-3.2}{Theorems}
\hrule
```

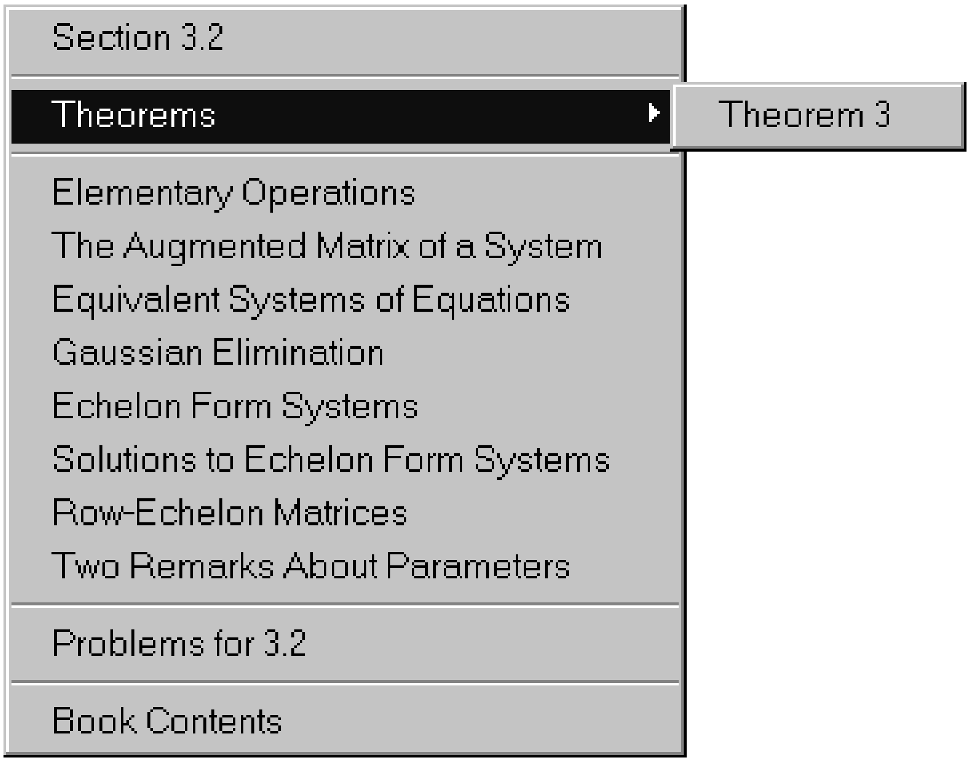

Figure 1 – A user-defined techexplorer menu

```
     \labelLink{subsec-3.2.1}{Elementary Operations}
     \labelLink{subsec-3.2.2}{The Augmented Matrix of a System}
     \labelLink{subsec-3.2.3}{Equivalent Systems of Equations}
     \labelLink{subsec-3.2.4}{Gaussian Elimination}
     \labelLink{subsec-3.2.5}{Echelon Form Systems}
    \labelLink{subsec-3.2.6}{Solutions to Echelon Form Systems}
    \labelLink{subsec-3.2.7}{Row-Echelon  Matrices}
    \labelLink{subsec-3.2.8}{Two Remarks About Parameters}
\hrule
    \labelLink{subsection-3.2.problems}{Problems for 3.2}
\hrule
    \docLink{lfmttoc.tex}{Book Contents}
}
```

This menu is shown in Figure 1. The `\hrule` commands put separator lines in the menu. Most of the selections are hypertext links within the section, but there is one submenu that lists the theorems in the section. (In this case, there is only one theorem.) The `\usemenu` command is used in menus to create submenus as above, but it is also used within the text to associate menus with text. In this particular book, clicking the right mouse button on any section heading brings up an appropriate context menus that allows you to easily move throughout the section.

An interesting and occasionally controversial subject is how techexplorer deals with fonts. On Microsoft Windows 95 and Windows NT, techexplorer directly uses TrueType fonts. Under UNIX, techexplorer uses PostScript fonts. Metafont fonts are not supported. The user can select any TrueType or PostScript font under Windows or UNIX, respectively, for use with `\rm`, `\bf`, `\it`, `\tt`, etc. Special symbols are gotten from various collections of fonts, such as the Monotype Math fonts shipped with Lotus SmartSuite, the WordPerfect math fonts, or the Lucida fonts sold by Microsoft. For UNIX we derived several symbol fonts from the Computer Modern and AMS Symbols fonts created by BlueSky Research and Y&Y and placed in the public domain under the auspices of the American Mathematical Society. In 1998 we plan to release TrueType versions of these derived symbol fonts for use under Windows. Eventually we expect to open the font model to allow users to map a symbol to an arbitrary character in a font of their choice.

techexplorer supports input files, and so it is possible to create collections of macros. LaTeX style files are typically not supported because of the current restrictions on macro definitions. Since techexplorer frequently represents document elements (such as `\section`) as primitive elements, it is not clear that the existing LaTeX style and package scheme makes sense for our use. We favor an approach similar to Cascading Style Sheets [8]. This also makes more sense for a math element embedded into an HTML document via the `EMBED` element where the math presentation should match the style of the surrounding text.

4. techexplorer architecture

The techexplorer architecture consists of a large cross-platform core that supports TeX and Mathematical Markup Language input. The core is extended on each platform to provide the necessary graphics user interface support and interfaces with the web browsers.

In order to do interesting things with these electronic documents, the Professional Edition of techexplorer that is under development provides a Java interface so that documents can be manipulated externally. For the interactive book project, we have implemented bidirectional connections between techexplorer and running Java applications. We plan to eventually provide a document object model programming interface so that techexplorer documents can be fully scripted within the browser.

Our Web Developers Kit (WDK) will also include a specification for our add-ins interface. This allows third-parties to have compiled C or C++ code dynamically used by techexplorer. This code could extend the core functionality or provide links to other applications, perhaps across the Internet. We plan to use this type of add-in for distance and distributed learning applications and for connections to symbolic math systems.

4.1. techexplorer core

It is important for techexplorer to exist on many operating system platforms. The design is carefully factored to isolate the platform- or library-specific areas so that as large a percentage of the C++ code as possible is cross-platform. This makes it possible for us to seamlessly use the Microsoft Foundation Classes under Windows and the Standard Template Library under UNIX.

Throughout the development of techexplorer, we maintained an "Object-oriented" approach to scientific and technical documents rather than an "output-oriented" one. Our algorithms abstract the document into fundamental components and encapsulate the font and graphics information needed for document composition and rendering. This allows for rapid dynamic display of documents across multiple platforms. Indeed, roughly 80 per cent of techexplorer's core parsing and formatting algorithms are ANSI C++ compliant cross-platform code.

In general, techexplorer primitives are higher-level than most in TeX. For example, we have "root" objects, "heading" objects, "menu" objects, and so forth. We do support many of TeX's primitives and basic macro definitions. Our goal is to eventually support all of AMS-LaTeX and a sufficient amount of pure TeX for the majority of documents.

Philosophically, we are strongly in the structural and semantic markup camp versus the presentation markup one. Thus we encourage the use of techexplorer primitives such as `\section` as opposed to defining a macro that means "make this text big and bold and leave some space before the following text." Some of our future research will be concerned with how we can use or at least model the cascading stylesheet technology that has been developed for HTML [8].

We have started to add support for the Mathematical Markup Language as devised by the HTML Math Working Group of the World Wide Web Consortium [10]. At the time of this writing, we support a subset of the presentation elements, but plan to provide full support. With this second markup language in addition to TEX, our goal to produce an interactive **tech**nical publishing tool via **tech**explorer is being realized.

We do not now support MML syntax within TEX documents or vice-versa. However, since our MML support is based on an XML parser that we developed, it would be easy to add this functionality [11].

4.2. Add-ins

techexplorer "add-ins" enable third-party developers to extend techexplorer via a C/C++ application programming interface (API). Add-ins blend seamlessly into techexplorer and can add a wide range of interactive and multimedia capabilities. For example, a computer algebra add-in could augment techexplorer for in-place manipulation of math objects.

The `\liveLink` control sequence is one mechanism for transporting an object from a techexplorer document into another application via the add-in API.

`\liveLink[` *protocol* `]{` *appInput* `}[` *defaultInput* `]{` *displayText* `}`

Here the *protocol* parameter identifies a particular add-in. It might be "Maple" or "Axiom" (for example) and the *appInput* would be the verbatim command to send to the corresponding external application. Either *appInput* or defaultInput can be empty. First, techexplorer attempts to send the *appInput* and then the *defaultInput*, if *appInput* fails or is empty.

As XML becomes more and more the language of choice for markup applications, we anticipate exposing our internal XML parser to add-ins so that they can render or otherwise operate on the XML data.

4.3. Java connectivity

The crux of Internet development has historically focused on server-side applications. Developers relied on mechanisms such as the Common Gateway Interface (CGI) and World Wide Web server Application Programming Interfaces (APIs) to achieve interactivity within a Web browser. Today, the emergence of tightly coupled client-side APIs and Internet programming languages have changed the paradigm of Web-based application development. The Web browser has become an operating system that can support applications with a new level of interactivity.

Netscape LiveConnect allows developers to create connections between HTML elements, Java, JavaScript and Navigator plug-ins [6]. The LiveConnect technology utilizes the Java Native Interface (JNI) to "bridge" Netscape plug-ins and HTML elements to the Java environment. LiveConnect allows plug-in developers to define and implement Java classes in their C/C++ plug-in code. Such native Java methods can be referenced from both JavaScript and Java applets.

The techexplorer Web Developers Kit (WDK) includes a Java class techexplorerPlugin and corresponding native methods for manipulating techexplorer documents. The techexplorerPlugin Java class is instantiated each time the techexplorer plug-in is loaded. Once techexplorer is loaded, the techexplorerPlugin instance becomes available to the Netscape Java environment.

Navigator creates an "object instance hierarchy" for every HTML document. The "descendants" of the object hierarchy are document-dependent and thus reflect the structure of the HTML page itself. By default, each HTML page provides objects which contain information about the Web browser (navigator object), window or frame (window object), document contained in a window (document object) and URLs (links and history object). Additional objects can be added to the hierarchy by using the `NAME` attribute of the HTML tag.

Figures 2 and 3 depict the techexplorer Web Developers Kit (WDK) in action. In this example, the techexplorer plug-in is identified by the "techexplorer" object name; the HTML `FORM` is identified by the "latex" object name; the HTML `TEXTAREA` is identified by the "string" object name; and similarly the HTML `BUTTON` element is identified by the "send" object name. Within this hierarchy, the value of the HTML text area can be obtained by following the `window.document.latex.string.value` descendent chain. When the "Submit to techexplorer" button is pressed, the corresponding HTML element calls the JavaScript `Submit()` function. The `Submit()` function subsequently calls the techexplorer `ReloadFromTeXString()` method that redisplays the techexplorer window.

Our plan is to ultimately expand the techexplorerPlugin class to support the World Wide Web Consortium Document Object Model (DOM) interface [9]. Our Java implementation of DOM will allow Java applications to dynamically access and update the content, structure and style of techexplorer documents.

5. Future directions

It is our hope that with a publicly available add-in architecture and Java interfaces, techexplorer will serve as the "glue" for developing novel interactive scientific and technical documents. By using techexplorer, such documents will be able to leverage Internet programming languages, as well as specialized applications from the sea of heterogeneous scientific processors and renderers.

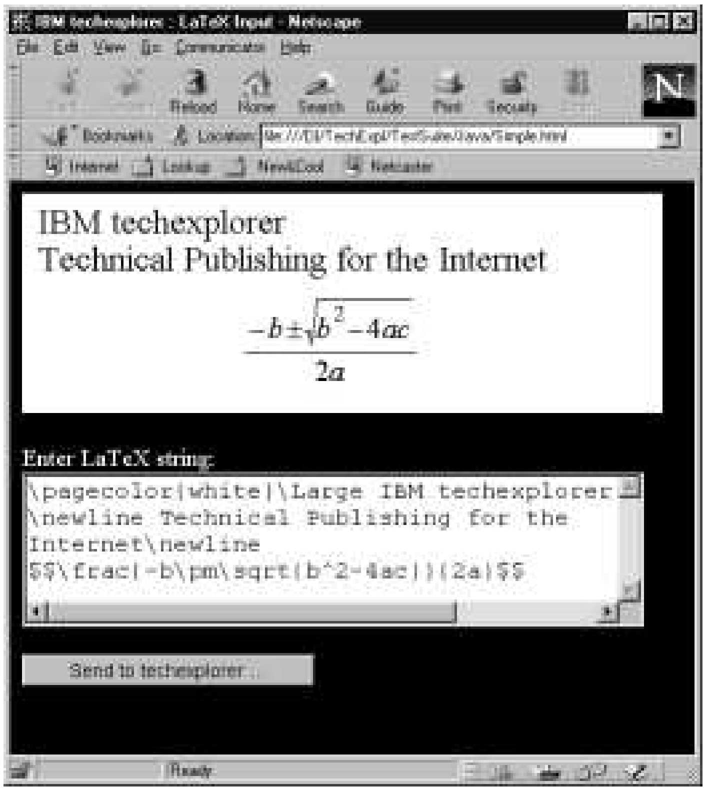

Figure 2 – A simple LaTeX editor

```
 <HEAD>
 <TITLE>IBM techexplorer : LaTeX Input</TITLE>
 </HEAD>
<BODY BGCOLOR="#000000" TEXT="#FFFFFF">


<SCRIPT LANGUAGE="JavaScript">
 function Submit( ) {
  window.document.techexplorer.ReloadFromTeXString(
     "techexplorer",
        window.document.latex.string.value );
 }
 </SCRIPT>


<EMBED TYPE="application/x-techexplorer"
        TEXDATA="\pagecolor{white}\Large
        IBM techexplorer\newline
        Technical Publishing for the Internet"
        NAME="techexplorer" MAYSCRIPT
        HEIGHT=40% WIDTH=100%>


<FORM NAME="latex">
       Enter LaTeX string:<BR>
       <TEXTAREA NAME="string"
         ROWS=4 COLS=40></TEXTAREA><BR><BR>
       <INPUT TYPE="button" NAME="send"
        VALUE="Send to techexplorer ..."
        OnClick="Submit()">
 </FORM>


</BODY>
 </HTML>
```

Figure 3 – A simple $\LaTeX$ editor: HTML Source

IBM and Waterloo Maple Inc. have developed a prototype add-in package that permits the transfer of math objects from techexplorer documents into Maple sessions. Using this add-in, we then developed a Java applet that can evaluate mathematical expressions by employing techexplorer as the conduit between Java and Maple. It is exactly these tightly coupled connections between external applications such as mathematical software packages and Java applications which will become the basis for the next generation of interactive scientific and technical publishing.

We plan to evolve the techexplorer scientific and technical publishing Web Developers Kit (WDK) to allow for the transmission of standard math presentation and semantic encoding protocols such as the World Wide Web Consortium HTML Math and OpenMath specifications. The HTML Math Content tags and OpenMath [7, 1] XML encoding detail the underlying mathematical structure of an expression, rather than any particular rendering for the expression. This intermediate semantic format will facility math object communication between varying math systems. In the future we expect users to employ the techexplorer Web Developers Kit (WDK) to transfer math objects between HTML Math or OpenMath compliant systems.

In 1998 we plan to introduce a "Professional Edition" of techexplorer that will include the Web Development Kit (WDK) with its add-in architecture and Java interfaces, printing, searching, and annotation.

Clearly, with the continued growth of a world-wide network and increased investments in scientific and educational content, traditional scientific markup languages will play a central role in online dissemination. The deployment of interactive scientific and technical documents using enhanced versions of these languages will be central to the success of the next generation of technical publishing. We in the Interactive Scientific Publishing group plan to continue enhancing our techexplorer products to support and help define professional-quality scientific online publishing in this next generation.

Bibliography

[1] J. Abbot, A. Diaz, R. S. Sutor. A Report on OpenMath. *SIGSAM Bulletin,* Volume 30, Number 1, Issue 115, New York: The Association for Computing Machinery (March, 1996).

[2] Bill Jacob. *Linear Functions and Matrix Theory.* Springer-Verlag, New York, 1995.

[3] Richard D. Jenks and Robert S. Sutor. *AXIOM: The Scientific Computation System.* Springer-Verlag, New York, 1992.

[4] IBM alphaWorks: http://www.alphaworks.ibm.com

[5] *IBM techexplorer Hypermedia Browser User's Guide:* http://www.ics.raleigh.ibm.com/pub/techug.html

[6] Netscape Corporation, *Using LiveConnect,* http://home.netscape.com/comprod/products/navigator/version_3.0/building_blocks/liveconnect/how.html

[7] OpenMath Consortium: http://www.openmath.org

[8] World Wide Web Consortium, Cascading Style Sheets (CSS): http://www.w3.org/Style/css/

[9] World Wide Web Consortium Document Object Model (DOM): http://www.w3.org/DOM/

[10] World Wide Web Consortium Mathematical Markup Language (MathML): http://www.w3.org/Math/

[11] World Wide Web Consortium Extensible Markup Language (XML): http://www.w3.org/TR/