Math on the Web: A Status Report

January, 2002

Focus: Authoring Tools

by Robert Miner and Paul Topping, Design Science, Inc.

View this paper on-line, where the links and references are live, go to **http://www.dessci.com/webmath/status.**

We plan on updating this report as the world of Math on the Web changes. Join our Math on the Web mailing list and we'll notify you when the report is updated: **http://www.dessci.com/webmath.**

Design Science
www.dessci.com
How Science Communicates™

Design Science, Inc. 4028 Broadway, Long Beach, CA 90803 USA, Phone: 562.433.0685 Fax: 562.433.6969

Math on the Web: A Status Report

Focus: Authoring Tools

by Robert Miner and Paul Topping, Design Science, Inc.

The last six months have seen very significant developments in Math on the Web. Effective, ubiquitous support for math notation in mainstream web browsers is finally becoming a reality. This edition of the Status Report is devoted to taking a closer look at the new generation of Math on the Web technology. We begin by examining recent breakthroughs in browser support, followed by a rundown of notable news and events for the last six months. In the Focus section, we conclude by looking at the status of authoring tools for this technology.

Big Strides Toward Math in Browsers

By most accounts, the rise to prominence of the World Wide Web started to take off in 1993. From the outset, critics were quick to point out that the utility of the web for scientific communication was severely limited by its lack of support for mathematical notation. Already by 1994, work had begun at the World Wide Web Consortium (W3C)[1], the standards body for the web, to develop an effective framework for Math on the Web. Since that time, work on Math on the Web has proceeded steadily in many quarters.

As veteran Math on the Web observers will tell you, however, the major milestones of the last five years have all addressed different facets of the problem. The frustrating result has been that the individual pieces didn't fit together into a complete solution. As a consequence, from the point of view of the average author, there has been little tangible progress in support for math in mainstream browsers. Starting with Internet Explorer 6 and the soon to be released Mozilla 1.0/Netscape 6 browsers, this situation is changing. The 6.x browsers implement a number of new standards-based technologies. Long in development, these new technologies make possible a quantum leap in math support.

The W3C has traditionally been a staunch supporter of math. Although most W3C member organizations support the idea of developing standards for scientific communication, most have little interest in actually implementing math-specific features themselves. As a consequence, the emphasis at W3C naturally turned toward the development of general-purpose extension mechanisms that could accommodate math rendering. While on the surface, native math support in browsers might seem preferable, a case can be made that the drive for general extension mechanisms actually serves the scientific community better. For one thing, dealing with math notation requires expert knowledge, and is better handled by companies focusing on that niche. For another, it permits competition between vendors of math renderers, which generally enhances quality.

The HTML Platform

The downside of using general extension mechanisms to handle math is that those mechanisms needed to be exceptionally powerful. Math is essentially a very complicated kind of text, and displaying text is the most basic and fundamental thing a browser does. Thus, developing mechanisms that could accommodate math has meant extending virtually every aspect of a browser's core rendering functionality.

The job of extending a web browser to handle math notation breaks into two broad subtasks. First, there must be a way of encoding math notation in the page. Second, there needs to be a way to teach the browser to display it, which is mostly a matter of hooking up add-on software of some sort. The first subtask is clearly something that needs to be handled in a standard way, so that an author can create a single document that works on all platforms. However, the second subtask is inherently browser-specific.

The task of encoding math notation in a web page was already largely solved in 1998 by the XML and MathML Recommendations, which respectively specify a general syntax for web documents and a specific vocabulary for describing math. Of course, the situation is actually a little more complicated since web pages containing MathML must also properly interact with standard ways of manipulating documents from scripts to make them dynamic, style information encapsulated in stylesheets, and so on. Nonetheless, the main outlines of what we might call the semantic extension mechanism have been worked out in a series of W3C Recommendations over the intervening years and are now in place.

In past editions of this Report, we have been calling the collection of W3C Recommendations which spell out the extension architecture for accommodating math in web pages "The HTML Platform". The main technologies are XML, HTML, and MathML for encoding content, XSL and CSS for styling and processing documents, and JavaScript and DOM for scripting of dynamic features in a page.

Exciting New Math Rendering Technologies

While MathML and the other constituent technologies of the HTML platform were being developed at W3C, much effort has been devoted to the second subtask, the software extension problem. A number of vendors have developed math rendering components for specific browsers and operating systems using Netscape Plug-ins, ActiveX controls, Java applets, and so on. However, in previous generations of web software, the integration between add-on software components and browsers has left a great deal to be desired from the point of view of math rendering.

Older software extension mechanisms tacitly assumed that add-on components would primarily be dealing with interactive content at the paragraph level in a document. As a result, applets, plug-ins and Active X controls don't work very well for rendering inline math notation interspersed with text. There were serious problems with alignment, sizing, and printing. Since the last Status Report in July 2001, there has been significant progress on improving the integration of math and text rendering in three different environments.

MathPlayer

Microsoft recently introduced a new technology called Behaviors[2], which allows low-level integration between an add-on component and Internet Explorer 6 on Windows. With Behaviors, it is possible to write add-on browser components that eliminate the earlier problems with alignment, sizing and printing. Using the new Behaviors technology, Design Science has developed a new MathML rendering component called MathPlayer. For IE users on Windows, MathPlayer promises much faster and more seamless MathML rendering than anything available until now. Since over 80% of the world browser usage is currently Internet Explorer on Windows, MathPlayer is a key ingredient in the browser math pie. MathPlayer is free in exchange for your email address. You can get it at http://www.dessci.com.

Better HTML Layout

If MathPlayer represents a new standard of performance for MathML rendering in browsers, the second major advance in rendering is at the other end of the spectrum. By taking full advantage of JavaScript and CSS control over HTML layout in the 6.x browsers, it has become much more feasible to produce legible, if somewhat crude, renderings of MathML expressions using only standard techniques of HTML layout and styling. While it was possible to use CSS and JavaScript in 4.x browsers to do math layout as well, the implementations of these technologies differed widely in browsers. In the new generation of software, the underlying standards have matured, and the implementation of the standards are more uniform and complete. Because the math rendering is being done with standard HTML techniques, it doesn't suffer from any of the integration problems that add-on component-based rendering does. It just isn't very fast or pretty. However, its existence as an acceptable fallback in standard browsers makes the benefits of MathML accessible to a much larger group of users.

MathML Support in Mozilla

The third significant development in the math rendering area is the announcement that MathML is now scheduled for inclusion in the 1.0 release of the Mozilla browser, currently slated for April 2002. The Mozilla approach to solving the add-on rendering component integration problem has been to build MathML support directly in the browser. The Mozilla announcement is significant because the commercial Netscape 6.x browser releases are closely based on the open source Mozilla code, and official inclusion of MathML in Mozilla increases the likelihood of math support in Netscape proper. The developments with Mozilla are also significant because they potentially have a major impact on Macintosh users. Microsoft's Behavior technology is not available in Internet Explorer for the Mac. As a consequence, MathML support in Netscape is probably the most likely avenue for high-quality, high-performance math support in a Mac browser.

The Universal Math Stylesheet

Given the advances in rendering software and coding standards, only one obstacle to ubiquitous and effective math support remains: different rendering technologies require bits of “glue code” to signal the browser how to handle the MathML equations it might encounter in a document. In some cases, this extra code takes the form of special declarations in the document header. In others, special wrapper code is required around each equation. In still other cases, a little code is required in both places. On the surface, this would seem to make it impossible for an author to publish a single document that simultaneously works in all rendering environments.

The solution envisioned in the HTML Platform is a standardized way of transforming parts of a document on the fly according to rules in a stylesheet. This powerful new stylesheet language is called Extensible Stylesheet Language (XSL), which became a W3C Recommendation in October 2000. XSL rules can take into account what browser is being used to view the page, and what add-on rendering components are installed. This enables authors to ignore the “glue code” that used to be necessary to fire up a specific rendering component to handle math notation. Instead, authors generate documents which are strictly standards-compliant, and at run time, the stylesheet running in the reader's browser adds whatever “glue code” is necessary to render MathML based on what is installed on the reader's system.

Internet Explorer 6 and Netscape 6 are the first browsers to fully implement XSL, the last major piece of the HTML Platform. To capitalize on the new technology, the W3C Math Working Group has recently released a “Universal Math” XSL stylesheet, developed by David Carlisle of $\LaTeX$ fame and an editor of the MathML 2.0 Specification. The stylesheet currently works with IE6 and Netscape 6.2, and produces legible renderings of strictly standards-compliant web documents on a wide variety of platforms.

The Universal Math Stylesheet searches through a list of possible rendering configurations and uses the first one that matches the reader's system. Authors can customize the order of the search, to specify a preferred rendering configuration on systems that have more than one available. In general, the stylesheet attempts to use native implementations and add-on renderers first. If that fails, it will generate HTML/CSS/JavaScript code on the fly to approximate traditional math layout in 6.x browsers.

The math rendered by the stylesheet ranges from crude but legible to very high quality depending on the combination of browser, operating system and add-on software. But for the first time, with the Universal Math stylesheet an author can be relatively certain that most of his or her readers will actually be able to see MathML equations in a web page.

Better Image-based Math support for Older Browsers

While superior solutions for Math on the Web are coalescing around the new 6.x browser technology, it is a fact that the vast installed base of 4.x browsers will continue to be a major force for several years. For this reason, images are still the best choice for reaching the largest audience. As is frequently the case in the history of technology, the mature and optimized solutions from the preceding generation of technology remain superior in practical application for quite some time as the bugs are worked out of new, immature technologies.

Handling Math on the Web via images is a good case in point. Web technology for images is highly advanced at this point, and Design Science MathType 5 is the last word in using images for Math on the Web. MathType 5 includes new MathPage technology which is a sophisticated Save As Web Page feature, that can produce either HTML + MathML or HTML + GIF documents from a Word document.

The HTML + GIF format uses JavaScript, basic CSS, and images to produce web pages with high-quality equation graphics. At the expense of being able to change the text size, equations are aligned and sized to match the surrounding text. Images are generated at several resolutions to match different display settings as well as printer resolution. Equations print at 300 dpi, or standard laser quality, which eliminates a long-standing weakness of using images for equations. MathPage technology is somewhat similar in concept to the JavaScript/CSS rendering produced by the Universal Math Stylesheet, which also seeks to use only native browser capabilities in an attempt to reach a broad audience. However, by using high-quality graphics and studiously avoiding new features only found in 6.x browsers, MathPage technology is able to simultaneously produce much higher-quality rendering while reaching the vastly larger audience using 4.x or later browsers.

While the progress toward ubiquitous, effective MathML support in browsers over the last six months has been enormous, it is still the province of early adopters, who enjoy installing the latest versions of software packages and don't mind troubleshooting the occasional glitch. By contrast, MathType's new MathPage technology offers a very easy and high-quality alternative for everyone else while the new technology matures.

News Round-up

This section spotlights important developments that have been announced since the last edition of the Status Report was published in July 2001. The list may not be complete, and the authors apologize in advance for any omissions.

- **MathPlayer 1.0 beta released.** Design Science announced[3] the release of MathPlayer, a MathML rendering behavior for Internet Explorer for Windows. MathPlayer offers significantly better performance and browser integration than previously available.

- **Universal Math Stylesheet released.** The W3C Math Working Group[4] made available an XSL stylesheet which attempts to determine the optimal way of displaying a document containing MathML given the browser and available add-on software on a reader's system.

- **MathType 5 released.** Design Science announced[5] the release of MathType 5, the professional version of the Equation Editor in Microsoft Office in October. The new release includes sophisticated new features for saving Word documents containing equations as web pages.

- **MathML to SVG prototype converter announced.** SchemaSoft made available[6] prototype software to convert MathML 2.0 to Scalable Vector Graphics 1.0 using XSLT 1.1, packaged as a Java executable. The intention is to facilitate "publication of MathML by conversion to a widely accessible vector graphics format" according to

Dr. Philip Mansfield, president of SchemaSoft and member of the W3C SVG working group.

- **MathML slated for inclusion in Mozilla 1.0.** The recently published Mozilla 1.0 Manifesto[7] makes MathML one of the default configuration features, and estimates a release date within six months. The current version of Mozilla is 0.9.6.

- **WebEQ 3 released.** Design Science announced[8] the release of version 3.0 of the WebEQ Developers Suite in December. This is the company's first major upgrade of WebEQ since acquiring the product and its development staff in June 2000.

- **LiveMath version 3.5 beta**. Theorist Interactive announced[9] version 3.5 beta of the LiveMath Plug-in, LiveMath Maker, and MathEQ Typesetter in September. The new version includes a Solaris version of LiveMath Maker. Work is in progress on a new version of the LiveMath Plug-in that will run in Internet Explorer 6 on Windows.

- **ActiveMath project announced.** ActiveMath[10], announced in November by DFKI[11], the German Research Center for Artificial Intelligence, has as its goal the development of “a web-based interactive learning system (for mathematics) that uses instruction as well as constructivist elements”. The project provides an architecture, basic knowledge representations, and techniques for new-generation online interactive mathematics documents (textbooks, courses, tutorials) and e-learning. ActiveMath uses the OMDoc format, an extension to OpenMath[12] standard for semantic encoding of mathematics.

- **jDVI released.** jDVI[13] is a new viewer for TEX DVI output. It can run either as an application or as a Java applet for displaying DVI files on the web. It supports most standard DVI viewer features as well as the ability to make hyperlinks, use color, and embed other applets within a document.

- **Question*mark* releases Perception 3.** Question*mark* announced[14] the release of version 3 of its online testing and assessment software in November, which includes MathML support among its new features.

- **WebCT licenses WebEQ.** E-learning solution provider WebCT[15] concluded arrangements to use Design Science WebEQ technology to add math support to its product lineup.

- **Math Forum moves to Drexel.** The Math Forum[16], the venerable math resource site whose Ask Dr. Math program pioneered math help for students on the internet, moved to Drexel University in September.

Focus: Authoring

While effective, ubiquitous support for math rendering in browsers is a necessary prerequisite for Math on the Web to achieve its full potential, it is not by itself sufficient. Documents must be created and published, and that means widely available, easy-to-use authoring tools are also required in order for Math on the Web to be useful for average authors.

Work on authoring tools has proceeded in parallel with work on browser support over the last several years. In fact, since math authoring tools are largely the work of individuals or organizations focused on math, rather than general web technology, progress on MathML support in authoring tools has generally out-paced progress on browser support. Nonetheless, without browser support, authoring tools have been effectively hamstrung, since there simply was no way to write out math expressions that were guaranteed to connect with a general web audience, other than images. Now that the situation regarding browser support for math is changing, mainstream authoring tool vendors are beginning to adapt their products accordingly. We expect to see major improvements in the authoring situation for Math on the Web over the coming year.

Varieties of Math on the Web

Math on the Web is a broad label, and a brief survey of the ways people are actually using Math on the Web suggests several natural divisions.

Static Math vs. Dynamic Math

The obvious distinction is that between static and dynamic math. Static math documents are those in which equations appear as part of the text, and do not change in response to interaction with the reader. Web versions of print documents all fall into this category. Dynamic math, by contrast, refers to any kind of interactive exposition involving math notation. Whereas static math is ideally fairly uniform from document to document, following traditional typesetting practices that have evolved over centuries, dynamic math is a new medium, and there are almost as many approaches to dynamic exposition as there are authors.

Static math represents the vast majority of Math on the Web when measured by volume. It stands to reason that simple and effective ways of authoring static math are key to the long-term success of the web for scientific communication. However, at the same time, it is clear that at least among certain audiences, part of the appeal of the web is the ability to do dynamic math. While publishing a web version of a print document (static math) adds convenience and accessibility, adding dynamic math to a document gives it impact. Especially in the area of education, it is the ability of an exposition to engage a student which makes it successful. As any professor who has sat through a long semester of lonely office hours knows, convenience and accessibility are not enough!

Articles, Assignments and Expositions

In addition to the static vs. dynamic dichotomy, which classifies documents according to media type, it is also useful to categorize documents by content. Most documents on the web containing math fall into one of three categories: research articles, assignments and other classroom documents, or instructional expositions of a topic. Generally speaking, documents within each category share an emphasis on one of the three major axes along which putting Math on the Web adds value -- accessibility, convenience, and impact.

For researchers, increased accessibility of Math on the Web is probably the dominant added value. Staying current and being able to find related work in a field are the critical needs of researchers. Increasingly, peer-reviewed journals are turning to the web to provide value-added features that increase accessibility — searching and indexing, maintaining errata, forward and backward reference tracking. Consequently, from the authoring point of view, researchers need tools that efficiently publish web versions of print documents in ways that maximize the ease with which other researchers can find them. Furthermore, research articles are fundamentally print documents (static math), regardless of whether or not they are distributed electronically, since readers nearly universally prefer a print document when intense study and concentration is required. Therefore, authoring tools for research on the web must make the production of very high-quality hard copy relatively easy.

The largest category by far of Math on the Web documents are those related to day-to-day course work, such as assignments, quizzes, practice tests, syllabi, etc. Like research articles, these documents must typically be prepared simultaneously in print and web form, since hard copies are typically handed out in class. However, unlike research articles, now the web version of the document exists primarily for convenience rather than accessibility; an instructor doesn't care much about reaching students in other classes, but does want an easy way to get the assignment to the student who missed class last Thursday. The implication for authoring tools is that producing a web version should take very little additional thought or time above and beyond what would be necessary to create the paper version. Also, since students are rarely willing or able to maintain a state-of-the-art Math on the Web rendering environment, authoring tools must generate web documents that don't require any special browser configuration, setup, or fat bandwidth.

Expository Math on the Web documents are much harder to quantify than research articles and course work. Compared to the other two categories, there are fewer examples, and the range of approaches is extremely varied. Nonetheless, a couple of clear authoring patterns are emerging. One is what might be called the "mathlet" — usually an applet devoted to interactively demonstrating a single concept, supported by several pages of static math exposition, often with heavy use of graphics. Another common pattern is the computer algebra notebook paradigm, where a piece of expository text has certain "live" equations within it that can be manipulated in various ways by the reader to gain a fuller understanding of the topic. Both these patterns rely heavily on dynamic math.

With expository Math on the Web documents, the most important thing is that they be engaging. In some cases, such as certain distance learning contexts, online exposition is forced to stand in for live instruction. In other cases, these documents have been created to supplement traditional classroom instruction in attempt to connect with students who, for whatever reason, just didn't get it during class. In either case, the intent is to have an impact on the reader above and beyond what is possible with text.

Looking at the dynamic math sites online today, it is clear that authors of these kinds of documents tend to be more technologically sophisticated, and willing to devote more time and effort to the authoring process to achieve a desired effect. Similarly, within limits, readers of dynamic math materials are probably more willing to put greater effort into browser configuration and endure longer download times. Consequently, at this early stage in the development of dynamic math, the main pressure on authoring tools is for added functionality. As this usage category matures, one can expect to see this emphasis shift toward ease of use, as more people become interested in taking advantage of the potential of dynamic math.

Authoring Tools for Research

The huge majority of scientific research documents are authored in one of two ways: as Microsoft Word documents containing Equation Editor or MathType equations, or as some flavor of TEX, with LATEX being the most common. (Hereafter we will use the term TEX generically to refer to all flavors unless a specific distinction needs to be made.) Within the mathematics and physics communities, TEX is the dominant format, while Word is more prevalent in most other research disciplines.

TEX Converters

As of today, the majority of research articles are published to the web as PDF files, prepared using pdfTeX[17]. However, looking down the road, the HTML + MathML format probably offers more opportunity for value-added accessibility services, and therefore, that format is our emphasis here.

Regardless of the ultimate output format, the overwhelming majority of TEX authoring takes place in a text editing environment of some kind. There are a number of TEX-specific editing products, such as WinTEX 2000[18], WinEdt[19], etc, as well as TEX support in general-purpose text editors such as Emacs[20] and BBEdit[21]. These products typically add features like syntax coloring for TEX commands, help with making braces match up, easy ways to run TEX and preview the results, etc. In all cases, however, the end result of the editing process is a TEX file, which is compiled into a DVI file for printing.

Two exceptions to this rule that deserve special mention are Scientific Word[22] and Textures[23]. Scientific Word provides a WYSIWYG TEX authoring environment wedded to a computer algebra kernel. Version 4.0 already provides a "Save As HTML" feature that generates images for equations. MathML support is planned for future versions.

Textures provides an interactive, integrated TEX editing environment as well. As of version 2.1, however, the only support for exporting documents to the web was the ability to save a typeset page as a JPEG image. Consequently from the point of view of authoring for the web, Textures is not very different from other text-based editors.

Given a TEX document, there are two basic strategies for converting it to HTML + MathML. The first analyzes the original TEX source file to create an HTML + MathML document. The second strategy involves converting the DVI output into HTML + MathML. The advantage of converting the original source is that in general it contains much more information about document and equation structure. A DVI file is more like a long list of characters and the sizes and positions at which to render them; thus, it is very difficult to recover structure from the DVI output. However, the appeal of converting a DVI file is it avoids all the issues surrounding different flavors of TEX and will work with even the most non-standard user defined macro packages.

The earliest and perhaps most well-known TEX-to-HTML converter is the LaTeX2HTML[24] package. It employs the first strategy of converting TEX source to HTML. While some experimental versions of LaTeX2HTML now generate HTML + MathML, output from LaTeX2HTML is primarily oriented toward HTML + images. A more recent source-level converter that does generate HTML + MathML is TtM.[25] TtM is a modified version of TtH which converts TEX to pure HTML, using a combination of fonts, tables, CSS to do math layout for equations. TtM uses its own parser to process TEX input and directly generate HTML + MathML. Omega[26], a modified and extended version of the TEX engine itself, can also generate MathML. It's approach also focuses at the source level.

TEX4ht[27] is probably the most powerful and sophisticated converter currently available. It uses a kind of hybrid approach, first performing analysis at the TEX source level which it uses to insert hints into the DVI file using special commands. It then processes the DVI output to generate the final document. TEX4ht is highly configurable and can be used to generate output in a wide variety of XML dialects in addition to HTML. The advantage is superior output, but the disadvantage is a fairly steep learning curve.

In general, currently available TEX to HTML + MathML converters are probably most accurately characterized as being in an experimental state. However, now that there is progress on rendering, it is not unreasonable to expect to see renewed interest in TEX conversion as well. To facilitate this work, a subgroup of the W3C Math Working Group chaired by Ivor Phillips of Boeing is developing a TEX conversion test suite and working with vendors on improving their TEX translators.

MS Word and MathType

A survey of technical publishers conducted by Design Science indicated that 75% of STM documents published are authored in Microsoft Word. For documents that require math notation, the only real choices are to use the Equation Editor included with Word or to upgrade to MathType, the professional version of Equation Editor. For authors that only require occasional use of math notation and are interested only in producing print documents, Equation Editor is likely sufficient. However, for Word authors making heavy use of mathematical notation or requiring web output, MathType is an almost essential tool.

Word provides a default “Save As Web Page” function that can be used after a fashion for documents created using Equation Editor. The output is essentially HTML in which equations have been replaced by images. However, the resulting output has a number of fairly severe problems, including:

- The HTML itself is extremely hard to work with and contains a large quantity of Microsoft-specific markup

- Equation images don't align properly with the surrounding text
- Equation numbering is lost
- Equations print at screen resolution

To address these problems, MathType 5 adds its own export-to-the-web functionality. MathType adds a new button to the Word toolbar which brings up an "Export to MathPage" dialog. From this panel, an author can configure a number of export options. The most important export configuration option is the choice between generating HTML + MathML or generating HTML + GIF images. We will discuss HTML + GIF further in the next section. Here we focus on HTML + MathML export.

MathType generates presentation MathML using a rule-driven translator mechanism. The rule sets are ordinary text files that sophisticated authors can customize to tweak the MathML being generated. A small number of MathType constructions have no MathML equivalents and cannot be translated. In these cases, the translator mechanism warns the author and omits the problem construct.

In the currently shipping version of MathType 5, authors are obliged to choose between nine MathML target platforms, each of which generates the extra glue code or document declarations required by specific add-on components, such as MathPlayer, WebEQ Viewer Control, or Techexplorer, or the MathML-enabled browsers Mozilla and Amaya. Regardless of the target platform, MathType translates its equations into the same MathML expressions; the difference lies entirely with the glue code needed up until now to have MathML render in a browser. With the advent of the Universal Math Stylesheet described in the first section of this report, Design Science is planning a maintenance release of MathType adding that as a target platform.

MathType's MathPage technology also processes the HTML markup for the rest of the document generated by Word. MathPage always removes most Microsoft-specific markup, but authors can choose whether to allow some optimizations for Internet Explorer 5 and above. Disabling the optimizations slightly degrades performance when viewed with Internet Explorer, but generates better cross-platform results. For some target platforms, this choice is disabled since, for example, MathPlayer is only available for Internet Explorer under Windows, and MathML support in Mozilla requires that the surrounding HTML conform to stricter XML syntax rules, a format called XHTML. In these cases, the MathPage exporter automatically cleans up the Word HTML as necessary.

Authoring Tools for Course Work

At the college level, most instructors are also engaged in research. This puts strong pressure on instructors to use the same authoring tool for course work that they use for research articles, since the initial investment required to learn to use two authoring tools is prohibitive. As a general rule, therefore, one finds that TEX authors use TEX for classroom documents, while Word authors use Word. As noted above, for course work documents, maximizing convenience for teachers and students is paramount. So, in this section, we briefly revisit TEX and Word as authoring tools with that in mind.

TEX Tools for Course Work

Most experienced TEX authors have extensive libraries of template documents that only require slight tweaking from semester to semester. Once the initial investment has been made (something that happened long ago for most TEX authors), the incremental effort required by the author is minimal. So, convenience is not much of an issue for TEX authors. To the student, however, convenience is defined in terms of being able to readily view and print documents using lowest common denominator computer equipment, which he or she frequently does not control.

From this point of view, TEX is not a particularly convenient format. As a consequence, TEX authors generally must employ additional tools to prepare course documents in more convenient formats. The main options are:

LaTeX2HTML

As noted above, straight HTML + GIF images is a good format for insuring the widest range of students can conveniently access a document. The most well-known TEX converter producing HTML + GIF images is LaTeX2HTML. However, there are many others as well.

pdfTeX

It is a simple matter for most authors to produce Adobe's PDF format using pdfTeX. Since the Acrobat Reader is widely available and comes pre-installed on the majority of new computers, viewing and printing PDF output is generally not a problem for students.

Techexplorer

Techexplorer[28] itself isn't, properly speaking, an authoring tool. However, since it will display a raw TEX file, provided the author sticks with standard TEX dialects, we include it here since it facilitates the use of a plain text editor as an authoring tool. As a consequence, the Techexplorer plug-in is a very convenient option for the author since no additional processing is required once the TEX source has been created. From the student perspective, however, Techexplorer is not so convenient since it requires installation, and in fact, it must be purchased to enable printing. However, in some situations where a teacher can guarantee access and installation to students, say via a computer lab, Techexplorer can be at least a viable alternative in some cases.

MathType Revisited for Course Work

Whether authoring research articles or course documents, using Word and MathType involves basically the same effort on the part of the author. Since these tools are visual tools, designed specifically to minimize the barriers to getting started with them, they require much less of an initial investment than TEX on the part of the author to learn how to use them. Consequently, in the world of course work where convenience is paramount, Word + MathType offers a substantial advantage to authors who don't already know TEX.

As far as creating convenient electronic versions of documents for students is concerned, the “Export to MathPage” technology in MathType 5 is again key. However, in this arena, it is the new HTML + GIF format, and not MathML format, which leaps to the fore. Generating these documents is superlatively easy for authors, requiring nothing more than clicking a button. However, the big win is from the student perspective. An HTML + GIF document displays in an ordinary web browser on nearly any platform just like any other web page, except that it now contains great looking math that prints nicely.

By using carefully crafted JavaScript and CSS directives, MathPage achieves platform independence without sacrificing quality and effectively addresses all of the issues listed above with Word's default Save As Web Page output. Another important point to note is that HTML + GIF eliminates the need for special fonts containing math symbols to be installed on the reader's system, which remains a serious issue with most MathML rendering software.

The key to the HTML + GIF format is the generation of images at several resolutions. This enables documents to display low-resolution images that match the screen resolution in a browser, while using high-resolution 300 dpi images when printing to a laser printer. The availability of different resolution images also enables the “MathZoom” feature that magnifies an equation at a mouse click to reveal fine detail that can often be difficult to make out at screen resolution. Another useful feature of HTML + GIF is that the images have additional information embedded in them which makes it possible to drag an equation from a web page into MathType for editing.

Authoring Tools for Dynamic Math

For interactive exposition, there is a whole panoply of tools, each generally aimed at a specific strategy. This area is still very new and rapidly changing. In general, the existing tools are fairly rudimentary. As a result, most dynamic math authoring that has taken place to date has been a matter of hand coding. However, there are three broad categories of tools that have some dynamic math capabilities.

Computer Algebra System Notebooks

Maple[29], Mathematica[30], and MathCad[31] all provide "Save As HTML" functionality for their notebook documents. In the case of Maple and Mathematica, interactivity is limited to the ability to cut and paste MathML expressions from the HTML output back into a notebook for evaluation or other symbolic manipulation. The MathCad output documents use the Techexplorer plug-in to connect to a local copy of MathCad on the reader's computer to do computations in place in the web page.

With all of these products, more sophisticated online interactivity capabilities and authoring tools are under development. Wolfram Research has already released part of such a solution in the form of webMathematica, a server version of the Mathematica computation engine. webMathematica can also function as an enhanced web server, interpreting special commands that authors can embed in an HTML page which request that the output of computations be inserted in the page. This is an example of a rapidly developing area of interest at the World Wide Web Consortium called Web Services, in which a number of computer algebra vendors are active. However, with the current generation of technology, authoring is still really in the province of the technically adept programmer coding by hand.

WebEQ Developers Suite

From the point of view of the HTML Platform, the proper way of doing interactivity not involving computation is to dynamically modify the document by using script code embedded in the page, triggered by the user clicking on buttons, entering text, etc. From the point of view of MathML, which originally addressed the issue of interactivity before the vision of the HTML Platform had really emerged, the way to do interactivity is to use MathML actions, which are encoded directly in the equation markup. There is already fairly extensive support in browsers and add-on rendering software for both kinds of interactivity, and much of the dynamic math currently available on the web already makes use of these capabilities. However, just as is the case with Web Services, authoring dynamic math using these techniques requires skill with programming and a strong background in web development. The one exception to this is the recently released WebEQ 3 Developers Suite, which makes authoring at least some dynamic math substantially easier then it has been until now.

The WebEQ Developers Suite is actually a collection of 5 tools:

- WebEQ Editor for authoring presentation and content MathML
- WebEQ Publisher for processing HTML pages containing math markup
- WebEQ Input Control which functions as an easy-to-use graphical equation editor in a web page
- WebEQ Viewer Control that displays MathML in any Java-capable browser
- WebEQ Equation Server which works behind the scenes to facilitate batch processing and processing via scripts on a server

The main audience for the Developers Suite is primarily web-savvy developers who are used to hand-coding scripts to wire together components. However, the Editor and the Publisher are both relatively easy-to-use graphical tools that make at least basic dynamic math authoring possible for authors who only have modest web skills.

WebEQ Editor gives authors a graphical way to insert MathML actions into equations. MathML actions trigger one of a handful of dynamic behaviors when a reader moves the mouse over a part of an equation or clicks on it. The available behaviors are changing the foreground or background color of an expression, toggling between two expressions on a mouse click (such as question mark and an answer,) linking from part of an equation to another document, or displaying a message in the status line of the browser. Several MathML renderers will display MathML actions; however, MathML actions authored with WebEQ Editor are optimized for display with the WebEQ Viewer Control. In particular, WebEQ Editor can automatically generate the applet code necessary to instantiate the Viewer Control in a web page.

WebEQ Publisher is essentially a converter program designed to scan through an HTML source document looking for math markup which it then processes and writes out into an output HTML document. The Publisher recognizes two kinds of input markup: MathML and WebTeX. WebTeX is similar to the math portion of LaTEX, with some changes and extensions. In particular, WebTeX introduces new commands such as \hilight for creating MathML actions. The Publisher can be used to translate WebTeX into MathML and write out the necessary wrapper code to display dynamic equations with the Viewer Control.

Proprietary Approaches

A number of companies have fielded self-contained proprietary approaches to doing dynamic math. Three that are especially worth mentioning are LiveMath, Mathwright[32], and Poliplus EqnWriter[33]. All these products are similar in that they provide an authoring environment and a browser plug-in or applet that displays their proprietary formats. The plug-in portions of all of these products basically function as mini computer algebra systems, and are primarily designed to run in a large rectangular region of a browser Window. The authoring portion of these programs creates something akin to a computer algebra notebook, which a student then manipulates in the plug-in window.

While these proprietary approaches have some merit, they are mostly of interest in situations where the author has a close relationship with the reader, and has some control over the setup of the reader's machine. The main advantage is that, because of the proprietary nature of the format, the authoring environment and the plug-in work well together. However, in the longer run, it is not clear whether these proprietary approaches will survive as more mainstream authoring tools begin to better serve the demand for dynamic math under the HTML Platform.

Conclusion

Many individuals and organizations have been working to establish a ubiquitous, effective framework for Math on the Web for nearly a decade. While progress has been steady, until recently, the successes along the way haven't come together into a useable solution for mainstream authors and readers. Over the last half of 2001, however, the pieces have finally started to come together: full implementation of the HTML Platform in 6.x browsers, new MathML rendering software, and the Universal Math Stylesheet to mediate between the two. As this new generation of software begins to be disseminated, the widespread use of MathML for scientific communication becomes truly practical. In anticipation of a critical mass of users and readers, advances in user-friendly authoring tools and interoperability between math-aware applications are already beginning to make their way into the marketplace. In future editions of this Status Report, we look forward to reporting on the advancement of accessibility, convenience and impact in scientific communication which MathML makes possible.

References

[1] World Wide Web Consortium, http://www.w3.org
[2] Microsoft Behaviors, http://msdn.microsoft.com/library/default.asp
[3] Design Science MathPlayer, http://www.dessci.com/webmath/mathplayer/
[4] Universal Math Stylesheet, http://www.w3.org/Math/
[5] Design Science MathType, http://www.dessci.com/company/press/releases/oct01.stm
[6] SchemaSoft, http://www.schemasoft.com/MathML/
[7] Mozilla 1.0 Manifesto, http://www.mozilla.org/roadmap/mozilla-1.0.html
[8] Design Science MathType, http://www.dessci.com/company/press/releases/dec01.stm
[9] Theorist Interactive LiveMath, http://www.livemath.com/
[10] The ActiveMath Project, http://www.mathweb.org/activemath/
[11] The German Research Center for Artificial Intelligence, http://www.dfki.de/
[12] The OpenMath Society, http://www.openmath.org/
[13] jDvi, http://www-sfb288.math.tu-berlin.de/jdvi/home.html
[14] Questionmark, http://www.questionmark.com/us/news/pressreleases/perceptionv3_november_2001.htm
[15] WebCT, http://www.webct.com/
[16] MathForum@Drexel, http://mathforum.org/
[17] pdfTeX, http://www.tug.org/applications/pdftex/index.html
[18] WinTeX 2000, http://www.tex-tools.de/main.html
[19] WinEdit, http://www.winedit.com/
[20] Emacs, http://www.gnu.org/software/emacs/
[21] BBEdit, http://www.barebones.com/
[22] Scientific Word, http://licensing.mackichan.com/
[23] Textures, http://www.bluesky.com/
[24] LaTeX2HTML, http://cbl.leeds.ac.uk/nikos/tex2html/doc/latex2html/latex2html.html
[25] TtM, http://hutchinson.belmont.ma.us/tth/mml/
[26] Omega, http://omega.cse.unsw.edu.au:8080/index.html
[27] TeX4ht, http://www.cis.ohio-state.edu/~gurari/TeX4ht/mn.html
[28] Techexplorer, http://www-4.ibm.com/software/network/techexplorer/
[29] Maple, http://www.maplesoft.com
[30] Mathematica, http://www.wolfram.com
[31] MathCad, http://www.mathsoft.com
[32] Mathwright, http://www.mathwright.com
[33] Poliplus Eqn Writer, http://www.poliplus.com

Design Science
www.dessci.com
How Science Communicates™