WME: Towards a Web for Mathematics Education

Paul S. Wang[†*] Norbert Kajler[‡] Yi Zhou[†] Xiao Zou[†]
[†]Institute for Computational Mathematics,
Kent State Univ., Kent, Ohio 44242-0001, USA
[‡]Ecole des Mines de Paris, 60 bd. Saint-Michel, 75006 Paris, France

January 18, 2003

Abstract

Reported is an approach for *Web-based Mathematics Education* (WME). The WME framework is a distributed system that aims to create a *Web for mathematics education*. Components of the WME framework include the *Mathematics Education Markup Language* (MeML) for page markup, regular Web servers to deliver pages, WME Page Processors enabling common Web browsers to receive MeML pages, and a variety of WME services (mathematical and educational) to supply power and interactivity to MeML pages. The MeML language enables the creation of efficient, effective, and dynamic mathematics education content. The WME architecture is designed so that mathematics education content (both static and dynamic – through remote WME services) can be built independently, maintained independently, deployed easily, and still interoperate and take advantage of one another on a global scale.

1 Introduction

Recent years have seen much increased research and development activities in making mathematics accessible on the Web and Internet. Publishing mathematical materials on the Web is made easy by MathML [17], an XML-defined language for markup of mathematical expressions with support for both presentation encoding (display layout) and content encoding (computation semantics).

The list of MathML-compliant software is growing fast. Among many others, *WebEQ* and *MathPlayer* [21] can display WebTeX and MathML in a browser. The *Amaya* Web browser demonstrates a prototype implementation of MathML which allows users to browse and edit Web pages containing mathematical expressions [27]. Together with the rest of the Web page, these expressions are manipulated through a WYSIWYG interface. The increasing

*Work reported herein has been supported in part by the National Science Foundation under Grant CCR-0201772 and in part by the Ohio Board of Regents Computer Science Enhancement Funds.

acceptance and software support for MathML were evident at the 2000 and 2002 *MathML International Conferences* [19, 20].

Mathematical content viewing on a Web page is static. On the Internet, end-users, especially those learning mathematics, can make good use of *dynamic access to mathematical computing*. To this end, the online exchange of mathematical data for computation becomes critical. Efforts to establish common formats include MathML content encoding, OpenMath, and MP.

Besides the widely-used MathML, MP, the *Multi Protocol* [8], is a format that uses a binary encoded annotated parse tree for efficiency. OpenMath [4] is a protocol for representing semantically rich mathematical objects, allowing them to be exchanged between programs, stored in databases, and published in electronic form.

"Internet Accessible Mathematical Computation" has been the subject of the *IAMC Workshop series*. At the Institute for Computational Mathematics (ICM/Kent), efforts have been made to build a *distributed IAMC framework* [36, 29, 30] which can support both interactive and transparent access to mathematical computation on the Internet/Web through the *Mathematical Computation Protocol* (MCP). For more information on IAMC, please refer to the Proceedings of the IAMC Workshops [11], the IAMC homepage [10], and the Workshop on *The Future of Mathematical Communication* [7].

Researchers have begun to make attempts to deliver mathematical education materials over the Web/Internet. Already, we can find many websites providing courses and tools for mathematics education. Such sites include WIMS [34], Livemath [15], Mathwright [24], WebMathematica [33], Calc101 [3], ActiveMath [1], Maple [16], and MathWeb [23]. WIMS, for instance, is a well-known site for Web-based mathematical education created by Xiao Gang. WIMS [34] offers great features such as simplicity, availability (no cookies, no plugins), smooth interface to a bunch of external servers (MuPAD, PARI/GP, Coq, Gnuplot, ...), and support for virtual classrooms allowing teachers to assign a set of exercises to be performed by each student. WIMS uses a case-oriented approach by providing a new CGI program for each new type of educational content page. Trouble with such an approach is lack of scalability and openness. Authoring educational content within the WIMS scope can be awkward but the most serious flaw of such an ad hoc approach is that the WIMS components (content pages and server-side programs) do not combine to form an open system within which to interoperate and to mutually reinforce. Linda Becerra, et al. [2] gave a good summary of Web tools for interactive computation.

It is perhaps time to consider a systematic way of creating and supporting mathematics education on the Web in an open and scalable way.

Towards this end, we present the design and architecture of a *Web-based Mathematics Education* (WME) framework. WME provides an authoring language (MeML), works with regular browsers, makes authoring simple and easy, allows systematic access to supporting *WME services*, and enables these independently developed components to interoperate seamlessly. In short, we hope the WME framework will help create a *Web for Mathematics Education*, that is, an environment in which mathematics education content, including *both* static and dynamic content, can all be built independently, maintained independently, placed on any Web server, delivered to any Web browser, and still interoperate and take advantage

of one another on a global scale.

Goals, design, architecture, and software prototypes of the WME framework are presented. The *Mathematics Education Markup Language* (MeML), as well as its client-side processing and server-side services are described.

2 WME Goals

The goal of our research is to design an infrastructure, called the WME framework, for mathematics education that allows easy and systematic development of:

1. mathematics educational content, and,

2. mathematics education support capabilities

both accessible and interoperable on the Web/Internet.

With the WME framework, educators can produce Web pages to teach specific mathematics lessons and topics. These pages can invoke support services such as example generation, answer checking, graph plotting, and interactive exploration for various mathematical subjects.

Thus, content pages and support services can be developed independently but can interoperate from anywhere on the Web. With the right design and implementation, the WME framework can become the infrastructure on which to build effective mathematics education systems on the Web. Figure 1 shows the WME objective.

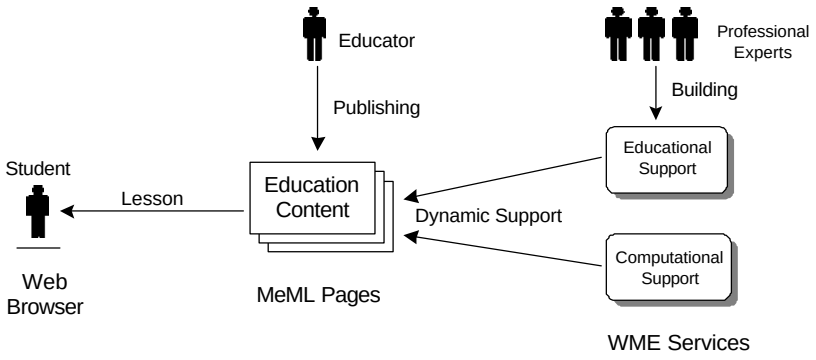

Figure 1: WME Goals

The WME framework has these characteristics:

- WME is distributed allowing everyone to put WME pages on the Web and to supply support services.

- WME combines existing technologies and emerging Web standards.

- Mathematics education content pages are easy to create, simple to edit, delivered by regular Web servers, and accessed via regular Web browsers powered by the *WME Page Processor*.
- Education pages may combine fixed information with dynamically generated content and interactions with the end user.
- WME services can supply a variety of useful content and functionalities to educational pages from anywhere on the Internet.

Clearly, for WME to work, it must have content-markup support, front-end support, and back-end support. More importantly, such support must be delivered within an architectural framework that uses appropriate technologies to integrate components, allowing them to interoperate in a seamless manner on the Web.

3 The WME Architecture

Figure 2 shows the WME framework architecture.

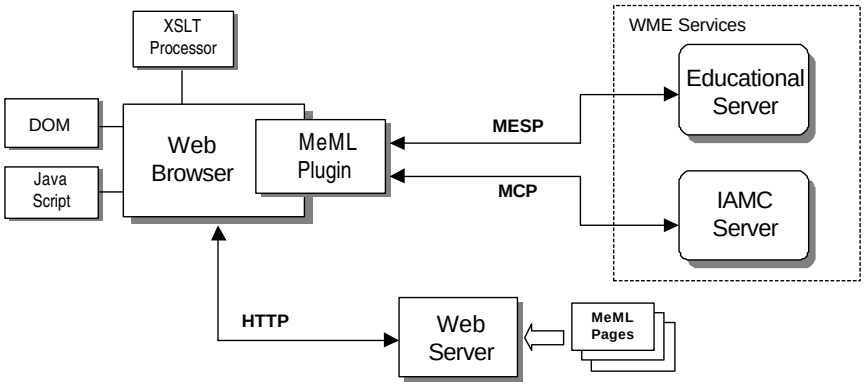

Figure 2: WME Architecture

WME achieves Web-based mathematical education through a combination of technologies.

- *Content-markup support*—The *Mathematics Education Markup Language* (MeML) is used to encode Mathematics education content pages (Section 4). MeML provides well-defined *mathematics computation elements* and *mathematics education elements* that can be used together with HTML and MathML elements for easy content markup. MeML also provides elements designed to access *WME services* that supply various mathematics education capabilities.
- *Regular Web servers*—MeML pages, with the `.meml` suffix, originate from normal Web servers with the content type `text/x-meml`.

- *Front-end support*—On the client side, common Web browsers can be used. A *WME Page Processor* (Section 5) in the form of a browser plug-in (the *MeML Plug-in*) provides control and processing for the MeML content, using the browser as a GUI.

- *Back-end support*—WME services (Section 6) can be supplied to MeML pages from anywhere on the Internet. The WME Page Processor can access WME services through the *Mathematics Education Service Protocol* (MESP) and through the *Mathematical Computation Protocol* (MCP) [30].

Thus, the WME architecture involves these major components: the Mathematics Education Markup Language (MeML), educational content pages written in MeML, standard Web servers and browsers, the WME Page Processor, and WME services.

Usage Paradigm

In the WME framework, an online courseware consists of a group of MeML pages with dynamic back-end support by WME services that can supply a rich set of computational and educational functionalities useful in many different MeML pages. Figure 3 shows the WME framework concept.

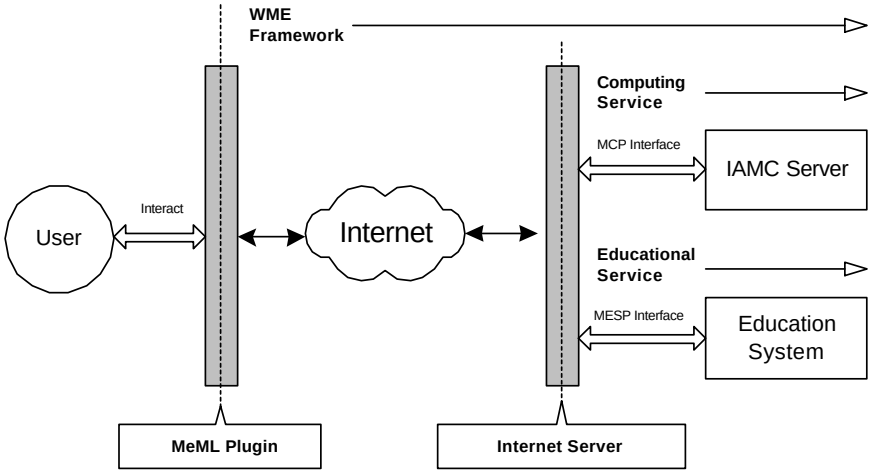

Figure 3: WME Conceptual Model

With WME, a new paradigm of Web-supported mathematics education emerges:

- Students can access mathematics education pages for a dynamic and interactive learning experience from anywhere on the Web, at anytime. They can study at their own levels and paces. Instructional materials may come largely from courses in schools, but the materials can be supplemented by pages from other authors globally.

- Software developers can focus on building WME services that can be accessed by URL from anywhere on the Web. General-purpose WME services such as plotting/graphing, terminology lookup, and computation execution are useful in many different situations.

Special-purpose services provide for narrowly focused areas. Each service can provide a well-defined set of capabilities within a particular scope. For example, there may be a *fractions service*, a *polynomial service*, an *algebraic equation service*, a *derivative service*, an *integration service*, and so on. As long as the services conform to a common interface, they can be used in any education content pages. We also envision a variety of back-end educational services including example (counter example) generation, answer verification, intermediate steps production, illustration generation, word problem generation, test generation, test grading, and performance evaluation.

- Educators author interactive online courseware as a set of educational pages. Such pages are written in MeML either directly or via authoring tools. Authors can focus on teaching mathematics rather than dealing with the underlying Internet/Web technologies. An author can easily include, mix and match, powerful interactive mathematical education features by accessing WME services made available by others.

- Educators deploy and maintain their courseware simply and easily. Authors can modify, revise, and change their MeML pages anytime from anywhere.

- Service developers can upgrade and improve WME services without affecting MeML pages as long as interface compatibility is maintained.

4 MeML

The *Mathematics Education Markup Language* (MeML) is central to the WME framework. Defined in XML, MeML provides markup elements to represent and structure mathematics education pages. An MeML page may contain XHTML, MathML, and MeML elements. MeML elements support mathematical expressions, formulas, lessons, examples, exercises, tests, interactive computations, experiments, explorations, and other educational functions. The elements are designed for expressive power and ease of use.

Currently, we have about 40 MeML elements to support the various markup needs for mathematics education. The elements can be divided roughly into five categories: *content elements*, *education elements*, *organization elements*, *computation elements*, and *interaction elements*.

- Content elements—for units of knowledge. Such tags include `<concept>`, `<skill>`, `<terminology>`, `<expression>`, `<equation>`, `<diagram>`, and `<theorem>`.

- Education elements—for objectives in education. Tags include `<example>`, `<test>`, `<exercise>`, `<assessment>`, `<diagnosis>`, and `<remediation>`. Contents for these elements can be supplied explicitly or can be generated. Figure 4 shows a generated example for fraction addition. The page also contains underlined links that lead to further details.

- Organization elements—for page content organization. These include `<abstract>`, `<lesson>`, `<lesson-opener>`, `<syllabus>`, `<roster>`, `<guide>`, `<hint>`, and `<summary>`.

- Computation elements—for explicitly invoking WME services to generate dynamic content. Such tags include `<computation>`, `<mathgraph>`, and `<wme>`.
- Interaction elements—for receiving and processing user input. These mainly include the `<interaction>` and `<userinput>` elements.

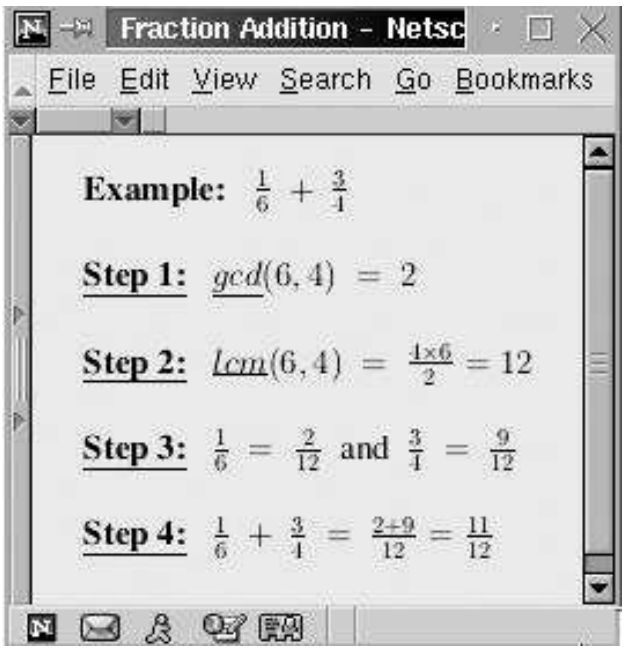

Figure 4: Generated Example

CSS (Cascading Style Sheet) rules and Javascript code can be associated with MeML pages in exactly the same way as in XHTML.

With MeML, an author can write HTML-like code to create mathematics education pages. Mathematical expressions can be encoded in MathML content (for computing), MathML presentation (for rendering), or infix (for convenience).

An MeML page can have both fixed and dynamically generated content. The latter can be supplied by any WME service identified by a URL. MeML pages can also specify interactions with the end-user (the student) for an engaging learning experience.

MeML allows infix notation as a convenience for authors and end-users to enter mathematical expressions. Infix expressions are translated into equivalent MathML content encoding automatically within the WME framework. Functions and mathematical elements defined in MathML can be used in the infix notation. Mathematical constants can be entered easily. For example, π is entered as `PI` and translated into `&pi;`. Several other simple rules apply.

A Sample MeML Page

Examples can help further illustrate MeML. Examples 1 and 2 below give a complete sample MeML page that teaches the *sin* function with interactive curve plotting.

Example 1: A lesson to teach the function *sin*

Defined first are some internal definitions. These are not displayed and therefore hidden from the end user:

```
<internal>
  <variable name="var" type="symbol"> x </variable>
  <variable name="from" type="real"> -10.0 </variable>
  <variable name="to" type="real"> 10.0 </variable>
  <expression name="func" encoding="infix"> sin(x) </expression>
</internal>
```

The variable named `var` has the symbol value `x`. The mathematical expression `func` has its value given by infix notation.

Then, the lesson begins with some textual content defined below. Note: `use`, a replaced inline element, inserts the value of a variable or expression.

```
<lesson title="The Sine Function" id="sinfun">
  <p>The function <em>sin</em> is important in trigonometry.
    <a href="#sincurve">Diagram 1</a> shows the <em>sin</em> curve.
    Notice that <use target="func" /> is periodic with a period of
    <expression encoding="infix" id="p2">2 * PI</expression>
  </p>
```

Diagram 1 for a *sin* curve is generated dynamically with the `mathgraph` element:

```
  <diagram title="Diagram 1" id="sincurve">
    <mathgraph id="plot" URL="http://icm.kent.edu/plot.wme">
      <parameter name="operation">plot2d</parameter>
      <parameter name="function"><use name="func" /></parameter>
      <parameter name="variable"><use name="var" /></parameter>
      <parameter name="range">
        <math id="plot-range">
          <list>
            <cn type="real"><use name="from" /> </cn>
            <cn type="real"><use name="to" /> </cn>
          </list>
        </math>
      </parameter>
    </mathgraph>
  </diagram>
```

The `mathgraph` code sends the `operation`, `function`, `variable`, and `range` values to the given WME service `plot.wme` to obtain the plot. The element `use` is applied multiple times here. Later we can assign new values to the *use variables* and obtain different plots with this same `mathgraph` element.

Example 2: Interactive experimentation

The lesson continues with an interactive exercise that encourages student learning:

```
  <exercise title="curve" id="practice" type="interactive">
  <p>Perform your own experiments. Give an expression such as
    <em>sin(x+PI)</em>, <em>sin(x-PI/2)</em>, and so on and see
    the resulting curve:</p>
    <interaction target="plot">
      <p>Expression:
        <userinput type="expression"
                   name="func" encoding="infix" />
        <userinput type="symbol" name="var"  />
        <userinput type="real" name="from"  />
        <userinput type="real" name="to"  />
        <userinput type="submit" value="go" />
      </p>
    </interaction>
  </exercise>
</lesson>
```

The `interaction` element contains `userinput` elements to obtain user input of various types. The collected input become values for variables providing a context to execute the `target` computation element, `plot` in this case.

The Web page generated by examples 1 and 2 is shown in Figure 5.

5 WME Page Processor

The *WME Page Processor* is a critical part of the WME framework implementation.

When it takes the form of a client-side plug-in (as it is the case in our current design), it is known as the *MeML Plug-in*. The MeML Plug-in executes in the context of a regular Web browser, enabling it to display MeML pages (Figure 6), support user interaction, and transparently access WME services specified in MeML pages for dynamically computed content and user interactions.

The MeML Plug-in processes MeML pages within the the Web browser in the following manner.

- The Web browser receives the `text/x-meml` content from the Web server.

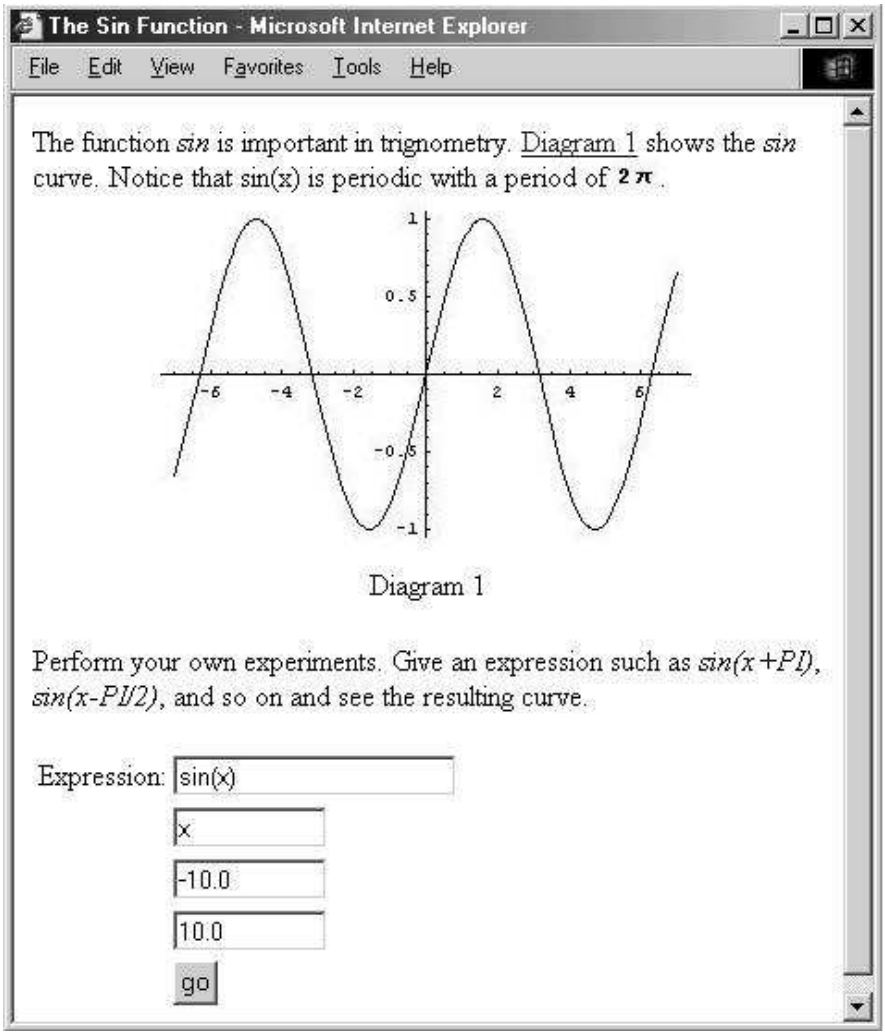

Figure 5: Teaching the Sine Function

- The XSLT[1] processor of the Web browser is used to parse and translate the MeML page into a regular XHTML page, often with embedded objects (via the XHTML `object` element). The resultant page can then be handled by the browser normally.
- The XSLT processor performs the required translation by accessing supplied MeML DTD and XSLT templates[2], and by invoking WME services through the MeML Plug-in.
- The MeML Plug-in supports Javascript access to local and remote WME services.

[1]XSLT [28] is a language for transforming XML documents into other XML documents.

[2]The MeML Plug-in automatically downloads the version of the MeML DTD and XSLT template file required by the MeML page if not already stored locally.

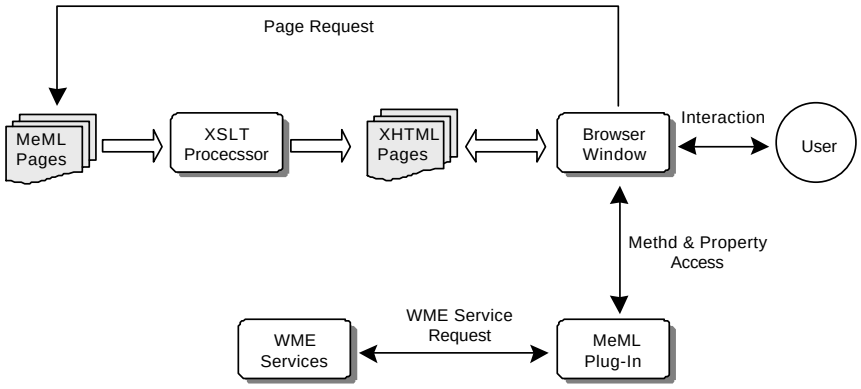

Figure 6: Accessing MeML pages

- The MeML Plug-in supports MeML-defined user interactions.
- The MeML Plug-in can interoperate with other browser plug-ins for increased power and flexibility.
- The MeML Plug-in can store and retrieve persistent data by accessing the the local file system as needed.

The prototype MeML Plug-in called *Woodpecker* is a client-side WME Page Processor we are developing as a research tool. It will help evolve the MeML language and establish a specification for other implementations of the MeML Plug-in.

Design of Woodpecker

Woodpecker is a prototype MeML Plug-in whose object-oriented design is shown in Figure 7. Woodpecker is implemented as an ActiveX control (`.OCX`) for Netscape Navigator (NN) and Internet Explorer (IE) on MS/Windows and as a NN plug-in on UNIX and other platforms.

As a piece of software, Woodpecker consists of a C++ program (the plug-in), a library of Javascript event handlers and other support programs, an *MeML XSLT template*, and an MeML DTD.

The Woodpecker design identifies major objects, defines relationships among the objects, the Web browser, the document (Web page), WME services, as well as the third-party WebEQ applet and MathPlayer plug-in.

Woodpecker *page-support objects* (pso) take the form of generated XHTML code (`form` for example) and embedded plug-in/ActiveX controls. Each support object, in the DOM spirit, exposes a set of well-defined methods and properties that constitutes its Javascript API. Page-support objects may generate specific events to interact with the Woodpecker Plug-in. Many page-support objects embed in XHTML via the `<object>` element. The

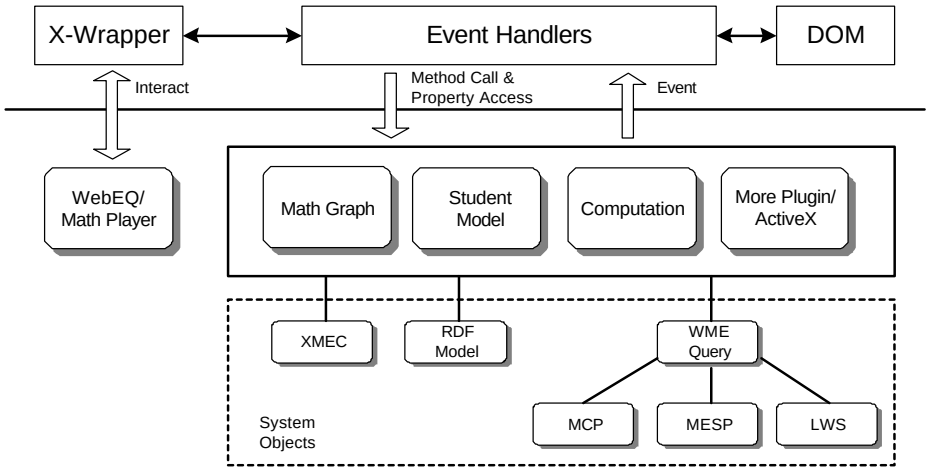

Figure 7: Woodpecker Design

embedding is defined in the MeML XSLT template and performed by the XSLT processor at page translation time.

The Woodpecker Javascript library supplies event handling and generation capabilities to page-support objects. The pso-to-library interface is currently ad hoc. A uniform specification of the pso-to-library interface can enable systematic development of pso's.

As the end-user interacts with the page through the browser, Woodpecker is activated through specific events to perform required tasks in support of the MeML-defined user interaction. Each event handler performs its preset actions, accessing methods/properties of page-support objects and modifying the page content through DOM.

The event handlers can apply changes to any target node on the DOM tree as long as the node or its location information is made available.

In addition to pso's, major Woodpecker modules include:

- The *WME service module*—to accesses local and remote WME services, as specified in the MeML page, and integrates results retrieved into the page.
- The MCP module—to implement the Mathematics Computation Protocol. It is essential for MeML Plug-In to interact with remote IAMC servers which can provide dynamic mathematics computing.
- The MESP module—to implement Mathematics Education Service Protocol. This protocol handles all educational service requests.
- The LWS module—to handle local WME service. It makes calling local services same as calling remote WME services.
- The WME Query module—to provide a uniform interface for pso's to call all types of WME services

- The XMEC module—to help convert among different mathematical data encodings. This module is an application of the *eXtensible Mathematics Encoding Converter* [38].

- The Record-keeping module—to record information such as user activities, progress, and correlations between student knowledge and education topics. The stored data files enable performance evaluation and diagnosis of learning difficulties.

The Woodpecker design also addresses service integration with other in-page objects and client-side plug-ins, such as the WebEQ applet, the MathPlayer plug-in, and other mathematics renderers/editors. Woodpecker aims to interface with such external programs in a similar way as it treats pso's. Because external programs vary in implementation method and API, we will use an *X-Wrapper*, written in Javascript, to hide the interface details and encapsulate their functions into a uniform interface.

Woodpecker Page Processing

Accessing an MeML page from an MeML-enabled browser results in the following sequence of events:

1. The MeML page is sent from the target Web server under the content type `text/x-MeML`.

2. The MeML-enabled browser calls XSLT to translate the page into a regular Web page. The MeML translation rules [37] are made available to the XSLT. The resulting page will typically contain generated pso's, and Javascript controls. It may also include dynamically computed content obtained from WME services.

3. The translated page is displayed in the browser window and ready for user interaction.

4. MeML specified user interactions are supported by event handlers triggered by the browser. The interaction may be supported by WME services. Results of user interactions can be displayed through the DOM interface.

The browser, not the MeML Plug-in, calls XSLT. This way, the MeML Plug-in can focus on support for MeML processing.

As a small example, the `diagram` element (with id `sincurve` in Section 4) is translated into the following XHTML code:

```
<div class="diagram" id="sincurve">
  <table>
    <caption style="caption-side: bottom"
             class="diagram"> Diagram 1 </caption>
      <tr><td>
        <object id="plot" classid="clsid:..." >
          <param name="operation" value="plot2d" />
          <param name="variable" value="gen_id_1" />
          <param name="function" value="gen_id_1" />
          <param name="range" value="gen_id_3" />
```

```
        </object>
      </td></tr></table>
</div>
```

Here, the `plot` object is a concrete pso example. The generated `id`s are internal variables usually with more complicated string names.

WME Page Processor Location

In the course of our research on the WME framework, we have explored two possibilities for deploying the WME Page Processor that provides support for MeML pages:

1. Placing the WME Page Processor on the server side.
2. Placing the WME Page Processor on the client side.

We considered the server-side approach, built a server module in Java, and performed experiments with an Apache server. The server module translates MeML pages into normal Web pages and returned the dynamically computed page back to standard Web clients. These efforts have been reported already [31].

Not quite satisfied with the server-side approach, we began to investigate the client-side approach as described earlier in this paper. Table 1 details the pros and cons of each approach.

In summary, both approaches to deploy the WME Page Processor are workable. But the server-side approach seems to create a bottle neck both in terms of speed and availability and in terms of persistent data storage. In comparison, the client-side approach is more personal, customizable, and better exploits the power of distributed computing. If WME becomes widely used, it is possible that some situations will call for the server-side approach which can co-exist with the MeML Plug-ins on the *Web for Mathematics Education.*

6 WME Services and MESP

The power and versatility of WME as a *Web for mathematics education* depend, in large part, on the number and quality of services available to educators and students through MeML pages.

There are two broad categories of WME services: *mathematics knowledge and computation services* and *mathematics education and pedagogy services.* A WME service receives a set of name-value parameters and returns a valid MeML/XHTML page fragment. Thus, WME service responses can be embedded in an MeML page directly.

Simple services can use CGI/HTTP and several such services are in operation at ICM/Kent including a fraction arithmetic service, an algebraic equation solving service, and a plot/graph service [12].

Other services may provide ready access to computation execution, definitions for mathematical notations, terminology concepts, theorems, rules, principles, formula, skills, patterns, and solution templates for typical problems.

A WME education service can supply dynamically generated education content such as examples, exercises, and tests. Optionally such a service may also provide diagnosis, assessment, and even automated coaching. Student performance and progress data can be recorded on the client side for better customization of the educational experience.

IAMC and MCP provide great enabling technologies for accessing a large variety of mathematical computations and for implementing WME services. But, to make WME services easy to access and widely usable we still need to address several issues including:

- A way for a WME server to provide a list of services available.
- A way to provide and retrieve a *service description* for any particular service.
- A way to distinguish between service types such as transaction-oriented and session-oriented services.
- A way to indicate the encoding for mathematical data.
- A formalized way to describe how to invoke a service.

We are investigating these issues and developing the *Mathematics Education Service Protocol* (MESP). MESP will be an application layer protocol on top of general-purpose communication protocols such as HTTP, MCP, SOAP, TCP, UDP, CORBA, and others. The purpose of MESP is to decouple MeML pages and WME services and make them interoperable.

WME services can also be useful at MeML page authoring time. An MeML authoring tool can access such services to obtain content materials for the author to view and select for inclusion as static material in the page being created. An authoring tool will also offer listings of available services that can be applied in the MeML page for dynamic contents and user interactions at page-display time.

7 MeML Authoring Tool

It is possible to construct MeML pages by writing MeML code directly. But that can be hard for most teachers. We are building an MeML authoring tool to make it much easier. The design of our prototype authoring tool, *MadMath*, seeks to provide ease, speed, and power to mathematics education content creators.

Components of *MadMath* include *Woodpecker* (as an ActiveX control or dynamic shared library), *MeML Parser*, *File Manager*, *MeML editor*, *Knowledge manager*, *Previewer*, *Configuration manager*, *View manager*, and *Project Manager*. Figure 8 shows its logical architecture.

Some of these components are described in a little more detail below.

- MeML Editor—The Editor is the GUI for MadMath.
- Page Processor—MadMath interfaces to Woodpecker to obtain results for previewing and to conveniently access WME services to supply page content at authoring time.

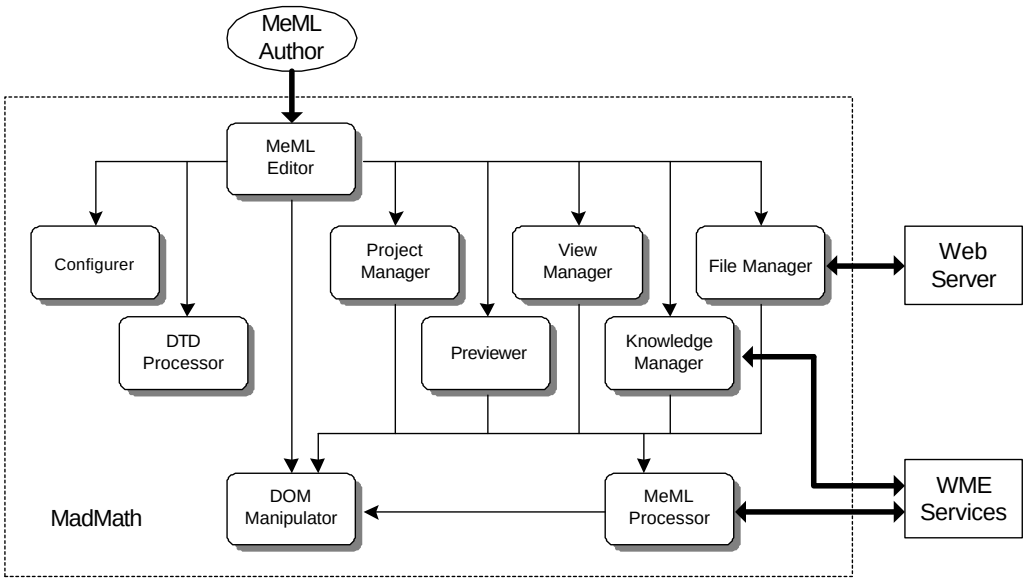

Figure 8: Design of MadMath

- MeML Parser—The parser checks incoming page content for compliance to the MeML DTD. It also builds the MeML tag names and attributes for visual editing functions such as drag-drop.
- File Manager—This is the MadMath interface to the local and remote file systems.
- Knowledge Base Manager—This component empowers the author. It helps locate information from MeML knowledge bases to be included in the page being constructed. Such information include terminology definitions, concepts, skills, rules, from WME services and MeML pages elsewhere.
- DOM Manipulator—The MeML page being edited is represented internally by a DOM tree and the Manipulator is responsible for accessing and modifying the DOM tree.
- Previewer—This allows the author to switch to the preview mode while constructing the page. A less capable authoring tool may leave this out. The author can always preview the page with an MeML-enabled Web browser.

8 Conclusions and Further Work

The WME framework designs a distributed system to enable Web-based mathematics education. WME empowers the teacher and eliminates many technical difficulties of on-Web mathematics education. WME is open because it uses standard Web/Internet technologies. WME is flexible because it allows educators to easily compose and create teaching materials and obtain automatically generated contents. WME is widely available because it is

served by standard Web servers and accessed by standard Web browsers with the MeML Plug-in. WME is powerful because it allows anyone to build WME servers that are immediately usable by educators within any MeML page. Finally, WME can elevate the quality of the mathematics education because students and teachers have access to servers and pages built by experts world-wide, and because mathematical ideas can be reinforced with realistic examples, interactive experiments, and instant visualizations. WME has the potential of becoming a valuable tool for teachers and students of mathematics and can be applied flexibly in many educational contexts.

So what does WME do that other approaches don't? The answer perhaps is that "WME aims to build a Web for Mathematics Education".

The whole WME architecture is designed so that mathematics education content (MeML Pages), computational and pedagogical services (WME services), can all be *built independently*, *maintained independently*, placed anywhere on the Internet, delivered to any browser, *and still interoperate and take advantage of one another*. Clearly, achieving the highest level of flexibility and scalability in the way mathematics education content can be created, deployed, and maintained, is a major concern of ours and an essential part of our research.

We have presented the design and architecture for WME. Much work is ahead before WME can be a successful reality. Refinements for the design, interface definitions, detailed specification of the MeML language and all its elements must be considered and tested. Emerging technologies such as SOAP[25], UDDI[26], and WSDL[35] may prove useful to i) locate and discover WME servers with desired capabilities and ii) interoperate with additional software available under the form of Web services [5]. Software prototypes must be developed further and made available for the research community and mathematics education experts for comments, experimentation, and critical evaluation.

Hopefully, the WME framework can become an enabling technology for practical mathematics education on the Web and will create new opportunities for educators, service providers, as well as further R&D.

References

[1] ActiveMath, `http://www.mathweb.org/activemath`.

[2] L. Becerra, O. Sirisaengtaksin, and B. Waller, "On Categories of Interactive Computational Web Tools," ATCM 2000, Proceedings of the Fifth Asian Technologies Conference in Mathematics, Chiang Mai, Thailand, December 2000.

[3] Calc101, `http://www.calc101.com`.

[4] O. Caprotti, D. P. Carlisle and A. M. Cohen. The OpenMath Standard 1.1 Draft, October 24 2002 `www.openmath.org/cocoon/openmath/standard/om11`.

[5] O. Caprotti and W. Schreiner, "Towards a Mathematical Services Description Language", ICMS 2002, Proceedings of the First International Congress Of Mathematical Software, pp. 442–452, Beijing, China, August 2002.

[6] Ezmath, http://www.w3.org/People/Raggett/EzMath.

[7] Workshop on *The Future of Mathematical Communication,* http://www.msri.org/-activities/events/9900/fmc99, Dec. 1999.

[8] S. Gray, N. Kajler, and P. S. Wang. Design and Implementation of MP, a Protocol for Efficient Exchange of Mathematical Expressions. *J. of Symbolic Computation*, 25(2):213–238, Feb. 1998.

[9] Hypermedia Browser techexplorer, http://www-3.ibm.com/software/network/techexplorer.

[10] http://icm.mcs.kent.edu/research/iamc (IAMC homepage), http://icm.mcs.kent.edu/research/iamcproject.html (IAMC project homepage).

[11] Proceedings of the IAMC 1999, 2001, and 2002 Workshops, http://icm.mcs.kent.-edu/research/iamc.html#iamcworkshop, July 1999, July 2001, and July 2002.

[12] Institute for Computational Mathematics, demos of mathematical computation http://icm.mcs.kent.edu/research/demo.html.

[13] JavaMath, http://javamath.sourceforge.net.

[14] S. Linton and A. Solomon, "GAP, OpenMath, and MCP", Proceedings of IAMC'99 Workshop, July 1999.

[15] LiveMath, http://www.livemath.com.

[16] Maple, http://www.maplesoft.com.

[17] Mathematical Markup Language, http://www.w3.org/Math.

[18] MathLink, J/link, http://www.wolfram.com/solutions/mathlink.

[19] MathML International Conference 2000, www.mathmlconference.org/2000, UIUC Illinois USA, Oct. 20-21, 2000.

[20] MathML International Conference 2002, www.mathmlconference.org/2002, Hickory Ridge Conference Center, Chicago IL. USA, June 28-30, 2002.

[21] MathPlayer and WebEQ, http://www.mathtype.com/webmath.

[22] MathScript, http://www.mathscript.com.

[23] MathWeb, , http://www.mathweb.org/mathweb.

[24] Mathwright, http://www.mathwright.com.

[25] SOAP (Simple Object Access Protocol), http://www.w3.org/TR/soap.

[26] UDDI (Universal Description, Discovery and Integration), http://www.uddi.org.

[27] W3C Amaya browser, http://www.w3.org/Amaya.

[28] W3C. XSL Transformations (XSLT), Version 1.0. See http://www.w3.org/TR/xslt, November 1999. W3C Recommendation.

[29] P. S. Wang, "Design and Protocol for Internet Accessible Mathematical Computation", Proceedings of ISSAC'99, ACM Press, pp. 291–298, 1999.

[30] P. Wang, S. Gray, N. Kajler, D. Lin, W. Liao, and X. Zou. "IAMC Architecture and Prototyping: A Progress Report," Proceedings of ISSAC 2001, pp. 337–344, July, 2001.

[31] P. S. Wang, N. Kajler, Y. Zhou, and X. Zou, "Initial Design of A Web-Based Mathematics Education Framework," Proceedings of IAMC'02 workshop, icm.mcs.kent.edu/reports/2002/ICM-200202-0001.pdf.

[32] A. Weber and W. Küchlin, "A Framework for Internet Accessible Software Components for Scientific Computing", Proceedings of IAMC'99 Workshop, July 1999, http://icm.mcs.kent.edu/research/iamc99proceedings.html.

[33] WebMathematica, http://www.wolfram.com/products/webmathematica.

[34] WIMS, http://wims.unice.fr/~wims.

[35] WSDL (Web Services Description Language), http://www.w3.org/TR/wsdl.

[36] W. Wu, "Experiments with Internet Accessible Mathematical Computation", Master's Thesis, Department of Mathematics and Computer Science, May 1998, ICM/Kent technical report ICM-199805-0003.

[37] X. Zou, Y. Zhou, and P. S. Wang, "MeML Page Translation Rules," ICM technical report ICM-200301-0003.

[38] X. Zou and P. S. Wang, "XMEC: An Extensible Mathematical Encoding Converter", http://icm.mcs.kent.edu/research/xmec.

	Client-side (Plug-in) WME Page Processor	***Server-side WME Page Processor***
Pros	△ No restriction on placement of MeML pages and no load on Web server, less Internet traffic. △ Handling one user at a time makes it easy to personalize and customize. △ Ability to use local file system for persistent educational data. △ Fast and efficient user interaction. △ Close integration with browser. △ Simple to cooperate with other client-side components such as MathML renderer and Mathematics editor. △ Simple to support user interactions with remote WME services. △ Easy integration/interaction with MeML authoring tool.	△ Standard browsers can be used directly with no plug-in to download or install. △ Transparent WME Page Processor upgrades.
Cons	▽ Users must download and install plug-in initially and for each new version. ▽ Porting for different browsers on different operating systems. ▽ Potential limitations on Internet access due to client-side settings and firewalls.	▽ Only Web servers with Page Processor can supply MeML pages. ▽ Increased server load creates scalability problems. ▽ Must keep track of individual users and sessions, making it difficult to personalize without extensive user account and database system support. ▽ Must have different implementations for different servers/OS. ▽ Difficult to support user interactions with remote WME services. ▽ Must maintain server-side database for persistent educational data. This can be hard and creates a large scalability problem. Even if a server installation will maintain such a database, its use with other MeML-enabled servers can be difficult.

Table 1: WME Page Processor Location Pros and Cons