Proceedings of the Second IASTED International Conference
EDUCATION AND TECHNOLOGY
July 17–19, 2006, Calgary, AB, Canada

MEML: SUPPORTING STRUCTURED, INTEROPERABLE AND DYNAMIC WEB-BASED MATHEMATICS EDUCATION

Xiao Zou Paul S. Wang
Computer Science Department
Kent State University
U.S.A
xzou@cs.kent.edu

ABSTRACT
The Mathematics Education Markup Language (MeML) is central to WME, a Web-based Mathematics Education system under development at Kent State University. With MeML, modules, lessons, manipulatives and tools become well structured, dynamic, easily interoperable, and customizable at different levels. A server-side MeML processor has been designed and implemented (in PHP) to dynamically translate MeML pages into regular Web pages for delivery to standard browsers. MeML and the server-side processor combine to support a WME model site designed to deliver classroom-ready modules and lessons to individual schools. Support is also provided to access Mathematics Education Services that can add power and functionality to lessons.

KEY WORDS
MeML, Customization, Interoperability, Web-based Mathematics Education,

1. Introduction and Background

At the Institute for Computational Mathematics, an interdisciplinary team of mathematicians, computer scientists, education researchers, Web developers, and middle school teachers is building a *Web-based Mathematics Education* (WME) system by an innovative combination of open Inernet technologies. WME can deliver, via the Internet or a LAN (wired or wireless), classroom-ready lessons that are well-prepared, interesting, effective, as well as interoperable. In addition to allowing multimedia content and hyperlinks, lesson pages feature in-page *manipulatives* to help students understand and explore mathematical concepts and skills through hands-on activities.

WME is different from existing approaches and aims to be a modern, practical, efficient and effective Web-based system to support mathematics education and learning [3]. The WME system conforms to open standards, works with regular browsers, delivers integrated and complete lessons, enables easy customization, provides systematic access to client-side and server-side support, and allows independently developed WME components to interoperate seamlessly. In short WME seeks to create a *Web for Mathematics Education*.

To deliver WME to schools, a WME model site is being built that contains education content (lessons and modules), teacher guides, and assessment tools. Lessons and modules are configurable and customizable on a per school, per class, and per teacher basis. Changing information is kept in a database which supports dynamic page generation.

Central to all of this is the Mathematics Education Markup Language (MeML) which provides structure to lessons and modules as well as making them interoperable. WME lessons are interactive and contain hands-on activities defined by manipulatives. These manipulatives are also customizable and can easily be deployed in different MeML pages.

Our approach with MeML has evolved since the publication of [18]. We now support MeML translation on the server side. The server-side MeML processor (implemented in PHP) becomes an important part of the WME model site and facilitates many of its important operations.

2. MeML Processing

As a newly defined XML language, MeML is not automatically accepted by regular Web browser. It need either client-side or server-side support. We have developed a client-side MeML processor -Woodpecker, which runs as an ActiveX control of Web browser [6, 7]. In this article, we are going to introduce a different approach - MeML Server Extension, which could run on Model site or any school site to translate MeML page into regular HTML page. Figure 2 shows the diagram on how the MeML page is delivered to client-side browser.

When a MeML page is loaded by MeML Server Extension, the Composer parses the page into an internal DOM tree. Then the Composer searches the customization table in local database to find out if there is

any updated content for every MeML tag. The search is based on compound index - (Course, Topic Module, Active Lesson, Tag, Tag ID). We know "Course" is associated to teacher and his/her class. "Topic Module" and "Active Lesson" can identify the unique MeML page source. So, updated content is guaranteed to be unique per class and teacher.

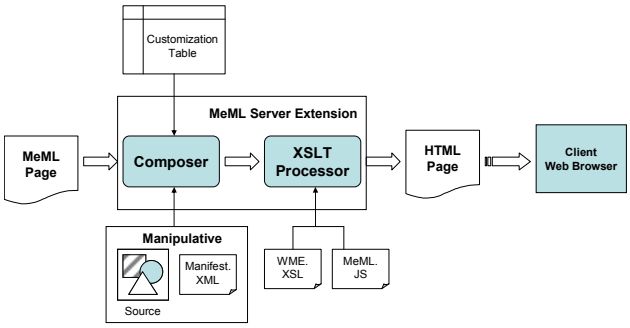


Figure 2: MeML Processing

For MeML tag "<manipulative>", the updated content is a new version of manipulative's configuration file – "manifest.xml". Composer will read the configuration and insert the manipulative source into DOM tree to finish deployment. For other tags, the content is a replacement to their sub tree in DOM structure. After this step, Composer will forward the updated DOM tree to XSLT [9] Processor, which will use XSL file "wme.xsl" to translate DOM tree into HTML page with supporting JavaScript code. A special JavaScript Library "meml.js" is also linked to the resulting HTML page to preparing client-side environment in user's web browser for the interaction between page components that are translated from MeML tags. Each MeML page may run in one of two different runtime modes – *Student Mode* or *Teacher Mode*. The XSL file "wme.xsl" is used to translate MeML page for Student Mode. If MeML page is going to run in Teacher Mode, MeML Server Extension will use another XSL file "wme_custom.xsl" to do the translation. The customization of MeML page by school teachers can only be finished in Teacher Mode. Once MeML Server Extension finishes the translation, the resulting HTML page will be delivered to client-side Web browser just likes the usual.

This MeML processing schema is different with our previous client-side approach "Woodpecker" [18] on three aspects:

- MeML Server Extension does not require any software installation on client machine. Woodpecker requires user to install a software package to work with Web browser.
- MeML page is processed at server-side in new approach. With Woodpecker, MeML page is processed at client-side when browser downloads it from Internet.
- Server-side processing could completely separate site organization and other site specific content from MeML page to improve dynamic support and interoperability. It is harder for client-side processing because client has less context information than server.

The server-side MeML processing is transparent to users. Teacher and students access the MeML page just like any other HTML page. At another hand, Content developers of MeML page will never need to concern about the technical detail hidden in the process. The only thing they need to do is authoring education pages in MeML and getting all kinds of dynamic support automatically.

3. WME Model Site

In order to study how the WME technologies can best be tailored and applied for actual mathematics education in schools and to demonstrate the features of all kinds of WME technologies, we build a WME Model Site for Kimpton Middle School (Munroe Falls, Ohio). But it is also ready for public access.

WME Model Site implements a variety of topic modules as well as lessons. The topics include *integers*, *percentages*, *proportions*, *length and area*, *number relations*, *fractions*, *probabilities*, and *understanding data*. Some of the topics are in the Ohio Academic Content Standards [11] which closely follows the NCTM standards [15]. We put these topic modules in class trial at Kimpton Middle School and receive many positive reactions.

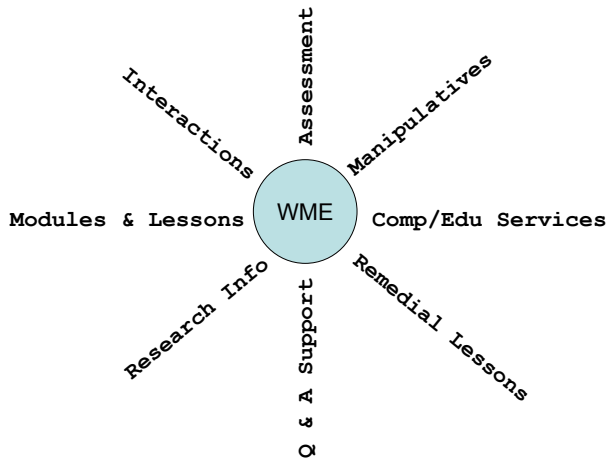


Figure 3: WME Model Site

The Model Site has necessary structures for site administration and course management, which are written in PHP and have MySQL as backend database server. But the topic modules and the education pages are all implemented by MeML. They have no direct relation with site administration. MeML content developer should be aware that MeML page should only contain education

materials and must not contain any site specific information like school logo, page navigation, account management, etc. This is the basic rule to exchange MeML pages among all kinds of school sites and it simplifies the creation of MeML page.

Figure 3 shows the crucial components and the features of WME technologies. Topic module is similar to a chapter in a book. It covers a topical area and includes an ordered sequence of interactive Active Lessons. Active Lesson is a self-contained unit that helps teaching a particular mathematical concept or skill, strengthen the student's understanding through hands-on activities, manipulation of tools, games, and answering questions. Manipulative [14] is a self-contained software package which runs in client browser. It provides hands-on exercise which is often designed as a game to help teaching a mathematical concept to students visually and interactively. A manipulative could be created in JavaScript, SVG, or Flash. An interactive assessment database [8] is provided to supply exercise and questions. Meanwhile, a group of computation server and education service are also built to provide constant backend support.

The central part of all of these components is MeML which implements and integrates them into a whole system. Topic modules and lesson pages are written in MeML. MeML dedicates special tags, like "<manipulative>", to embed all kinds of virtual manipulative into lessons cleanly and easily. It is also straightforward to use "<wmeservice>" tag to introduce various WME services into lessons and enable the interaction between page components and WME services.

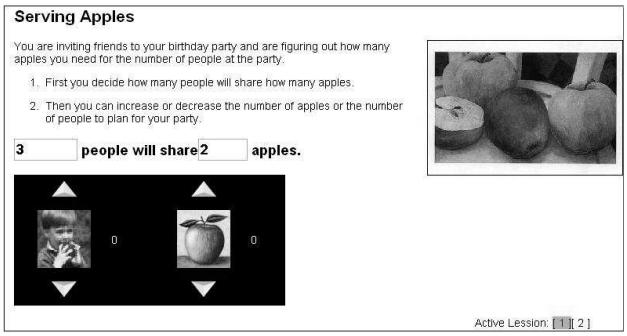
4. Structure of Sample MeML Page

MeML page is a composite of MeML, XHTML, MathML [10], and SVG. Table 1 shows a sample MeML page. The resulting page displays as shown in Figure 4.

Table 1: Sample MeML Page

```
<meml title="Serving Apples">
<reference type="css" src="[LPDIR]/css/headline.css" />
<reference type="manipulative" name="ExactPortion" />

<vsection id="instruction">
<p> <img id="apples" editable="yes" style="float:right; padding-
left: 2em; " src="[LPDIR]/img/apples.jpg" width="250"
alt="Apples"/>
<span id="intro" editable="yes">You are inviting friends to your
birthday party and are figuring out how many apples you need for
the number of people at the party.</span></p>
<ol><li>
First you decide how many people will share how many apples.
</li>
<li>Then you can increase or decrease the number of apples or
the number of people to plan for your party.
</li></ol>

<manipulative id="first" ref="ExactPortion" />
</vsection>

<vsection id="questions" display="no">

<exercise id="qset" title="C&amp;amp;I Questions" ordered="yes">
<question id="q1" type="word">
If there are 10 people, how many apples do you have to prepare?
</question>

<question id="q2" type="word">
If there are 7 apples, how many people can you serve?
</question>
</exercise>
</vsection>

</meml>
```

Every MeML page starts from root element "<meml>" and expand its hierarchical structure in the sub tree of root. As above code indicates, each MeML page is composed of one or more viewable section, (see "<vsection>" tag). Each viewable section could use a full page to display its content. The top image of Figure 4 displays the "instruction" section. The bottom image of Figure 4 displays the "questions" section and it is invisible initially. Student can only see one section at a time. Viewable section by default supports the switching between different sections. Because of this separated view, we call each viewable section as a Lesson Page (LP).



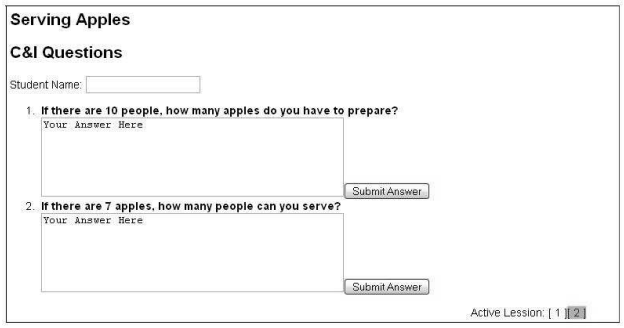


Figure 4: Sample MeML page

Content developer can develop their teaching materials in any order they want in viewable sections. In our example, we insert a Manipualtive in "instruction" section. Although the actual source of manipulative is very complicate, we only use two lines to make it work.

```
<reference type="manipulative" name=" ExactPortion " />
<manipulative id="first" ref=" ExactPortion " />
```

The Design of MeML puts a lot of effort like this to simplify the authoring work of MeML page while still getting powerful functionality and abundant dynamic support.

5. Interoperability Support

Interoperability is one of most important feature of Web-base Mathematics Education. MeML supports three types of interoperability:

- The interoperability between various page components which are translated from MeML tags;
- The interoperability between MeML and backend WME Service;
- The interoperability between MeML and manipulatives.

The interoperability between page components is enabled in two methods – by default or by events. The following example shows the interaction between system element "<userinput>" and several computation elements by default.

Table 3: Interactive Computation in MeML

```
<variable id="var1" type="real"> x </variable>
<compute id="comp1" >
   <expression id="exp1" encode="infix">
    x^2+2*x-1
   </expression>
</compute>
<userinput type="real" name="var1" />
```

In this example, we declare a variable "*x*" which is of type "real" and has a MeML id "var1". Then we define a compute block using "<compute>" tag, which is going to evaluate expression "$x^2 + 2x - 1$" once the value of "*x*' is given. After translation, the "<userinput>" tag will create a math expression input tool to get an assignment for "x". Once user assign a value to variable "*x*", the operation will immediately trigger the evaluation of expression "exp1" in compute block and display the result in page.

The interoperability between MeML tags and WME Services is also simple. Here we still take code in Table 3 as example. We know MeML Server Extension will not install client-side software package in user's machine locally. So any mathematics computation is not going to happen in user's browser. It has to be done by remote WME Compute Engine, especially for advanced symbolic computation. Just adding one line of code

```
<wmeservice id="Engine"  type="computation"
  url="http://squirrel.cs.kent.edu/WMEService/compute.php" />
```

will link WME Computation Service with the MeML page. The URL points to the location that provides computation service. The compute block will interact with computation service automatically.

Above code looks very simple, though the implementation work behind is much more complicate. Table 4 shows how we define the interface of computation service in WSDL [12].

Table 4: WME Computation Service

```
<portType name='WMEComputePortType'>
  <operation name='StartSession'>
     <input message='tns:NetworkID'/>
     <output message='tns:ComputeResponse'/>
  </operation>
  <operation name='Compute'>
    <input message='tns:ComputeRequest'/>
    <output message='tns:ComputeResponse'/>
  </operation>
  <operation name='EndSession'>
    <input message='tns:ComputeRequest'/>
    <output message='tns:ComputeResponse'/>
  </operation>
</portType>
<binding name='WMEComputeBinding'
           type='tns:WMEComputePortType'>
<soap:binding style='rpc'
          transport='http://schemas.xmlsoap.org/soap/http'/>
<operation name='StartSession'>
<soap:operation
   soapAction='urn:wme-compute-service#StartSession'/>
<input>
  <soap:body use='encoded'
    namespace='urn:wme-compute-service'
    encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
</input>
<output>
  <soap:body use='encoded'
    namespace='urn:wme-compute-service'
    encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
</output>
</operation>
<operation name='Compute'>
    <!-- Similar Definition  -->
</operation>
<operation name='EndSession'>
   <!-- Similar Definition  -->
</operation>
</binding>

<service name='WMEComputeService'>
<port name='WMEComputePort'
     binding='WMEComputeBinding'>
<soap:address
location='http://squirrel.cs.kent.edu/WMEService/compute.php'/>
</port>
</service>
```

The interface definition is complicate. However, it is just part of whole WME Computation Service and we have not yet shown the program code that accesses the interface. All of these technology details are hidden from end users with just one MeML tag and it happens transparently to users.

The interoperability between MeML tags and manipulatives is a different story. Manipulative by its

nature is actually an external source to MeML page. As a special software component, manipulative must follow the WME Client-side Manipulative Architecture, which defines an interface that each manipulative must support. The interface contains following instance methods.

- **reset()** – Re-initializes the manipulative instance.
- **getArg(***name***)** – Returns the value of the named parameter.
- **setArg(***name*, *value***)** – Sets the named parameter to the given value.
- **getProperty(***name***)** – Returns the named instance property.
- **setLocation(**dir**)** – Sets working directory of manipulative.
- **deploy()** – Generates HTML content for an instance of manipulative and inserts the content into current page through DOM.

With these instance methods, manipulative can be deployed in MeML page automatically. Figure 5 shows another example of manipulative "MealOrder", which belongs to topic module "percent" and active lesson "Eating Out".

```
<reference type="manipulative" name="MealOrder" />
<manipulative id="menu" ref="MealOrder" />
```

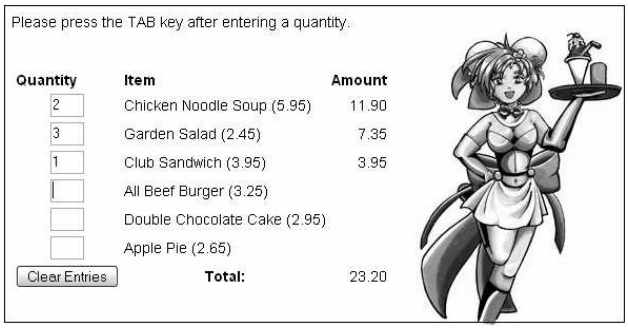


Figure 5: Manipulative Example

Except the exposed methods, each manipulative could also expose some internal attributes to other MeML content. For example, the manipulative above will expose an attribute "Total", which reflects the total cost of ordered meal in the menu. If other MeML tags, like an expression in a compute block, need to use this attribute as a variable to do some calculation, we can declare a new variable as below:

```
<variable id="var2" type="real" ref="wme.menu.Total">
 total </variable>
```

And then we can use variable name "total" in any expression for calculus. To understand how it happens, we should explain the meaning of variable reference "wme.menu.Total". The first part "wme" references to a global JavaScript object which is created by system JavaScript library "meml.js". This global object is the real master that controls the interaction between page components and the communication to WME Services. Here the MeML ID of the manipulative is "menu". The deployment of manipulative "Meal Order" will create a sub object with ID "menu" in "wme" global object. And "Total" is the exposed attribute of object "menu". In this way, global object will know how and when to update the value of variable "var2" to support the interaction we hope.

6. Customization Support

Education pages in MeML could be downloaded to anywhere on Internet and be exchanged among teachers. Therefore, the customization of MeML page is very important to suite math teachers' class requirement. The customizable content in MeML page includes Text, Image, Math Formula, Question, and manipulative. MeML Server Extension provides a special function to guarantee the customization is unique per class and teacher. As we always emphasize, only the page in Teacher Mode can be customized. When MeML page is accessed in Teacher Mode, the resulting HTML will have a group of tools, in-line forms, and pop-up menu to support in-page customization in addition to all features of Student Mode.

6.1 Customizing Text and Images

Text and Image are usually provided via HTML tags. To enable the customization to the HTML content, MeML provides a general tag attribute "editable". Using a piece of code from Table 1 (Sample MeML Page), author of MeML page could permit the customization to text as shown in Figure 6 through following code:

```
<span id="intro" editable="yes">You are inviting friends to your birthday party and are figuring out how many apples you need for the number of people at the party.</span>
```

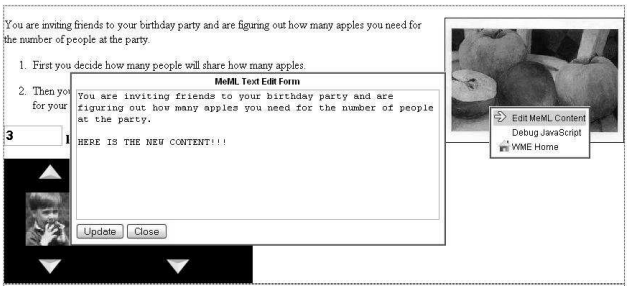


Figure 6: Customize MeML Page

Teacher who downloads the page and wants to do customization could access the page in Teacher Mode, right-click on the customizable content to get pop-up menu, and choose "Edit MeML Content" option. A pop-up menu and a Text Edit Form will display to customize

the text in first paragraph of MeML page – "Serving Apples", which belongs to Topic Module "Proportion". Using similar operation on other editable content, the right edit form for that specific type of content will show up. The customization of Image is supported in similar way and all implementation detail is hidden in the process. After teacher commits the customization, the changed content will be saved in the customization table of server database. Again, the storage is based on compound index - (Course, Topic Module, Active Lesson, Tag, Tag ID).

6.2 Customizing Mathematics Expressions

Any math formula given by MeML element "<expression>" is customizable or editable. Table 3 shows an example on how to use "<expression>" tag. The customization is supported by WME tool – MathEdit, which will be loaded into MeML page by MeML Server Extension during translation stage. Figure 7 shows the interface of MathEdit. MathEdit is the work of Kahraman Cem Karadeniz.

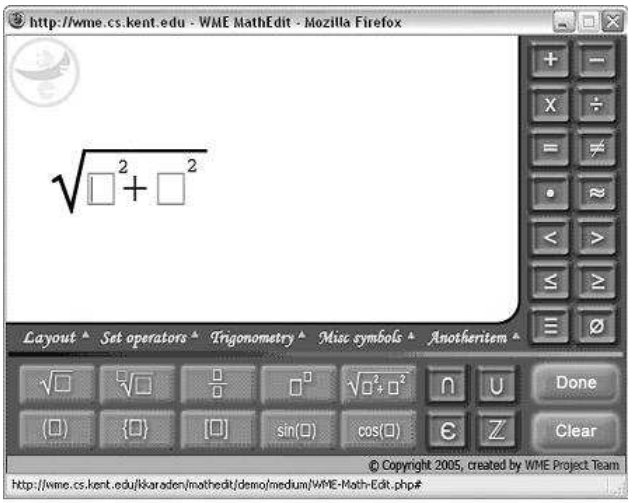


Figure 7: MathEdit Tool

6.3 Customizing Manipulatives

Customization of manipulative is different with other MeML components for manipulative is external source to MeML page. Each manipulative comes with a configuration file "manifest.xml", which supplies initializing parameters to manipulative. Its content can affect the display, computation, and outcome of manipulative. It is also the base to customize Manipualtive as math teacher wishes. Table 5 shows the configuration of manipulative "MealOrder", which is displayed in Figure 5.

Table 5: Configuration of Manipulative

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<manipulative name="MealOrder" src="MealOrder.js"
        memo="Caculate Percent of Meal Order" >
<parameter name="image" type="file" value="waitress.jpg"
        memo="Sample image file" />
<parameter name="menu" type="associate array"
        memo="Menu">
   <name>Chicken Noodle Soup</name>
   <value>1.95</value>
   <name>Garden Salad</name>
   <value>2.45</value>
   <name>Club Sandwich</name>
   <value>3.95</value>
   <name>All Beef Burger</name>
   <value>3.25</value>
   <name>Double Chocolate Cake</name>
   <value>2.95</value>
   <name>Apple Pie</name><value>2.65</value>
</parameter>
</manipulative>
```

If teacher chose to customize the manipulative, a Manipulative Edit Form will be given. The form will show the configuration in "manifest.xml" and let teacher change it. The new configuration will be saved in database and it is not permitted to overwrite original "manifext.xml". Figure 8 shows a sample Maipualtive Edit Form about how to customize this manipulative. There are two instances of manipulative "MealOrder" in one MeML page. The Manipulative Edit Form shows the configuration of first instance.

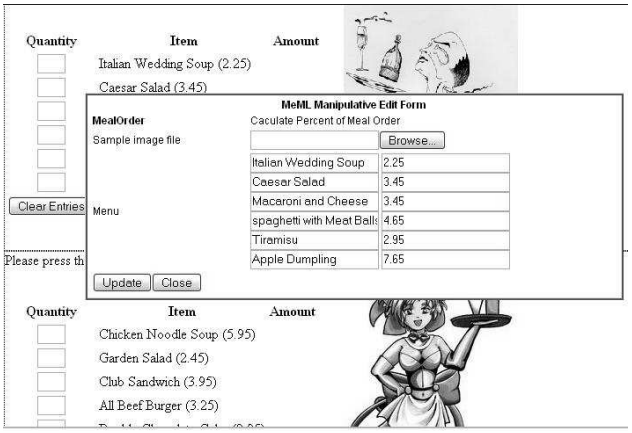


Figure 8: Customizing Manipulative

Content developer can create as many instances of one manipulative as they want in single MeML page. Each instance will behave in its own way without interference with others. The customization to each instance is also safely separated.

7. MESP Support

We have not explained at protocol level about how the information exchange happens between MeML page at client-side and server-side WME Service. Here we introduce Mathematics Education Service Protocol (MESP). Unlike general purpose protocol like HTTP [20], SOAP [12], and TCP/IP, MESP is an education-oriented, event-driven protocol that serves information exchange between MeML page and WME Service, and between

WME Services themselves. In real implementation, MESP can be bound to HTTP or SOAP to provide protocol services. The code example in this paper is using SOAP to complete MESP functions.

The significant difference between MESP and other general purpose protocols is that MESP defines a Common Education Model (CEM). CEM defines a group of education-oriented objects like Student, Teacher, Course, Compute Engine, Account Certificate, Assessment Database, Terminology Dictionary, etc. as well as the Events and relations among the objects. Each CEM object could be mapped to a MeML page component or a WME Service. When a MeML page is loaded by MeML processor, a virtual interaction scenario of CEM objects is created according to the content of MeML page. Then the scenario will be delivered to WME Client-side Environment to guide all kinds of in-page interaction after MeML page is loaded by user's browser. So, CEM is actually an abstraction of all kinds of resource that could play a role in education progress. And a MeML page actually contains the information of how these resources cooperate with each other to finish the educating task. MeML processor is responsible for finding a bind for each CEM object appearing in MeML page. WME Client-side Environment is responsible for the final bind operation and it will control the interaction according to the scenario. Figure 9 shows the relationship among CEM, MeML page, MESP Agent, and WME Services.

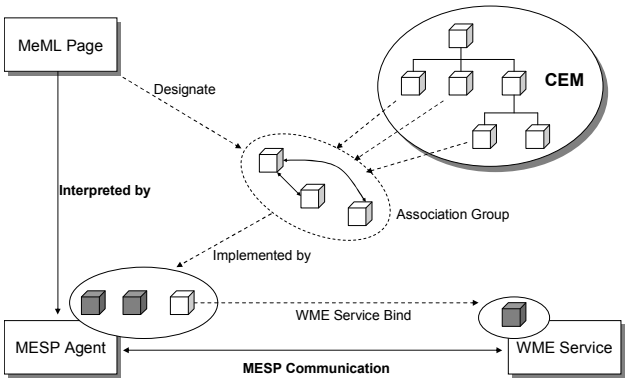


Figure 9: MESP and CEM

There are two types of MESP Agent considering different approaches of MeML processing schema. The client-side approach uses Woodpecker as MESP Agent to maintain WME Client-side Environment. The server-side approach depends on system JavaScript library "meml.js".

8. MeML Server Extension; Conclusion

MeML Server Extension is a server-side MeML processor written in PHP, which could be installed on WME Model Site or school site. It provides a set of function calls to site administrator and developers to handle MeML pages. The usage of MeML Server Extension is shown in Figure 10.

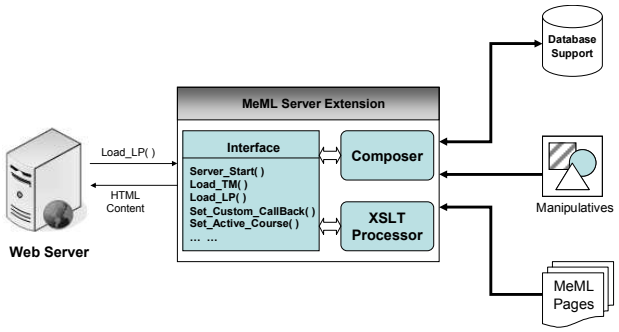


Figure 10: MeML Server Extension

School site developers may create their site content and decide site structure in any way they want. When they want to deliver education pages in MeML to client, they could simply call the load-page function to translate MeML source into HTML content. The resulting HTML content could be inserted into anywhere and merged with school pages smoothly. To enable MeML Server Extension, site administrator just need set the configuration about the installation directories of Topic Module and manipulatives. The configuration could also be set at runtime by calling function of Extension's Interface.

With the installation of MeML Server Extension, the in-page customization of MeML page is turned on automatically. Since customization requires database support to save personalized info collected from teachers, MeML Server Extension will install a built-in customization table to local database. There is no extra work for school site administrators. But, if they want to manage the customized content directly, they could also define their own database table. The only thing they need do is supplying three call-back functions to MeML Server Extension Interface.

Acknowledgement

Work reported herein has been supported in part by the National Science Foundation under Grant CCR-0201772 and in part by an Ohio Board of Regents Research Challenge Grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any funding agency.

References

[1] Michael Mikusa, Paul S. Wang, David Chiu, Xun Lai, Xiao Zou, "Web-based Mathematics Education Pilot Project", *Proceedings of the 2004 Conference on Information Technology in Education (ITE'04)*, Elizabethtown, PA; September 18th, 2004

[2] Paul S. Wang, Yi Zhou and Xiao Zou, "Web-based Mathematics Education: MeML Design and Implementation", *Proceedings of IEEE/ITCC'2004*, Las Vegas, Nevada, April 5-7 2004, pp. 169-175.

[3] P. Wang, M. Mikusa, S. Al-shomrani, D. Chiu, X. Lai, X. Zou, "Features and Advantages of WME: a Web-based Mathematics Education System", IEEE Southeast Conference, April 8-10, 2005

[4] P. S. Wang, “Design and Protocol for Internet Accessible Mathematical Computation”, *Proceedings of ISSAC'99*, ACM Press, pp. 291--298, 1999.

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

[6] Paul S. Wang, Norbert Kajler, Yi Zhou, and Xiao Zou. “Initial Design of A Web-Based Mathematics Education Framework”, Internet Accessible Mathematical Computation 2002 Workshop, Lille, France.

[7] Paul S. Wang, Norbert Kajler, Yi Zhou, and Xiao Zou. “WME: Towards a Web for Mathematics Education”, *Proceedings of ISSAC 2003*, Philadelphia, Pennsyvania, August, 2003, pp. 258-265.

[8] S. Al-shomrani, P. S. Wang. *Building DMAD: A Distributed Mathematics Assessment Database for WME*, Proceedings of the IEEE Southeast Conference, IEEE, Ft. Lauderdale, FL, April 8-10 2005, pp. 630-635.

[9] Chris V. S, Nitin Keskar, *XSLT Developer’s Guide* (McGraw-Hill/Osborne, 2002)

[10] Pavi Sandhu, *The MathML Handbook* (Charles River Media, 2002).

[11] Center for Curriculum and Assessment. *Academic Content Standards: K-12 Mathematics*, Ohio Department of Education, Feb. 2002.

[12] Eric Newcomer, *Understanding Web Services: XML, WSDL, SOAP, and UDDI* (Addison-Wesley Professional, 2002)

[13] The Advanced Distributed Learning Initiative, Office of the Secretary of Defense. “Sharable Content Object Reference Model”. www.adlnet.org

[14] National Library of Virtual Manipulatives for Interactive Mathematics, matti.usu.edu/nlvm/nav/.

[15] National Council of Teachers of Mathematics. *Principles and Standards for School Mathematics*, www.nctm.org/standards, Reston, Va., 2000.

[16] Simonson, M., Smaldino, S, Albright, M., and Zvacek, S. “Teaching and learning at a distance: Foundations of distance education” (2nd ed.). (2003) Upper Saddle River, NJ: Merrill.

[17] Chris Auld, Paul Spencer, etc. *Practical XML for the Web* (Glasshaus, 2002).

[18] Xiao Zou. "Support for Online Mathematics Education: MeML and WME Services", Proceedings, IEEE SoutheastCon, Fort Lauderdale, Florida, April 2005

[19] Powell, T. A. *HTML & XHTML: the complete reference* (Emeryville, Calif., McGraw-Hill/Osbborne, 2003).

[20] David Gourley, Brian Totty, *HTTP: The Definitive Guide* (O'Reilly Media, Inc. 2002).