Web-based Mathematics Education: MeML Design and Implementation

Paul S. Wang, Yi Zhou, Xiao Zou

***Abstract*—The *Web-based Mathematics Education framework* (WME) aims to create a Web for mathematics education. WME empowers mathematics teachers, learning content developers, as well as dynamic mathematics computation and education service providers, to deliver an unprecedented mathematics learning environment to students and educators. Main WME components include the *Mathematics Education Markup Language* (MeML), the MeML processor (Woodpecker, browser plug-in), and on-Web Mathematics Education Services. MeML provides effective and expressive markup elements to represent and structure mathematics education pages that may also contain XHTML and MathML elements. Woodpecker enables regular Web browsers to process MeML pages and to interact with a wide variety of mathematics computation and education services deployed on the Web.**

***Index Terms*—Web-based, Mathematics, Education, Markup, XML, Web, Services**

I. Introduction

THE past few years have seen much increased research activity in making mathematics computation accessible on the Web and Internet. With MathML presentation encoding [7] and pending support by popular Web browsers, *viewing of static Mathematical content* on the Web is no longer difficult.

Internet users, especially those learning mathematics, can also benefit from *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* [9], is a format that uses a binary encoded annotated parse tree for efficiency. OpenMath [2] 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 recent *IAMC Workshops* [19]. At the Institute for Computational Mathematics (ICM/Kent), efforts have been made to build a *distributed IAMC framework* [13], [14] 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 [19], the IAMC homepage[1], and the Workshop on *The Future of Mathematical Communication* [24].

At ICM we see Web-based mathematics education as a major application of IAMC that has enormous potential. Ad hoc mathematics education facilities offered by such websites as the National Library of Virtual Manipulatives [22] and WIMS [25], work individually but do not interoperate. Our group at ICM, together with collaborators, has been working on the design and architecture of a *Web-based Mathematics Education* framework (WME) [16]. WME provides an authoring language (MeML), works with regular browsers, makes page creation 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.

Institute for Computational Mathematics, Kent State University, Kent OH, 44242, USA, email: icm@mcs.kent.edu

[1] `icm.mcs.kent.edu/research/iamc.html`

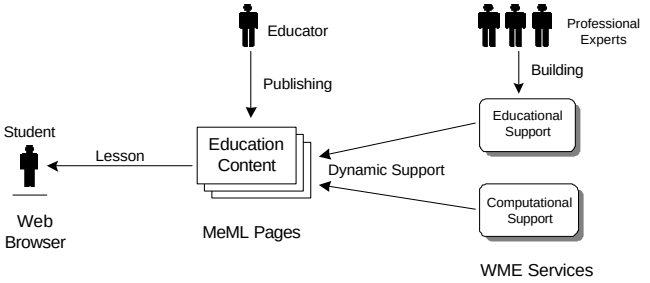

Fig. 1. WME Concept

The Mathematics Education Markup Language (MeML) is the centerpiece of WME. MeML aims to provide an effective and expressive means for authoring and delivering mathematics education content on the Web (Figure 1). MeML is at the core of the WME framework (Figure 2) [16] under investigation at ICM. MeML works with other components in the WME framework to deliver dynamic and interoperable mathematics education content and diverse online services to students on the Web. The requirements for MeML can be listed as follows.

- Delivering static content, dynamically generated content, and live interactions for learning and experimenting with mathematics on the Web
- Providing good support for structuring and organizing mathematics education content
- Being compatible with XHTML and MathML for intermixed usage
- Allowing marked-up pages to be delivered to regular Web browsers for end-user viewing and interaction
- Enabling access and interaction with independently developed and deployed mathematics computation and mathematics education services anywhere on the Web
- Supporting easy cross-referencing and exchange of marked-up content
- Being easy to learn and use, and supplying expressive power for authoring mathematics education materials

- Being flexible and easily extensible for adding features and future growth

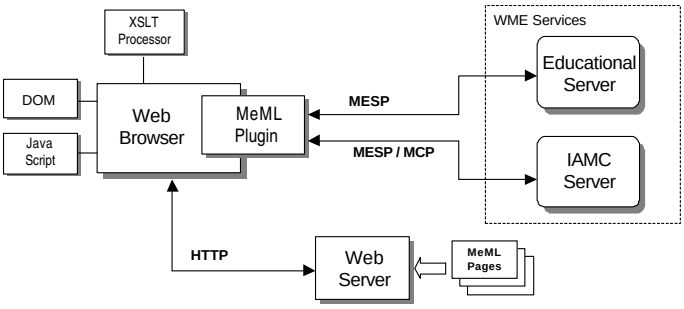

Fig. 2. WME Architecture

Thus, MeML is designed to facilitate authoring, delivering, storing, processing, and exchanging mathematics education materials over the Web, including live interactions of students with on-Web computational and educational services.

We design MeML as an XML-based markup language[2] that contains enough domain knowledge of mathematics computation and education to be efficient and effective. MeML elements are designed to structure teaching materials as well as to specify learning and educational activities such as providing motivation, demonstrating practical application, exploration of ideas, doing exercises, taking tests, and evaluating results.

Additional MeML elements support accessing on-Web services and live interactions of the student with such services. A variety of educational services can be made available such as application demos, example (counter example) generation, answer verification, intermediate steps production, illustration generation, word problem and test generation, test grading, and student performance evaluation.

II. Related Work

There are many XML-based languages [4] defined since the XML recommendation was released by W3C in 1998. Those that may be considered related to MeML include: TEI-Lite [1], DocBook [12], ISO 12083 [8], LMML [6], MathML, OpenMath [5], OMDoc [10], MathBook [3] and SCORM [21].

TEI-Lite is used to define the online books for SGML documents. DocBook and ISO 12083, like TEI-Lite, are also DTDs (Document Type Definition) for structured books. They provide comprehensive ontology and structures for online books. LMML provides DTDs for Web-based computer-science-related learning documents. MathML provides DTDs for content and presentation encoding of mathematical expressions. OpenMath is another DTD representing mathematical objects and a set of symbols (content dictionary). OMDoc is an extension of OpenMath to provide a DTD for mathematical documents. MathBook is intended to be a DTD simplified from DocBook and OMDoc for Web-based math learning. Although none of these XML-based languages come anywhere close to satisfying the MeML requirements, they do provide good references and models for the MeML work.

The MathChat effort by a different member of our group complements WME and provides a classroom setting for live math-oriented discussions while learning mathematics.

III. Overview of MeML Elements

The MeML DTD [18] defines the syntax of MeML elements. By examining current school textbooks, existing computer-aided mathematics education software, NCTM[3] Principles and Standards, online materials for learning mathematics, and by consulting experts on mathematics education, we can carry out the design of MeML in a spiral approach.

Currently, we have about 46 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>`, `<variable>`, `<identity>`, `<equation>`, `<diagram>`, and `<theorem>`.
- Education elements—for objectives in education. Tags include `<lesson>`, `<example>`, `<exercise>`, `<homework>`, `<test>`, `<assessment>`, `<diagnosis>`, and `<remediation>`. Contents for these elements can be supplied explicitly or can be generated.
- Organization elements—for page content organization. These include `<abstract>`, `<syllabus>`, `<roster>`, `<guide>`, `<hint>`, and `<summary>`.
- Computation elements—for explicitly invoking WME services to generate dynamic content. Such tags include `<computation>` and `<mathgraph>`.
- Interaction elements—for receiving and processing user input. These mainly include the `<interaction>` and `<userinput>` elements.

IV. A Complete MeML Page

To get a feel of what an MeML page looks like, let us give an example that shows a complete MeML page. The example is an educational page that teaches double angle identities in trigonometry.

The page is constructed to show MeML constructs rather than how to teach mathematics. The lesson starts with several identities (lines A-B), an explanation for them (lines C-D), and an example with step-by-step solutions (lines E-F).

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl"
href="meml.xsl"?><!DOCTYPE meml
 SYSTEM "http://icm.mcs.kent.edu/MeML.dtd">
<meml title="Trigonometry"><lesson>
 <lesson.content title=
     "Trigonometry:Double-Angle Identities">
 <ul id="dai"><li>
    <identity><expression encoding="infix">     (A)
     sin(2*\theta) = 2*sin(\theta)*cos(\theta)
     </expression></identity></li>
  <li><identity><expression>
     cos(2*\theta)=cos(\theta)^2-sin(\theta)^2
```

[2] See `http://icm.mcs.kent.edu/research/wme.html` for related work.

[3] the National Council of Teachers of Mathematics, USA

```
    </expression></identity></li>
   . . .
  </ul>                                          (B)
```

/clearpage

```
<explanation>                                    (C)
<p>An identity is an equality that holds for all
  possible (allowed) values of the variable.
  In the above identifies the variable is
  <variable>\theta</variable> and it can take on
  all real values.  ...</p><p>By substituting
  <expression>\theta/2</expression> for
  <expression>\theta</expression> on both sides
  of an identity you can derive
  <em>half-angle identities</em> ... </p>
</explanation>                                   (D)
<example title="Using double-angle identities"> (E)
<problem><p>Find <expression>sin(2*\pi/3)
</expression> by using <a href="#dai">
double-angle identities</a>.</p></problem>
<solution><step><expression>
     sin(2*\pi/3)=2*sin(\pi/3)*cos(\pi/3)
  </expression></step>
  <step><expression>= 2*(squareroot(3)/2)*(1/2)
      </expression></step>
   <step><expression> = squareroot(3)/2
   </expression></step>
   </solution>                                   (F)
</example></lesson.content>
```

Continuing with the end of the lesson, you can find two interactive exercises (lines G-H) and a homework with a due date (lines I-J). The exercise can be done online and the student input is sent to a designated WME service, `trig.mesp`, for verification. The interactive `<computation>` element, upon submission, sends the given parameters, including user input, to the URL-designated WME service and obtains a response which is displayed. Normally parameters in `<computation>` are not displayed to the user, unless specifically requested with the `display="ture"` attribute.

The student submits the homework to the teacher by email. Figure 3 shows the browser display of this MeML page.

```
<lesson.ender>
<exercise type="interactive" id="ciseTrig">
<ol>
 <li>Compute <computation type="interactive"
   id="exer1" wmeurl="mesp:http:
    //icm.kent.edu/mesp/trig.mesp">          (G)
  <parameter name="expr" display="true">
   <expression>sin(4*\pi/3)</expression>
  </parameter> and fill in your answer
  <parameter name="ans"><userinput type=
   "expression" name="func" encoding=
   "infix" /></parameter>
  <parameter name="command" display="button"
   value="Verify">verify
  </parameter></computation></li>
 <li>Compute <computation type="interactive"
    id="exer2" wmeurl="mesp:http:
        //icm.kent.edu/mesp/trig.mesp">      (H)
  <parameter name="expr" display="true">
     <expression>cos(\pi/4)</expression>
  </parameter> and fill in your answer
  <parameter name="ans"><userinput type=
   "expression" name="func" encoding=
   "infix" /></parameter>
  <parameter name="command" display="button"
   value="Verify">verify
  </parameter></computation></li>
</ol></exercise>
<homework emailto="teacher@school.edu"
  due="08/08/2003 10:00AM"                       (I)
  hwid="Trig identities 1">
  <ol><li id="assignment">Simplify the
  expression <expression>2*sin(\pi/5 -
\pi/2)^2 - 1</expression></li></ol>              (J)
</homework></lesson.ender></lesson></meml>
```

In this example, the very first line is the XML processing instruction specifying the XML version (1.0), and the character encoding (UTF-8). The stylesheet tag indicates the XSLT to use for processing this page. The `DOCTYPE` element declares the MeML DTD to which this MeML page conforms. The `meml` is the root element. Here, it contains one child element *lesson* which encloses the *lesson.content* and the *lesson.ender* elements.

The *lesson.content* has a title *Trigonometry: Double Angle Identities*. Five identities are listed using regular xhtml elements `ul` and `li`. Allowing mixed use of XHTML and MathML elements simplifies MeML while supplying the proven powers of XHTML and MathML to the author. In general we allow HTML elements to contain MeML elements and MeML elements to contain XHTML and MathML elements. Nonsensical mixing of elements is prevented by the usual inline vs. block element rules and by specific restrictions for each MeML element.

Each trig identity is given by an infix `<expression>`. The notations `\theta` and `\pi` specify Greek symbols. Alternatively, expressions can be given in MathML.

V. MeML Page Processing Overview

An MeML page may contain MeML, XHTML, and MathML elements. An MeML page is made acceptable to a Web browser by translating MeML into XHTML and MathML. The Mathematical expressions are translated into MathML and displayed either by MathPlayer [20] or directly by the MathML-enabled Web browser.

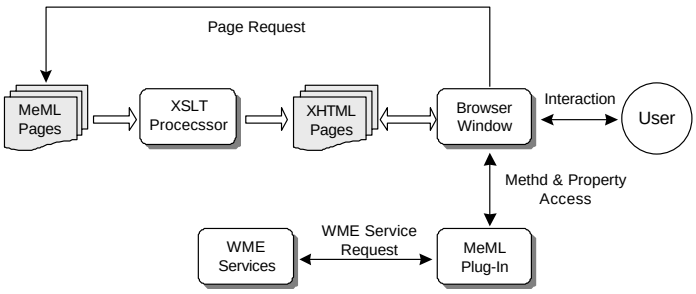

Fig. 4. MeML page Access

The translation can be performed by a *WME Page Processor* on the server side or on the client side [16]. Currently, we think the client side page processor solution has more advantages. Figure 4 shows the processing steps for an MeML page. Our prototype client-side WME Page Processor, Woodpecker, has

1. an XSLT processor and XSLT templates and
2. a collection of Web browser plug-ins called MeML plug-ins.

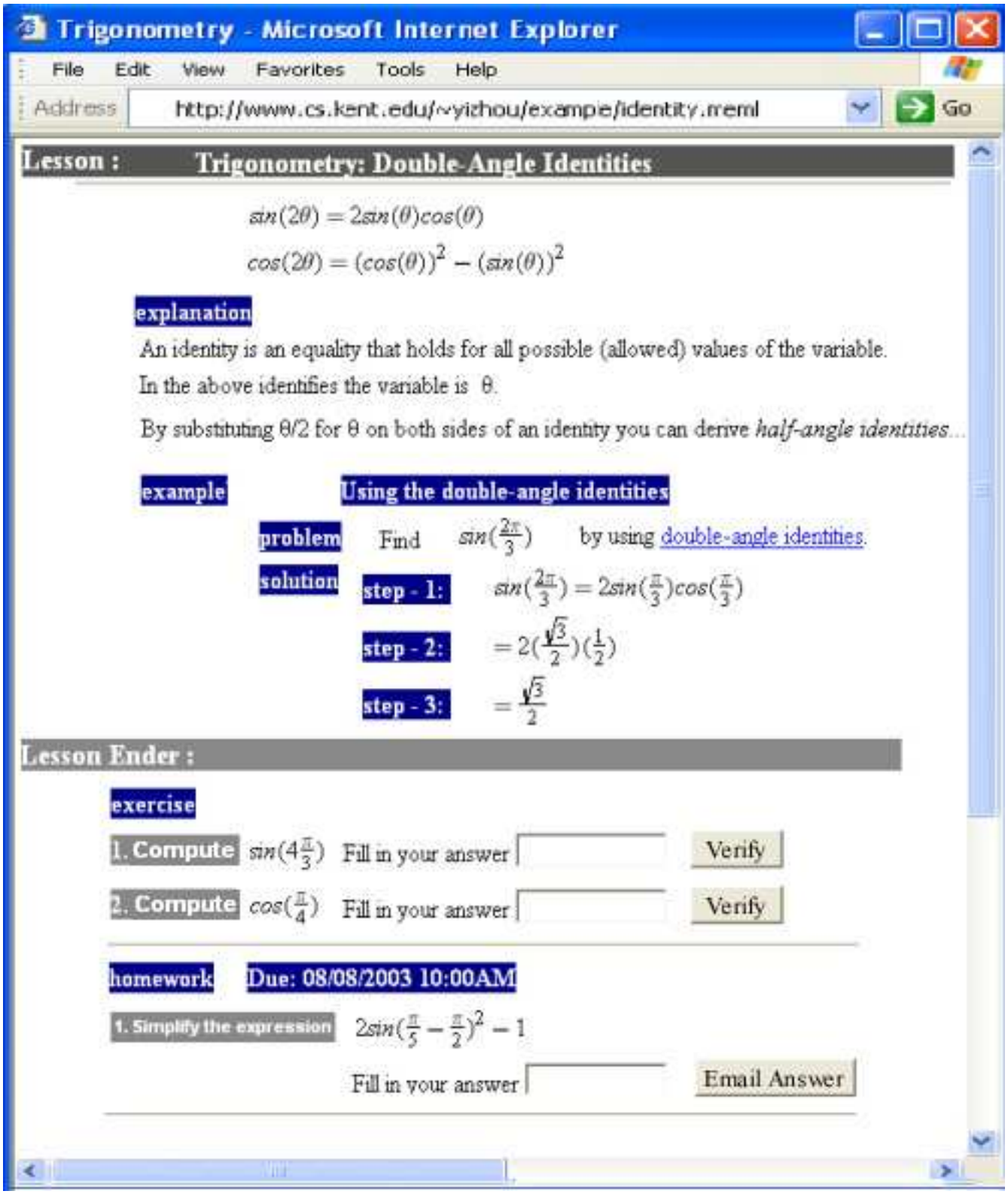

Fig. 3. Identities in Trigonometry Page

The translation is done by an XSLT processor. The translated XHTML page may also have embedded *event handlers* (in JavaScript) and references to MeML plug-ins.

When a Web browser loads the translated page, references to MeML plug-ins will be instantiated as in-page objects. Depending on the type of browser, these object instances may take the form of ActiveX Controls (in Internet Explorer) or Plug-ins (in Netscape Navigator). Some important MeML plug-ins are *MathComputation*, *MathGraph*, and *MathExpression*. When a user interacts with the page, a series of in-page events may

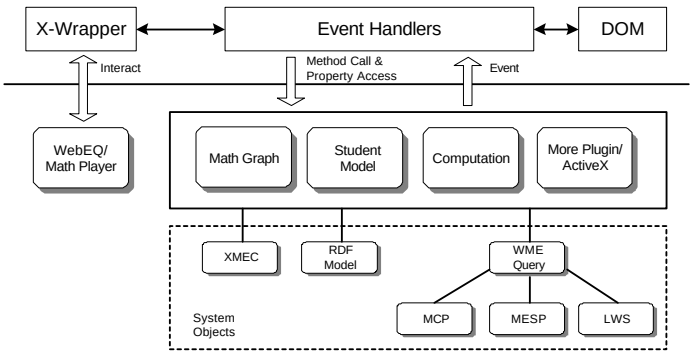

Fig. 5. Woodpecker Design

be generated from input-control elements (such as XHTML `<input>` elements) and MeML plug-ins (visible or hidden). The JavaScript event handlers treat these events by interfacing to appropriate MeML plug-ins to support the functionality of specific MeML elements. Figure 5 shows the architecture of Woodpecker.

VI. MeML Element Processing Example

Let's illustrate MeML page processing by following an example. We'll begin by looking at the `<computation>` element in detail.

The `<computation>` tag provides a generic way to invoke any WME service, identified by a URL (line 1), for any particular *command* (line 2) supported by that service. WME services follow the MESP (Mathematics Education Service Protocol) that may sit on top of widely used communication protocols such as HTTP, TCP, MCP [14], and SOAP [23]. Arguments required by the particular command, `expand` in this case, are again given by `<parameter>` elements (line 3).

```
<computation id="Compute1" wmeurl=
   "mesp:mcp://symbolicnet.org/polynomial.mesp"> (1)
  <parameter name="command">expand</parameter>
  <parameter name="polynomial">                  (2)
  <expression id="exp1" encoding="infix">
  (x+1)*(x-1) </expression></parameter>          (3)
</computation>
```

If the expression `exp1` has been defined earlier in the MeML page, it does not need to be repeated and the notation `<use name="exp1" />` can be given instead.

The following is part of the previous XSLT code for the element `<computation>`:

```
<xsl:template match="computation">
 <object>
  <xsl:attribute name="id">
   <xsl:value-of select="@id" />
  </xsl:attribute>
  <xsl:attribute name="classid"><xsl:text>
   clsid:DD406EB1-8F51-46A9-AC5C
    -AA0C5214F60E</xsl:text>
  </xsl:attribute>
    <xsl:attribute name="onclick">
    <xsl:text>DetectEvent(</xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>)</xsl:text>
  </xsl:attribute>
  <param name="wmeurl">
   <xsl:attribute name="value">
     <xsl:value-of select="@wmeurl" />
   </xsl:attribute>
  </param>
  <param name="command"><xsl:attribute
    name="value"><xsl:value-of
    select="parameter[@name='command']" />
   </xsl:attribute>
  </param>
  <xsl:for-each select=
      "parameter[not(@name='command')]">
   <param><xsl:attribute name="name">
     <xsl:value-of select="@name" />
     </xsl:attribute>
     <xsl:attribute name="value">
      <xsl:text>ID{</xsl:text>
      <xsl:if test="self::node()[use]">
      <xsl:value-of select=
      "node()/@name" /></xsl:if>
      <xsl:if test=
          "self::node()[expression]">
      <xsl:value-of select="node()/@id" />
      </xsl:if> <xsl:text>}</xsl:text>
     </xsl:attribute>
   </param>
   </xsl:for-each>
  </object>
  <xsl:apply-templates
   select="parameter/expression" />
</xsl:template>
```

After the first processing step of WME Page Processor, the `<computation>` tag will be translated into the following result.

```
<object id="Compute1" classid="clsid:
    DD406EB1-8F51-46A9-AC5C-AA0C5214F60E"
    onclick="DetectEvent(Compute1)">
  <param name="wmeurl" value= "mesp:mcp://
        symbolicnet.org/polynomial.wme" />
  <param name="command" value="expand" />
  <param name="polynomial"
        value="ID{exp1}" />                (4)
</object>
```

The polynomial `(x+1)*(x-1)` is transmitted to the *MathComputation* plug-in (line 4) via the element `id` through DOM.

Thus, the `<computation>` tag is mapped to an instance of MeML *MathComputation* plug-in which is a hidden object. The `classid` points to the plug-in's registration ID. A set of JavaScript event handlers placed in the `<head>` element of the translated page serves all in-page plug-ins. The following code segment shows the generation of the event handlers.

```
<xsl:template match="/meml">
  <script type="text/javascript">
  <xsl:comment>
  function DetectEvent(ObjectID)
  {  var OID = getElementById(ObjectID);
     switch (OID.Event) {
       case "AccessDOMObject":
          AccessDOMObject(ObjectID,
          OID.DestID); break;
       case "UpdateDOMObject":
          UpdateDOMObject(ObjectID,
          OID.DestID); break;
        /*  Other events' list  */
     }
  }
  function AccessDOMObject(SourceID, DestID)
  {  var sID = getElementById(SourceID);
     var dID = getElementById(DestID);
     sID.SetParameter(sID.ParamName,
                      dID.value);
  }
```

```
  /* Other event handlers -
     UpdateDOMObject(), etc. */
  </xsl:comment></script>
</xsl:template>
```

After the translated page is loaded into a Web browser, this MeML plug-in instance will fire an event `AccessDOMObject` to the browser to obtain `exp1`. The event handlers will catch the event, retrieve the expression's text value, via the DOM interface, and send the value back to the MeML plug-in. With the expression, MeML plug-in will send an MESP request to the remote *WME Service* [16] using the specified URL (line `1`). When the computation result is returned from the WME service, the MeML plug-in will fire a `UpdateDOMObject` event. Finally, event handlers will update the page by displaying the result through DOM. The `<computation>` tag can also specify target elements in the page where the dynamically computed results will be inserted into the page. With this kind of round-trip between Web browser and remote WME Services supported by Woodpecker, an MeML page can provide rich dynamic content and user interactions.

The general mechanism for `<mathgraph>` is similar to that for `<computation>`. But they have very different functions. The `<mathgraph>` tag is mapped to the MeML plug-in *MathGraph*, which is a visual object in the Web page, and *MathGraph* generates 2D and 3D figures in its allocated area in the browser window.

The `<expression>` tag is mapped to a hidden MeML plug-in *MathExpression* which is responsible for manipulating all in-page mathematical expressions. `MathExpression` also contains an implementation of XMEC (eXtensible Mathematics Encoding Converter) [17]. The converter can convert in-page expressions among four primary encodings: MathML, MP, Infix, and OpenMath. This capability is also handy when calling WME services or generating appropriate display in the browser.

VII. Future Work

WME is an ambitious vision and there is much work ahead. The set of MeML elements and their functions will continue to evolve to better support quality teaching and learning of mathematics. To make WME services interoperable and easily accessible on the Web/Internet, we are defining MESP , the *Mathematics Education Service Protocol*, and creating support libraries for it. The software components of the WME framework, of which the MeML language is a part, will continue to be tested and refined. A project involving experts of mathematics education in the education department and middle school mathematics teachers is being formed to put MeML and WME prototypes to trial and to refine the system to better support the real needs of mathematics education in practice. An MeML authoring tool, *Madmath*, is being designed to make authoring mathematics education pages in MeML much easier.

References

[1] Lou Burnard. *An Introduction to Text Encoding for Interchange* `www.tei-c.org/Lite`

[2] 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`.

[3] H. Cuypers and H. Sterk. *Mathbook, web-technology for mathematical documents*. Proceedings of the BITE 2001 conference. See also from `www.riaca.win.tue.nl`

[4] Berthold Daum, and Udo Merten. *System Architecture with XML*, Morgan Kaufmann Publishers.

[5] Mike Dewar (Web Master). *OpenMath*. The OpenMath Society, `www.openmath.org/`

[6] Burkhard Freitag. *Learning Material Markup Language*, `www.lmml.de`

[7] Max Froumentin, Team Contact for the Math Working Group. *MathML*. `www.w3.org/Math`

[8] Dianne Kennedy. *ISO 12083*. `www.xmlxperts.com/12083.htm`

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

[10] Michael Kohlhase. *OMDoc: Towards an Internet Standard for the Administration, Distribution and Teaching of mathematical Knowledge*. Proceedings of Artifi cial Intelligence and Symbolic Computation Springer LNAI 2000.

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

[12] Norman Walsh. *DocBook XML 4.2*, `www.oasis-open.org /docbook/xml/4.2/index.shtml`

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

[14] 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.

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

[16] Paul S. Wang, Norbert Kajler, Yi Zhou, and Xiao Zou. *WME: Towards a Web for Mathematics Education*, Proceedings of ISSAC 2003 (to appear).

[17] Xiao Zou and Paul Wang, "XMEC: An Extensible Mathematical Encoding Converter", `http://icm.mcs.kent.edu/research/xmec/`

[18] Yi Zhou. *MeML DTD*. `www.cs.kent.edu/~yizhou/MeMLDTD.pdf`

[19] Proceedings, IAMC 99, 01, 02, and 03 Workshops, summer 1999, 2001, 2002, and 2003, `icm.mcs.kent.edu/research /iamc.html#iamcworkshop`

[20] MathPlayer and WebEQ, `www.mathtype.com`.

[21] The Advanced Distributed Learning Initiative, Offi ce of the Secretary of Defense. *Sharable Content Object Reference Model*. `www.adlnet.org/`

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

[23] SOAP (Simple Object Access Protocol), `www.w3.org/TR/soap`.

[24] Workshop on *The Future of Mathematical Communication*, `msri.org/activities/events /9900/fmc99`, Dec. 1999.

[25] WWW Interactive Mathematics Server, `wims.unice.fr/ wims/`