Proceeding of The 11th Asian Technology Conference in Mathematics， HongKong, December 2006, pp. 271-279.

MathEdit, A Browser-based Visual Mathematics Expression Editor

Wei Su[1], Paul.Wang[2], Lian Li[1], Guanyu Li[1], Yanjuan Zhao[1]
[1]Lanzhou University, Lanzhou, Gansu,730000, China
[2] Kent State University, Kent, Ohio 44242-0001, USA

Abstract. MathEdit is a browser-based interactive editor for mathematical expressions being developed jointly by researchers at Kent State University (USA) and Lanzhou University (China). MathEdit is an open source software written in JavaScript and DOM. It makes use of browser support, either natively or via a plug-in, for MathML display. The dynamic nature of MathEdit makes it useful for mathematical web sites to enable users to create/edit/submit mathematical expression. MathEdit is also an important tool for WME (Web-based Mathematics Education) and MAG (Mathematics Assessment Grid), both are on-going research projects at Kent and Lanzhou. We present the design and implementation of MathEdit, its user interface, program architecture, and API. Also discussed is the flexibility and extensibility of MathEdit.

1. Introduction

This research collaboration between ICM at Kent State University and the Department of Computer Science at Lanzhou University (LZ) started in 2004 when LZ got interested in adopting WME (Web-based Mathematics Education) [11, 12, 21] for use in China. WME is a modern distributed system on the Web for mathematics education. The approach is to provide each participating school with a website that is comprehensive, well-organized, dynamic, interactive, hands-on and ready to use by teachers for mathematics teaching in the classroom.

An interesting sub-project for WME is to create a simple to use editor for users to input mathematical expressions. This editor must be easily integrated with the Web-based lessons and assessment materials in WME and provide an intuitive GUI for entering and editing mathematical expressions. The editor must also be flexible, customizable and extensible to address different user groups at various levels.

ICM and LZ jointly created MathEdit [5, 20], an editor that is easily interfaced to WME but is self-contained and free-standing so it can be used for many other purposes. MathEdit is implemented in standard JavaScript [16] and uses the Document Object Model (DOM) [2] to represent the mathematical expression being created/edited. MathEdit can produce MathML code--content, presentation, and composite codes. Its GUI offers visual navigation of sub-expressions and an *expression template palette* for expression input.

Several mathematical expression editors have been developed in the past ten years. Early editors adopted proprietary formats making interoperability with other programs a problem. For Web publication, mathematical expressions are usually created then converted to an image [4, 17, 22, 23]. The standardization of MathML [8, 18] by W3C has been a positive development forward and people began to create mathematical expression editors based on MathML [6, 7, 9, 13]. Systems such as WebEQ [19] and the IBM MathML expression editor [15] have been widely used in science computing and education. These systems still need a proprietary mathematics rendering engine, such as MathPlayer [3] for displaying mathematical formulas. More recently, Web browsers such as Amaya [1], Firefox [10] and other Mozilla-derived browsers began to provide native support for MathML.

Browsers on which MathEdit works include IE (with the MathPlayer [3]plugin), Firefox, Netscape Navigator, and Mozilla. MathEdit makes use of browser support for MathML and is different from earlier editors. Because MathEdit is open and coded entirely in standard JavaScript, it runs inside any MathML-capable Web browser and can be included in any Web page that requires mathematical expression input. MathEdit also allows a mixture of infix and template-based input. Furthermore, MathEdit has a well-defined customization and extension mechanism.

2. MathEdit Overview

MathEdit is an interactive visual mathematical expression editor. Running in a Web browser, it allows you to create and edit mathematical expressions with a convenient and intuitive graphical user interface. With MathEdit, users can easily enter mathematical expressions as answers to questions in mathematics lesson pages for example. Web page authors can also use MathEdit to create mathematical expressions to be included in their XHTML documents. Figure 1 shows the MathEdit authoring environment.

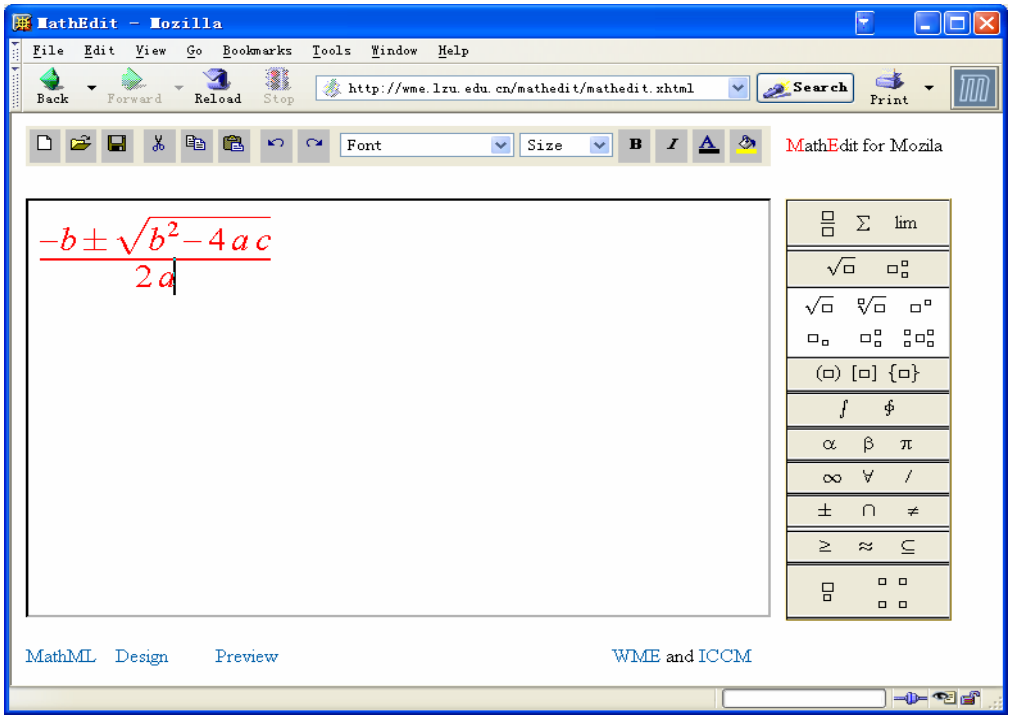

Figure 1: MathEdit Authoring Environment

MathEdit is implemented in standard JavaScript and DOM. MathEdit runs within any standard Web browser that supports JavaScript and DOM.

2.1 Main Functionalities

While its main function is the interactive creation and editing of arbitrary mathematical expressions, MathEdit also provides other important capabilities.

The following lists the most important functionalities.

* Create a new or edit an existing mathematical expression interactively with a convenient GUI
* Direct editing of MathML code
* Set mathematical expression format and style
* Customize toolbar, palette and expression template
* Import/Export MathML
  - Capture and retrieve the MathML markup from other applications or webpage.
  - Open an existing MathML file stored in the local file system or at the originating Web server
  - Save MathML in a local or remote file
  - Return the result mathematical encoding (content, presentation, and composite) to the parent window

2.2 Operational Modes

MathEdit provides three different operational modes: *MathML mode*, *Visual Edit mode*, and *Preview mode*. Generally, the *visual Edit mode* is for users to create and edit mathematical expression interactively. In *Preview mode*, the expression is displayed in its final form and no editing is allowed. Previewing is usually the last step before a user is done creating or editing an expression.

The MathML mode allows the user to view, enter, and edit MathML source code directly. A user can copy and paste MathML code in the MathML mode.

2.3 Usage Scenarios

MathEdit is designed to work with dynamic Web pages providing end users with the ability to interactively enter mathematical expressions. The expression entered by the user is returned to the invoking Web page for display or posting to the server.

Here is a typical usage scenario:

Step 1: The user accesses a Web page using a suitable browser, say Firefox.
Step 2: The Web page, together with MathEdit (JavaScript code), is delivered to the user's browser which displays it.
Step 3: The user now fills out a form on this page which contains an entry for a user-created mathematical expression.
Step 4: The user clicks on an "Enter Expression" button which invokes MathEdit in a pop-up window for the user to enter the desired expression.
Step 5: The user clicks a "Done" button to close the pop-up window causing the finished expression to be returned to the form and displayed together with the rest of the form.
Step 6: The user submits the form after filling it all out. The form data, including the mathematical expression, coded in composite MathML, is posted to a specified server.

Alternatively, an off-line Web page can be run locally that can save the created MathML code into a local file. The generated code can then be used for many purposes. For example, Web authors

may create mathematical expressions to be included in Web pages as static content this way.

3. MathEdit Architecture and Components

Figure 2 shows the end-user view of the MathEdit architecture. User actions, mouse clicks and keyboard input, are treated as commands. Commands invoke JavaScript functions in the command processing module. Each function basically adjusts the DOM tree of MathML content markup kept internally for the mathematical expression being constructed or edited. The code conversion module converts the content tree to MathML presentation code for browser display. For reading an existing expression, we use the Mozilla XML Dom parser under the Mozilla family browsers and the MSXML parser under Microsoft IE to conveniently process the MathML input. The *Mathematical Expression Toolbar* is customizable and extensible via adding and deleting templates (Section 4). New special characters, such as ∞, Greek Characters, etc, can also be introduced by the user through a combination of keyboard and mouse operations.

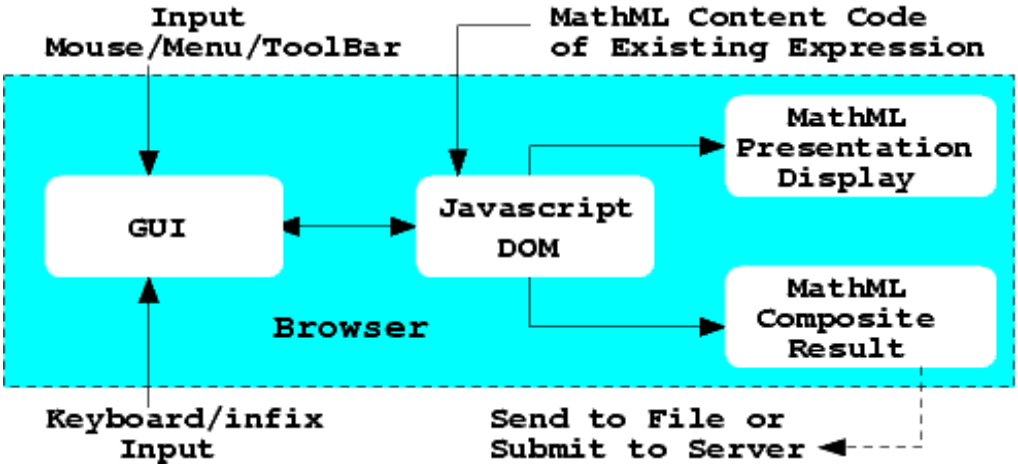

Figure 2: MathEdit Architecture

4. Internal Design and Implementation

MathEdit is coded entirely in standard JavaScript and runs in regular Web browsers. Here we describe several important implementation features.

4.1 Template

MathEdit supports two forms of user input, infix and template. Infix is convenient for users who are familiar with the notation. For others, the common approach is to use graphical templates [4,14,19] that can be selected to enter particular kinds of mathematical expressions such as fractions, square roots, powers, and so on (Figure 3). After selecting a template, the internal DOM tree and the display are updated with required expressions indicated by *placeholders*. The user can select each placeholder to enter the desired expression. Each input infix expression is parsed and a DOM tree fragment for it is created for placement on the DOM tree.

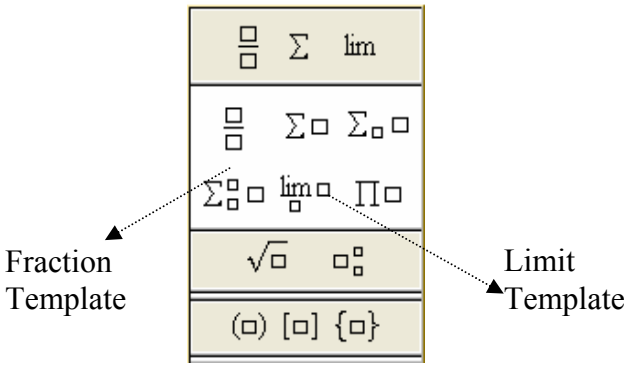

Figure 3: The Template Palette

4.2 Sub-expression Navigation

Convenient visual navigation of the displayed expression is essential for the user to pick a location within the expression to make changes. MathEdit keeps track of the *sub-expression* and displays a color background to visually identify it to the user. The arrow keys are used to move the current *sub-expression* up to the parent node, down to the first child, or left/right to sibling nodes. With the current expression selected, a user can *insert before*, *insert after*, *edit*, *delete*, or *replace* the current expression (Figure 4).

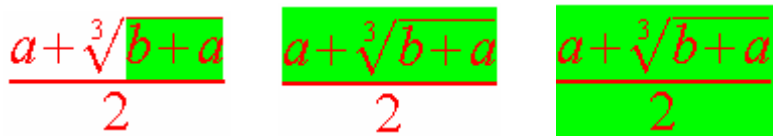

Figure 4: Sub-expression Navigation

4.3 Command Processing

Each template corresponds to a particular DOM tree structure. Infix input is parsed to produce the corresponding DOM tree structure. For this purpose we have obtained the infix parser from Dr. Xaio Zou which is written in JavaScript and originally produced MathML content codes. We have modified this parser to directly construct DOM tree nodes.

Thus, each user command results in a DOM tree fragment that is placed on the DOM tree which represents the entire expression being edited. Figure 5 shows the DOM trees in creating $\frac{\sqrt{a+b}}{2}$.

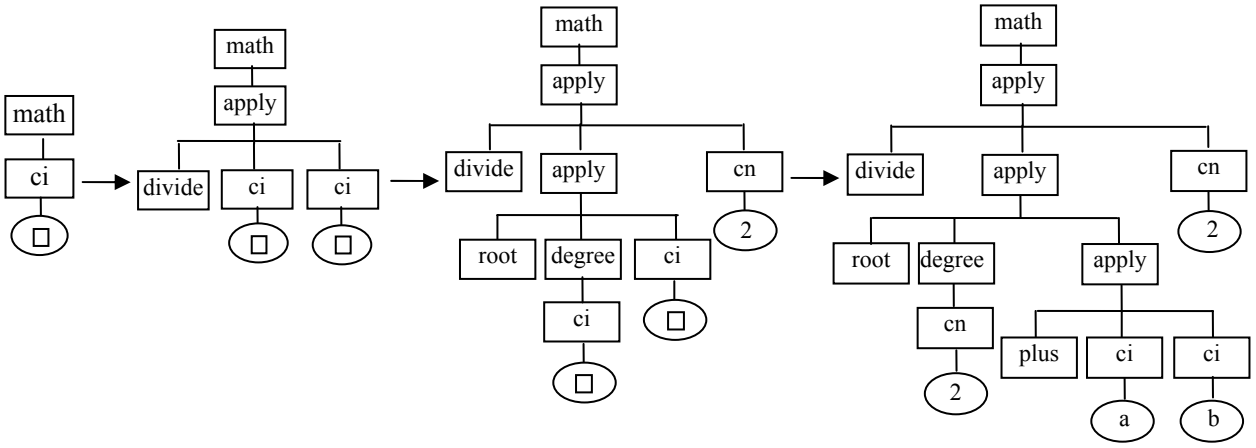

Figure 5: Command Processing

4.4 Editing DOM Tree

Modifying and replacing DOM tree nodes in MathEdit are straightforward. Deleting is more complex. Depending on the current expression selected, the delete operation may or may not be allowed. If the current expression is a *required operand* then it cannot be deleted. For example deleting the numerator or denominator of a fraction is disallowed. So is the exponent of a power or an argument of a function such as *sin*.

5. Extensibility and Customization

In designing any software, it is important to carefully balance ease of use, power and functionality, and the needs of different user groups.

For MathEdit, we have a broad spectrum of potential users: students, teachers, scientists, engineers, and researchers. The education community and publishers of electronic textbooks are two important groups that can take advantage of putting mathematical formulas on the web.

In secondary education, students and teachers need to be able to create mathematical content quickly and easily through an intuitive and natural user interface. It is well-known that if the tool becomes more complex than the educational content, tool usage can become a distraction rather than a help.

Requirements for mathematical notations differ for different users. For example, at the elementary to middle school level the sign for division is ÷ rather than /. For high school, notations for log, trigonometric functions, the Greek symbol π, and the plus-minus sign (±) as in the roots of a quadratic equation are needed. At higher levels, additional Greek symbols (β for example), notations for limits, integrals, vectors, tensors, and so on will have to be included. For engineers, the square root of -1 is usually denoted as *j* rather than *i* as in mathematics.

The key in MathEdit to solve this usability vs. complexity issue is to make the tool easily extensible to add functionalities and customizable to suit the needs of different user groups.

Customizable configurations in MathEdit include GUI properties, the toolbar, the input palettes and

other properties. GUI properties include font size, color, window background color, window size, highlight color, etc. The toolbar contains the file menu, the edit menu, and the format menu. The MathEdit *palette* displays a set of icons for the available input templates. The main palette may contain template icons as well as sub-palettes that pull down with a mouse click.

The customizable configuration information is stored in the `config.xml` file. At initialization time, MathEdit will load and parse the server-side `config.xml` file. Application developers can directly edit the configuration file to customize MathEdit. For certain end uses, we also provide a dialog window to change the configuration information. Customized results are saved back to the server.

In addition, the API library of MathEdit provides functions for changing the configuration parameters, and for reloading and saving configuration information from/to local filesystem or server. In WME, teachers can customize and deploy MathEdit as needed for their individual classes.

Using the MathEdit customization mechanism, both the mathematics expression toolbar and the list of mathematical symbols can be easily extended and modified. Figure 6 shows four customized examples, which respectively are used for middle school, high school, triangle function curricula and set operation curricula.

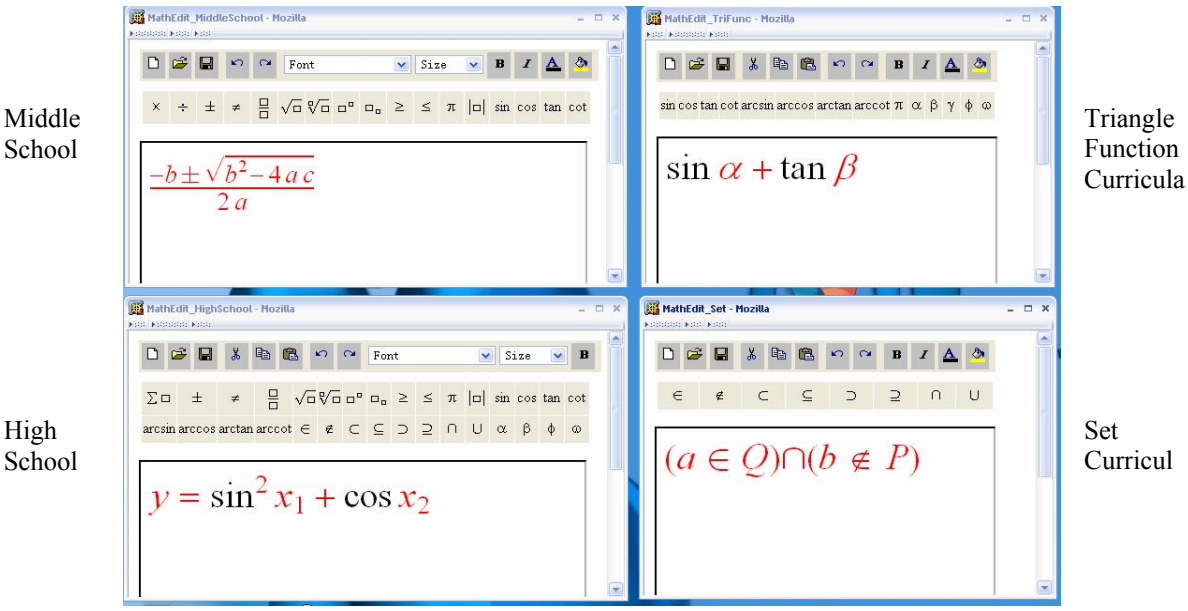

Figure 6: Customization Example

6. MathEdit API library

The MathEdit API, supported by a set of JavaScript functions, allows applications to create an editor instance (associated with a pop-up window or in-page frame), sets any initial expression in it, configure its template palette and other GUI features, and set/retrieve the MathML code it contains.

Figure 7 shows a WME lesson with a MathEdit in-page frame. Code similar to the following can be used in the Web page header to load the MathEdit API library.

```
<script type="text/javascript" src="matheditAPI.js"></script>
```

Here are parts of code to set the displaying toolbar and template palette:

```
Var MathEditWin= new Mathedit.setIframe(“mathedit1”);
MathEditWin.hiddentoolbar(“all”);
MathEditWin.showtemplate(“fraction”);
MathEditWin.showtemplate(“divide”);
MathEditWin.showtemplate(“times”);
MathEditWin.showtemplate(“sub”);
MathEditWin.showtemplate(“bracket”);
```

When a user clicks on the “show me” button, the MathML of user’s expression will be retrieved through `mathedit.getContentMathML()` call. The MathML will be used to compute and display the results.

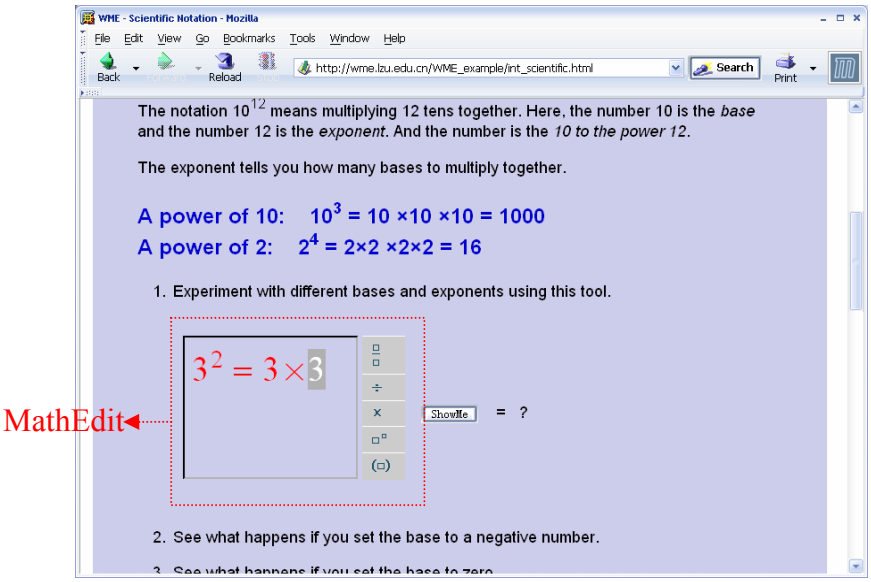

Figure 7: MathEdit API Example

7. Future Work

We want to put MathEdit to more extensive use in the WME system, to collect user feedback, and to make improvements. Extensions to the API are also anticipated. Once the feature set of MathEdit becomes stable, we want to re-implement it with total object orientation.

We will also add *Copy* and *Paste* support to MathEdit so users can obtain/set the MathML code in the MathEdit through the GUI. Other encoding formats such as LaTeX and infix notations may be supported.

Interfacing MathEdit to a plotting program for mathematical functions can also be very useful.

8. Acknowledgments

We'd like to thank Dr. Xiao Zou of Kent State University for making available his mathematics encoding converter for use in this project.

The material reported here is based upon work supported in part by the National Natural Science Foundation of China under Grant No. 90612016, 60473095 and the National Science Foundation of USA under Grant CCR-0201772. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

Reference

[1] Amaya Homepage http://www.w3.org/Amaya/.
[2] Document Object Model http://www.w3.org/DOM/.
[3] Document of MathPlayer http://www.dessci.com/en/products/mathplayer.
[4] Document of MathType http://www.dessci.com/en/products/mathtype/.
[5] K. Cem Karadeniz *WME MathEdit Design & Requirements Analysis,* technical report ICM/Kent.
[6] LI Junguo ZHANG Lipang. *A XML Based Presentation Technique for Mathematical Expressions on Web Page,* Acta Scientiarum Naturalium Universitatis Pekinensis , Vol. 39 , No. 5 (Sept ,2003).
[7] Luca Padovani. *Interactive Editing of MathML Markup Using TEX Syntax,* TEX, XML, and Digital Typography 2004.
[8] MathML Conference 2002 http://www.mathmlconference.com/2002/.
[9] Mathmled http://www.newmexico.mackichan.com/MathML/mathmled.htm.
[10] Mozilla MathML Project http://www.mozilla.org/projects/mathml/
[11] Paul S. Wang, Norbert Kajler, Yi Zhou, and Xiao Zou. *WME: Towards a Web for Mathematics Education*. Proceedings of ISSAC, ACM Press, August 2003, pp. 258- 265.
[12] 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 SoutheastCon 2005.
[13] Samuel S. Dooley. *coordinating Mathematical content and presentation markup in interactive mathematical documents.* Proceeding of the 1998 International Symposium on Symbolic and Algebraic Computation, ACM SIGSAM, ACM Press.
[14] Samuel S. Dooley. *Editing Mathematical Content and Presentation Markup in Interactive Mathematical Documents,* Proceedings, ISSAC 2002.
[15] Samuel S. Dooley. *Users Guide for the Integre MathML Equation Editor* http://www.integretechpub.com/mathex/.
[16] Standard ECMA-262 http://www.ecma-international.org/.
[17] The W3C MathML software list http://www.w3.org/Math/Software/.
[18] W3C Math http://www.w3.org/Math/.
[19] WebEQ Documentation http://www.dessci.com/en/products/webeq.
[20] Wei Su, Guanyu Li, Yanjuan Zhao, Lian Li. *Initial Design of Mathematics Assessment Grid*, Presentation at Symbolic Computation in Education'2006.
[21] Wei Su, Lian Li, and Paul S. Wang. *Lesson Page Structure and Customization in WME,* Proceedings of the 2005 IAMC Workshop, Beijing, China, July 24 2005.
[22] Yasutomo Nakayama. *Mathematical Formula Editor for CAI* ,CH1'89 PROCEEDINGS.
[23] Y. Doleh and P. S. Wang. "*A System Independent User Interface for an Integrated Scientific Computing Environment*" Proceedings of the ISSAC'90, Addison-Wesley (ISBN 0-201-

54892-5), Aug. 1990, pp. 88-95.