How to achieve user-friendly interactivity in Mathematics Exercises on the web

Giovannina Albano[1], Bernardo D'Auria[1,2], Aldo Pagano[2], Sandro Viglione[2]
[1]DIIMA – University of Salerno
[2]CRMPA c/o University of Salerno

Abstract

In this paper we want to show some problems that can arise in developing Mathematics Exercises on the web and how we have tried to face them. We have developed a web application, MoMAM@th, that lets the users to solve on-line exercises about basic mathematics. The main idea behind MoMAM@th is to help the student in the learning process by decomposing the exercises in simple steps and asking her/him to answer questions by filling only some blank boxes contained in the formulas. In the first and current version of MoMAM@th the output was given by images while the student has to insert input using Mathematica syntax. Our experience in e-learning application suggests that in order to produce e-learning mathematics exercises of good quality it is important to furnish the student user-friendly tools. The user has to pay attention to the content rather than the way s/he has to write mathematical expressions. Hence, we are now upgrading the application by using MathML expressions and editors in the web pages and we realized that this process isn't immediate as the available tools to read and write mathematics on the web seem not to be directly suitable for application with a non-trivial structure. We tried to overcome such difficulties using javascript and WebEq, allowing also common actions such as copy and paste. Finally we want to stress the importance to improve the available plug-ins in order to satisfy the needs of the new application for distance learning.

Introduction

We live in a rapidly changing and increasingly complex world where knowledge is the key factor for economic growth, so as individuals we need to learn faster and more often. As pointed out from European Community [WP-ET-03-05.2.5], for all aspects of life - education, work, leisure - modern society needs to provide efficient and inclusive support for its citizens to learn continuously if it is to remain competitive and socially aware. Of particular importance is the need to ensure that opportunities are open to all within this evolving Knowledge Society.
ICT has helped enormously in making learning resources more accessible to local communities, to schools, to universities and to individuals for work or for pleasure, producing the so called computer-based learning environment. The Internet advent gave a powerful communication structure to easily access to any kind of information and knowledge, and e-learning can be roughly defined as the ability to provide information to anyone, anytime, anywhere. But e-learning cannot be reduced only to furnish material, as it needs, through technology, turning standalone educational applications into web-based ones. Pedagogical questions have been posed: it was necessary to know the instructional theories that technology has tried to implement. E-learning arises from the contact between computer science and education and researchers from these two branches received new motivations for their studies. A substantial change in methods and techniques approaches to teaching/learning is needed. This leads to an approach based on descriptive, intuitive, experiential and inductive aspects.
Current and future e-learning solutions are required to be based on a user-centred integrated approach that recognises the inseparability of pedagogy, organisation, ubiquitous access, technology and infrastructures [WP-ISTC-FR]. In such view three main characteristics are stressed

and fostered by e-learning: the constructivist feature of learning - the learner has an active role in acquiring his “own” knowledge; the tailored instruction – individual tutoring situations better fit the needs of the learner; learning for all - all solutions should be technically feasible and economically affordable.

In this context we have tried to design and to implement an e-learning application, that is MoMAM@th, to support mathematics learning, giving a qualitative improvement to the didactical properties of the traditional textbook, as briefly described in the following.

The process that leads the learner to solve mathematics exercises (or more in general to acquire competencies) can be split into three phases: the acquisition of the information, the personal elaboration and the verification [AlDaSa03]. MoMAM@th is structured in such a way that the learner to follow the student in all the previous didactical phases, and our efforts were and are to make it simple and intuitive in order to let users to mainly focus in the learning process.

The learner constructs his/her knowledge performing exercises split into “elementary” steps: at each step the software suggests elementary operations to do, giving the possibility of consulting theory references, and the student is required to interact with the software completing some blank boxes which suppose to do computations or to take into account theoretical results. The student can access the next step when s/he correctly responds to the question in the current step, or when s/he choices to see the solution.

MoMAM@th is accessible through a common web browser: the student accesses to a personal working area through login and password where the available topics and various types of related exercises are presented by means of a user-friendly interface with menu and buttons.

Beyond the exercises classified with respect topics and subjects, the student is able to make a simulation of exam, that is a session with exercise randomly generated from chosen topics: the student is required to give directly the final answer and, if s/he fails, s/he can access to the step by step learning path.

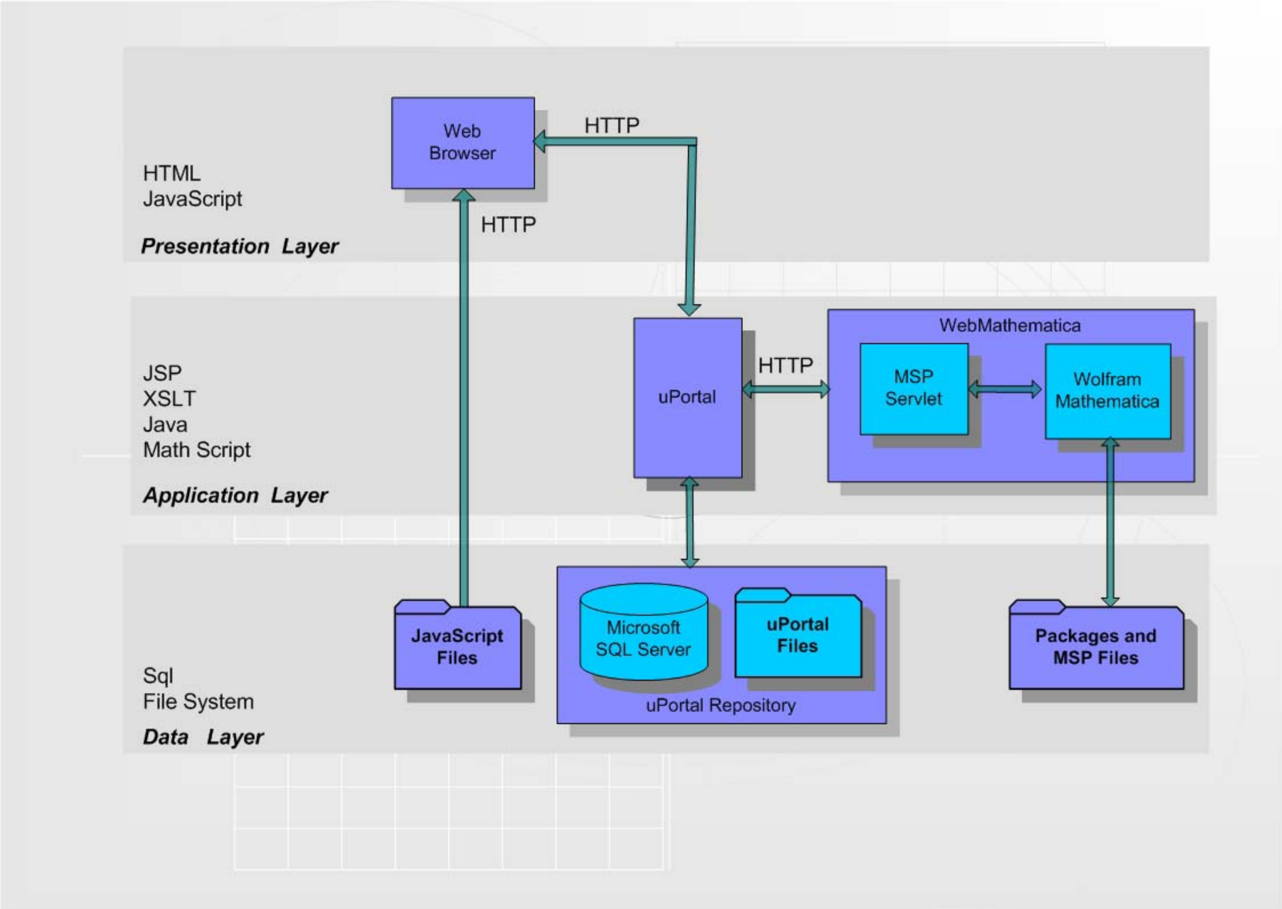
A teacher support in asynchronous way by e-mail is available and synchronous tools (e.g. chat) are planned.

What we are focus on in this paper are the difficulties arose from the fact that we deal with mathematics, whose language is hardly symbolic, and textual mode is not suitable, nor sufficient, to communicate. Moreover the introduction of a particular syntax, like in a hard CAS, is not well envisaged because it implies an extra effort for learners and the risk is that people concentrate attention to learn syntax rather than the mathematics knowledge.

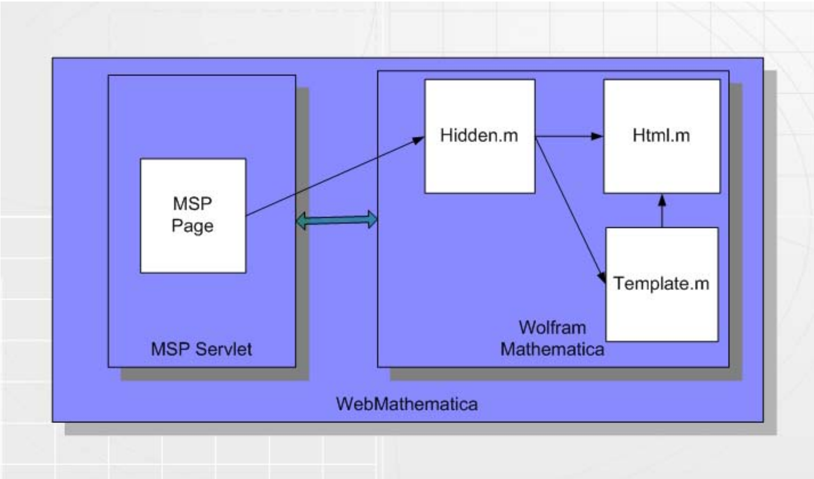
In the following we will discuss technical limitations of MoMAM@th in the communication of mathematics on the web and which are our ideas to solve them. Further research needs to be carried out to ensure quality (in terms of user friendliness) and efficiency.

An idea of mathematics exercise structure on the Web

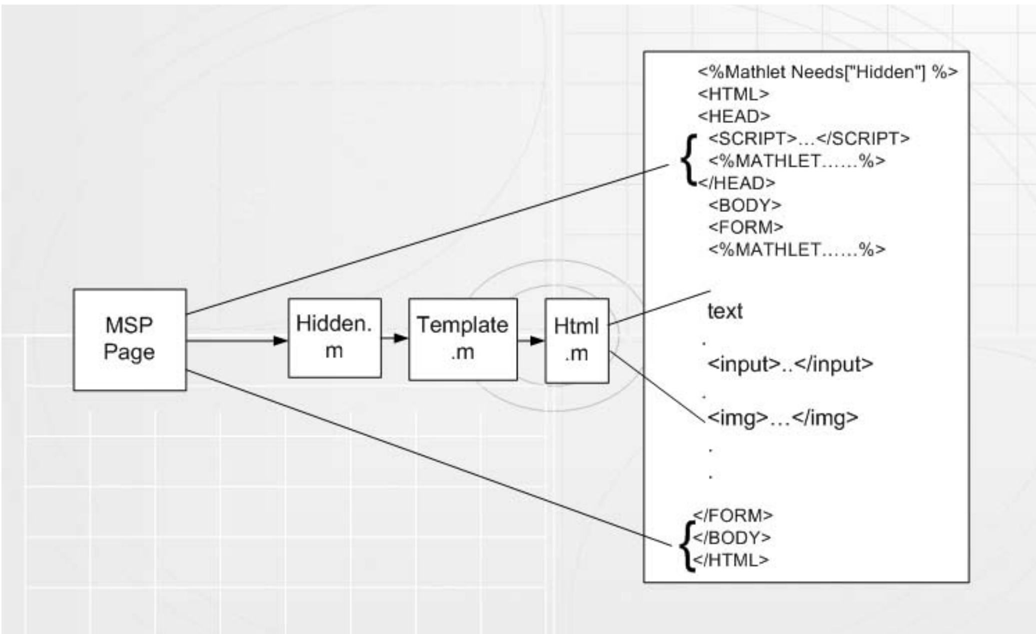
The logical architecture of MoMAM@th is distributed on three layers: the data access layer contains the database and the file system that store all system data; the application layer where the information are elaborated according to the user’s inputs; the presentation layer that is responsible of the delivery. In this section we don’t want to describe all the components of each layer but only to describe the technologies, software and standards used in MoMAM@th.



MoMAM@th is mainly based on two applications, uPortal and WebMathematica©.
uPortal is a free and open source application, developed in java by the JA-SIG group, and it is used to create web portals, i.e. to organize the presentation of resources on the web. uPortal lets to organize the web-page in sub-areas, called "channels", managed independently by software. The technologies it uses are a combination of JSP page, Servlet, XSLT and CSS stylesheets.
In MoMAM@th we used the channels to collect exercises all belonging to a particular mathematical subject (such as Limits, Derivatives, etc.). Moreover these topics additionally are grouped in different sessions relative to particular branch of mathematics (e.g. Calculus, Linear Algebra etc.). We use XSLT and CSS to modify and control the layout of the exercise generated by WebMathematica.
WebMathematica is the new product of the Wolfram Research that lets to insert mathematical computation made by Mathematica within web pages. It uses special web pages, MSP pages, that use a special script code, the Mathlet instructions, to communicate with the Mathematica Kernels. The Kernel can access to data posted to the MSP page, elaborate them according to the script instructions and then dynamically insert output, including graphs, in the returned page.
In MoMAM@th, uPortal receives the input data submitted by the user, and then, like a proxy, it forwards them to WebMathematica with a post to the MSP page of the specific exercise.
The WebMathematica servlet acquires a Mathematica session for the request among a pool of pre-initialized sessions and starts in processing the script of the MSP page.
In order to give a standard structure to the exercises and moreover to simplify their development we organized the function called in the script code in some packages, as it's shown in the next figure.



The MSP pages is able to exclusively call the functions contained in the Hidden.m package. They manage the structure and the sequence of the steps the exercise is made by. The functions are called for structuring the exercise in steps, for the input and the output of the data and to analyze the learner answers. For instance in Hidden.m there is the logic to decide the how much times the learner has to retry an erroneous answer before being able to jump to the solution. Moreover, according to the step number, the functions in Hidden.m provide to call the respective functions of the package Template.m. The package Template.m contains all the logic relative to the particular exercise.
Finally, in order to simplify the format of the output data, the Html.m page collects some utility functions.



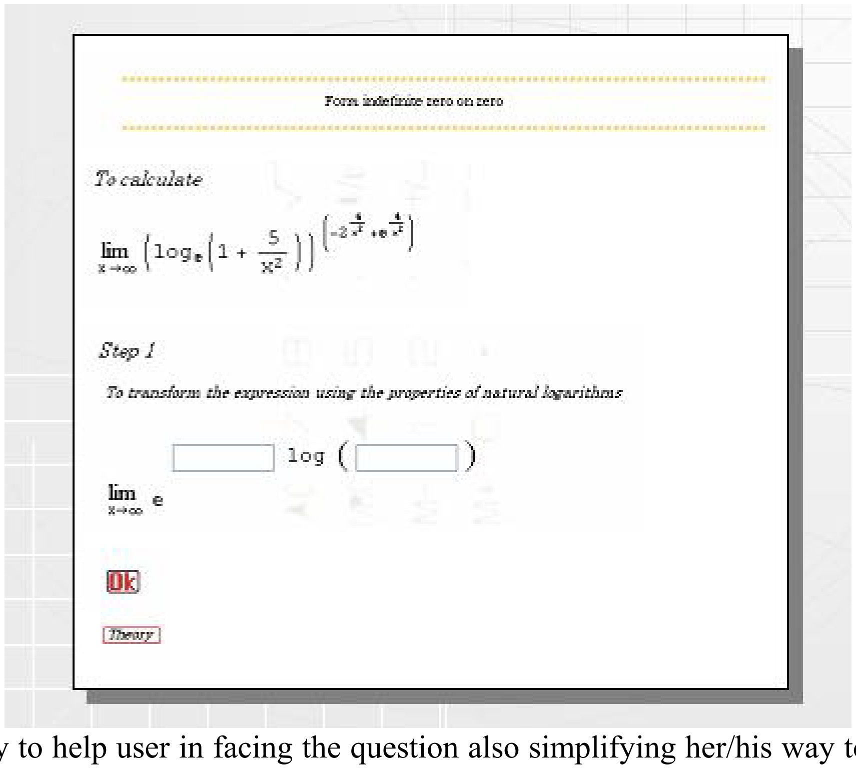
In the figure above the structure of a MSP page is roughly shown. It is mainly an HTML page, but for the mathlet script that make calls to the Hidden.m functions.
In the figure, at the beginning of the page, the mathlet instruction is also shown, needed to load the package Hidden.m. (i.e. the tag <% Mathlet Needs ["Hidden"]%>).
The resulting HTML page, representing one step of some exercise, will contain text and formulas, the latter generated as images in the classic mathematical formalism, produced automatically by WebMathematica.

The MSP page contains also a link to a JavaScript code file. We used JavaScript functions to transform the exercises input/output data from a HTML format to that one intelligible by the Template.m (and vice versa), and also to build exercises that need to adapt their layout according to user needs, directly client side.

Our experience

MoMaM@th user interface

The main goal of MoMAM@th is to be a collection of non-trivial mathematics exercise available on the web. The words ‘non-trivial’ mean that the exercises do not limit themselves to be a simple right or wrong controller of user answers, but they try to follow the student in the learning process by being decomposed in incremental steps.
In order to be an useful and helpful learning tool, MoMAM@th needs to let the learner focus himself on the mathematical problem hence avoiding to bore her/him with the difficulties to understand the MoMAM@th environment. This implies that MoMAM@th interface must be very easy and tidy.
What we did is to show formulas in symbolic syntax, by using gif image, and to show the contents to the essential. In the following feature a sample exercise is reported where it is possible to see the exercise statement together with the first step to do. Moreover there are some text boxes that user has to fill, some command buttons and a link to the theory.



MoMAM@th try to help user in facing the question also simplifying her/his way to answer. As it’s visible in the figure the answer is given filling some text boxes that are part of a formula. Hence the formula is made by fixed part, visible as gif images (that are also used to draw square roots and fraction lines) and by editing parts that at the moment are simply text boxes.
This implies that the fixed parts are shown by using symbolic formalism while the user as to answer the question filling text boxes by using textual expressions.
Sometimes, such as to define the domain of existence of a function, we used also other html tools like buttons and combo boxes to simplify the user input.

Limitations and first solution

Our solution seems far from being the best one. At the moment in order to create a formula in the way described before we need to do a lot of work at the level of "Template.m" package. This means that who creates the exercise has also to take into account how to generate the structure that user will use to provide the answer, and often the result is not very appealing from an aesthetic point of view. In the browser if the formula is too long and overcomes the maximum horizontal window size, it doesn't distribute on more lines but causes the horizontal scrolling of the browser window. In this way the learner cannot see the formula fully but has to use the horizontal scroll bar to see all the mathematic expression.

Moreover, many times the format of the answer is not easy to respect; as an example, if the exercise needs to receive as answer the base of a vector space, the user should give in input something similar to {{1,2,3},{5,6,4},{2,3,4}}, which is not very intuitive.

MoMAM@th has been experimented at the Faculty of Engineering of the University of Salerno. It has been used as complementary tool for the first year mathematics students. The students attending traditional lectures have used the software for their own extra exercise sessions.

Some students sent teacher e-mails asking for help on how to write textually mathematic expressions by Mathematica syntax, while someone else sent comments about the fact that in MoMAM@th it's not possible to make copy and past from of the graphical part of the formula.

Hence student feedbacks have underlined two more and important limits that the MoMAM@th exercises have: they don't allow the learner to write the formulas in symbolic notation and obviously it is not allowed to make copy and cut between graphical and textual part of a formula.

A partial solution to help the learner to fill more quickly the formula text boxes has been to preserve some values user entered in the case her/his answer was wrong. Indeed in this case the same question is shown again to the user and the system leaves the content of the text boxes with correct values erasing only the wrong ones. Moreover, thanks to the browser history, the user is able to access the values of the text boxes of the previous steps.

Problem overcome

To overcome the previous limitations the ideal solution would be to let the user be able to directly edit formulas in the browser using the symbolic mathematics notation. Moreover as shown before, the formulas of the exercise aren't a simple series of mathematical symbols.

They could contain an inner structure that we will call *template*, with some parts fixed (read only parts) and some editable. The latter should be described by some operators, one example, the simplest, could be the *placeholder* (equivalent to the Mathematica's placeholder) that let's the user to put in every type of mathematical expression.

More complex operators could control the data inserted by the learner letting or avoiding her/him to put some type of mathematical symbols or operator.

For example, considering the previous example about the base of a vector space the user could:

1. insert the complete set of vectors
   In this case the user has to fill a single placeholder and s/he is free to write any kind of expressions. The system has no control on the input before evaluating the answer server-side.
2. fill the components of a varying number of vectors, defining also their dimension
   In this case the user can insert only vectors together with their components, deciding also the dimensions of the vectors is unique for all of them. The system controls which objects the learner can use and their properties directly client-side.
3. fill the components of a fixed number of vectors (whose dimension is right), leaving some vectors empty
   The system shows the user a given set of empty vectors whose dimensions are fixed, the user can only fill the components. In this case the system maintains full control of the user

but limits the possible answers, it cannot verify if the user has understood the dimension of the vector space.

4. fill some components of a fixed number of vectors, leaving some vectors and some components of the vectors empty.
   The system shows, as in the previous case, empty vectors. However in this case their dimensions are more than the right one. Hence the user answer results to be complete even if to give it (i.e. to leave empty some components in the vectors) is less intuitive.

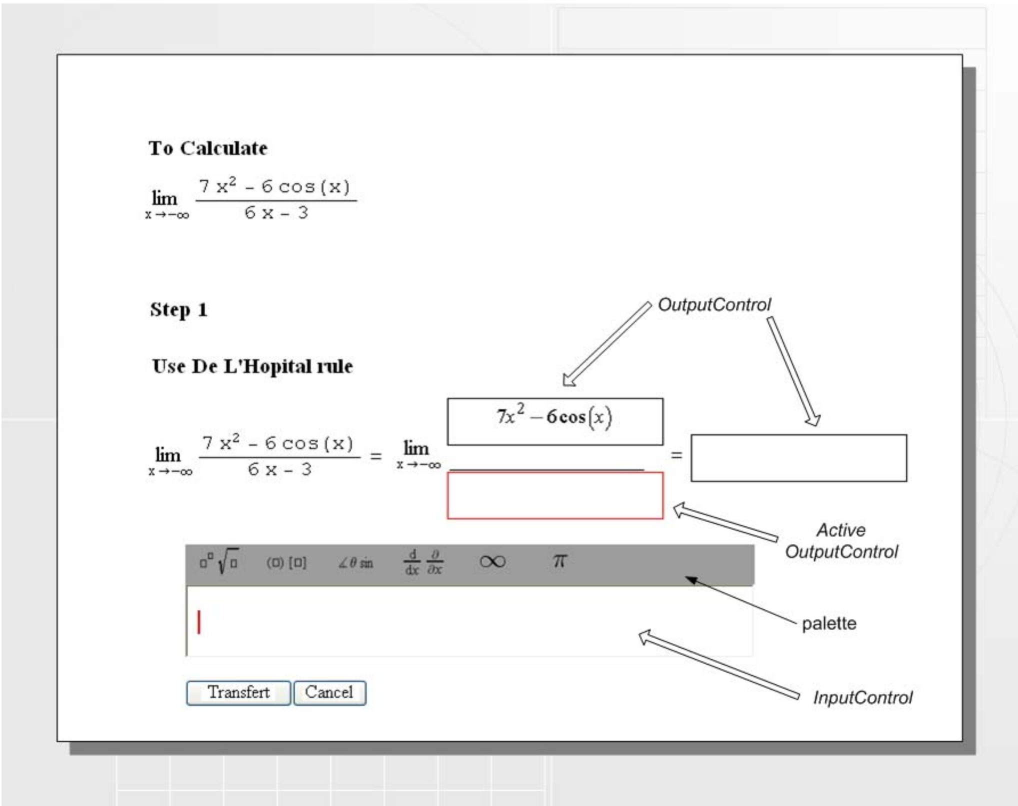
The better way surely is the second one. This one, however, imply that the user could insert *only vectors*, all with the *same* but *not fixed* dimension, and whose components can be *only numbers*. Hence, in this case, the input formula should know its structure by using special operators that know what the user has to insert. These operators are able to do syntactic verifications of the input directly client side and while the user digits.

The actual technologies and their limitations

The problem with the current technologies for putting mathematics on the web is that they are too complicated and ad hoc. It often takes expert knowledge and programming skill to get different math software packages to work together, but no single approach provides a complete answer. The remedy is obviously standards. MathML support is already widespread and it plays a pivotal role for dynamic math in Web pages. MathML equations can be manipulated using JavaScript, Applets and plug-ins can send MathML-encoded equations to server-side software such as computer algebra systems or scientific visualization software. However both MathML and plug-ins do not support *template* and *placeholder* as described by us. MathML deals only with complete formulas and not with formulas that may contain parts to fill, and more precisely parts that need to be filled in a specific way.
To obviate to this lack, a first solution could be to substitute each potential *placeholder*s inside a formula with a plug-in to write by symbolic notation (the natural extension of our current use of text boxes). Such solution, however, results to be unacceptable from a aesthetic point of view because of the interface of available plug-ins that is huge thanks to existence of a palette for the mathematical symbols. The presence of more then one plug-in inside a formula has resulted in html page that overcomes the maximum horizontal window size, so the learner cannot fully see all the formula but.
Another and better solution is to use a single plug-in to write and one or more plug-ins to visualize the written formulas, in symbolic notation. This has the advantage that the visualizing plug-ins occupy just the space required to show their mathematical contents. They naturally need to communicate with the writing plug-in to get the data to show. This is our solution and we will explain it more in detail in the next paragraph.
With the plug-ins technology we have obtained a good solution to put mathematic exercises on the web, but it is the custom solution.
However, the ideal solution would be that standard MathML was directly able to deal with formulas containing a *template* described by special operators. In the last years the didactic contents used for the e-learning have become more and more interactive and widespread used. In the case of the mathematics, MoMAM@th application constitutes a valid example of such interactive contents.
The standard MathML, that surely will be used in e-learning application about scientific domains, will have to be suitable to support the kinf od interactivity and structures described before. The possibility to use a standard will assure the development of exercises for the web that are independent from client plug-ins and CASs.

Our Solution

In this paragraph we describe which will be our next improvement of MoMAM@th system that will lead to the complete editing of formulas in symbolic notation.
To reach our scope we have used two WebEquation applets and the MathPlayer Plug-in of the Design Science.
The *InputControl is* used to modify the editable parts of a formula and let the learner to write text and mathematic symbols choosing them by a graphic palette.
The *OutputControl is* used to visualize them in the formula, and the Math, while the MathPlayer plug-in is used to display the fixed part of the formula.
In this way all the formulas are transferred by using MathML and so using textual information and this reduces also the time needed to download the pages. The figure below shows a prototypal example.



It shows an example of how *InputControl* and *OutputControl* are used to build an exercise. The three *OutputControl* represents what we defined as *placeholders*. To complete a part of the formula, the learner selects one *placeholder* and writes the mathematic expression in the *InputControl*. The active *OutputControl* is evidenced by a red border, so the learner can understand which part s/he is editing. When s/he ends to digit the formula, pushing the 'Transfer' button, transfers the content of the *InputControl* to the active *OutputControl*. If the user select an *OutputControl* that is non-empty, its data are transferred in the *InputControl* letting her/him the possibility to modify them. Using the *InputControl*, the learner has also the possibility to do operation like copy and paste.
WebEquations applets furnish a series of API to access the formulas that is represented through a MathML tree. We used Java Script to call the API functions in order to manipulate and move the expression in the *InputControl* and *OutputControl*.
Our solution still has same problems: we use many applets, their dimensions server-side fixed cannot be modified client-side by the usual browser commands. Moreover it needs to write a lot of

Java Script code to manage the applet communications. Indeed one limitation of WebEquation controls is that they can be controlled by JavaScript code but they are not able to issue events. This obliged us, for example, to capture the focus of the *OutputControl* by means of added object, because the applet does not manage the event *onmouseclick*.

Conclusion

What we have stressed in this paper is the fact that to use mathematics expressions on the web is not jet an easy task especially if we want to use them for e-learning purposes. Indeed we need to show the user a user-friendly interface to let her/him to focus on the content and not on the appearance. What we suggested is that probably also the standard for math on the web, i.e. MathML, needs to be evolved in order to include new constructs, that we call *templates* and *operators* (e.g. the *placeholder*), and that are very useful for didactic aims. Nevertheless we showed our temporary solution that seems to be a first step towards the ideal one.

References

[AlDaSa03] Albano, G., D'Auria, B., Salerno, S. "Stepwise maths exercises on the web", Proceedings of IMS 2003 (International Mathematica Symposium)

[AAVV02] E-Learning Report – December 2002

[MiTo01] Miner, R., Topping, P. (Design Science, Inc), "Standards-based Math on the Web", January, 2001, avaible at http://www.dessci.com/webmath/status/status_Jan_01.stm

[MiTo02] Miner, R., Topping, P. (Design Science, Inc), "Authoring Tools", January, 2002, avaible at http://www.dessci.com/webmath/status/status_Jan_02.stm

[WP-ET-03-05.2.5] ICT enabled learning in the 6th FP –Working Party on Education and Training, Information Society D/3/BH D(2001)

[WP-ISTC-FR] New Learning - Priorities for Work in Technology supported Learning under the Sixth Framework Programme – Working Party on Education and Training, Final Report to ISTC (2001)

Design Science, Inc. (MathType, WebEQ, MathPlayer), http://www.dessci.com

IBM Techexplorer, http://www-4.ibm.com/software/network/techexplorer/

MathML, http://www.w3.org/Math/

TeX (TUG: TeX Users Group), http://www.tug.org

Tex4ht, http://www.cis.ohio-state.edu/~gurari/TeX4ht/mn.html

uPortal, http://mis105.mis.udel.edu/ja-sig/uportal/

Wolfram Research (Mathematica, WebMathematica), http://www.wolfram.com/