WExEd - WebALT Exercise Editor
for Multilingual Mathematics Exercises

Arjeh Cohen* amc@win.tue.nl
Hans Cuypers* hansc@win.tue.nl
Karin Poels* k.j.p.m.poels@tue.nl
Mark Spanbroek* m.j.m.spanbroek@tue.nl
Rikko Verrijzer* r.verrijzer@tue.nl

*Department of Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands

***Abstract*— Recently, the computer program WExEd was developed within the WebALT project. It enables one to create and edit interactive mathematics exercises that can be automatically translated to and subsequently played in different languages (currently English, Spanish, Finnish, Swedish and Italian). It makes use of OpenMath, a standard for encoding the semantics of mathematical objects in XML, and software developed within WebALT. WExEd does not only encode the mathematics of the exercise in OpenMath but also the text. This is possible because, in general, mathematics exercises are built from a well defined and relatively small set of words. In this paper we give a presentation of WExEd.**

I. Introduction

The number of languages in the world is extremely large. This implies that publishers or authors of exercises in a certain language can only use these exercises in the countries where this language is spoken. If the exercises could be automatically translated to different languages, then a publisher could extend his scope dramatically.

Automatically translating exercises is hard because of large lexicons and complex grammars. It becomes less hard when the exercises are constructed within a well defined context from only a small and specific part of a language's lexicon. In general, this is the case for mathematics exercises, which can be as simple as "Calculate the derivative of $\sin(x)$", but also the words from more advanced exercises may come from a rather small and specific set of the language's lexicon.

OpenMath [1] is an emerging standard for representing the semantics of mathematical objects in XML. It allows mathematical objects to be exchanged between computer programs, stored in databases or published on the worldwide web. Each mathematical symbol (e.g. $\cos$) has its own OpenMath representation and OpenMath representations are grouped in so-called Content Dictionaries (CD's). Every combination of mathematical symbols can be written as one OpenMath object. By ways of illustration, the OpenMath representation of $\cos(1/x)$ can be seen below. Note that the mathematical symbol "cosine" has the OpenMath representation "$\cos$", which is in the CD "transc1" on transcendental functions.

```
<OMOBJ>
  <OMA>
    <OMS cd="transc1" name="cos"/>
    <OMA>
      <OMS cd="arith1" name="divide"/>
      <OMI>1</OMI>
      <OMV name="x"/>
    </OMA>
  </OMA>
</OMOBJ>
```

WebALT (Web Advanced Learning Technologies), an e-learning project funded by the EU, uses OpenMath to automatically translate mathematics exercises into natural languages. To do so, not only mathematics is encoded in OpenMath, but also text so that every sentence in the exercise can be written as one OpenMath object. To encode text in mathematics exercises in OpenMath, a mathematical lexicon and OpenMath representations for all words in this lexicon were created within WebALT. In order to translate the OpenMath object corresponding to a sentence into some natural language, grammar generation rules corresponding to the newly created OpenMath representations were formulated and an existing grammar formalism was integrated. The tool that does these automatical translations will be further described in the following section.

For creating and editing interactive multilingual math-

ematics exercises we developed the editor WExEd - WebALT Exercise Editor. Exercises created with this editor can be automatically translated to various natural languages and can be made interactive. The latter is done by making use of, for example, random variables and connecting with a Computer Algebra System (CAS) such as Mathematica [2] or Maple [3]. With random variables, a student can solve multiple instances of the same exercise and with the connection with a CAS, the student's answer can be interpreted and verified. Because of the capability of reading and understanding the student's answer, WExEd can prompt the student with a following exercise or task depending on the student's answer.

In this document we regularly speak about *sentences* or *exercise sentences*. By this we mean sentences in an exercise such as e.g. the problem statement or the predefined answers in a multiple choice exercise.

In this document we describe WExEd in Sections II, III and IV; in Section II we indicate the software WExEd makes use of, in Section III we describe its functionality and in Section IV we give the current status. Finally, in Section V we give the description of a WExEd demonstration.

II. Software Components included in WExEd

The components developed within WebALT that are integrated with WExEd are described in Section II-A and II-B. In Section II-C we describe tools developed by Technische Universiteit Eindhoven that are integrated with WExEd.

A. TextMathEditor

The TextMathEditor is a Java application that is developed for WebALT by Maths for More [7]. With this editor, one can create and edit multilingual mathematics sentences. These sentences are encoded as a single OpenMath object, i.e. both mathematics and text are encoded in OpenMath, where the text must follow a restricted grammar. Such a grammar defines sentence constructs that have a corresponding OpenMath representation in such a way that they can be automatically translated to various natural languages. The editor stores the OpenMath representation of the created sentence, with natural language generation hints as (OpenMath-defined) attributes.

When a user (e.g. an author) creates an exercise sentence, all possible words with which he can start the sentence are shown to him (Fig. 1).

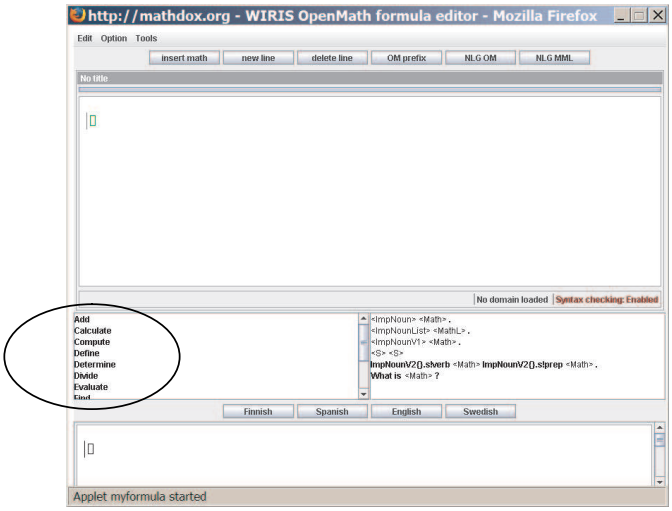

Fig. 1. TextMathEditor: a user can choose a verb to start with.

From then on, the user is given the possible words/mathematical objects with which his sentence might be continued (Fig. 2).

The output of the TextMathEditor is an OpenMath object. Suppose that the user created the sentence "Calculate the gcd of 18 and 24". Then the output of the TextMathEditor is as follows.

```
<OMOBJ>
  <OMATTR>
    <OMATP>
      <OMS cd="nlg" name="mood"/>
      <OMS cd="nlg" name="imperative"/>
      <OMS cd="nlg" name="tense"/>
      <OMS cd="nlg" name="present"/>
      <OMS cd="nlg" name="directive"/>
      <OMS cd="nlg" name="calculate"/>
    </OMATP>
    <OMA>
      <OMS cd="arith1" name="gcd"/>
      <OMATTR>
        <OMATP>
          <OMS cd="nlg" name="render"/>
          <OMS cd="nlg" name="formula"/>
        </OMATP>
        <OMI>18</OMI>
      </OMATTR>
      <OMATTR>
        <OMATP>
          <OMS cd="nlg" name="render"/>
          <OMS cd="nlg" name="formula"/>
        </OMATP>
        <OMI>24</OMI>
      </OMATTR>
    </OMA>
  </OMATTR>
</OMOBJ>
```

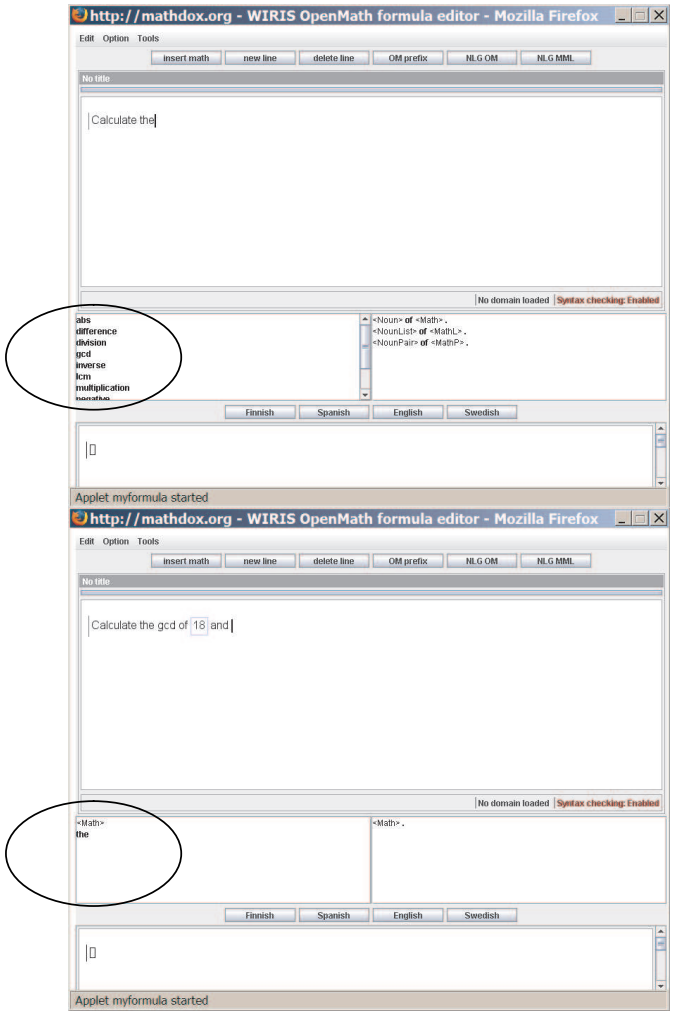

Fig. 2. TextMathEditor: a user can choose the word or mathematical object to continue.

The TextMathEditor offers additional functionalities:

- Completion grammar: the editor displays the different ways that a sentence may end. This is illustrated at the right hand side of the ovals in Figs. 1 and 2.
- Error highlighting: highlights at which part of the sentence there is an error.

The TextMathEditor is developed in such a way that the user can choose the language in which to edit the exercise sentence.

B. Natural Language Generator

The Natural Language Generator is being developed for WebALT by Lauri Carlson and his team at the University of Helsinki [5]. It makes use of the Grammatical Framework (GF) [6] which has been developed at Chalmers university (Gothenburg) by Aarne Ranta. The GF grammar environment and compiler have been implemented in Haskell. There is support for a Java runtime engine.

The Natural Language Generator generates natural language for mathematics sentences which are encoded in OpenMath. Therefore, the input of the generator is an OpenMath object representing the sentence together with a set of natural language generation hints (the output of the TextMathEditor). The output of the generator is the natural language rendering of the given sentence. The target languages for WebALT include English, Finnish, Swedish, French, Italian, Spanish, Catalan, Dutch and German.

Let us illustrate the Natural Language Generator. Suppose that the input of the Natural Language Generator is the OpenMath object illustrated above. The language generation hints say that the mood is ”imperative”, the tense is ”present” and the main directive is ”calculate”. We also see that the mathematical symbol ”gcd” is rendered as text (this means that the symbol ”gcd” is verbalized) and the integers 18 and 24 should be rendered as a formula (if not, the Natural Language Generator would generate ”eighteen” and ”twentyfour” respectively in English). This leads to the following output (given that the Natural Language Generator translates to English).

”Calculate the greatest common divisor of 18 and 24.”

C. Answer Processing Tools

When a student gives an answer to an exercise, he/she is presented with a mathematics editor that outputs an OpenMath object; the answer of the student is presented to the system in OpenMath format. To verify whether the student’s answer is correct, e.g. whether the answer is equal to the correct answer as predefined by the author of the exercise, the answer has to be sent and processed by a CAS. Because a CAS cannot read OpenMath, the OpenMath should be translated into something that the CAS can read, e.g. the CAS’ own language. This translation is done by so-called *phrasebooks*.

Technische Universiteit Eindhoven developed phrasebooks for various CAS’s like Mathematica, Maxima, Wiris and Maple. WExEd uses these phrasebooks to translate the student’s OpenMath answer to something the specified CAS can read. After the answer processing by the CAS, the result is translated back to OpenMath by the phrasebooks.

We note that phrasebooks are not only used by WExEd to process the student’s answer but also to generate e.g. the values of random variables or mathematical expressions.

III. Functionalities of WExEd

The formal language underlying WExEd is the LeActiveMath Exercise Language [8] which is a collection of XML tags together with the restrictions imposed on their use. This language is such that exercises are:

- Interactive: have means of exchanging information with the student by being able to read the student’s answer and to give feedback and/or hints.
- Automatized: have means of checking whether the student gives the correct answer by connecting for example to a CAS. The CAS can calculate the correct answer and subsequently compare the student’s answer with the correct answer.

- Adaptive: takes appropriate action depending on its knowledge of the student and/or the student's answer to a particular question; exercises are made up of *interactions* which consist of information given to the student (e.g. a question) on the one hand and information provided by the student (e.g. an answer to that question) on the other hand. Depending on the student's answer he/she is directed to a following interaction (which could be the same question, a question that will teach the student how to solve the previous question, a new question, feedback, etc.).
- Reusable: (random) variables can be defined in an exercise so that students can do multiple instances of the same exercise.

The exercise language, as any XML-language, is extensible in the sense that new tags can be added at any moment to facilitate new features.

WExEd supports multiple choice exercises and open exercises. To create multilingual exercises, the exercise author is presented with the TextMathEditor every time the input of a sentence is required (e.g. the exercise problem statement, the choices in a multiple choice exercise, feedback or hints). In this way, the exercise merely consists of OpenMath objects that can be translated to some natural language by the Natural Language Generator.

The editor is integrated with an exercise repository (an XML database containing interactive exercises) and a presentation layer; exercises in the repository can be played with the MathDox player developed by RIACA [9]. Before an exercise can be actually played, the exercise containing only OpenMath objects should be translated to some natural language. This is done by sending the exercise to the Natural Language Generator. The resulting exercise contains both mathematics (as OpenMath objects) and natural language (in the language predefined by the user).

The software developed is based on free software, namely on the XML database eXist [10] and the Orbeon Presentation-Server [11] which handles the presentation.

IV. Current Status WExEd

WExEd is still under development, which means that not all functionality is implemented yet. At this moment, the following features still need to be worked on:

- (Random) variables: an author cannot yet set variables for an exercise using the editor. At the moment, this is only possible by editing the source code of the exercise.
- Elaborate GF grammar: the GF grammar currently used in the TextMathEditor is a very simple, provisional, one. This means that the set of multilingual sentences that can be edited with the TextMathEditor is very limited; the author can only choose from a relatively small set of words. A more complex GF grammar is being developed.

V. Demonstration

In the following sections we describe a software demonstration of WExEd, as to be given at WebALT2006 [4], the first WebALT conference and exhibition.

During the demonstration we will navigate through the exercise repository surrounding WExEd to demonstrate the nested structure of folders and exercise files.

The demonstration is divided into the following three parts. In the first part, an exercise is created with the editor WExEd and played in various languages. In the second and the third part, we will play exercises that were created by editing the source code. This showcases the advanced future functionalities of WExEd.

A. Create and Play New Exercise

In this part, the functionality of WExEd and the TextMathEditor is demonstrated by creating a multilingual exercise with WExEd. We explain this part step by step and illustrate it with various screenshots.

1) We first create a new, empty, exercise in the exercise repository surrounding WExEd (Fig. 3).

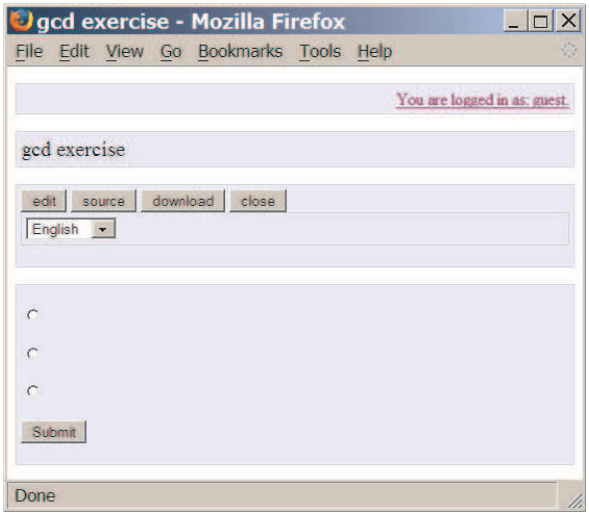

Fig. 3. WExEd: an empty exercise.

2) The exercise is opened and we choose to edit it. The editor now shows all existing interactions which is only one for an empty exercise; the default interaction *question* (Fig. 4).

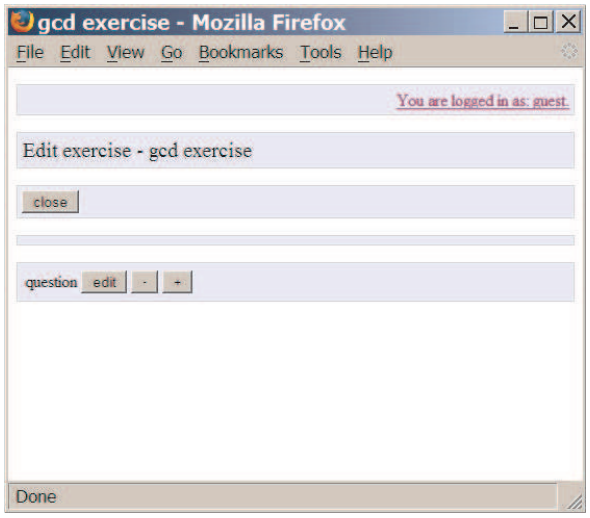

Fig. 4. WExEd: the default interaction ”question”.

3) We choose to edit this interaction.
4) We write the exercise problem statement using the TextMathEditor (Fig. 5, note that on the background the empty fields of the multiple choice exercise can be seen). This makes the exercise multilingual.

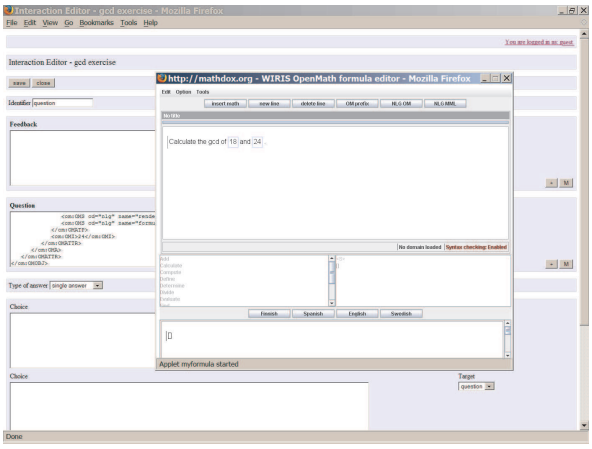

Fig. 5. WExEd: using the TextMathEditor to create the question.

The exercise we create is a multiple choice exercise and a student can be directed to different interactions for every choice. If the student answers correctly (incorrectly) then we want to send him to the ”correct” (”incorrect”) interaction. For this, we create new interactions (remember that we now only have the *question* interaction), see Fig. 6.

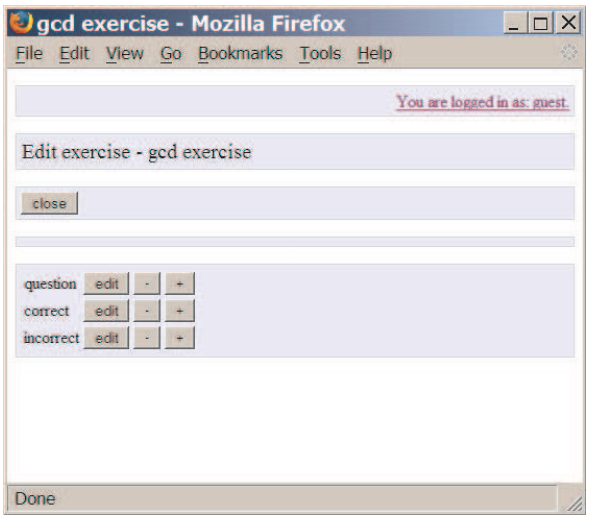

Fig. 6. WExEd: creating new interactions.

We fill in the various choices using the TextMathEditor and connect a different interaction to every choice (Fig. 7).

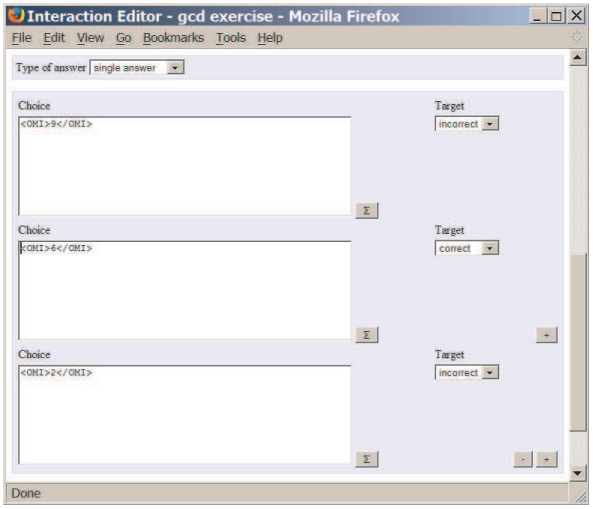

Fig. 7. WExEd: linking interactions to the multiple choices.

5) We then play the multiple choice exercise in multiple languages (Fig. 8).

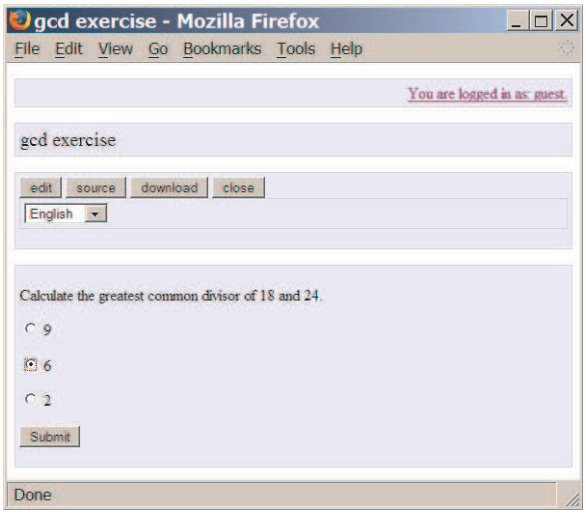

Fig. 8. WExEd: playing the multiple choice exercise.

Fig. 10. WExEd: inserting a query and linking connections.

6) We edit the exercise to make it an open exercise, which is done by changing the type of answer (Fig. 9).

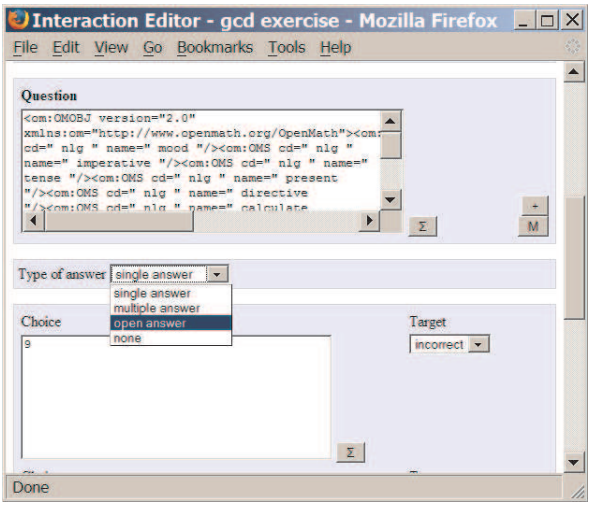

Fig. 9. WExEd: the open exercise.

Using the WIRIS math editor [7], we insert a query that determines the correct answer to this open exercise (Fig. 10).

The student can give a correct or an incorrect answer to the open exercise. We connect two different interactions to these two scenarios (see Fig. 10).

7) The open exercise is played in multiple languages (Figs. 11 and 12).

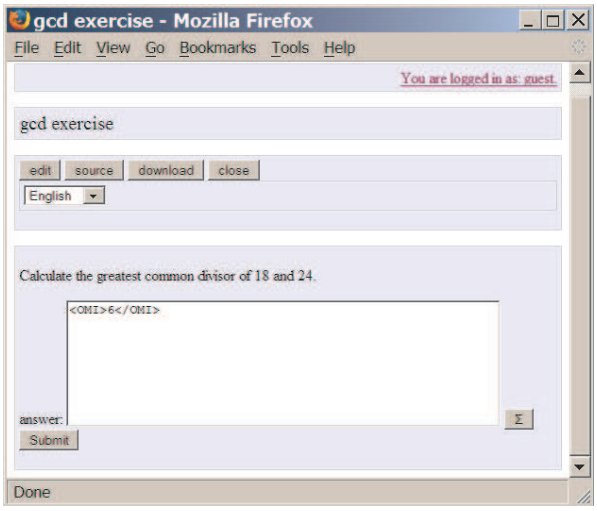

Fig. 11. WExEd: playing the open exercise.

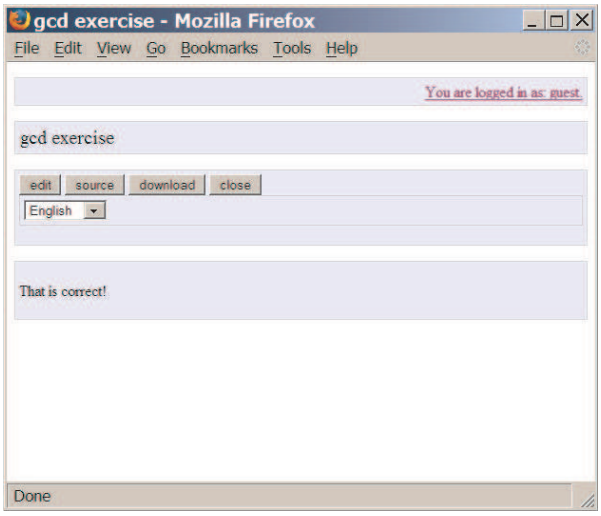

Fig. 12. WExEd: feedback to student's answer.

B. Demonstrate and Play Existing Advanced NL Exercise

In this part, the functionality of the Natural Language Generator and the future functionality of the TextMathEditor are demonstrated. This is done in the following way:

1) We start with the exercise created in the first part of the demonstration.
2) Before the conference, we change this exercise by source code editing. We only change the wording of the exercise problem statement and possibly the query defining the answer so that as a result, the exercise contains natural language that cannot be edited with the TextMathEditor but can be translated with the Natural Language Generator.
3) The exercise is played in multiple languages (at least 2).

C. Demonstrate and Play Complex Existing Advanced NL Exercise

In this last part, the richness of the LeActiveMath Exercise Language and the future functionality of WExEd are demonstrated. This is done in the following way:

1) Before the conference, we created an advanced and interactive exercise by editing the source code. The exercise will showcase future functionalities of WExEd, like random variables, media items, etc. to make the exercise richer. An example of such an exercise is one in which the student is asked to do a differentiation. If the student gives an incorrect answer then he/she is guided through the exercise step by step by many different interactions until he/she can answer the original question.
2) The exercise is played in multiple languages (at least 2).

VI. Final Remarks

In this document we described WExEd, the WebALT Exercise Editor with which interactive multilingual exercises can be created and edited, and software integrated with WExEd. As mentioned before, this is software which is not yet finished. It will be further developed since the WebALT project takes another year.

References

[1] http://www.openmath.org/
[2] http://www.wolfram.com/
[3] http://www.maplesoft.com/
[4] http://webalt.math.helsinki.fi/webalt2006/
[5] http://www.rosetta.helsinki.fi/english/index.htm
[6] http://www.cs.chalmers.se/ aarne/GF/
[7] http://www.mathsformore.com/
[8] http://www.leactivemath.org/
[9] http://www.riaca.win.tue.nl/
[10] http://exist.sourceforge.net/
[11] http://www.orbeon.com/