

AUTOMATIC GENERATION OF CLOZE QUESTIONS

Mikuláš Gangur

*Department of Economics and Quantitative Methods, Faculty of Economics, UWB University
Husova 11, Pilsen, Czech Republic*

Keywords: Quiz, LMS Moodle, Cloze question, XML, Automatic generation, XSLT, LaTeX.

Abstract: Teachers need to test their students in the most effective way i.e. they need the check to be as easy as possible, with a unique test for each student. This paper shows an example of how to generate a unique test containing cloze questions in a selected LMS. The system of cloze questions consists of a question together with an answer to it in a single task. We propose cloze questions to be generated in the XML structure. The creation procedure written in any programming language or in any appropriate environment is the basis of the generating process. The following items can be used as the procedure inputs - a task with a parameter as an input value, a problem solving function, or an XML template. The XML templates are the formulas of a proposed universal question format for the cloze type of questions according to the XML structure. In this way it is also possible to use other types of questions e.g. numeric or short answer questions. In our example we used the Matlab system to calculate the results of the given problems created from randomly generated input parameters (another environment with similar math libraries can be used in the same way). The randomly generated input parameters of the task are attached to the pattern of the task assignment. These randomly generated input parameters together with the results (calculated by means of the function solving given problem) represent the inputs into the generation process according to the chosen template. The output from this part of the generating process is a file with questions in a universal format. In the next stage we can translate the generated file with the help of the XSTL transformation rules to the appropriate style of the selected LMS. The paper shows an example of a translation into the XML Moodle format which is used for importing questions into the LMS Moodle or a translation into the LaTeX format, which is appropriate for creating questions in PDF. This format is suitable for creating both a teacher's version (with answers) and a student's version (without answers) and it even supports the test completion directly in a PDF document or HTML format suitable for web presentations.

1 INTRODUCTION

Study quizzes are an important part of any e-learning course. There are two types of quizzes – learning quizzes and test (exam) quizzes. Every teacher attempts to create automatic self-tests based on questions that stimulate practising the desired topic and contain unique questions for each student. The questions should also be the so called learning questions i.e. they should include some instructions how to solve a given problem. For instance the LMS Moodle (Moodle, 2007) offers quiz questions with the *Calculated answers*, and like this a collection of input parameters is generated. These parameters are inserted automatically into the question text and are unique for each student. Currently it is possible to enter a prescription (function) in the process of inputting the parameters to one output (problem

solution). However the above mentioned example has some limitations. Let's illustrate what we consider the main limitations of this question type:

- The collection of input parameters is generated manually i.e. the possibility of automatic random generation doesn't exist according to the given rules or a functional prescription.
- The system doesn't allow the inclusion of some more difficult functional prescriptions that would process the requested operations above input data and, in the same way, it is no possible to enter some more complicated numerical calculations (iteration etc.)
- Only one output value can be the answer to this question type

In the next described approach of the automatic generation of questions tries to solve the above

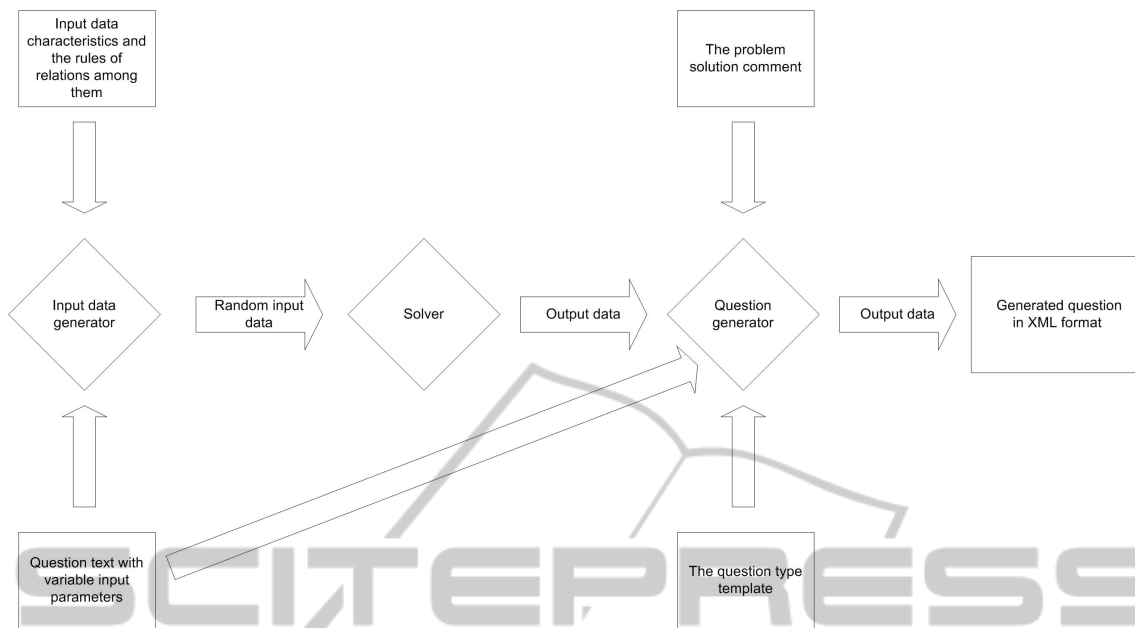


Figure 1: The process of questions generation to XML format (Source: own).

mentioned limitations. The proposed solution follows the processes published in the technical report (Fikar, 2007) and (Bakošová, 2007).

There are applications for the automated test generation, for example Random Test Generator-PRO (RTG-PRO, 2009) or Test Maestro II (TM-II, 2011). These applications as well as the majority of Learning Management Systems use the principle of randomly generated tests from the pool of questions in the question bank. The preparation of questions and building of such question bank is a difficult and time consuming job. The automation and simplification of this job is not solved in the above mentioned systems. In the paper we present a universal principle of the automatic generation of questions and the use of some particular questions that require numeric answers, the questions with short answers and especially the questions with embedded answers (cloze question). The cloze questions allow more questions of different types within one question. We propose such a structure of questions that has embedded answers (cloze question). The applied principle simplifies and streamlines the questions bank building. The outputs of the procedure can consist in creating the questions meant for import to the question bank or in creating a set of randomly selected automatically created questions for the whole test. Both the outputs can be generated within the demanded structure suitable for applied system. The proposed procedure is presented on an example of the collection questions creating. These questions are suitable for the import into the

LMS Moodle or into the PDF format as a quiz with variants both for students (without solution) and for teachers (with solution).

2 THE PRINCIPLE OF THE AUTOMATIC GENERATION OF QUESTIONS

The core of the generating system is a functional prescription of the solved problem (solver) together with the input data generator. The solver is a function with a variable number of the input parameters depending on a particular problem. The generator allows automatic generation of “suitable” input data on the basis of the designated rules. These rules describe the relations within the input data. The generator algorithm is often implemented as a backtrack procedure of the problem solution from the expected randomly generated problem results back to input data. The solver and input data generator can be implemented in any programming language. The generator output is input data collection and the output of solver is the collection of requested output data. The system, proposed in this way, allows overcoming the above mentioned limitations.

The data, generated by means of the described procedure, are used as an input to the question generator together with the template of the question text and with the structure (template) of the

universal output XML format. The generator allows processing a question requiring a numeric answer (NUM), a question with a short answer (SA) and a question with a multiple choice answer (MC). The possibility of processing a question with the embedded answer is very important. This type of question can consist of all the 3 above mentioned question types.

The question generator inserts the generated input data into the question text and then the generator inserts this text together with any possible comments to the template of the requested question and the output results, calculated by the solver as answer (problem solution), as well. In case of a NUM problem the generator inserts the answers only for one question; in case of a close question it inserts the answers for more questions. The applied approach allows to determine more answers for a NUM question with the level of validity. The output of the question generator is a XML file in the universal proposed format that is possible to transform to the format of the selected LMS by means of a particular question template.

In the paper the output to the Moodle XML format is shown. The universal output format is also suitable for the processing methods that we describe below as another possibility how to process the generated document. The whole process is described in Figure 1.

3 AN EXAMPLE OF THE GENERATION PROCESS

First of all let us present an example of a simple question with a numeric answer, a short answer or a multiple choice answer. The first input to the whole generation process is the question text with variable input parameters. An example of such input text is shown in the below listing. The variable input parameters are marked with symbols ## on both sides. In the given listing variables ##loan##, ##period2## can be found for example.

```
A property worth $##loan## is sold for
$##advance_payment## down and equal
payments at the ##end_begin## of
##period2## for the next
##number_years## years. Find the
##period4## payment if the interest
rate is
j<low_index>##interest_frequence##</low_index> = ##ratio##~%.
```

The input data generator generates the input data

and it sends them to the solver. The output data i.e. the solution of a problem and the generated input data are the results of the solver process and they are the input of the question generator together with the question type template and the comments on the problem solution. The result of the question processing is the universal XML structure of the final question. The next listing shows the main parts of one generated question. We can see the main tag <question>, tag <questiontext> with the question text and the tag <answer> with a question numeric answer. The variable input parameters in the question text were substituted by the values generated by the input data generator.

```
<question type="numerical" score="1">
  <name>
    <text>General annuity 10 - 1</text>
  </name>
  <questiontext format="html"><text>
    A property worth $19000 is sold for
    $1800 down and equal payments at the
    beginning of year for the next 5 years.
    Find the year payment if the interest
    rate is j<low_index>12</low_index> =
    3.15 %.</text></questiontext>
    ....
  <answer><answertext>3660</answertext>
  <tolerance>1</tolerance>
    ....
</answer>
  ....
</question>
```

The result of the question generator can be transformed into the requested output format in the transformation process with the help of the XSLT processor and the XSL template file according to the output format (Kosek, 2007). The next listing shows one of the possible output formats – Moodle XML format. The transformation of tag <low_index> to html tag <sub> is obvious.

```
<question type="numerical" score="1">
  <name>
    <text> General annuity 10 - 1</text>
  </name>
  <questiontext><text>A property worth
  $19000 is sold for $1800 down and equal
  payments at the beginning of year for
  the next 5 years. Find the year payment
  if the interest rate is
  j<sub>12</sub> = 3.15 %.
  </text></questiontext>
  ....
  <answer>3660
    <tolerance>1</tolerance>
  ....
  </answer>
  ....
</question>
```

4 CLOZE QUESTIONS

Cloze questions in comparison with simple questions with numeric or short answers consist of more sub questions of a simple type (numeric, short answer, multiple choices). The main control structure of the cloze questions is saved in the main control file. The following listing shows the structure of such a file. The file contains the information about the sub files with the main question text and with comments. One of the most important aims of the control file is the question type determination (numeric, multiple choices, cloze).

```
<question type="cloze">
  <text_file>Example_question.txt
</text_file>
  <text_commentary>Example_comment.txt
</text_commentary>
</question>
```

In contrast with the simple question type (numeric, multiple choice or short answer) the description of the embedded answers is a part of the submission text, too. The below listing shows two embedded answers. One of them is a multiple choice question and the second one is a numerical type question.

```
<subquestion type="multichoice"
id="1"><text>Possibilities.xml</text></
subquestion>
```

```
What is criteria value?<subquestion
type="numerical"
id="2"><text></text></subquestion>
```

The result of the generation process is the text in the universal XML format proposed by the author. The listing shows a part of this XML output text.

```
<question type="cloze" score="3"
name="no">
<name><text>Invest comparison 7b -
1</text></name>
<questiontext><text>
A company is able to borrow money
. . .
</text></questiontext>
. . .
<subquestions>
  <subquestion type="multichoice"
score="1"><name>
<text>Subquestion 1</text></name>
<questiontext format="html"><text>What
criteria are chosen for comparison?
</text></questiontext>
. . .
  <single>true</single><answers>
```

```
<answer fraction="0"><text>IRR -
Internal Interest Rate</text>
<feedback><text>...</text></feedback>
</answer>
. . .
</answers>
. . .
</subquestion>
. . .
</subquestions>
</question>
```

The question in the following example consists of the main question text and 3 sub questions. Two of them are multiple choice answers and the last one is a sub question with a numeric answer with the calculated value of the criteria. This file in the proposed XML format is the input for the following XSLT process that can transform it to a different requested format. The listing shows an example of one of such formats - Moodle XML format.

```
<question type="cloze" score="3">
<name><text>Invest comparison 7b -
1</text></name>
<questiontext>
<text>A company is able to borrow money
. . .
What criteria are chosen for
comparison? {1:MULTICHOICE:~IRR -
Internal Interest Rate#It is too
complicated if we have interest rate of
alternate investment~IRR of one
investment and then IR of other
investments#It is too complicated if we
have interest rate of alternate
investment~Future value of
investments#Future value is not
. . .
length of payments is important, too}
What is criteria value?
{1:NUMERICAL:=122536.8987:0.01}
. . .
</text>
</questiontext>
. . .
</question>
```

5 THE USE OF OTHER OUTPUT FORMATS

The proposed universal XML output format allows transforming the generated questions to different formats in the following XSLT process (see Fig 2).

We presented an example of the Moodle XML format. We will carry out the operation with the help

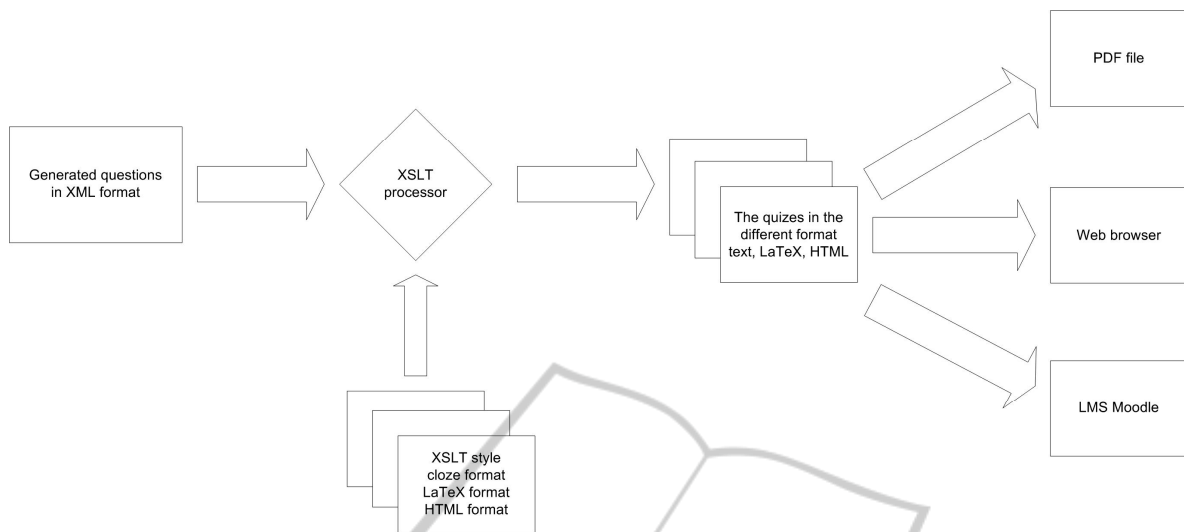


Figure 2: The follow-up transformation of the XML format to other output formats (Source: own).

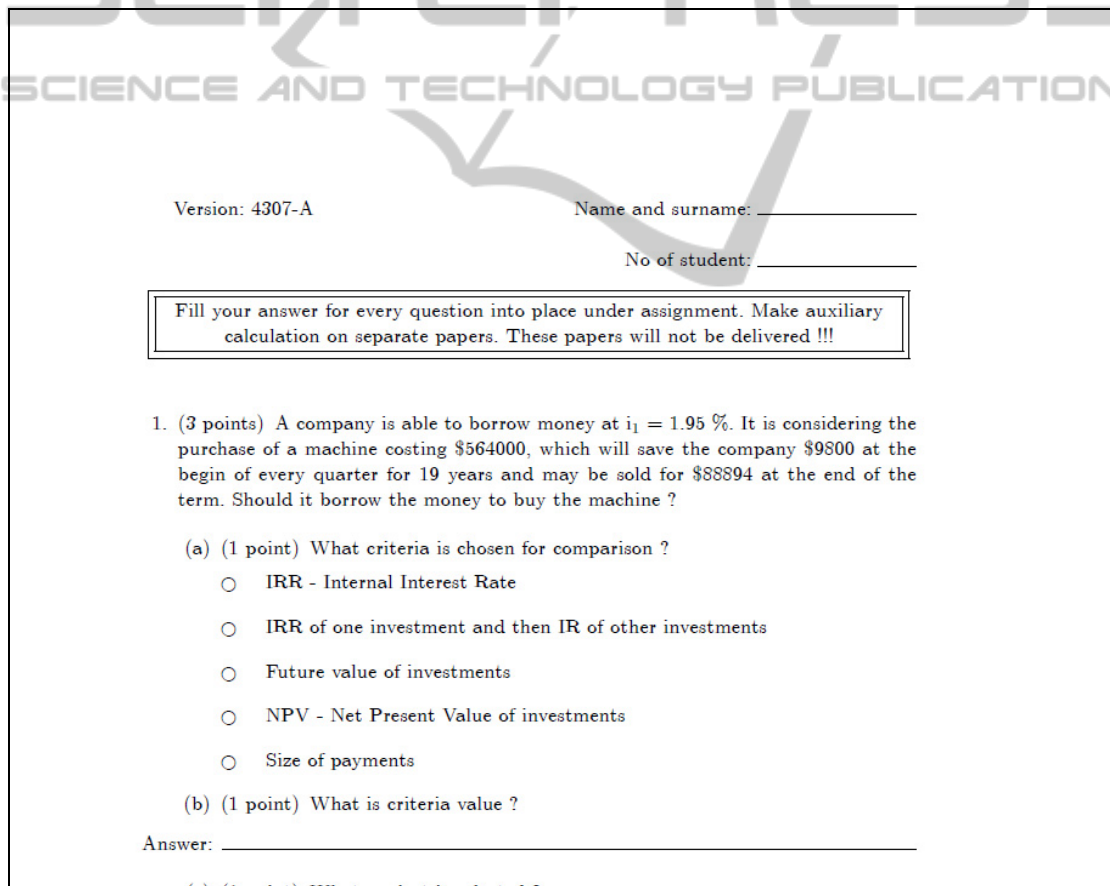


Figure 3: The final result of XSLT transformation to LaTeX (PDF) format (Source: own).

of a XSL template file for any requested format. We have just prepared some XSL template files for the Moodle XML format in Czech and English and for the LaTeX format (also in Czech and English). The

part of the final result of the LaTeX template use and the result of the following PDF transformation is shown in Fig 3.

6 CONCLUSIONS

The above described system for the automatic generation of exercises introduces a useful tool that can easily help prepare single exercises but also complete quizzes in different variants. The contribution of such a system is a possibility of using the cloze questions (exercises with embedded answers), that allow applying more than one question in one exercise. The proposed universal XML format allows that the following XSL format may be transformed to different required formats. In this way the system output can be used for different systems and different purposes. As a practical example two language templates are proposed and used for the generation of the Moodle XML format and for the import into the LMS Moodle. In the same way other two language templates for the LaTeX quiz format are used and then a final PDF quiz is generated from this LaTeX output. This quiz is prepared both as a teacher version (with results) and a student version.

The proposed XML format uses only restricted features of QTI (QTI, 2008). The construction of the described system enables an exchange between the template of the applied XML format and the template with the structure of the complex QTI format. This way the generation of a single question set in the QTI format is allowed as well as the generation of whole test in QTI format.

The above described generator was employed for example in a course of financial mathematics. 120 typified exercises were implemented and, by means of the generator, 12,000 unique exercises were prepared for practice in the LMS Moodle.

REFERENCES

- Bakošová, M., Fikar, M., Čirka, L., 2007. E-learning in course on process control. *Proceedings of eLearning Conference and Competition 2007*, pp. 191-197, Hradec Králové 2007, ISBN 978-80-7041-573-3
- Fikar, M., 2007. On Automatic Generation of Quizzes using MATLAB and XML in Control Engineering Education. Technical Report fik07xml, OIRP UIAM FCHPT STU, 2007 (on-line) (cit. 2010-10-10) Available at: http://www.kirp.chtf.stuba.sk/publication_info.php?id_pub=348
- Kosek, J., 2007 XML pro každého.(on-line) (cit. 2007-11-11) Available at: <http://www.kosek.cz/xml/index.html>
- Moodle, 2007 - A Free, Open Source Course Management System for Online Learning, (on-line) (cit. 2011-01-11) Available at: <http://moodle.org>
- RTG-PRO, 2009 Random Test Generator- PRO, (on-line) (cit. 2011-01-28) Available at: http://www.hirtlesoftware.com/p_rtgpro.htm
- TM-II, 2011 Test Maestro II Details, (on-line) (cit. 2011-01-28) Available at: <http://www.rredware.com/tmdetails.htm>
- QTI, 2008 IMS Question & Test Interoperability Specification (on-line) (cit. 2011-01-28) Available at: <http://www.imsglobal.org/question/>