UDC 004.853
Original Scientific Paper

Model-Driven Analysis towards Interoperability of Assessments in LMS

Zuzana Bizonova *zuzana.bizonova@fri.uniza.sk*
University of Zilina, Slovakia
Faculty of Management Science and Informatics

Katarina Pilatova *katarina.pilatova@fri.uniza.sk*
University of Zilina, Slovakia
Faculty of Management Science and Informatics

Abstract

In this article we are focusing on interoperability of two aspects of LMS systems - test question types and assessments as such. The proposed strategy is based on MDA, especially on the Platform Independent Models (PIM). On the higher level of abstraction (PIM) it is possible to find commonalities and differences of architectures of various systems and these are the basis for the common generalized model of the assessments. In the three steps methodology we are adding specificities of PIM models of candidate systems with various architecture – Moodle, Olat and Claroline. The correctness of the final common model (General PIM) is proved in the implemented system for exchange of tests between the existing systems.

Keywords: Model Driven Architecture, PIM, Interoperability, Assessment, LMS

1. Introduction

This contribution presents an enriched Model Driven Architecture (MDA) [9] - based approach towards interoperability of assessments and assessment questions in Learning Management Systems (LMS). Interoperability of data, especially assessments, among various LMS systems has become an important issue for the society, as these data stay isolated in the systems and cannot be re-used in others. There have already been some standards created, such as IMS QTI [1] or SCORM [6] but there are limitations to their use. There is number of highly used non-standard-based LMS systems and it is necessary to make their assessments interoperable with the rest of the tools with various architectures too. As the result of the analysis of currently widely used LMS systems, there are features of these systems that are not considered in the above mentioned standards and that need to be taken into account when exchanging the information among systems.

The proposed strategy is based on the fact that on the higher level of abstraction of the architectures of existing systems the viewer is able to see the differences and commonalities of various systems – this can be reached by creation of Platform Independent Models (PIM) as described by MDA [10]. The PIM models of assessments of existing LMS systems can be compared on the higher level of abstraction [11] and our aim is to create a common model for all the systems based on the commonalities of the systems – a generalized model. Throughout this model it is possible to transfer assessment data among different systems.

In this article we are focusing on two aspects of LMS systems – in the first part on test question types and in the second on the tests (or assessments) as such. For both of them our final goal is to create a common generalized model in three steps: in the first we are searching for the common standard that could possibly cover all the main commonalities of the assessments used in systems. In the second we are analyzing the existing systems in their higher levels of abstraction (Platform Independent Models in MDA) to see the commonalities that we add to the generalized model. Finally we create mapping rules between the real systems and the generalized models – to be able to translate them within each other.

The three existing systems used for this analysis were chosen based not only on their popularity, but mainly because their architectures were different. This way we are able to cover more points of views in the resulting generalized model.

2. Test Question Types

To create a generalized model for test questions is a demanding task. Basically there are question types that are common for most of the systems – like a true/false question or a short answer question but on the other hand, almost all the systems have a very unique and specific question types – like a KPrim question in OLAT [7] or calculated question in Moodle [3].

2.1. Choosing the most general and flexible relevant standard

Luckily, there is already a standard that covers a whole wide range of question types – IMS Question and Test Interoperability [1]. It is based on the idea that a question is a combination of certain attributes and their values (for example number of possible answers, number of correct answers, interaction type etc.). This strategy resembles to the RGB method [2] that is used in a monitor of a computer – we are able to display any colour on the monitor with a combination of a certain intensity of red, green and blue (Figure 1). In IMS QTI it is possible to define any question type with a combination of values of the attributes given by the standard.

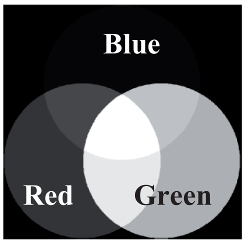

Figure 1. RGB – mixing of red, green and blue light with different intensity for displaying all kinds of colours on a monitor. Similar principle is used for definition of various question types in IMS QTI.

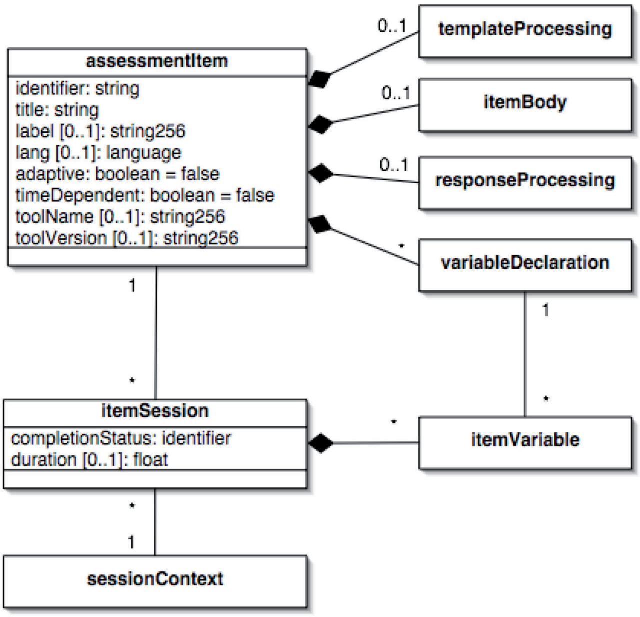

Figure 2. Assessmentitem – test question as defined in IMS QTI 2.0

As with a colour in RGB, here a question can be defined by a combination of values of the IMS QTI 2.0 attributes (Figure 2). In the following lines let us define a few question types in the terms of IMS QTI 2.0 values:

Single choice question type

```
assessmentItem. cardinality = single
assessmentItem. baseType  = identifier
assessmentItem.max = 1             //maximum choices
responseProcessing .responseType = simpleChoice
responseProcessing.interactionType = choiceInteraction
if a response is fixed: responseProcessing.fixed = true
```

Multiple choice question type

```
assessmentItem. cardinality = multiple
assessmentItem. baseType  = identifier
assessmentItem.max = (x)           //maximum choices
responseProcessing.responseType = simpleChoice
responseProcessing.interactionType = choiceInteraction
```

KPrim (OLAT) question type

```
assessmentItem. cardinality = multiple
assessmentItem. baseType  = identifiers
assessmentItem.max = 4             //maximum choices
responseProcessing.responseType = simpleChoice
responseProcessing.interactionType = choiceInteraction
```

2.2. Analysis of PIM models of candidate systems

A list of question types of candidate systems can be seen in the following table (Table 1).

Questions	**Moodle**	**OLAT**	**Claroline**
Matching	yes	no	yes
True/false	yes	yes (single choice)	yes
Short answer	yes (short answer and numerical)	yes (gap text)	yes
Multiple choice	yes	yes	yes
Essay	yes	yes (free text)	no
Particular questions	Calculated, Numerical, Random short answer	Kprim questions	nothing

Table 1. Test question types in Moodle, OLAT and Clarolin .

- **Multiple Choice -** In response to a question, the respondent chooses from multiple answers. There are two types of multiple choice questions - single answer and multiple answer.
- **Short Answer -** In response to a question, the respondent types a word or a phrase. There may be several possible correct answers, with different grades.
- **True/False -** In response to a question (that may include an image), the respondent selects from two options: True or False.
- **Matching -** A list of sub questions is provided, along with a list of answers. The respondent must "match" the correct answers with each question.

Moodle [3] has a few types of specific questions (Figure 3) that can be matched within a combination of above selected types. For example a *numerical* question is a short answer question just with a tolerance factor for answers (± n) added among the attributes. Another type - *Embedded answers* – is a passage of text that has various answers embedded within itself, including multiple choice, short answers and numerical answers – this can be specified as a combination of more question types. A *random* question or a *random short-answer* question is only defining that the question is randomly chosen from a specified set of questions.

Calculated questions – are specific for Moodle. The main purpose of a calculated question is to create multiple versions of a question with different numerical values. Calculated questions offer a way to create individual numerical questions by the use of wildcards ({x},{y}) that are substituted with random values when a quiz is taken. For example if you want to create number of "Calculate the area of the rectangle" problems to drill your students, you could create a question with 2 wildcards: {base} and {height} and put *"correct answer formula = {base} * {height}"*.

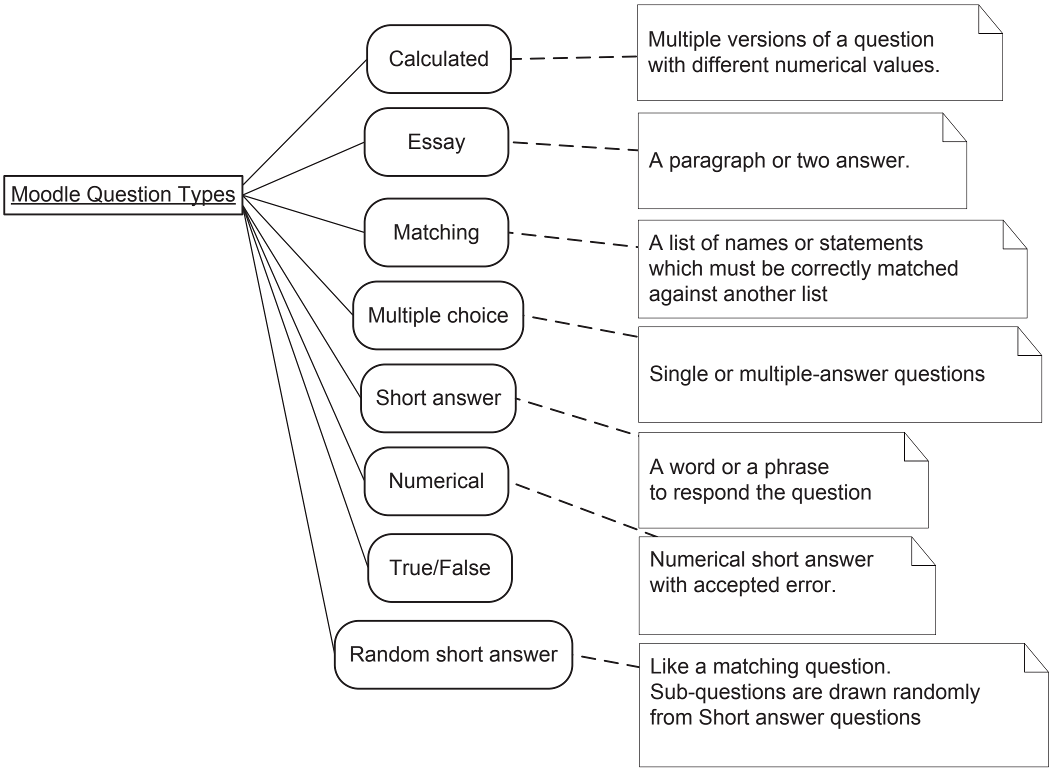
Moodle Question Types
Calculated
Essay
Matching
Multiple choice
Short answer
Numerical
True/False
Random short answer
Multiple versions of a question with different numerical values.
A paragraph or two answer.
A list of names or statements which must be correctly matched against another list
Single or multiple-answer questions
A word or a phrase to respond the question
Numerical short answer with accepted error.
Like a matching question. Sub-questions are drawn randomly from Short answer questions

Figure 3. Types of questions in Moodle quizzes.

A specific question for OLAT (Figure 3) is a KPrim question. Kprim is a very special type of multiple choice questions. It is also called k-type question.

K-type questions alway have four answer fields and the student must say for each answer whether it is correct or not. If all answers are correct the student gets the full amount of points, for three correct answers half the points otherwise no points at all.

It is used worldwide especially for medical examinations (you can find some examples on the internet [4]). Kprim questions are usefull when a student has to solve a problem where several, here four elements influence the solution.

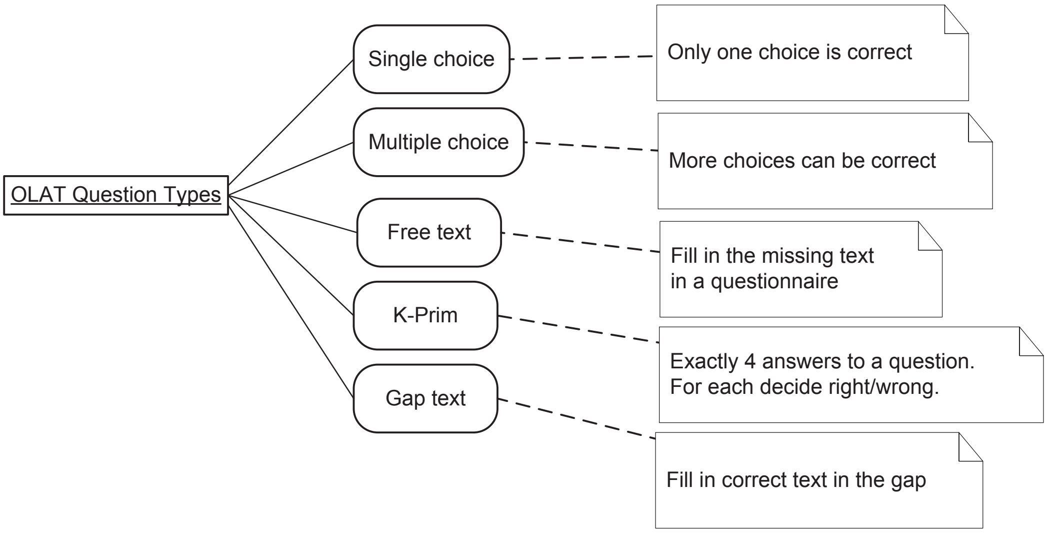
OLAT Question Types
Single choice
Multiple choice
Free text
K-Prim
Gap text
Only one choice is correct
More choices can be correct
Fill in the missing text in a questionnaire
Exactly 4 answers to a question. For each decide right/wrong.
Fill in correct text in the gap

Figure 4. Types of questions in OLAT tests.

2.3. General PIM

The base for our generalized model is the IMS QTI 2.0 standard that can mostly cover all of the common and specialized question types from candidate systems. Most of the categories and names remain the same as it was in IMS QTI 2.0 to be easily recognized in the model by the experts. Each question in the model is represented by the *G_assessmentItem* class that contains the main characteristics of the question: *a title, an identifier, the cardinality* (number of necessary responses), *baseType* (type of the response), the *image of the question*, *minimum* (lowerBound), *maximum* (upperBound) bounds and default score (*defaultValue*) for the question, text of the question (*prompt*), shuffling of the responses or not and maximum of the choices or matches for the question (*max*).

The G_correctResponse class determines the correct values of the responses. The *G_mapEntry* defines what score is added to which response to the question.

All the possible responses are listed in *G_response*, identified by an identifier, described by *responseType* (e.g. choice, text, match) and *interactionType* (e.g. choiceInteraction, orderInteraction) with a fixed predefined answer. If it is a matching question we can determine for each single response what is the maximum of matches (*max*) and to which match set it belongs to.

To get the outcome score of the responses (with their lower bound and upper bound for numerical questions) it is possible to define the particular type and score for each response in the *G_responseProcessing* class. If the particular response doesn't match any of defined types, we can set the response value to 'else' and the score to get for all the undefined responses.

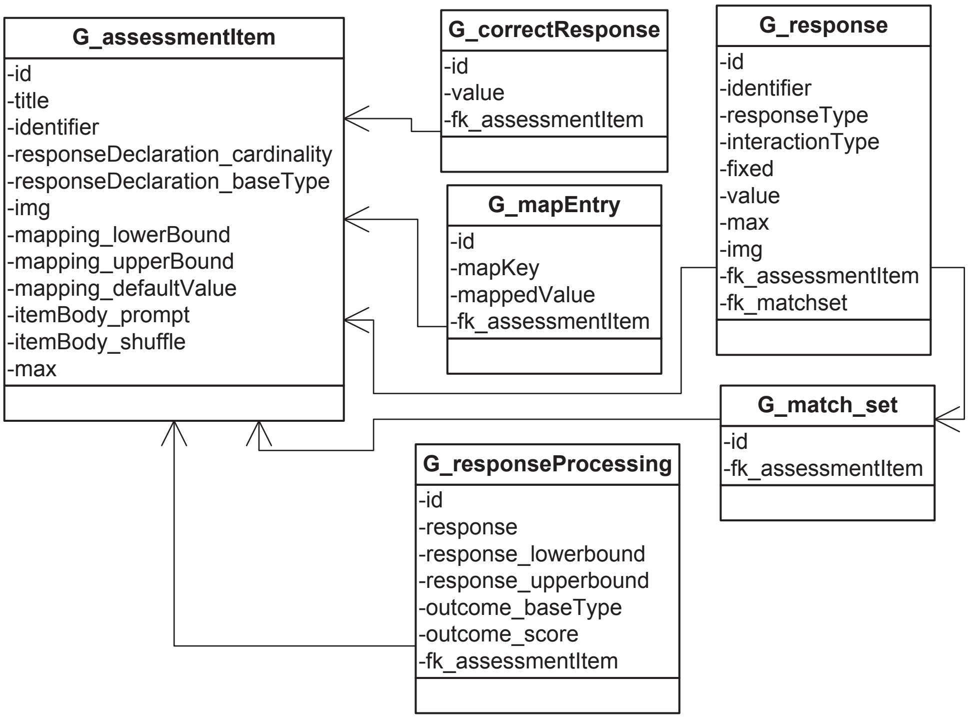

Figure 5. General PIM for tests is strongly inspired by IMS QTI 2.0

It is possible to define most of the question types of all the LMSs with categories defined in the model:

Single choice
G_assessmentItem.responseDeclaration_cardinality = single
G_assessmentItem. responseDeclaration_baseType = identifier //response.identifier
G_response.responseType = simpleChoice
G_response.interactionType = choiceInteraction
G_assessmentItem.max = 1 //maximum choices
if a response is fixed: G_response.fixed = true

Multiple choice
G_assessmentItem.responseDeclaration_cardinality = multiple
G_assessmentItem. responseDeclaration_baseType = identifier //response.identifier
G_assessmentItem.max = (x) //maximum choices
G_response.responseType = simpleChoice
G_response.interactionType = choiceInteraction

Ordering
G_assessmentItem.responseDeclaration_cardinality = ordered
G_assessmentItem. responseDeclaration_baseType = identifier //response.identifier
G_response.responseType = simpleChoice
G_response.interactionType = orderInteraction

Matching
G_assessmentItem.responseDeclaration_cardinality = multiple
G_assessmentItem.max = (x) //maximum matches
G_assessmentItem. responseDeclaration_baseType = directedPair
G_response.responseType = simpleAssociableChoice
G_response.interactionType = associateInteraction

```
G_response.max = (y)    //maximum matches for one response
G_response.fk_match_set = (id) //matching set
```

Short text entry

```
G_assessmentItem.responseDeclaration_cardinality = single
G_assessmentItem. responseDeclaration_baseType  = string
G_response.interactionType = textEntryInteraction
```

Longer text entry (essay)

```
G_assessmentItem.responseDeclaration_cardinality = single
G_assessmentItem. responseDeclaration_baseType  = string
G_response.interactionType = extendedTextInteraction
```

Numerical

```
G_assessmentItem.responseDeclaration_cardinality = single
G_assessmentItem. responseDeclaration_baseType  = float
G_response.interactionType = textEntryInteraction
G_responseProcessing.response_lowerbound = (lowerbound)
G_responseProcessing.response_upperbound = (upperbound)
```

KPrim

```
G_assessmentItem.responseDeclaration_cardinality = multiple
G_assessmentItem. responseDeclaration_baseType  = identifiers    //response.identifier
G_response.responseType = simpleChoice
G_response.interactionType = choiceInteraction
G_assessmentItem.max = 4        //maximum choices
```

2.4. Mapping rules of test questions

We can define almost all the question types of the candidate systems in the terms of the question types of the General PIM. It is only not possible to define the randomized types like calculated and random short-answer matching in this model. Instead we can use the numerical type and matching type. Following tables (Table 2, 3, 4) are describing the relations betweem General PIM question types and question types in the candidate systems.

Moodle question types	**G_model question types**
Calculated	Numerical
Essay	Longer text entry
Matching	Matching
Multiple choice	Multiple choice/Single choice
Short answer	Short text entry
Numerical	Numerical
Random short answer-matching	Matching
True/false	Single choice

Table 2. Moodle question types

OLAT question types	**G_model question types**
Single choice	Single choice
Multiple choice	Multiple choice
Free text	Longer text entry
KPrim	KPrim
Gap text	Short text entry

Table 3. OLAT question types

Claroline question types	G_model question types
Matching	Matching
Short answer	Short text entry
Multiple choice	Multiple choice
True/false	Single choice

Table 4. Claroline question types

3. Assessments

After analysis of the models for test question types it is necessary now to have a look at the whole module – the test.

3.1. Choosing the most general and flexible relevant standard

As the analysis of the question has been performed already and it has been based in IMS QTI 2.0 standard, to keep reach compatibility, our research in the test module architecture will continue in the same direction. The first step is performed by accepting IMS QTI 2.0 as the foundation of the General PIM.

3.2. Analysis of PIM models of candidate systems

3.2.1. Moodle (quiz)

The model of a test (called quiz) in Moodle consists of a *quiz* class (Figure 6) that belongs to a certain *course*. Each quiz contains a certain number of *questions* with their correct and incorrect answers. Attributes of each general *question* can be enriched by a specialized class – *question_calculated*, *question_numerical* etc.

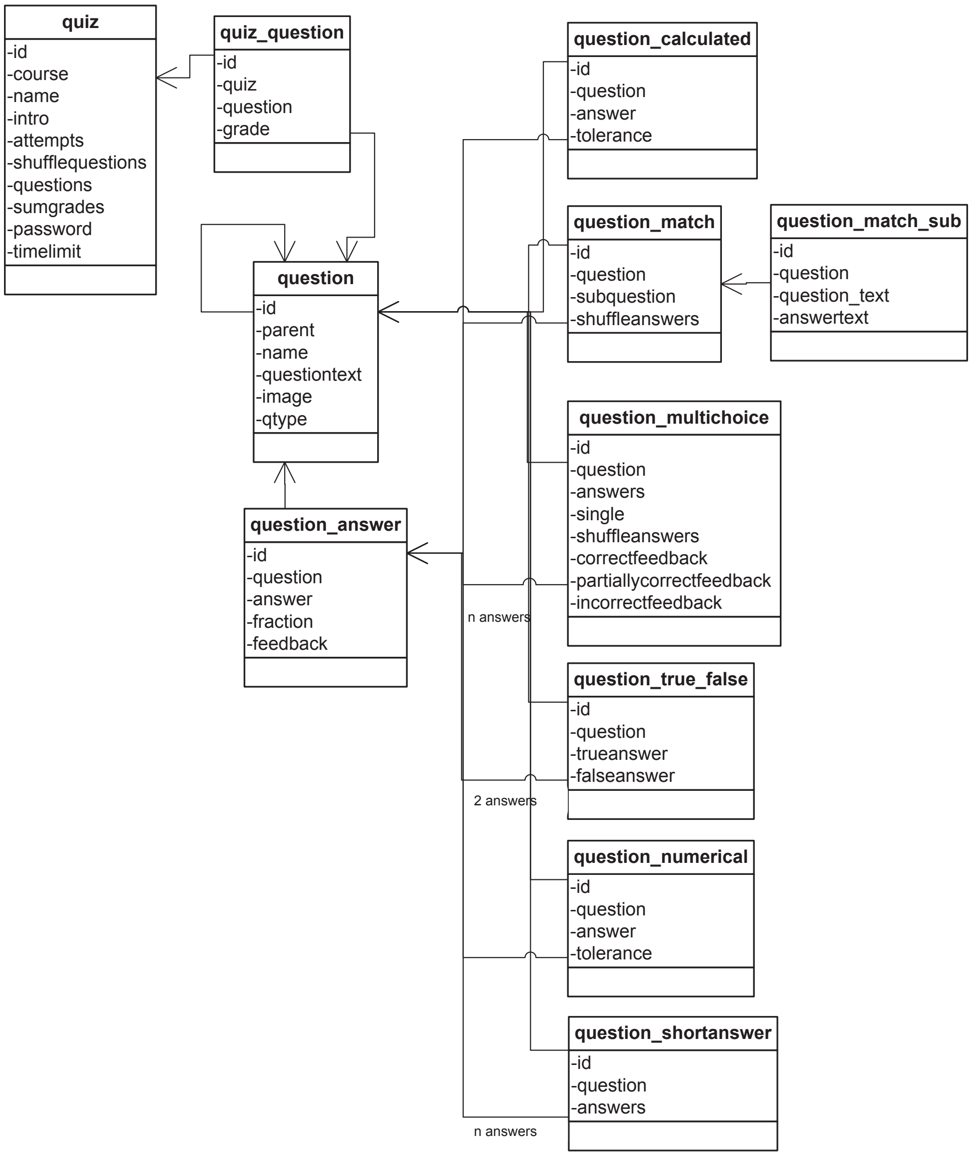

Figure 6. Quizzes in Moodle

3.2.2. OLAT

OLAT uses IMS QTI 1.2 standard for description of tests and questionnaires. The standard is different from its version 2.0 in a few elements. At the Figure 7 we can see the overview of the test defined in IMS QTI 1.2.

Each test and questionnaire in OLAT is described by a qti.xml file and it is represented in the application structure in the *OlatResourceImpl* class, where the *resid* identifier is identical with the name of the folder specified for the test (located in *OLAT/olatdata/bcroot/repository*). This folder contains a qti.xml file describing the test itself and possibly a media folder with images or sounds used in the test.

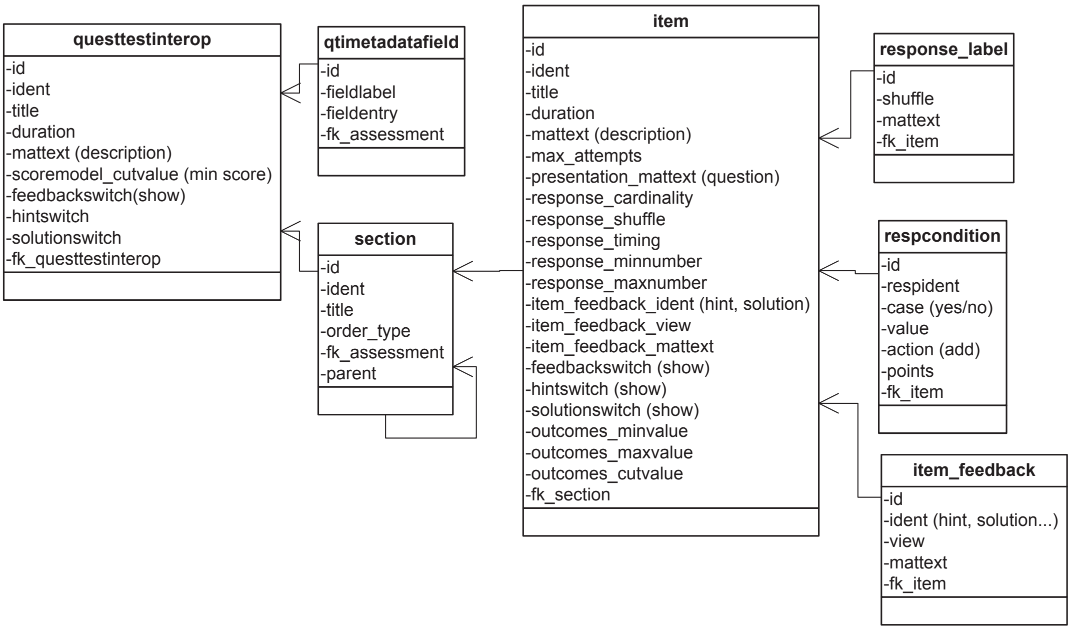

Figure 7. Tests in OLAT – IMS QTI 1.2

3.2.3. *Claroline*

The test in LMS Claroline [8] (Figure 8) is called *exercise* (*qwz_exercise*). As with every test it consists of questions (*qwz_question*) that can be of four types – true/false (*qwz_answer_truefalse*), matching (*qwz_answer_matching*), multiple choice and short answer.

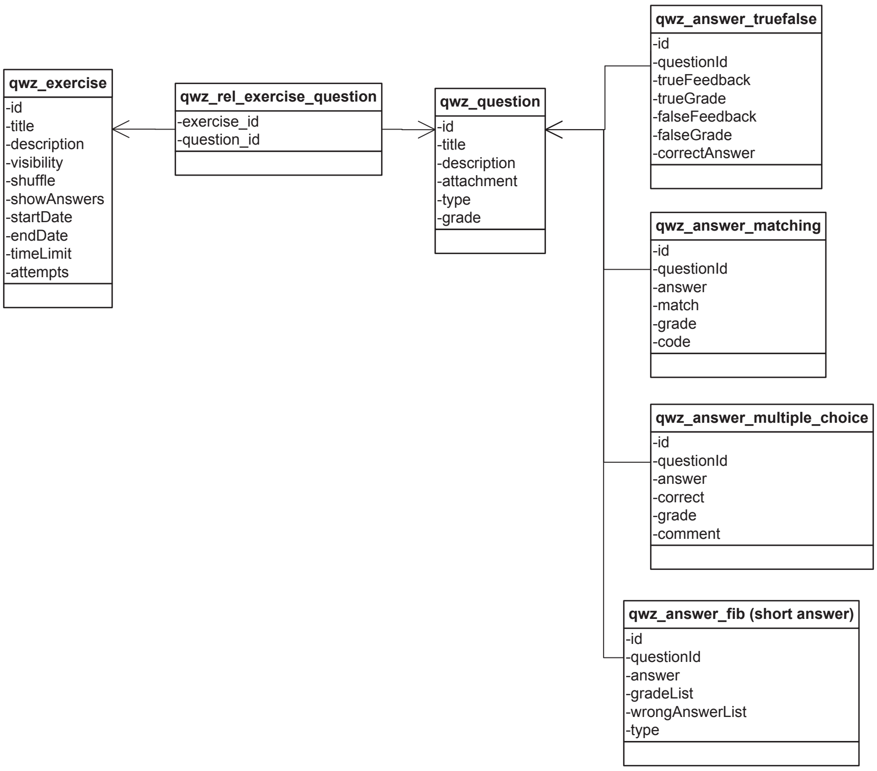

Figure 8. Testing module in LMS Claroline

3.2.4. G_model

The generalized model is strongly inspired by IMS QTI 2.0 (Figure 9) standard that defines interoperable tests, quizzes and questionnaires.

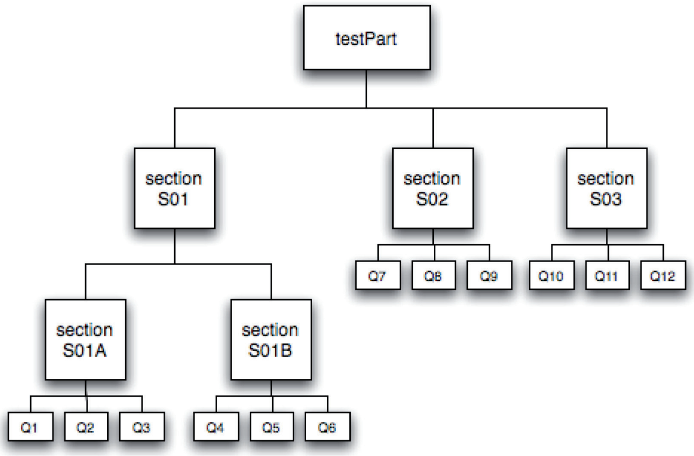

Figure 9. Sections in the test structure of IMS QTI 2.0.

All the classes are subclasses of *G_repository_entry*, therefore basic metadata like a title or a description does not need to be specified. This class contains already enough metadata about the material, therefore there is no need of defining extra metadata class as it is originally in IMS QTI (*qtimetadatafield*).

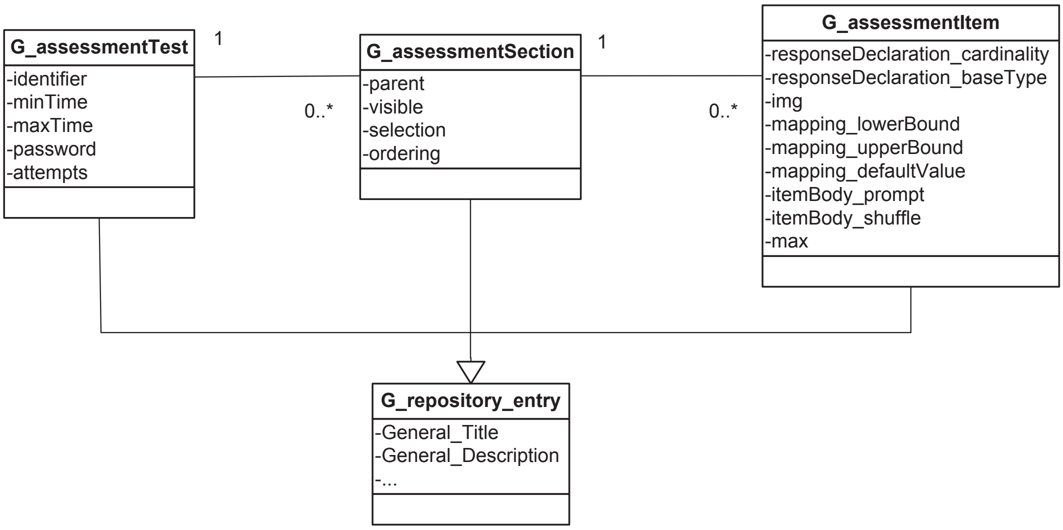

Figure 10. Test module in the General PIM (based on IMS QTI 2.0) [5]

Here is the list of the classes defined in General PIM for test module:
G_assessmentTest – the whole test is defined by this class. Unlike IMS QTI 2.0 there are a few more attributes influenced by other LMS (e.g. Moodle): *attempts, description* and *password*. These attributes are important extensions of the proposed model as these were highly used in the current systems.
G_assessmentSection – The items contained in each test are arranged into sections and sub-sections. It is possible to set *ordering* and *selection* of these items:

- *Selection* - The rules used to determine the order in which the children of the section are to be arranged for each instance of the test. If no selection rules are given we assume that all elements are to be selected.

- *Ordering* - The rule used to arrange the child elements of a section following selection. If no ordering rule is given we assume that the elements are to be ordered in the order in which they are defined.

G_assessmentItem – The questions of the test. More explanation of this class is given in the previous chapter.

3.3. Mapping rules

Mapping of the attributes from the generalized model to the models of the candidate systems for the test module are listed in the following tables – Table 5 for the tests, Table 6 for the sections and Table 7 for questions in general.

G model	**Moodle**	**OLAT**	**Claroline**
Generatl_Title	Title	name	title
Mintime/maxtime	Timelimit	duration	timeLimit
Attempts	Attempts	-	attempts
General_Description	Intro	Mattext	description

Table 5. The representation of a test: G_assessmentTest to Moodle.quiz and OLAT.questtestinterop and qwz_exercise in Claroline.

G model	**Moodle**	**OLAT**	**Claroline**
General_Title	Quiz.Title	section.Title	qwz_exercise.title
Ordering	Quiz.Shufflequestions	section.Order_type	qwz_exercise.shuffle
Parent	-	section.Parent	-

Table 6. The representation of a section: G_assessmentSection to Moodle.quiz and OLAT.section. There are no sections in Moodle itself, neither in Claroline.

G model	**Moodle**	**OLAT**	**Claroline**
General.Title	name	title	title
General_Description	questiontext	mattext	description
img	image	attachment	attachment
baseType	qtype	(defined by the combination of values in IMS QTI)	type
max	question_answer.fraction	outcome_maxvalue	grade

Table 7. The representation of a question: G_assessmentItemSection to Moodle.question, OLAT.item and qwz_question.Claroline.

Metadata that is not explicitly defined in the mapping rules can be defined in the *G_plus_metadata* list that is a part of the *G_repository_entry* class.

Concrete question types were analyzed in the previous chapter therefore here we have ended the analysis of testing modules in a generalized framework of a LMS with the general definition of a question.

4. Conclusion

In this contribution we have introduced the unique analysis of assessments and assessment questions in LMS systems with various architecture and belonging standards with the aim to create a common generalized model. This model and the mapping rules between the generalized model and existing models is a basis for transfer of the assessments between wide ranges of non-standard-based LMS systems with various architectures. The model is based on the IMS QTI model and enriched by important attributes and relations that are missing for use in the current systems. This publication not only introduces the generalized model but as well mapping rules from the three widely used LMS (Moodle, Claroline, OLAT) to this model and this way we also introduce lacking features of the former IMS QTI model and our generalized model. The main goal of this article is not to substitute the IMS QTI standard, but for example to suggest enhancements and to analyze and describe needs of the current systems and their mappings to the centralized model. Most of the LMS are not based on standards and to support interoperability we need to find ways to express their relations via a centralized model.

The same principle has been used for further aspects of LMS systems such as student results, learning objects, forums, wikis and similar and it has been published in the wider study, in a PhD thesis [12].

The proof of correctness of the results has been provided by implementation [12] of the application for the transfer of various data between tested LMS systems with various architectures – Moodle, OLAT and Claroline. Any other systems can be added to the system by description of their mapping rules into the generalized model General PIM described in this article.

This system is planned to be used in the integrated e-education system [13], [14] that is implemented in the University of Žilina to enhance the interoperability of the assessment content with other institutions and support reusability.

References

[1] IMS Global Learning Consortium Ltd. IMS Question and Test Interoperability Assessment Test, Section, and Item Information Model, Version 2.1 Public Draft revision 2 Specification

[2] Hunt, R. W. G. *The Reproduction of Colour*, 6th ed., Chichester UK: Wiley–IS&T Series in Imaging Science and Technology, 2004.

[3] Moodle LMS system. http://docs.moodle.org, downloaded: June, 13th 2007.

[4] Dr. Soto's PACU Web Didactics. http://www.thesotos.net/anesthesia/pearls/quiz.html, downloaded: July, 1st 2007.

[5] IMS Global Learning Consortium Ltd. IMS Question and Test Interoperability Assessment Test, Section, and Item Information Model, Version 2.1 Public Draft revision 2 Specification, http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_infov2p1pd2.html#element10511, downloaded: July, 11th 2007.

[6] Welsch, E. *SCORM – Clarity or Calamity?,* http://www.allbusiness.com/services/educational-services/4449495-1.html, downloaded: June, 10th 2007.

[7] Gnägi, F. *Olat 4.0 – Overview of functions*, University of Zurich http://www.olat.org/downloads/material/OLAT_4_0_Overview_of_functions_v15.pdf, downloaded: June, 24th 2007.

[8] Consortium Claroline. *Claroline .NET project*, Version 1.8.9, http://www.claroline.net/, downloaded: March, 26th 2008.

[9] Kleppe, A., Warmer, J., Bast, W. *MDA Explained The Model Driven Architecture: Practice and Promise,* Addison Wesley, 2003.

[10] Mellor, S., Scott, K., Uhl, A., Weise, D. *MDA Distilled: Principles of Model-Driven Architecture*, Addison Wesley Professional, 2004.

[11] Grob, H.L., Bensberg, F., Dewanto, B.L. Model Driven Architecture (MDA): Integration and Model Reuse for Open Source eLearning Platforms, *eLeed journal of education*, 1(1), 2005.

[12] Bizonova, Z. *Model Driven E-learning Platform Integration*, PhD thesis, University of Zilina, Slovakia, 2008, available at: http://www.etwinning.sk/MiddleI/thesis.doc, implementation: http://www.etwinning.sk/MiddleI, downloaded: October, 4th 2008.

[13] Drozdová, M., Dado,M. Integrated e-Education System Design. In: *Current Developments in Technology-Assisted Education*, pages 494 – 498, Sevilla, Spain,2006.

[14] Dado, M., Drozdová, M., Matiaško, K., Vajsová, M. E-Learning experiences at the University of Žilina. In: *Science days 2006 "Netzbasiertes Lernen im Umfeld der Telekommunikation"* , pages 72-80, Leipzig, Germany, 2006.