# INCREASED QUESTION SHARING BETWEEN ELEARNING SYSTEMS

RALPH ATTARD
http://otq.raland.net
raland@raland.net

BERNADETTE-MARIE BYRNE
School of Computer Science
University of Hertfordshire
b.m.byrne@herts.ac.uk

## ABSTRACT

*Online assessments are an integral part of eLearning systems that enhance distance and continuous education. Over two-hundred and fifty eLearning applications exist and educational institutions are often trapped with a particular vendor due to lack of test question content sharing features. This paper deals with issues related to interoperability options as well as documenting an approach to address such issues. Inspired by the JISC funded MCQFM project, an approach to the conversion of test question banks between QTI and QML formats is documented through the development of a web-based software system. Specifically, this software parses QML/QTI questions stored in XML documents, identifies their format and loads such questions in a high-level object representation of each question type, and then models an XML output in the other format.*

**Keywords:**
*eLearning, objective test questions, question sharing*

## 1 INTRODUCTION

Various eLearning systems exist that feature objective test questions in the form of exams and quizzes. Typical eLearning systems store exam questions internally using proprietary formats. The formats are also used for import and export purposes. Some of these systems also support a standard format, either Question Markup Language (QML) or Question and Testing Interoperability specification (QTI), making it a rarity that a product supports both.

This paper improves interoperability between eLearning systems through the development of a software system that aid users in migrating between systems that store question banks in different formats. The paper also identifies the test question types supported by the QML and QTI formats.

A web-based utility has been developed to support the conversion of eleven (11) objective test question types that are common to both QML and QTI formats. The software ensures full conformity to standards and enables organisations to migrate their existing test question base to a different software package without the need to re-input their questions, questions often written after much effort and thought.

## 2 ELEARNING SYSTEMS STANDARDS

Several standards and specifications exists for assessment and evaluation, most of which originated as proprietary formats and have been adopted by several parties. Agea et al (2009) have worked on a study funded by the eContentplus programme and analysed various standards, particularly the IMS QTI specification and those formats derived from proprietary tools that are in use by ICOPER participants: Moodle XML, HotPotatoes, OpenMark, Blackboard, DocBook, FML, QAML, and SuML. The study showed that the QTI de facto standard was the one that supported most question types even though its use is not highly adopted due to lack of interoperability between learning management systems.

In another study, Bennett (2007) proposes that the best way to share questions is to ditch standards, such as QTI and QML, and implement plain text representations for the five (5) most popular question types. Bennet explains that this would cover 95% of the questions which could be stored in a database with simple metadata and visualise them using

This paper is based on a master's thesis that was carried out following the award of a STEPS scholarship and has been part-financed by the European Union - European Social Fund (ESF) under Operational Programme II - Cohesion Policy 2007-2013, "Empowering People for More Jobs and a Better Quality of Life".

creative tools. Having said this, various proprietary languages exist and software houses have their reasons to use them. In the subsequent sections, formats pertinent to this project are described in detail:- QML and QTI.

## 2.1 Objective test question types

Apart from objective test question types supported by proprietary formats, most question types are either supported by QTI, QML or both. The Questionmark's manual highlights twenty (20) types of questions in QML: captivate/robodemo, drag-and-drop, essay, explanation, fill in banks, hotspot, flash, likert scale, matching, matrix, multiple choice, multiple response, numeric, pull-down list, ranking, select a blank, spoken response, text match, true/false, and yes/no questions. Further to this, the QTI specification highlights twenty (20) types of questions: true/false, single response, multiple response, order, associate, match, gap match, inline choice, text entry, extended text, hot text, hot spot, select point, graphic order, graphic associate, graphic gap match, position object, slider, drawing, and upload questions.

Notwithstanding that the above mentioned lists are quite long, some question types are common to both QML and QTI, and therefore overlap. It is also true that some of the questions types are the same amongst both formats even though they carry different names. With this in mind, the following section consolidates a common list of question types.

## 2.2 Standards and common question types

QML is a proprietary language that originated in 1998 as part of the Questionmark's flagship product Perception. This product supported a large number of question types on its own; however in 1999 the IMS Global Consortium started working on QTI which was based on QML. At first, Questionmark put in their experience in this new language to create an assessment specification, QTI, which can be adopted by all those working electronic assessments, however this evolved over the years to cater for various variants and new question types to be more complete. The following eleven (11) question types are common to both formats: Extended Text Questions, Upload Questions, Inline Choice Questions, Matching Questions, Multiple Response Questions, Position Object Questions, Select Point Questions, Sequencing Questions, Single Response Questions, Text Entry Questions, True/False (Yes/No) Questions**.**

## 2.3 Differences in XML formats

Below is an example of how XML samples have been compiled for each question type in both formats to serve as blue-prints for the software. Elements common to both formats have been highlighted and numbered for easy referencing.

### 2.3.1 QML

```
<QUESTION ID="4923048238058661⓪" DESCRIPTION="Sample: multiple response①" TOPIC="new"
STATUS="Normal">
  <CONTENT TYPE="text/html"><![CDATA[<SPAN style="FONT-FAMILY: 'Courier New'; FONT-
SIZE: 8pt; mso-fareast-font-family: 'Times New Roman'; mso-ansi-language: EN-GB; mso-
fareast-language: EN-US; mso-bidi-language: AR-SA; mso-no-proof: yes">Who of the
following became a president of Malta?⑥</SPAN>]]></CONTENT>
  <ANSWER QTYPE="MR" SHUFFLE="YES⑤" SUBTYPE="VERT" MAXRESPONSE="4">
    <CHOICE ID="ChoiceA⑦">
      <CONTENT TYPE="text/html"><![CDATA[George Abela⑦]]></CONTENT>
    </CHOICE>
    <CHOICE ID="ChoiceB⑧">
      <CONTENT TYPE="text/html"><![CDATA[Guido de Marco⑧]]></CONTENT>
    </CHOICE>
    <CHOICE ID="ChoiceC⑨">
      <CONTENT TYPE="text/html"><![CDATA[Edward Fenech Adami⑨]]></CONTENT>
    </CHOICE>
    <CHOICE ID="ChoiceD⑩">
      <CONTENT TYPE="text/html"><![CDATA[Ralph Attard⑩]]></CONTENT>
    </CHOICE>
  </ANSWER>
  <OUTCOME ID="ChoiceA George Abela" ADD="1②" CONTINUE="TRUE">
    <CONDITION>"ChoiceA②"</CONDITION>
    <CONTENT TYPE="text/html"><![CDATA[Became president in 2009]]></CONTENT>
  </OUTCOME>
```

```
  <OUTCOME ID="ChoiceB Guide de Marco" ADD="1③" CONTINUE="TRUE">
    <CONDITION>"ChoiceB③"</CONDITION>
    <CONTENT TYPE="text/html"><![CDATA[Became president in 1999]]></CONTENT>
  </OUTCOME>
  <OUTCOME ID="ChoiceC Edward Fenech Adami" ADD="1④" CONTINUE="TRUE">
    <CONDITION>"ChoiceC④"</CONDITION>
    <CONTENT TYPE="text/html"><![CDATA[Became president in 2004]]></CONTENT>
  </OUTCOME>
  <OUTCOME ID="ChoiceD Ralph Attard" ADD="0" CONTINUE="TRUE">
    <CONDITION>"ChoiceD"</CONDITION>
    <CONTENT TYPE="text/html"><![CDATA[Never.]]></CONTENT>
  </OUTCOME>
</QUESTION>
```

### 2.3.2 QTI

```
<assessmentItem identifier="4923048238058661⓪" title="Sample: multiple response①"
adaptive="false" timeDependent="false">
  <responseDeclaration identifier="RESPONSE" cardinality="multiple"
baseType="identifier">
    <correctResponse>
      <value>ChoiceA②</value>
      <value>ChoiceB③</value>
      <value>ChoiceC④</value>
    </correctResponse>
  </responseDeclaration>
  <outcomeDeclaration identifier="SCORE" cardinality="multiple" baseType="identifier"/>
  <itemBody>
    <choiceInteraction responseIdentifier="RESPONSE" shuffle="true⑤" maxChoices="3">
      <prompt>Who of the following became a president of Malta?⑥</prompt>
      <simpleChoice identifier="ChoiceA⑦" fixed="false">George Abela⑦</simpleChoice>
      <simpleChoice identifier="ChoiceB⑧" fixed="false">Guido de Marco⑧</simpleChoice>
      <simpleChoice identifier="ChoiceC⑨" fixed="false">Edward Fenech
Adami⑨</simpleChoice>
      <simpleChoice identifier="ChoiceD⑩" fixed="false">Ralph Attard⑩</simpleChoice>
    </choiceInteraction>
  </itemBody>
</assessmentItem>
```

## 2.4 Software System

The software parses questions stored in XML documents, identifies their format – QTI or QML – and loads such questions in a high-level object representation of each question type, and then models an XML output in the other format. This has been developed as a class library using C#.NET based on Microsoft .NET Framework 3.5 and exposed as an ASP.NET web application.

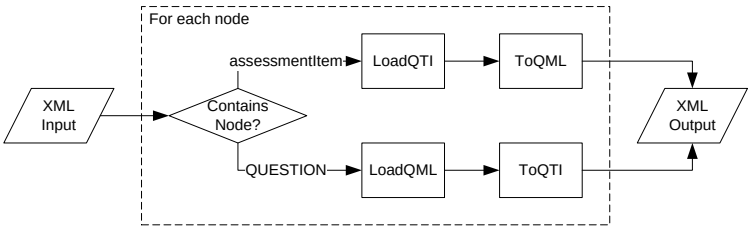



**Figure 1: Process Flow**

### 2.4.1 High-level objects

The below high-level class diagram shows the inheritance hierarchy together with the object associations and data contracts that could be used instead of the generic option and response objects. This object-oriented (OO) approach was chosen to ease development and provide consistency. An interface class was used to define required methods and have been applied to an abstract class. The purpose of having a non-implementable abstract class was to define data members and methods common to all those classes that inherit it, whilst that those methods imposed by the interface and vary per child were not implemented. This abstract class also included two generic objects as data members which were exposed through virtual properties, and therefore allows children to change their exposure by redefining their properties to use custom data contracts.

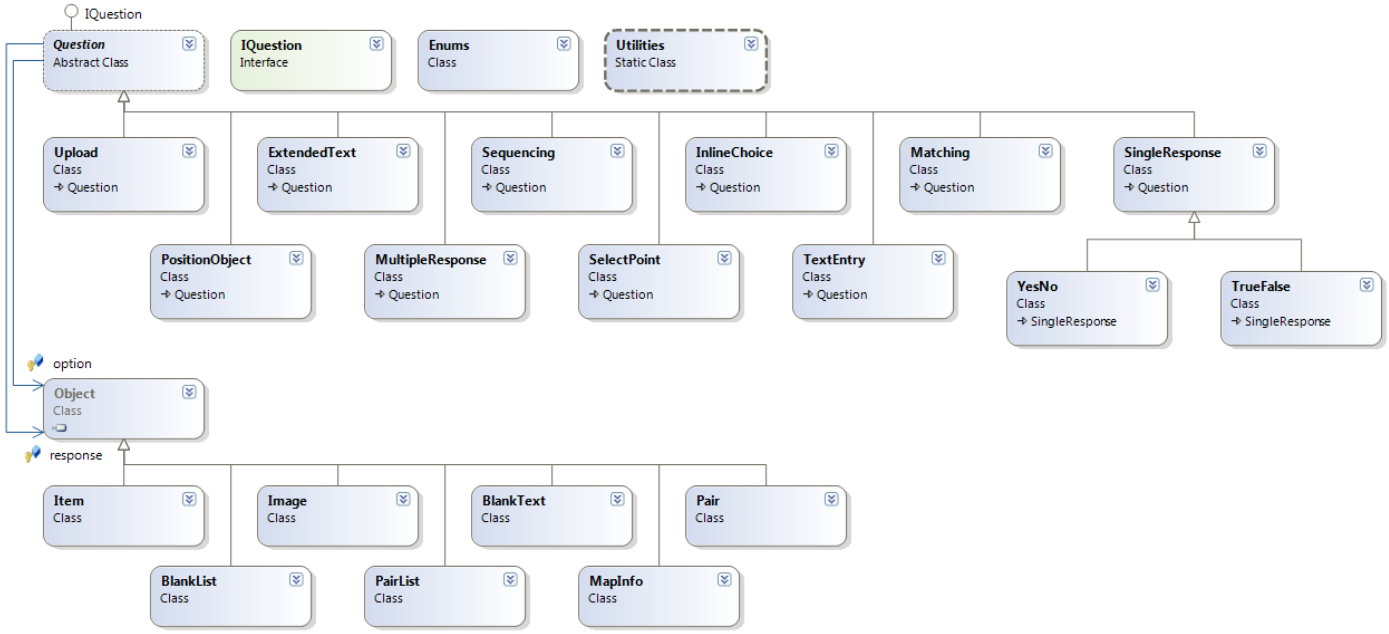



**Figure 2: Class Diagram: High-level**

### 2.4.2 Data Contracts

Seven classes that resemble custom complex data types have been created. Each data member has a `private` access modifier and is exposed through a `public` property that serves as its getter and setter. These data contracts are used to make up classes that represent each objective test question.

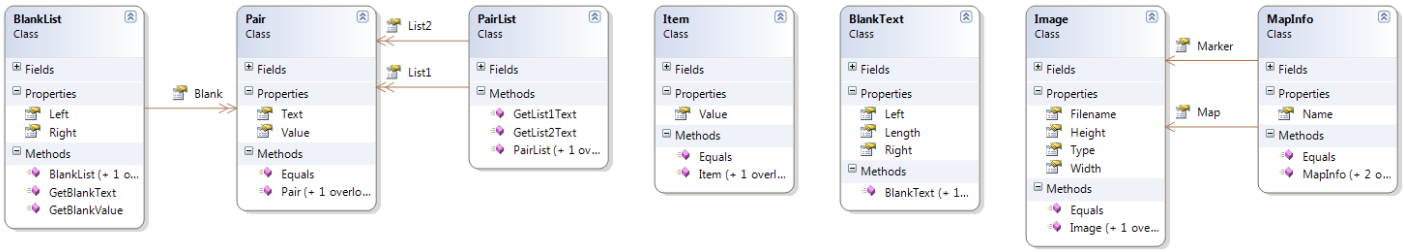



**Figure 3: Class Diagram: Contracts**

### 2.4.3 Question Objects

Fourteen classes, two of which are an abstract class and an interface, are used to represent the chosen question types. This object oriented approach was chosen to provide a high-level object that is valid for all possible objective test question types. Inheritance is used to customise the structure through method overriding whilst allowing the necessary flexibility by parameter hiding to vary the data type of the option and response data members according to the type of question data.

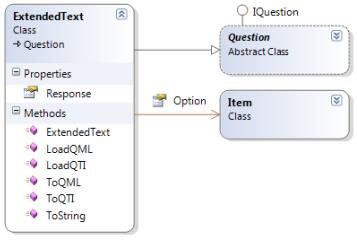



**Figure 4: Class Diagram: ExtendedText**

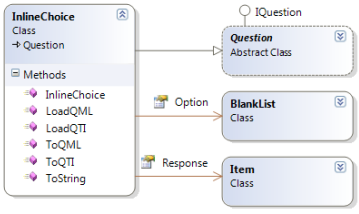



**Figure 5: Class Diagram: InlineChoice**

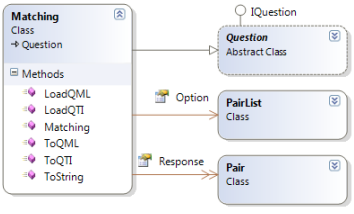
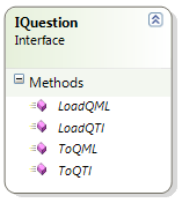



**Figure 6: Class Diagram: Matching**



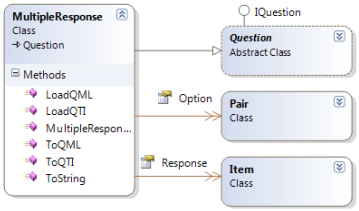




**Figure 8: Class Diagram: MultipleResponse**

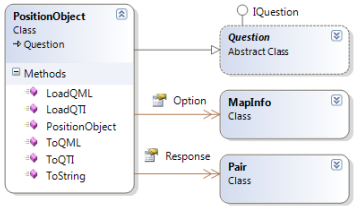



**Figure 9: Class Diagram: PositionObject**

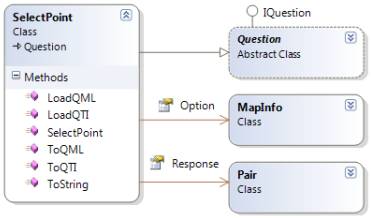



**Figure 10: Class Diagram: SelectPoint**

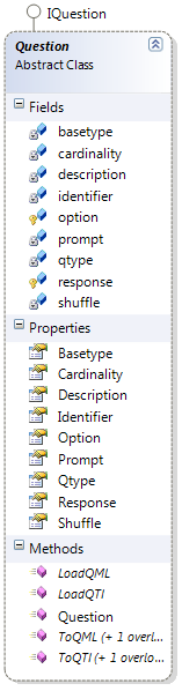



**Figure 7: Class Diagram: Question**

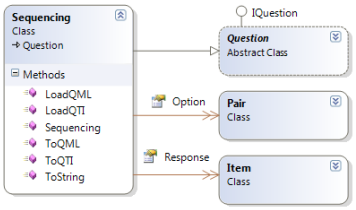



**Figure 11: Class Diagram: Sequencing**

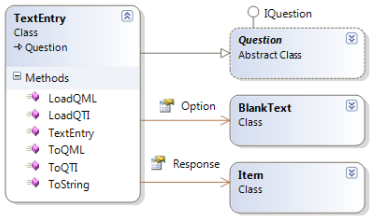



**Figure 12: Class Diagram: TextEntry**

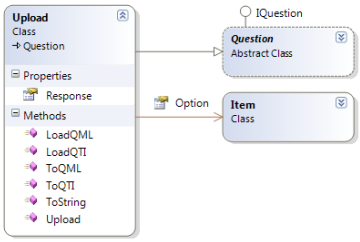



**Figure 13: Class Diagram: Upload**

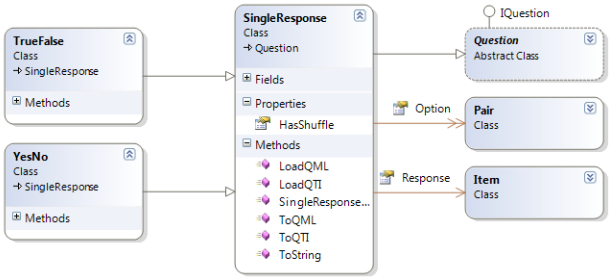



**Figure 14: Class Diagram: SingleResponse, TrueFalse, YesNo**

## 3 CONCLUSION AND FUTURE WORK

This paper showed a way to further interoperability options within eLearning systems to promote question sharing. The software produced demonstrates a simple approach to conversion of objective test questions. Test question banks are parsed from QML and QTI documents and stored into high-level object models. Once the models are filled, their content can be outputted in different formats. Currently, the software only supports the conversion of questions and answers between two standard formats. This could be extended both in a vertical and horizontal manner. Vertical expansion can cater for the inclusion of marking/scoring elements to enable automatic marking when questions are migrated between eLearning systems, whilst that horizontal expansion can provide support for other proprietary standards to further increase interoperability between eLearning systems. The only skills required to maintain this type of approach are object-oriented skills to build the models, and XML manipulation to parse and construct documents.

## 4 REFERENCES


- AGEA, A. et al, 2009, *Analysis of existing specifications and standards for assessment and evaluation and their usage in Europe*, Educanext, http://www.educanext.org/dotlrn/clubs/icoper/new-lors/Deliverables/Deliverables_-_Submitted/D6.1/D6.1.final.doc, [Accessed 09/08/2010].
- BENNETT, S., 2007, *MCQFM: Towards a Web 2.0 Approach to Objective Testing*, e-Learning Focus, http://www.elearning.ac.uk/features/mcqfm, [Accessed 13/05/2010].
- IMS GLOBAL, 2006, *IMS Question & Test Interoperability* Specification, IMS Global Learning Consortium, http://www.imsglobal.org/question, [Accessed 14/05/2010].
- QUESTIONMARK, 2005, *Perception Version 4, Concepts of Authoring*, University of Southampton, http://www.southampton.ac.uk/isolutions/computing/elearn/CAA/Perception/v44_concepts_of_authoring.pdf. [Accessed 17/08/2010].