R2Q2: Rendering and Reponses Processing for QTIv2 Question Types

Gary Wills[‡], Hugh Davis[‡], Swapna Chennupati[‡], Lester Gilbert[‡], Yvonne Howard[‡], Ehtesham-Rasheed Jam[‡], Steve Jeyes[†], David Millard[‡], Robert Sherratt[†] and Gavin Willingham[‡]

[‡]Learning Technologies Group, University of Southampton, UK.

[†]e-Services Integration, University of Hull, UK

Abstract

IMS QTI is a popular and important standard for e-learning assessment. The second version of the standard (QTIv2) works alongside other IMS standards, but take-up has been slow, with problematic implementations and no definitive reference software. The R2Q2 project aims to produce a set of loosely coupled web services that will provide definitive reference software for QTIv2. In this paper we describe how we have learnt from previous development efforts in order to produce a first architecture and initial implementation, we also describe the results of our interviews with the wider QTI community to identify what is believed to be important for our planned final reference implementation.

Introduction

E-learning assessment covers a broad range of activities involving the use of machines to support assessment, either directly (such as web-based assessment tools, or tutor systems) or indirectly by supporting the processes of assessment (such as quality assurance processes for examinations). It is an important and popular area within the e-learning community [6, 1, 2] Within this broad view of e-learning assessment, the domain appears established but not mature, as traditionally there has been little agreement on standards or interoperability at the software level. Despite significant efforts by the community, many of the most popular software systems are monolithic and tightly coupled, and standards are still evolving.

One of the more popular standards that has emerged is Question and Test Interoperability (QTI) developed by the IMS Consortium[1]. The QTI

[1] IMS QTI homepage: http://www.imsglobal.org/question/

specification describes a data model for representing questions and tests and the reporting of results, thereby allowing the exchange of data (item, test, and results) between tools (such as authoring tools, item banks, test constructional tools, learning environments, and assessment delivery systems) [10]. Wide take-up of QTI would facilitate not only the sharing of questions and tests across institutions, but would also enable investment in the development of common tools. QTI is now in its second version (QTIv2), designed for compatibility with other IMS specifications, but despite community enthusiasm there have been only a few real examples of QTIv2 being used, and no definitive reference implementation [8,9].

In the last few years there has been a trend away from tightly coupled monolithic systems towards Service-Oriented Architectures (SOA). SOAs are an attempt to modularise large complex systems in such a way that they are composed of independent software components that offer services to one another through well-defined interfaces.

One way to promote QTIv2 is through a reference implementation of the standard written within the service-oriented paradigm. In the UK, the Joint Information Systems Committee (JISC) is financed by all the Further and Higher Education funding councils within the country, and is responsible for providing advice and guidance on the use of Information and Communications Technology (ICT) for learning and teaching. Part of their strategy is the development of a SOA framework for e-learning [5,7], and of reference models that describe how different areas of e-learning can be supported by the framework.

For the assessment domain, the reference model is FREMA (Framework Reference Model for Assessment)[2]. The FREMA project has defined a number of high level service profiles that describe how services can work together within the assessment domain to fulfil particular use cases [4]. Several of these use cases require questions to be rendered, answers taken, and feedback to be generated. The corresponding services provide an ideal opportunity to create a reference implementation of the core functionality of QTIv2 that fits within the broader FREMA context.

This paper will report on the progress of the R2Q2 project. R2Q2 is a JISC funded project that aims to bring the SOA approach and QTI standard together to develop a set of Web Services that will render and respond to questions written in the QTI standard.

Service Oriented Architectures

A service approach is ideally suited to more loosely coupled systems, where individual parts may be developed by different people or organizations. Wilson *et al.* [7] discuss in detail the advantages of using a SOA:

---

[2] FREMA homepage: http://www.frema.ecs.soton.ac.uk/

- **Modularity**: As services are dynamically coupled, it is relatively easy to integrate new services into the framework, or exchange new implementations for old.

- **Interoperability**: Due to standardization of the communication and description of the services, third party services can easily be incorporated as required.

- **Extensibility**: Due to the relative ease with which services can be incorporated into a system, there is less danger of technology 'lock-in'.

Due to the nature of the loose coupling in a SOA, applications can be developed and deployed incrementally. In addition, new features can be easily added after the system is deployed. This modularity and extensibility make SOA especially suitable as a platform for an assessment system with evolving requirements and standards. Services are also appealing in terms of their ability to be reused, as they have well-defined public interfaces. In R2Q2 we will be developing web services that are built on widely used standards such as SOAP and WSDL. It is our hope that this will make it easy for other members of the community to use the services, and further develop them.

Question and Test Interoperability

The IMS QTI Specification is a standard for representing questions and tests with a binding to the eXtended Markup Langage (XML, developed by the W3C) to allow interchange. Figure 1 shows a short example of a question expressed in this format, taken from the IMS QTI examples. This example is a simple multiple choice question, illustrating the core elements: *ItemBody* declares the content of the question itself, *ResponseDeclaration* declares a variable to store the student's answer, and *OutcomeVariables* declares other resulting variables, in this case a score variable to hold the value of the result.

In R2Q2 we focus on rendering and responding to the 16 different types of interactions described in version 2 of the QTI specification (QTIv2). These are:

1) Choice
2) Order
3) Associate
4) Match
5) Inline Choice
6) Text Entry
7) Extended Text
8) Hot Text
9) Hotspot
10) Select point
11) Graphic
12) Graphic Order
13) Graphic Associate
14) Graphic Gap Match
15) Position object
16) Slider

```
<?xml version="1.0" encoding="UTF-8"?>
<assessmentItem xmlns="http://www.imsglobal.org/xsd/imsqti_v2p0"
    identifier="choice" title="Unattended Luggage"
    adaptive="false" timeDependent="false">
    <responseDeclaration identifier="RESPONSE" cardinality="single"
                         baseType="identifier">
        <correctResponse>
            <value>ChoiceA</value>
        </correctResponse>
    </responseDeclaration>
    <outcomeDeclaration identifier="SCORE" cardinality="single"
                        baseType="integer">
        <defaultValue>
            <value>0</value>
        </defaultValue>
    </outcomeDeclaration>
    <itemBody>
        <p>Examine the following sign:</p>
        <p>
           <img src="images/sign.png" alt="NEVER LEAVE LUGGAGE UNATTENDED"/>
        </p>
        <choiceInteraction responseIdentifier="RESPONSE"
                           shuffle="false" maxChoices="1">
            <prompt>What does it say?</prompt>
            <simpleChoice identifier="ChoiceA">You must stay with your
                 luggage at all times.</simpleChoice>
            <simpleChoice identifier="ChoiceB">Do not let someone else look
                 after your luggage.</simpleChoice>
            <simpleChoice identifier="ChoiceC">Remember your luggage when
                 you leave.</simpleChoice>
        </choiceInteraction>
    </itemBody>
    <responseProcessing template =
    "http://www.imsglobal.org/question/qti_v2p0/rptemplates/match_correct"/>
</assessmentItem>
```

Figure 1: Example QTIv2 question (abridged for simplicity)

The list of different question types can be combined with templated question or adaptive response, providing an author with numerous alternative methods for writing questions appropriate to the needs of the students. Templated questions include variables in their item bodies that are instantiated when a question is rendered (for example, inserting different values into the text of maths problems). Adaptive questions have a branching structure, and the parts that a student sees depends on their answer to each part of the branch. In total these allow for sixty four different possible combinations.

Previous Work

One of the earliest successful projects in the area of rendering and response using the QTI standard was the Assessment Provision through Interoperable Segments project (APIS) [8]. This was later reused in the ASSIS project as a core service, called *QTIRun* [9]. The APIS project aimed to implement a modular item rendering engine in line with QTIv2. Whilst the APIS and ASSIS projects have provided a launch pad from which many other projects have benefited, there are a number of short-comings in their final implementations.

- *QTIRun* is implemented as a single Web Service, and in order to preserve the statelessness of the render/response functions, the service calls pass excessively large amounts of XML data around the system.
- Despite this the interactions between services remained tightly coupled, compromising extensibility, such that if a different render engine was required (or a different response engine) the code would have to be re-written.
- A lack of documentation has resulted in confusion over the type of QTIv2 questions served by *QTIRun*. In fact the *QTIRun* service only deals with a limited subset of QTIv2 question types.

The aim of the R2Q2 project is to learn from the experiences of APIS and ASSIS and produce a genuinely loosely coupled SOA for flexibility and extensibility. The project uses an agile software engineering methodology in which every stage is carefully documented, the main points of which are published weekly on the project website in the form of a blog.

R2Q2 Design

The first stage of the design was to examine what had been built before in the APIS project and identify the lessons described above. Also in the initial stages of the project we interviewed people outside of the project team who are actively developing Web services for assessment and/or developing the QTI specification. An overview of the services the R2Q2 system will provide is shown in Figure 2.

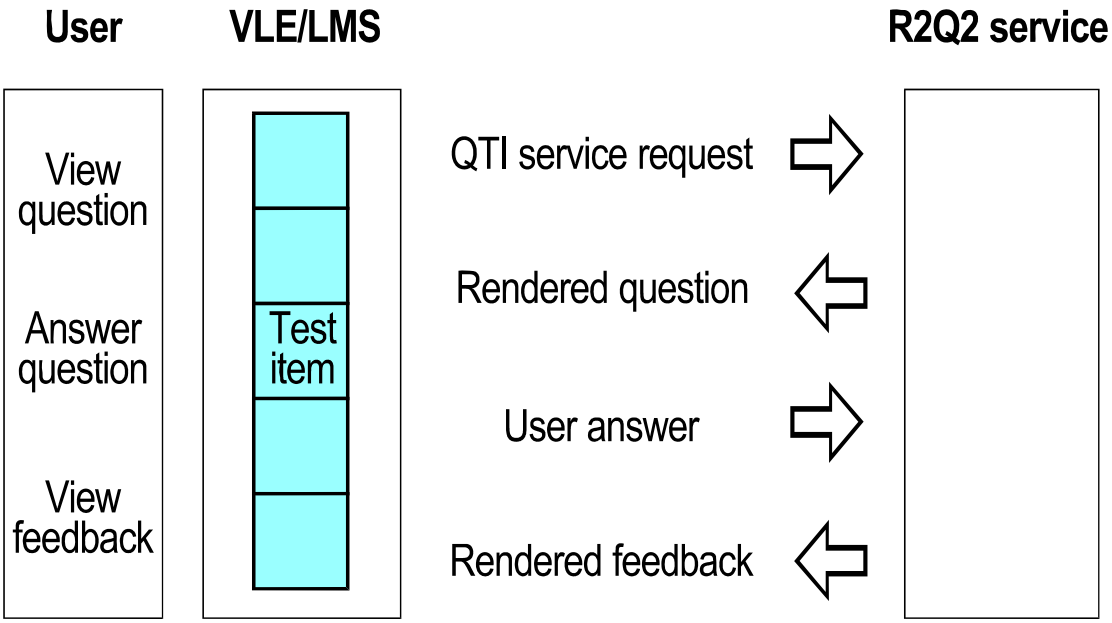

Figure 2 R2Q2 Overview

In the R2Q2 project we aim to provide a service that is more reliable than *QTIRun*, with definitive render and response processing engines for QTI version 2 question types. This is achieved by taking the single *QTIRun* Web Service and refactoring it such that the main functions are divided between

several co-operating Web Services. In our first development iteration we have focused on what we believe are the core functions (see Figure 3), allowing us to extend the service once we have validated the system.

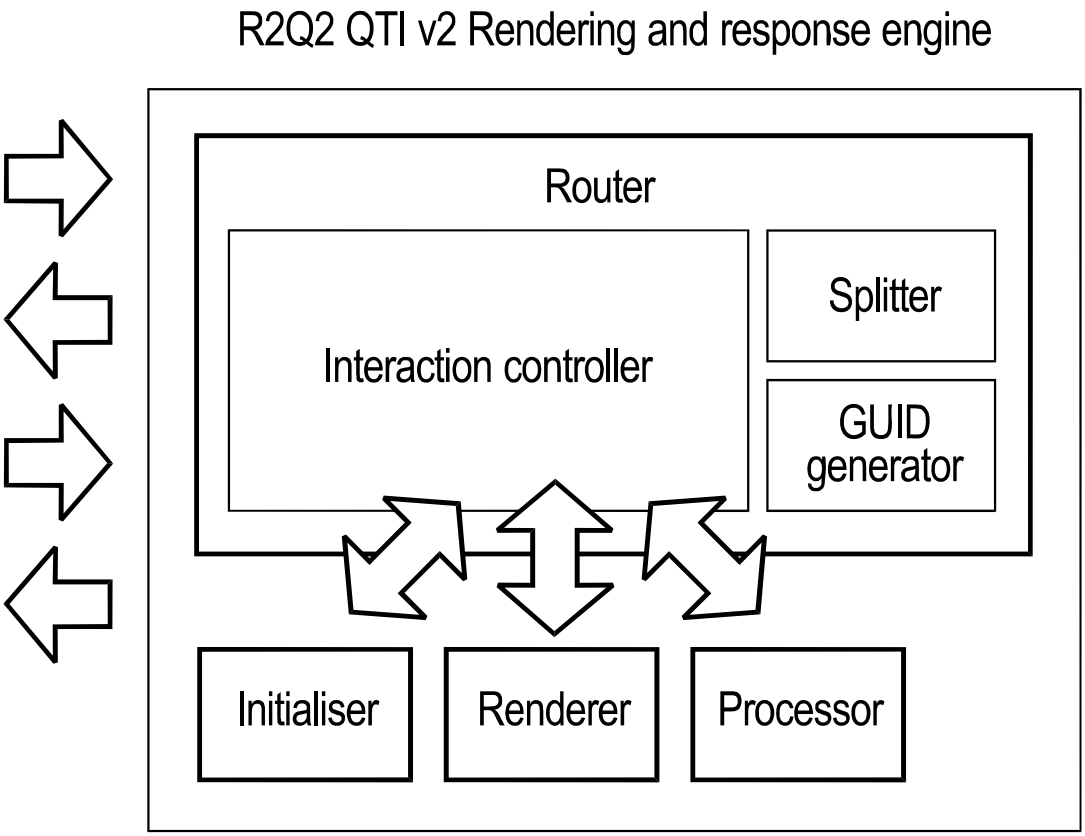

Figure 3 The R2Q2 Architecture

The R2Q2 engine is a loosely coupled architecture comprising of three interoperable services. All the interactions with and within the R2Q2 engine are managed by an internal component called the Router.

The Router is responsible for parsing and passing the various components of the item (QTIv2) to the responsible web services. It also manages the interactions of external software with the system, and it is therefore the only component that handles state. This enables the other services to be much simpler than QTIRun, and they can maintain a loosely coupled interface but without the need to exchange large amounts of XML.

The Processor service processes the user responses and generates feedback. The Processor compares the user's answer with a set of rules and generates response variables based on those rules. The Renderer service then renders the item (and any feedback) to the user given these response variables.

Future Development

Figure 3 shows the core services where R2Q2 is used as a stand alone service. However, R2Q2 is also designed to be dropped into applications such

as a test engine or authoring tool. The second iteration of the design will therefore develop the services that will allow the R2Q2 engine to be integrated into other community projects. From the interviews we have conducted there are several areas that the interviewees felt needed attention:

- Authors would like to be able to batch-process questions and answers.
- While the specification only gives examples in XHTML, it would be good to have a rendering process for questions using Flash.
- Some management of service loading and subsequent performance is required (as many users may attempt to take a given test at the same time).
- The use of the Remote Question Protocol (RQP) needs investigation as it may allow R2Q2 to be easily integrated into a VLE such as Moodle.
- Good documentation is essential if this tool is to be used by others.
- A single install process is important for community take-up of any tool.

Conclusions

At a recent conference the UK assessment community confirmed that kick-starting the use of the IMS Question and Test Interoperability version 2 specifications was a high priority. Whilst earlier versions of the specification provided most of the functions needed by practitioners, to ensure future interoperability it was considered essential that tools migrate to this new standard. However there was little incentive to move towards the new specification as existing public implementations are incomplete. The conference concluded that there needed to be a robust set of tools and services that conformed to the QTIv2 specification to facilitate this migration.

A central function that many systems require is that of rendering questions and responding to users answers. The R2Q2 project aims to produce a core set of web services to provide this functionality.

To ensure wide-spread take up of the specification, however, R2Q2 will need to be integrated into authoring tools, test engines, VLEs and LMSs, amongst other applications, to achieve the aim of migrating the community to this new standard.

References

[1] Bull J., and McKenna C. Blueprint for Computer Assisted Assessment. Routledge Falmer, 2004.

[2] Conole, G. and Warburton, B. "A review of computer-assisted assessment". ALT-J Research in Learning Technology, vol. 13, pp. 17-31, 2005.

[3] Cooper, A. and Reimann, R. About Face 2.0: The Essentials of Interaction Design. John Wiley & Sons, 2003.

[4] Davies, W. M., Howard, Y., Millard, D. E., Davis, H. C. and Sclater, N. Aggregating Assessment Tools in a Service Oriented Architecture. In Proceedings of 9th International CAA Conference, Loughborough. 2005

[5] Olivier B., Roberts T., and Blinco K. "The e-Framework for Education and Research: An Overview". DEST (Australia), JISC-CETIS (UK), www.e-framework.org, accessed July 2005.

[6] Sclater N., Howie K. User requirements of the ‘‘ultimate’’ online assessment engine, Computers & Education, 40, 285–306 2003.

[7] Wilson S., Blinco K., and Rehak D. Service-Oriented Frameworks: Modelling the infrastructure for the next generation of e-Learning Systems. JISC, Bristol, UK 2004.

[8] APIS [Assessment Provision through Interoperable Segments] - University of Strathclyde–(eLearning Framework and Tools Strand) http://www.jisc.ac.uk/index.cfm?name=apis, accessed 30 April 2006.

[9] Assessment and Simple Sequencing Integration Services (ASSIS) – Final Report – 1.0. http://www.hull.ac.uk/esig/downloads/Final-Report-Assis.pdf, accessed 29 April 2006.

[10] IMS Global Learning Consortium, Inc. IMS Question and Test Interoperability Version 2.1 Public Draft Specification. http://www.imsglobal.org/question/index.html, accessed 9 January 2006.