

MINIBIX: ITEM BANKING WITH WEB SERVICES

**Steve Lay, William Billingsley, Raymond Chan,
Dan Sheppard**

Minibix: Item Banking with Web Services

Steve Lay <swl10@cam.ac.uk>
William Billingsley
Raymond Chan
Dan Sheppard
CARET, University of Cambridge

Abstract

The Minibix system was developed from an existing prototype item bank system in use for high-stakes testing at the University of Cambridge. The system has been developed over the last year with support from the JISC e-Learning Programme. This project has redeveloped the system based on version 2 of the IMS Question and Test Interoperability (QTI) specification and is publishing the resulting system under an open source license.

In this paper, we propose a simple service model for describing the authoring, banking, test construction and delivery of assessment content. The item banking model is implemented by the Minibix system and will be demonstrated in conjunction with authoring, test construction and delivery systems developed by the sister projects: AQuRate (Kingston University) and AsDel (University of Southampton).

These services, as part of a wider e-Framework, could enable tool integration on a scale suitable for interacting with large-scale item banks. Private banks are already used routinely in high-stakes summative assessment but open repositories of items for formative use are now becoming available. For example, the E3AN item bank for Electrical and Electronic Engineering or the item bank for the Physical Sciences recently announced by the HEA.

REST

During the development of the Minibix system a number of approaches to implementing the web-service interfaces were investigated. In the end, a simple "REST"-ful approach was taken. This approach builds directly on existing web protocols to provide services that are easy to understand and implement. The approach works well when the service is inherently resource based, like the collection of pages that make up a website, and gives rise to some simple ways to integrate powerful item functions into other web applications.

Open Source

The Minibix project worked with the companion projects to adopt a common approach to open source development and licensing. This has, in theory,

enabled code to be shared more easily amongst the projects. In this paper we report on our experiences and the lessons learned.

Background

The QTI specification is a technical format for the exchange of assessment content and results. Although widely implemented, version 1 of the specification (Smythe, Shepherd, Brewer & Lay, 2002) has not led to the seamless interoperability desired by the community. Version 2 (Lay & Gorissen, 2006) was developed to address the technical flaws that were identified as hampering interoperability in practice. In particular, version 2 strengthens the role of IMS Content Packaging as the required way to transfer QTI-based content between systems. The development of QTI v2 has provided a solid platform on which to start building web services to support the integration of diverse assessment applications.

The Minibix system was born out of work at Cambridge Assessment (formerly University of Cambridge Local Examinations Syndicate). Cambridge Assessment has a long history of expertise in test development and uses a large-scale production environment that includes an item banking system called LIBS. The production process for their English for Speakers of Other Languages (ESOL) tests has been described in some detail (Saville, 2003).

There seems little doubt that a move to electronic publishing (and even delivery) will have a significant impact on test development processes. Research at Cambridge Assessment into Item Banking in XML (IBiX) led to a prototype application to support a new Thinking Skills for Admissions (TSA) test. The TSA test is used by many Cambridge University colleges as part of the process of interviewing prospective students (Harding, 2004). The system draws on a large item bank of pre-calibrated items delivering both online and paper-based forms of the tests with rapid delivery of results to college admissions tutors. The prototype item bank, which covers only a small part of a full production system's scope such as that implemented by LIBS, has come to be known as Mini-IBix or simply "Minibix".

In the UK, the Joint Information Systems Committee (JISC), in particular through its E-Learning Programme, has supported a number of projects that have taken forward work on transforming digital item banks. The SPAID project (Young, MacNeill, Adams & McAlpine) produced a system that began to address metadata tagging and packaging of QTI content. This work was followed up in more detail by the Scottish Qualifications Authority (SQA) with a broader look at the framework of services required to support item banking (McAlpine, Tierney & Zanden, 2006). Both Cambridge Assessment and the SQA are concerned mainly with high-stakes summative assessment. The use of item banks for formative assessment, and in particular to encourage the exchange of questions within the UK HE community, was investigated in depth in the IBIS report (Sclater, 2004) with practical experience gained in the COLA project (Sclater & MacDonald, 2004). There remain concerns about whether the conditions for interoperability of this type of content actually exist

(Sclater, 2007). Despite the scepticism, Sclater does acknowledge that a move to a service-oriented approach such as that adopted by the e-Framework could help stimulate this type of exchange.

It is also worth noting that the release of the item bank for the Physical Sciences, recently announced by the HEA, demonstrates a continued commitment to creating a community of sharing based on interoperability standards. A proprietary equivalent already exists for QuestionMark Perception users in the form of their SWAP service. The current age of social software suggests that creating a market based on the free exchange of content contributed as part of an initial central investment (e.g. HEA) and sustained by contributions that do little more than build the reputations of the contributors, might be feasible. There is still no sign of the type of micro-payment system considered in the IBIS report.

Minibix Project

The JISC-funded Minibix project was part of the e-Learning Programme, the general aim being to improve e-Learning through "*a technical infrastructure that supports flexibility, diversity and extendibility*". The project, along with AQuRate and AsDEL, was focussed on building and testing software tools to be released under an open source license. It is envisaged that providing free exemplar code that can be copied or incorporated directly into third party applications will help reduce the level of investment required to produce tools that utilise the QTI specification, breaking the cycle of content sharing being held back by a lack of capable tools which are not developed because the investment is not justified given the lack of content sharing!

A key objective was the integration of authoring, banking and delivery tools through service interfaces, something that goes beyond the scope of the QTI specification itself. JISC is a founding member of the e-Framework, an international initiative to help promote interoperability through a service-orientated approach.

Service Model

The service model we have developed has therefore been designed with the needs of the e-Framework in mind. The e-Framework describes itself as promoting "*service-oriented approaches to facilitate technical interoperability of core infrastructure*". The assessment domain has been slow to develop sustainable service-oriented approaches (SOA). The Remote Question Protocol (RQP) was developed as part of the Serving Maths project (Delius, 2005) and provided an early example of an attempt to link assessment systems together to enable specialist processing to take place. RQP was developed before QTI version 2 and had to cope with the absence of a unified format for assessment content. The domain in which it was developed, mathematics, remains poorly served by the current range of question types supported in general assessment systems. As a result, RQP includes the

negotiation of the content format to be used. In practice, the impact on interoperability is therefore limited.

The R2Q2 project (Wills et al, 2006) was a forerunner to the AsDel project for assessment delivery. R2Q2 was aimed at providing a SOA to assessment delivery using QTI version 2. The resulting service model teases apart the processes involved in managing the presentation and response processing of a QTI-based assessment. However, although the analysis provides a very useful design pattern for developers of delivery systems the need to express the interfaces developed as web services is questionable. The component parts in the model would typically be run together as part of a unified system and early indications from the current round of projects suggest that artificially using web-based services to implement these interfaces has a significant effect, reducing the capacity of any resulting system.

Both RQP and R2Q2 do provide a possible model for managing response processing externally. There is a strong case to be made for using a SOA to allow specialist systems to carry out response processing. Systems that support algebraic manipulation, advanced parsing or statistical processing of free text responses are likely to be written using specialist computer programming languages not suited to developing the main body of the assessment delivery system. The idea of using a SOA to tackle integration of components from diverse architectures was raised following experience with the PyAssess project (Lay, 2007). This project looked at SOA models for response-processing in some detail.

One other significant development in the use of SOAs for assessment interoperability is the IMS Global Learning Consortium's guidelines on Tools Interoperability (TI). Unlike the models described above, this work looked at the interface between a *learning environment* and an external assessment system. The early demonstrations of TI were encouraging, although the depth of assessment information exchanged was very limited (amounting to little more than a returned score). IMS are now undertaking a new activity in support of the TI agenda with a view to publishing a full specification.

The work undertaken by the assessment toolkit projects complements the above approaches. It looked at the system involved in the creation and publishing of assessment content, including making it available to the assessment delivery system.

The e-Framework describes sets of co-operating applications and services as Service Usage Models (SUMs). The scope of the QTI SUM is illustrated in Figure 1.

Question and Test Interoperability SUM

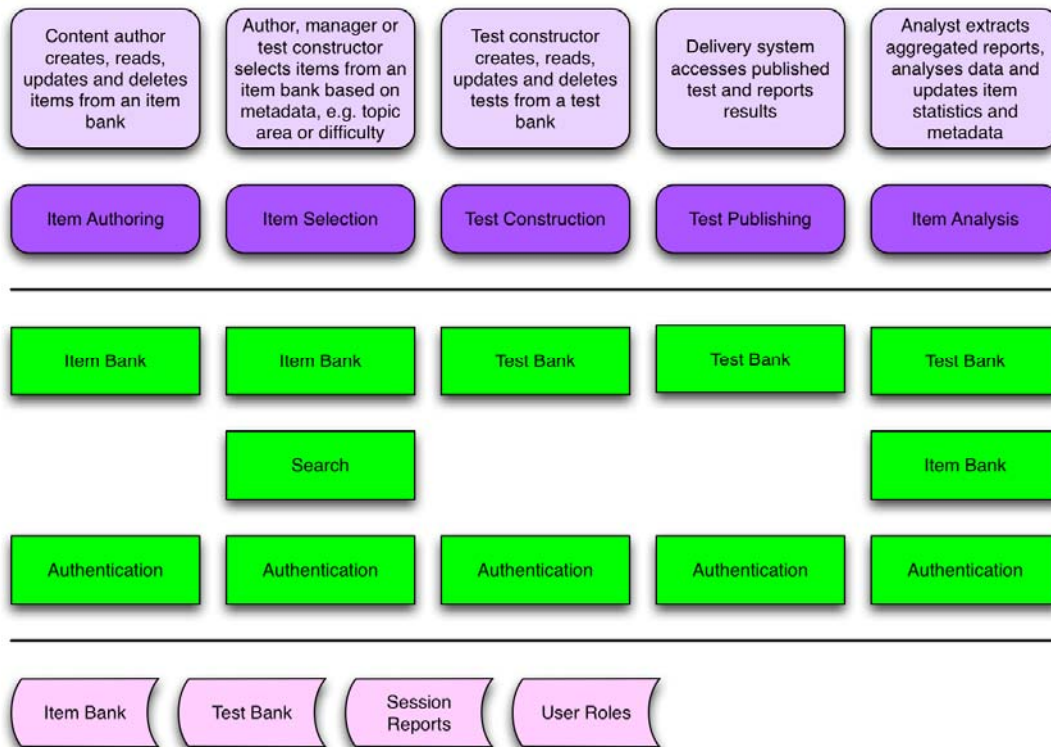


Figure 1

In this high-level view of the model, the general business processes are illustrated with the supporting services along with the main data stores present in the system. In this diagram, "Item Bank" and "Test Bank" are classed as *services*. In practice, both services may be provided by the same software system.

The item authoring and test construction processes involve the upload and maintenance of the item and test content in the banks. This type of Create, Read, Update and Delete process is supported by an interface commonly abbreviated to "CRUD". The e-Framework hasn't been designed to capture services at the level of CRUD but, in the case of item banking, these interfaces provide a gateway to a rich and varied set of workflow processes. The challenge is to provide a model flexible enough to allow business-specific workflow processes while retaining interoperability amongst the various tools.

Figure 2 describes the interaction between the author of an item and the item bank workflow processes using the Business Process Modelling Notation.

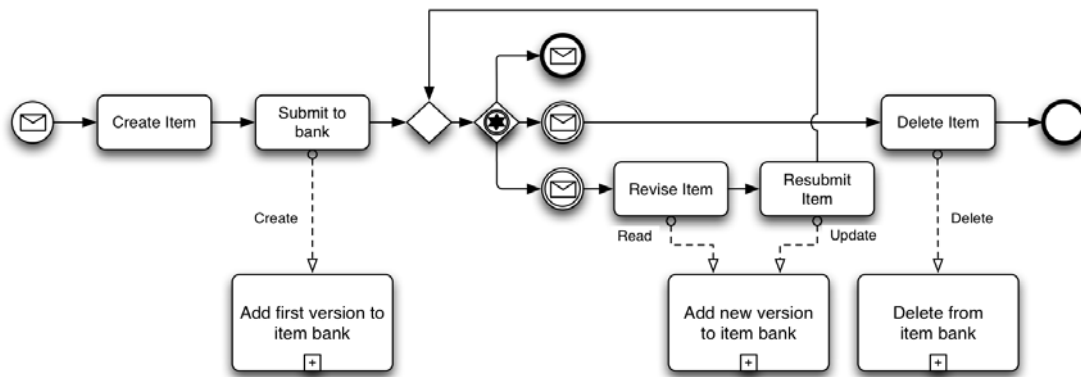


Figure 2

In this picture, the triggers that control the authoring process are indicated as message based events. For example, in a high-stakes process maintaining a large item bank the process might be initiated by a formal invitation to write material and terminate with a message indicating acceptance of the finished product. During the process, requests to revise (or even delete) the item might also occur. The messages sent to the item bank itself, indicated by the dotted arrows, trigger sub-processes (shown in a collapsed form) that are likely to vary from formal quality control workflows to something simpler such as a human moderator checking to prevent abuse of an open item exchange. The model we have developed for Minibix supports a flexible approach to implementing these workflows.

System View

In the system-oriented view of the model (Figure 3) the traditional item bank is at the centre, dividing its functions into those of item banking, test banking, test construction and analysis. These divisions are not artificial (although in practice item and test bank functions are often shared) as the supporting construction and analysis processes are typically carried out using specialist software.

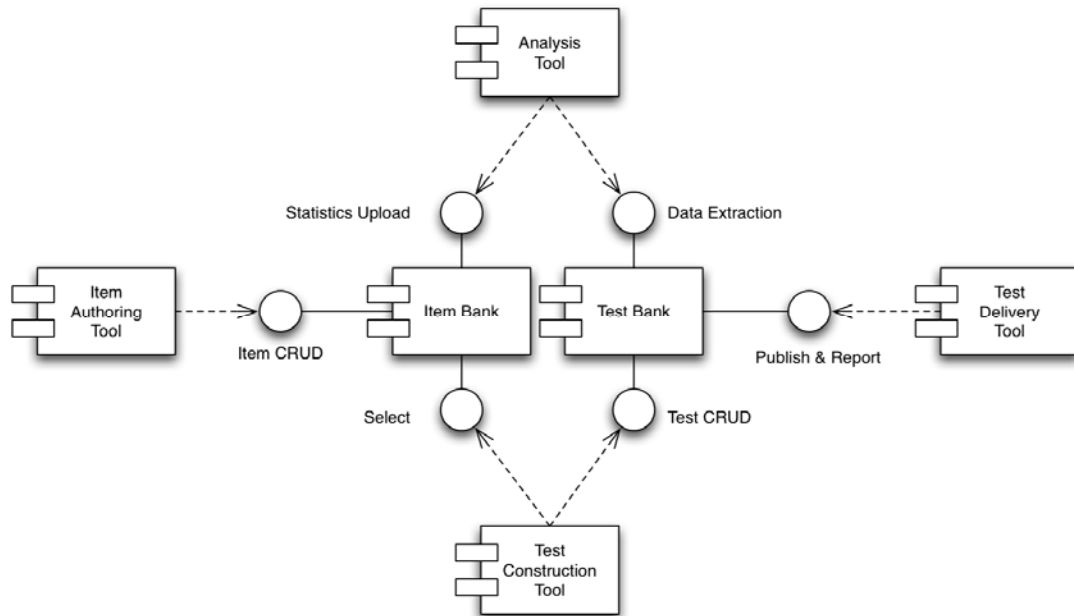


Figure 3: A Model for Creating, Managing and Publishing Assessment Content with Web Services

This model is expressed using Unified Modelling Language (UML) notation. The circles indicate the available interfaces to the central banking system components. An interface is just a defined set of messages, the expected behaviour and resulting responses. These interfaces are combined to form the services required to support the business processes illustrated above.

The interfaces play a critical role in creating an abstraction that allows tools that implement (provide) or use (consume) the interface to work interoperably without the need for tight integration based on specific knowledge of each other's technical design.

REST

The creation of software that provides or consumes interfaces requires that a mechanism is agreed for the passing of messages and responses. There are a variety of methods to choose from when passing messages over the internet. The Minibix project explored a number of approaches in collaboration with the AQuRate and AsDel project teams.

SOAP is a method suitable for implementing a wide range of interfaces, including those that manipulate complex data objects. The programmer typically uses a third party SOAP library compatible with their development platform (e.g., PHP, Java, etc.) to construct and send the messages and responses. To the programmer, SOAP can provide a very natural way of implementing web services, seamlessly turning a complex request to a remote network application server into a simple function or method call with very little additional programming effort. Unfortunately, there remain significant compatibility issues with the various SOAP libraries that limit the types of message that can be reliably exchanged.

At the opposite extreme is a group of methods which are collectively, if loosely, described as REST-based methods. REST is a description of the simple resource-based model of the world-wide-web. The client is envisaged as traversing a network of internet nodes or states (each with a URL) occasionally updating them or creating new ones. The client is provided with a representation of the current node (typically a web page) from which the model gets the name **RE**presentational **St**ate **T**ransfer or REST. This simple model is fundamental to our view of the web; without it concepts such as 'back' (return to previous node) and 'home' (return to start node) would be meaningless.

The REST model works very well for resource based systems, including file systems and databases. Therefore, the projects chose to explore a REST approach to implementing the item and test bank services. Figure 4 shows a REST based view of a Minibix server.

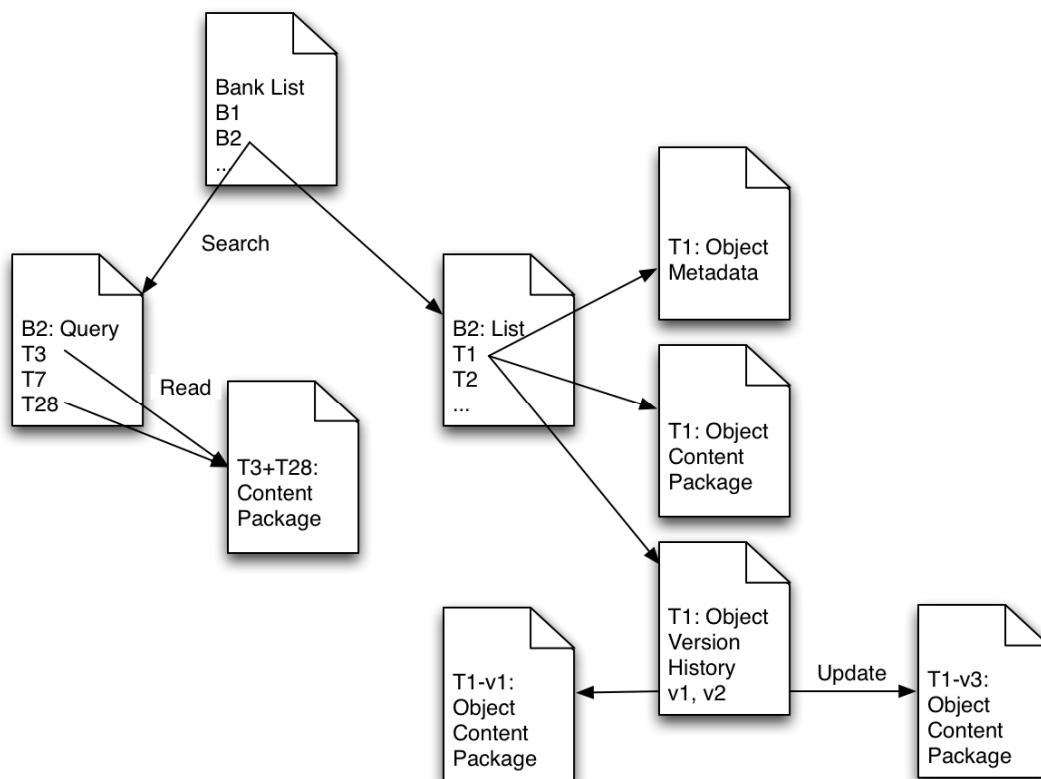


Figure 4: A RESTful view of an item bank stored in Minibix

Using simple HTTP client libraries enables a wider range of possible client platforms to develop systems with. Requests are constructed as simple URLs, for example, to obtain a list of all the objects in the bank called "B1" you can use a simple request of the form:

`http://server.domain.com/minibix/B1`

Likewise, to search for a list of all objects in B1 that match the query "physics" you use a simple URL like this:

<http://server.domain.com/minibix/B1?query=physics>

Adding new objects follows a similar pattern to simple web forms. Indeed, it is possible to construct a simple form on an HTML page that uploads a content package to the item bank. Individual objects are given 'tickets' as an acknowledgement each time a new object is created. These tickets can then be used directly in the URLs, for example:

<http://server.domain.com/minibix/B1/T3/metadata>

This URL returns the metadata for the object with ticket T3 in bank B1.

Open Source Development

The QTI specification is complex and developing software that meets the breadth of use cases supported, to a level of quality suitable for end users, will not be possible without building on the outputs of existing projects. To this end, the three projects have agreed to publish their combined outputs through the Sourceforge environment under the "New BSD" license. This license, which has very liberal terms, was chosen to enable the maximum flexibility in the way the source code is used in future. The project goal of "kick-starting" the adoption of QTI version 2 could have been hampered if we had restricted future developers from distributing tools commercially. We hope that these steps will provide a useful starting point for future QTI-based tool development.

Although the right type of license and collaboration tools are necessary for maintaining an ongoing open source development effort they are not sufficient. The creation of some type of community around the development is required. Rhatz (OSS Watch) refers to this as the "usual panacea" when attempting to sustain (or expand) an open source development project (Rhatz, 2005). He lists the typical outcomes:

- stay small (remains a nerd tool)
- gather users but no new developers (frustrated users)
- fragment when primary leader loses interest (unattractive for new people)
- develop power but with minimal documentation (no way to find the power)
- grow within an expert community (high price for admission)
- go commercial (stops being free)
- simply die

To investigate options for our own project futures, the three projects organized a joint workshop in February 2008 where the issue was discussed amongst developers and users of assessment systems.

Our first observation was that the attendees were skewed towards people who see themselves as users, rather than developers. There is no doubt that

the possibilities thrown up by computer-assisted assessment are wide ranging. But there remains a dichotomy between those who wish to apply software to improve an existing process and those who feel that computers should be used "for what they are good for" and should not simply be used to move paper-based assessment onto screens. This indecision as to where to direct efforts, combined with the fact that the community appears to have little developer resource, does indicate a danger of falling into the second of Rhatz's categories. On the other hand, it could simply be a further symptom of the lack of direction already discussed (Sclater, 2007).

During a small break-out session to look specifically at sustainability (of a community) some further problems were raised.

Why is code reuse so hard to achieve?

One factor was the perceived risk of building on code of unknown quality. Although a development 'from scratch' may take longer than one that builds directly on pre-existing work the overall development effort is perceived to be more predictable. This is a frustrating suggestion given that much of the development effort has been geared towards taking smaller steps in 6-12 month projects. A possible conclusion to draw is that code outputs need to be of the type which don't require future modification but can be used 'as is', in the same way that other basic utilities are built upon when developing any application.

Web services provide one way to achieve a kit of parts approach but the services offered have to be at an appropriate level. As discussed already, performance will be unacceptable if web services are relied upon for basic operations. The Minibix developers found this to be the case when investigating the possibility of using the services provided by the SPAID system. Although functional, these services were not appropriate when developing code that relies heavily on manipulating IMS content package files. Minibix is therefore structured so as to encourage reuse at the code library level. Web services are reserved for the message passing envisaged between the tools identified in the system model. In a similar move, the AsDel project has packaged much of the basic functionality of a delivery system into the JQTI library.

Security

A second problem is one of security. It is important to have a robust support strategy when deploying software in a high-stakes testing environment. The security of open source software relies on the vigilance of the community around its development and use. While this is absent, investing in an open source solution is seen as very risky.

Conclusions

The fundamental question of what people want from open source in the assessment community remains open. Complete open-source assessment tools aimed at early adopters are desirable if they enable experimentation with new features not supported in commercial offerings. The success of Moodle

has largely been based on this type of approach and has now provided a system mature enough for a market in commercial support to emerge. Could this type of success be repeated in the e-assessment community?

References

Delius, G. (2005). *Serving Maths*, JISC Project, no publication available. URL: <http://www.jisc.ac.uk/deletsm.html>

Fielding, R. (2000). *Architectural Styles and the Design of Network-based Software Architectures*, PhD dissertation, University of California, Irvine. URL: <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

Harding, R. (2004). *Thinking Skills Tests for University Admission*, Proceedings of the 8th CAA Conference, Loughborough: Loughborough University. URL: <http://hdl.handle.net/2134/1949>

Lay, S. (2007). *PyAssess Project: a toolkit for QTI2 migration*, Featured in e-Learning Focus, JISC online journal. URL: <http://www.elearning.ac.uk/features/pyassess>

Lay, S. & Gorissen, P. (2006). *IMS Question & Test Interoperability Specification Version 2.1*, IMS Global Learning Consortium, Public Draft (revision 2) Specification. URL: <http://www.imsglobal.org/question/index.html>

McAlpine, M., Tierney, B., & Zanden, L.v.d. (2006). *Itembanking Infrastructure: A Proposal for a Decoupled Architecture*, Proceedings of International Workshop in Learning Networks for Lifelong Competence Development, TENCompetence Conference. March 30th-31st, Sofia, Bulgaria: TENCompetence. URL: <http://hdl.handle.net/1820/848>

Rhartz, S. (2005). *What is an open source software community?* Presentation to Building Open Source Communities, 4th July 2005, Edinburgh. Available online from OSS Watch, University of Oxford. URL: <http://www.oss-watch.ac.uk/events/2005-07-04/index.pdf>

Saville, N. (2003). *The Process of Test Development and Revision in UCLES EFL*, Studies in Language Testing 15 (Edited by Weir, C. & Milanovic, M.), Chapter 2, p57-120, Cambridge University Press, ISBN 0 521 01331 3

Sclater, N. (2004), Editor *Item Banks Infrastructure Study (IBIS)* HEFCE Report. URL: <http://www.toia.ac.uk/ibis/>

Sclater, N. (2007). *The Demise of eAssessment Interoperability?* Proceedings of the Workshop on Exchanging Experiences in Technology Enhanced Learning - What Went Wrong? What Went Right? CEUR Workshop Proceedings, Vol 317, ISSN 1613-0073. URL: <http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-317/>

Sclater, N. & MacDonald, M. (2004). *Putting interoperability to the test: building a large reusable assessment item bank*, ALT-J, Research in Learning Technology, Vol. 12, No. 3

Smythe, C., Shepherd, E., Brewer, L., & Lay, S. (2002). *IMS Question & Test Interoperability Specification Version 1.2*, IMS Global Learning Consortium, Final Specification. URL: <http://www.imsglobal.org/question/index.html>

Wills, G., Davis, H., Chennupati, S., Gilbert, L., Howard, Y., Jam, E. R., Jeyes, S., Millard, D., Sherratt, R. and Willingham, G. (2006). *R2Q2: Rendering and Responses Processing for QTIv2 Question Types*, Proceedings of the 10th International CAA Conference, Loughborough University, UK. URL: <http://eprints.ecs.soton.ac.uk/12835/>

Young, R., MacNeill, S., Adams, D. & McAlpine, M. (2005). *SPAID (Storage and Packaging of Assessment Item Data)*, JISC Project Final Report. URL: http://www.jisc.ac.uk/uploaded_documents/SPAIDfinalreport.doc