Architecture of a SCORM-Compliant Assessment Authoring Tool

Kiran Parmar[1], Rachid Anane[2] and Robert J. Hendley[1]

[1]*School of Computer Science, University of Birmingham, UK*
{k.parmar, r.j.hendley}@cs.bham.ac.uk
[2]*Department of Computer and Network Systems, Coventry University, UK*
r.anane@coventry.ac.uk

Abstract

Compliance with standards is often put forward as a fundamental requirement for the widespread adoption of e-learning frameworks. Among the advantages that accrue from the conformance to standards interoperability and reuse have acquired special significance. This paper is concerned with the presentation of a SCORM-compliant assessment-authoring tool. The tool is designed and implemented as a stand-alone system, which can be integrated into a SCORM-compliant learning management system as a plug-in. A layered approach to system design was adopted as a way of satisfying a hierarchy of requirements. Convenience and usability in creating assessment content are enhanced by hiding the complexity of the underlying system, making transparent the conformance to e-learning standards and minimising user interaction. The system is put into perspective through a comparison with the QTI approach to assessment creation.

1. Introduction

On-line assessment plays a critical role in evaluating the learner's knowledge acquisition and the effectiveness of the e-learning process [1, 2]. Unlike ordinary learning content on-line assessment is marked by a high level of interaction. In addition, it requires accuracy in the recording of answers and evaluation of results, as well as secure and controlled access to content.

The usefulness and effectiveness of on-line assessment, as a key component in e-learning, depends on the support that assessment tools provide for reusability and interoperability. Reusability refers to the ability to share and reuse learning content in courseware creation. This is a feature which is essentially relevant to assessment authoring tools. Interoperability, on the other hand, refers to the ability of learning content to operate consistently across and within different platforms and learning management systems (LMS). A consistent approach to addresssing these two issues requires that assessment material and LMSs conform to common standards [3].

The Question and Test Interoperability (QTI) specification, proposed by the IMS Global Learning Consortium (www.imsproject.org), has become a *de facto* standard for on-line assessment [4]. The specification is used primarily for exchanging assessment material. This specification has attracted much attention and an intensive effort has been devoted to the development of QTI-compliant authoring tools [5, 6]. As QTI is a complex specification, many systems implement only a subset of the standard, called QTILite.

In contrast with prevailing practice, which favours QTI, a SCORM-compliant assessment-authoring tool is proposed. The focus of this paper is on the design approach and the architectural components that support this conformance. The choice of the SCORM standard is motivated by a number of factors. The Sharable Content Object Reference Model (SCORM) standard [7] is more mature than QTI and is seen as the convergence of many efforts at creating an adequate e-learning standard. SCORM deals explicitly with the dynamic interactions between learning content, including assessment, and the host learning management system.

From a pedagogical point of view the SCORM standard spans most of the phases of the e-learning cycle, namely learning needs analysis, curriculum design, curriculum delivery and curriculum evaluation [8]. The scope of QTI on the other hand is more limited and is relevant primarily to the evaluation phase. Moreover, a SCORM compliant authoring tool will ensure that interoperability and consistency with respect to the LMS is articulated along one single standard for all types of learning content. This will ease the integration of learning content into a SCORM compliant learning management system. Finally, SCORM as a more mature technology is the basis of many systems and provides a more comprehensive vehicle for e-learning migration [9, 10].

The remainder of the paper is organised as follows. Section 2 gives an introduction to the SCORM standard. Section 3 presents the architecture of the system and the components that ensure conformance. Section 4 highlights some salient features of the implementation. Section 5 details the interaction and assessment creation process. Section 6 addresses issues raised by the choice of SCORM and Section 7 presents some conclusions.

2. SCORM

The SCORM framework introduces two core components, the content aggregation model (CAM) which deals with the packaging, sequencing, navigation and description of learning content and the run-time environment (RTE), which handles the execution of the learning content and the learner's environment [11, 12].

A learning object, or shareable content object (SCO) in the SCORM vocabulary, is defined as stand-alone learning content built around one or more learning objectives [13]. Conformance to the SCORM standard has significant implications for curriculum design and development. Any SCORM-compliant courseware should run on any SCORM-compliant learning management system, such as WebCT. The main consequence of the design of a SCO as a self-contained unit is a clear separation between navigation within a SCO, and navigation between SCOs, which is the responsibility of the LMS, and ultimately the course developer. This restriction is, however, necessary for the achievement of the four goals of compliance with the SCORM standard: reusability (dis-aggregation and reuse), interoperability (horizontal interoperability), accessibility (search and discovery) and durability (vertical interoperability).

The packaging specification used by SCORM is the IMS Content Packaging specification but includes additional constraints on how to package assets, content organisations and SCOs. The specification allows learning content to be deployed in learning management systems or authoring tools where a standard method can be used to manipulate and modify the learning content. In order to package learning contents an XML manifest file needs to be produced. The file describes the structure of the learning contents (the content organisation map) and any associated resources such as image files (assets); the physical files that are required for use in the learning content are also specified in the manifest file with a clear indication of their name and location; and finally any meta-data related to the learning content. A manifest file is a focal point of activity and is crucial to the implementation of SCORM-compliant systems

The SCORM RTE defines a set of functionalities, which enable a SCO to be launched in the LMS and to exchange data with it. This requires the specification of a launch mechanism, an API and a data model. The LMS relies on the launch mechanism to determine the sequence of delivery of the learning content. The API defines the set of methods that can be invoked by the SCO. These methods are available in JavaScript and include, in particular, *LMSInitialize()* to start the LMS and *LMSFinish()* to stop the LMS. They also enable the SCO to read *(LMSGetValue())* or to write *(LMSSetValue())* data. The set of data that can be accessed and exchanged is defined by the run-time data model. This includes information about the learner and the set of interactions, and allows the storage of the learner's assessment score or the tracking of the learner's progress through course material.

It is this combination of expressive static structures and dynamic processing that confers to the SCORM standard its power and its appeal.

3. System Architecture

A layered approach was adopted in the design of the proposed system, as a reflection of the step-by-step creation of assessment and the generation of SCORM structures. Conceptually, the creation of assessment identifies two main processes, authoring and SCORM conformance. These processes are managed by a number of layers, which define the system architecture. This includes the input layer, the data conversion layer, the deployment layer and the security layer.

- The input layer consists of the program user interface for entering the data required for the creation of an assessment. This layer is concerned with the authoring aspect of the system.
- The data conversion layer converts the data into an e-learning standard format in order to ensure SCORM compliance.
- The deployment layer packages and creates all the necessary files needed for deployment.
- The security layer allows a user to save and load an assessment, within specific security constraints. An assessment can be modified and access restrictions assigned to it. The temporary state files that are generated are called assessment state files. This layer interacts directly with all the other layers.

Although a user interacts primarily with the input layer, all the other layers are called upon gradually in the generation of assessment. Once an assessment has been created, the assessment deployment package file can then be uploaded into the learning management system and made available to the learner. The different layers are further described in the following sections.

3.1 Input Layer

The input layer consists essentially of the user interface. Two options are available for assessment development, the creation of a new assessment or the loading of an existing assessment. A hierarchical approach to assessment creation was followed: the user is taken through a number of steps, which define specific tasks:

- The package creation: this initial stage obtains package information (assessment title and assessment location) and security related information from the user
- The assessment creation: the assessment developer calls upon the data conversion layer applications to create assessment questions.
- The deployment: this prompts the assessment developer to choose the location, the file name and type of the SCORM deployment package file.

3.2 Data Conversion Layer

The data conversion layer (Figure 1) is responsible for the conversion of the assessment question data, as the main body of the SCORM package, into a format that can then be translated directly into package files by the deployment layer. This contributes to the gradual building up of the SCORM profile to ensure reusability and interoperability.

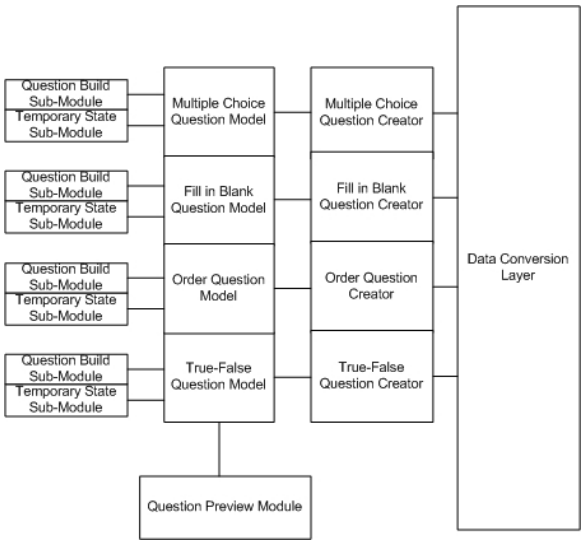

Figure 1. Data Conversion Layer

The layer consists of a number of 'question creators', associated with question types such as 'multiple choice' or 'true-false' questions. Each question creator has its own user interface, which collects the data from the assessment developer in order to build the question. The question creator is also linked to a question model, whose function is to transform the data obtained through the user interface into part of a SCORM package. This will enable the deployment layer to generate the necessary files. In addition to these functional units, the data conversion layer contains a module for previewing questions. Each question creator is generated from a basic XML structure to which it will add the data from the question models (Figure 2).

```
<question type="">
        <questionText>
            [Questions question text, here]
        </questionText>
        <answerList>
                <answer correct="[true or false, here]">
                    [the possible answer, here]
                </answer>
                .
        </answerList>
</question>
```

Figure 2. Question creator template

3.3 Security Layer

The main role of the security layer is to control access to learning content and to manage state information (Figure 3). These functions are performed by the password management and the encryption sub-modules. There are two distinct encryption processes, one for the password table and the second for the assessment state file. The password management sub-module maintains a table of assessment file locations and related passwords. For each created assessment the input layer presents to the security layer the corresponding assessment file locations and passwords. This information, as a pair, is inserted into the table with the passwords encrypted with the SHA-1 encryption algorithm. The password management sub-module can then make the password table available for queries. The security layer provides also a password user-interface, which enables the user to view all the assessments stored in the password table and to change the corresponding passwords, when required.

The second function of the security layer is to manage assessment states. An assessment state consists of the data required in order to restore an assessment following its modification by a user. The security layer manages the creation and the storage of assessment states in files, and the loading and translation of assessment states back into the program models. An assessment state file is

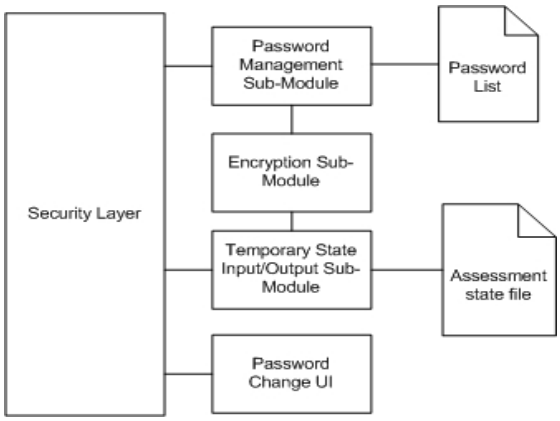

Figure 3. Security Layer

effectively made up of the SCORM package information provided by the assessment developer and all the information generated by the question models in the data conversion layer. The encryption of the assessment state file is performed in the same manner as for the password table.

3.4 Deployment Layer

The deployment layer consists of the SCORM Package module. This stores a package model, the SCORM Data Model, which is responsible for storing all package data for a particular assessment (Figure 4). A second responsibility of the deployment layer is to create all the files required by a SCORM package. Package data is added to the SCORM package by each step defined in the input layer. Once a question creator has provided the additional information required by the SCORM package to the SCORM package module, the build sub-module is then able to create the package files. These files include the IMS manifest file for storing the assessment structure, the physical files used by the assessment and the location of all these files.

The data in the SCORM Data Model is used to create this manifest file. The schema files created by the build sub-module are needed to validate the package while the JavaScript API file enables the assessment questions to communicate with the

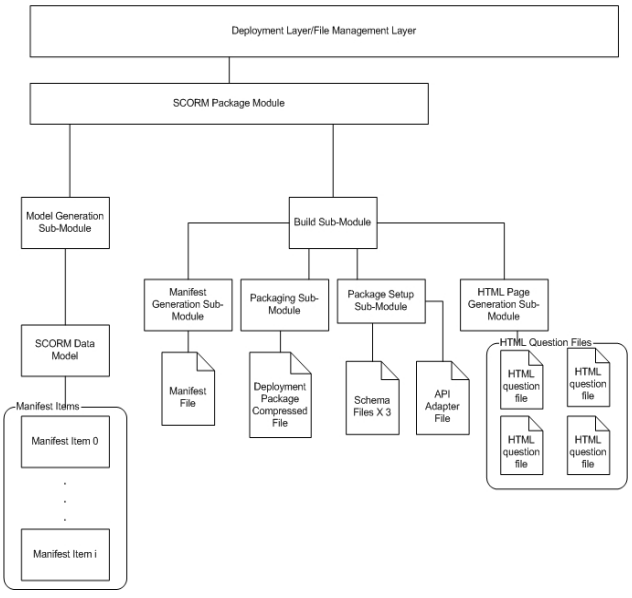

Figure 4. Deployment Layer

SCORM run-time environment. In addition, the build sub-module provides services for generating both the JavaScript files and the HTML files required for constructing a basic question for the question creators in the Data Conversion Layer. When the generation of the required files is complete, the build sub-module compresses all the above files into one SCORM deployment package.

3.5 Extensibility

One of the primary aims of this architecture is to allow new question types to be 'plugged-in' and to expand the range of the question creators available to the assessment designer. This was also the main reason for choosing the SCORM standard as it does not specify how to create the assessment questions, but only how to report the results of answers to questions. The architecture enables designers to create a 'question creator' by designing the questions user-interface and a question model so that they can be integrated with the structure of the authoring tool.

The design of the 'plug-in' question creators allows each question creator to handle the saving of its model to the assessment state file and also to restore the saved data so that the question can be amended by the assessment developer. Each question creator generates the question files it requires for the assessment to be presented to the learner, which is then submitted to the SCORM package.

4. Implementation

The system was implemented in Java and involves the application of the refinement process to the functions defined in the different layers. Since the Deployment layer implements the main SCORM features, the details of the implementation will be illustrated by focussing mainly on this layer.

The deployment layer, and more specifically the SCORM Package Module, performs two fundamental functions. The first is concerned with the specification of the sequencing code in the manifest. This involves the generation of the JavaScript code for each question created by the assessment developer. The second deals with the building of the manifest file in order to allow the assessment data to be deployed in the learning management system.

The JavaScript code is based on a template held in a file. For each question the file is augmented with a specific function, the *calculateUserScore,* which is provided by the corresponding question model. A number of functions contribute to the processing of the questions. The *loadQuestionPage* function is used to initialise each question, as a SCO. The *calculateUserScore* function obtains an answer from the user and checks its validity, while the *calculateScore* function communicates the result of the question to the learning management system. Progression through questions is achieved by the combination of the *calculateScore* and *unloadQuestionPage* functions. They disable the

```
- <imsss:sequencing>
    <imsss:controlMode choice="false" flow="true" forwardOnly="true" />
  - <imsss:rollupRules>
    - <imsss:rollupRule childActivitySet="all">
      - <imsss:rollupConditions>
          <imsss:rollupCondition condition="attempted" />
        </imsss:rollupConditions>
        <imsss:rollupAction action="completed" />
      </imsss:rollupRule>
    </imsss:rollupRules>
  - <imsss:objectives>
    - <imsss:primaryObjective objectiveID="PRIMARYOBJ" satisfiedByMeasure="true">
        <imsss:minNormalizedMeasure>1.0</imsss:minNormalizedMeasure>
      </imsss:primaryObjective>
    </imsss:objectives>
  </imsss:sequencing>
</item>
<imsss:sequencing>
```

Figure 5 Sequencing rules

current question (SCO) and inform the learning management system that the current question has been answered, and that the following question in the assessment can be presented.

The construction of the manifest file combines the IMS Content Packaging specification with additional constraints required by the SCORM Content Packaging specification. These constraints cover the design of the overall structure of the manifest file and the design of the sequencing code in the manifest. The file is made up of a number of components. In the first part, the schema file elements indicate where the schema files are located for the validation of the manifest XML. In the second part, the meta-data elements describe the learning content data. The third part represents the organisations section and lists all the questions used in the assessment under item XML tags/elements. A question title is given to each item tag/element. The item attributes provide information to the learning management system and indicates which resource item in the resources section is associated with the question. In the organisation section, the sequencing section is used to specify rules, which guide the learning management system in ordering and presenting the assessment questions to the learner. The tracking model in the SCORM run-time environment performs this function. The rules remain static for each assessment package created by the assessment developer.

An example of sequencing rules is shown in Figure 5. The sequencing rules state that each question can only be seen once, and that a strict linear sequencing of the questions must be enforced, when presented to the learner. The rules, in this case, state also some constraints on the score of the user. This makes use of the score model in the SCORM run-time environment to communicate constraints on the learners score. The value, which follows the *minNormalizedMeasure* tag indicates the minimum percentage score that the learner needs to obtain in order to pass the assessment. This value is set by the assessment developer in the pass rate combo-box of the assessment creation wizard. Finally, in the fourth part, the resources section lists all the files used by each question in the assessment, and specifies their location.

5. Deployment

The design of the user interface was shaped by the need to enforce a clear separation of concern between authoring requirements and SCORM compliance. This approach is aimed at facilitating the assessment creation process. It was decided, therefore, to use a template wizard instead of a single long and indiscriminate form to create assessments. This provides a means of segmenting the data obtained from the user, and of progressing seamlessly through the assessment creation. Once the user has opted to create a new assessment, a three-stage wizard process is initiated. The deployment of the wizard is controlled by the input layer. For convenience, the wizard offers a familiar Windows-like user-interface.

5.1 Data acquisition

The package creation wizard is used for the acquisition from the user of the package information (assessment title and assessment location) and security related information (Figure 6). The assessment title is presented to learners when they register for the assessment. The location indicates where the package files are stored, and the password entered is forwarded to the security layer for validation of access to the assessment.

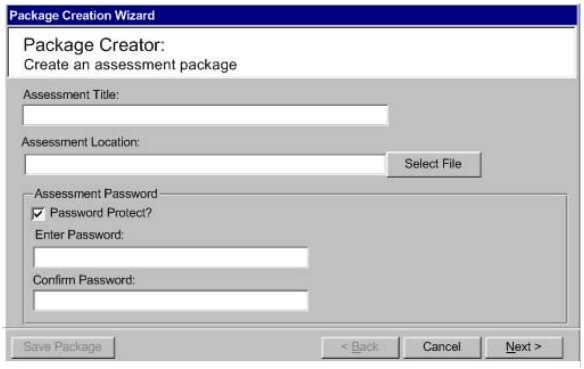

Figure 6. Package Creation

In the assessment creation wizard, the assessment developer interacts with the data conversion layer applications in order to create assessment questions (Figure 7). This is achieved by the interface in this wizard by launching a question creator for each type of question. This wizard displays all the created questions, and allows the developer to specify the order in which the questions are presented to the learner when an assessment is taken. The assessment developer can then set the percentage pass rate that

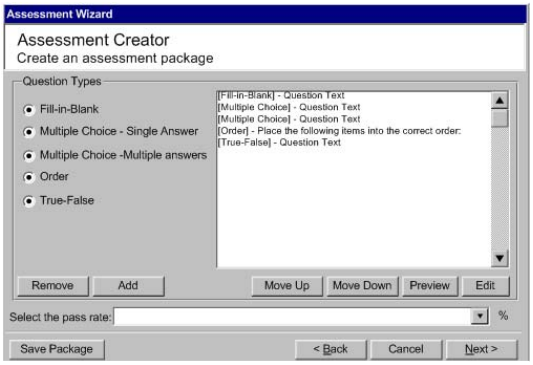

Figure 7. Assessment Creation

needs to be achieved by the learner. At this stage the user is allowed to save the assessment in its current state so that it may be modified.

The third and final stage of interaction is managed by the deployment wizard (Figure 8). This enables the assessment developer to select the location and the name of the file and the type of the SCORM deployment package file. Once the finish button is clicked, all the created questions are compressed into one file and placed in the specified location.

At any stage in the wizard the assessment developer is able to move to the previous or the next stage in the process. The data gathered by each wizard is used to build up gradually the profile of a SCORM package. This SCORM package is then used by the deployment layer to create the files and the SCORM manifest file in order to package the learning content.

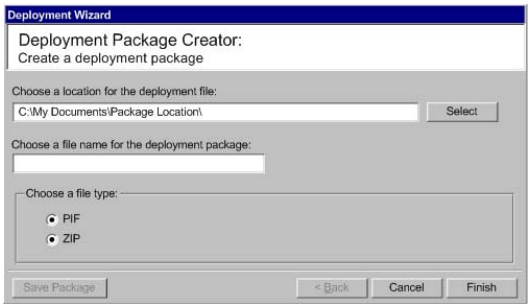

Figure 8. Deployment Package

5.2 Loading an existing assessment

The system allows existing assessments to be uploaded and modified, a feature that involves the input layer and the security layer. The input layer interacts with the security layer to obtain the assessment file (assessment state file), which was previously saved and which contains the temporary state of the current questions. In order to grant access the security layer requires a password from the user if the assessment state file is password protected. Once the user has been successfully authenticated, or if the file has no password protection it is passed to the input layer. The user-interface is then recreated from the assessment state file and the assessment can then be amended.

6. Discussion

Although the authoring tool is important in itself, it is primarily the conformance to the SCORM standard and the architecture that ensure this conformance that need to be addressed. In this section the system is put into perspective through a brief analysis of the two standards, QTI and SCORM. There is a general consensus that QTI provides a useful vehicle for assessment creation and dissemination across diverse platforms. QTI presents a uniform format, supports the development of question/test databases and facilitates adaptation in the learning process. In addition, QTI is didactically neutral, a characteristic that may however lead to a mismatch between the QTI standard and its application to a specific domain [13]. For example, QTI requires the specification of all the features of devices of interaction, the mode of evaluation as well as the resulting data [14]. It has been pointed out that strict adherence to QTI may also give undue emphasis to technology at the expense of pedagogy [15].

The QTI specification was designed to address the authoring aspects and integration of assessment material into learning content. Since the dynamic unfolding of assessment requires the processing of user input and reaction to it, a run-time environment is required to handle the high level of interaction between learner and system. This run-time environment is often implicit or absent, when compared to SCORM. This absence requires the transformation of assessment content into a suitable form that can be rendered using various technologies [6].

The representation of QTI language in XML favours implementations that rely on related technologies to render the QTI content and structure. A QTI player can be implemented as an XSL style-sheet, which transforms XML content into HTML. This transformation occurs at the application level [5]. Another approach to providing support for QTI and bridging the gap between standard and application relies on the development of templates tailored to specific subjects and topics. The template creation process is however laborious and requires a detailed knowledge of the QTI standard. It does not guarantee that the resulting learning object is SCORM compliant.

A more direct approach is proposed in the implementation of <e-QTI>, an assessment engine, which provides native support to QTI [16]. This system relies heavily on the use of templates to hide the complexity of QTI. The development of LMSs that are SCORM compliant is also encouraging the conversion of QTI into SCORM. The need to

transform QTI into intermediate forms to ensure interoperability may introduce additional levels of complexity in implementation and interaction. The different levels of concerns can be illustrated by the complementary roles of QTI and SCORM. A designer may opt for QTI format as a mode of transfer and exchange of learning objects, from which SCORM compliant concrete representations can be generated. This representation can be in HTML or Flash.

With respect to reusability and interoperability, QTI is primarily concerned with reusability (authoring level, bias towards mode of assessment and presentation), while interoperability (operational level) is a secondary concern since it is not addressed explicitly and since it can be satisfied in different ways, and may not even involve an LMS. SCORM on the other hand deals with learning objects as self-contained units of instruction; reusability is facilitated by the ability to aggregate and disaggregate SCOs and repurpose them. The RTE of SCORM supports interoperability, with respect to the LMS, explicitly. SCORM in this respect is more consistent and addresses both issues in a holistic manner.

SCORM is technology-agnostic and does not make any assumptions about the nature or the behaviour of the interaction devices. The SCORM level of concern determines the packaging of a learning object, the use of metadata for describing an object and the communication between learning object and LMS. It does not cover the internal structure and behaviour of the object, and is not biased towards a particular technology such as HTML, Flash or Java applets. For instance, the interaction types such as slide rule and hot spots in assessments, when specified in SCORM are more abstract than in the QTI specification and implementation. Although this particular characteristic offers a lot of scope for the dynamic aggregation of courseware the resulting sequence of SCOs may lead to the occurrence of a mosaic effect. The variety of styles and implementations of the different sources can be a serious impediment to effective reuse and instruction [17].

From a pedagogical point of view SCORM is seen as biased towards a single learner, and does not address the issue of collaboration, a view reinforced by the way SCOs are launched in a Web-based run-time environment [18]. Attempts at overcoming these limitations include real time personalisation through the provision of dynamic access to alternative sources [19]. In contrast, QTI is seen as pedagogically neutral.

The proposed SCORM-compliant scheme has the advantage that it deals with the overriding concerns related to QTI and addresses explicitly important issues such as security, reusability and interoperability. The system provides controlled access to the material, encrypts passwords files and assessment files. Reusability is relevant to the SCORM-compliant authoring tool, while interoperability refers to the SCORM-compliant LMS for its realisation. Both these concerns are addressed through adherence to the same standard. Furthermore, in contrast with the embedded assessment modules of a traditional LMS such as WebCT [20] or Moodle [21], this implementation promotes modularity; this is achieved through the development of a stand-alone assessment module, which can be incorporated with other LMSs and interoperate with them, a feature promoted in the SAKAI project [22].

In a similar context to the provision of QTI editors [5], the authoring process depends on the availability of adequate templates for handling question types. Unlike QTI-based systems, however, the proposed scheme integrates seamlessly with a SCORM-compliant LMS. The system also provides support and guidance through a carefully designed wizard. The use of templates, as a mode of interaction, is an effective means of dealing with the complexity of standards in general and QTI in particular. The role of the user interface is to shield the user from the complexity entailed by conformance to standards, while as the same time ensuring some form of backward compatibility.

Extensibility, as promoted by the proposed system, takes full advantage on the focus of SCORM on interaction rather than on implementation. The SCORM standard does not specify how to create the assessment questions but only how to report the results of questions. Assessment designers can create new questions types using various technologies and incorporate them easily in the scheme by conforming to prescribed protocols. Although the aim of the proposed scheme is to ease the creation of instances of assessment questions, and to support extensibility through the incorporation of question creators it does not, however, address explicitly the generation of new types of question. This issue is the focus of the further work. The RELOAD [23] tool is being investigated as an aid in the enhancement of the functionality of the authoring tool, and easing the generation of SCORM templates.

The layered approach to system design offers a number of advantages. First, it reflects the hierarchy of requirements, namely assessment creation and SCORM conformance. Second, architectural enhancements may be confined to their respective layer and therefore the effect of changes can be contained. And third, it allows for seamless integration of learning content with the LMS. Despite the limited functionality offered by the proposed system when compared to fully-fledged authoring tools such as Authorware [24], it has the merit of offering potential users simplicity, convenience and SCORM conformance.

7. Conclusion

The work discussed above has addressed the fundamental issues of reusability and interoperability in the assessment creation, through compliance with the SCORM standard. Unlike the more common QTI based practice this approach offers greater flexibility and consistency in assessment creation and deployment, as SCORM is technology neutral and works at a more abstract level in the specification of modes of interaction. It also promotes a greater integration with other SCORM compliant learning content. Conformance to standards requires however additional information and may add complexity in the development of learning content. A layered approach to the design and an adequate user interface, based on template wizards, has ensured that the user is shielded from the underlying complexity.

8. References

[1] Dalziel J., Enhancing Web-based learning with computer assisted assessment, *Proceedings of the 5th International Computer Assisted Assessment (CAA) Conference*, Loughborough University, UK, 2001.

[2] Chang W., Hsu H., Timothy K. Smith and Wang C., Enhancing SCORM metadata for assessment authoring in e-Learning, *Journal of Computer Assisted Learning, Volume 20 Issue 4*, August 2004.

[3] Kim W. and Shih T. K., On Reusability and Interoprability for Distance Learning, *Journal of Object Technology, Vol 3, No. 8,* September-October 2004.

[4] IMS Global Web-Site, "IMS Question and Test Interoperability",*http://www.imsglobal.org/question /qti_item_v2p0pd/index.html.*

[5] Giacomini-Pacurar E., Trigano P. and Alupoaïe S., A QTI editor integrated into the netUniversité web portal using IMS LD, *Journal of Interactive Media in Education, in Advances in Learning Design, ISSN: 1365-893X*, vol. 1, pp9-19, 2005.

[6] Vogten, H. Martens, H. Nadolski, R. Tattersall, C. van Rosmalen and P. Koper, R. CopperCore Service Integration, *Sixth International Conference on Advanced Learning Technologies (ICALT),* July 2006, pp378- 382.

[7] Advanced Distributed Learning Website, "SCORM ContentAggregationModelv1.3.1", *http://www.adlnet.org /screens/shares/dsp/displayfile.cfm?fileid=996*

[8] Anane R., Crowther S., Beadle J. and Theodoropoulos G, eLearning Content Provision, *The 15th IEEE International Workshop on Databases and Expert Systems (DEXA04*), Zaragoza, Spain, August 2004, pp420-425.

[9] Hinny Kong, W. K.H. Lim and L. Wang, SCMP: An E-Learning Content Migration and Standard Conformance Approach, *Journal of Distance Education Technologies, 4(2),* 2006.

[10] Kong H.P., Lim W.K., Wang L. and Gay R., SCMP: An E-Learning Content Migration and Standardization Approach, *Journal of Distance Education Technologies, 4(2),* 1-9, April-June 2006.

[11] ADL Website, "Two Minute SCORM Overview for Developers", *http://www.adlnet.org/index.cfm? fuseaction=print&prntTPK=12&prntTypeID=10.*

[12] Advanced Distributed Learning Website, "SCORM Run-Time Environment v1.3.1", *http://www.adlnet.org/ screens /shares/dsp_displayfile.cfm?fileid=996*.

[13] Helic D. Template-based Approach to Development of Interactive Tests with IMS Question and Test Interoperability, *Proceedings of ED-MEDIA 2006*, pp2075-2081, AACE, Charlottesville, USA, 2006.

[14] Ostyn C., SCORM Interactions, *http://www.ostyn. com/standards/docs/SCORMInteractions.pdf*.

[15] Davies W.M and Davis H.C., Designing Assessment tools in a Service Oriented Architecture, 1st *European Learning Grid Infrastructure (EleGI) Conference*, Vico Equense (Naples), Italy, 15-16 March 2005.

[16] Martinez I., Moreno-Ger P. and Sierra J., <e-QTI>: a Reusable Assessment Engine, <e-QTI>: A Reusable Assessment Engine*, 5th International Conference on Web-based Learning (ICWL),* 2006, pp134-145.

[17] Canale E. and Ip A., A Layered Approach to the Re-Use of Content and its Presentation, *The Tenth Australian World Wide Web Conference*, Seaworld Nara Resort, Gold Coast, July 2004.

[18] Welsh, E. (2003). SCORM: Clarity or Calamity? *http://www.onlinelearningmag.com/onlinelearning/magaz ine/article_display.jsp?vnu_content_id=1526769.*

[19] Abdullah N.A and Davis H.C., A Real-time Personalization Service for SCORM, *5th IEEE Int. Conference on Advanced Learning Technologies (ICALT*), Kaohsiung, Taiwan, 2005, pp61-65.

[20] WebCT, "Some Key Features of e-Packs and the Teaching & Learning Principles They Support", *http://webct.com/service/ViewContent? contentID=20846880.*

[21] Dougiamas M. and Taylor P.C. 2003. 'Moodle: Using Learning Communities to Create an Open Source Course Management System', *EDMEDIA 2003*. Available at *http://dougiamas.com/writing/edmedia2003*.

[22] SAKAI, *http://sakaiproject.org/.*

[23] *http://www.reload.ac.uk/.*

[24] Macromedia Web-Site, "Authorware: Features", *http://www.macromedia.com/software/authorware/produc tinfo/features/static_tour/power.html*