

Towards Open Standards: the Evolution of an XML/JSP/WebDAV Based Collaborative Courseware Generating System

Changtao Qu, and Wolfgang Nejdl

Learning Lab Lower Saxony, University of Hannover
Deutscher Pavillon, Expo Plaza 1
D-30539, Hannover, Germany
{qu, nejdl}@learninglab.de

Abstract. In this paper we present the evolution of a collaborative courseware generating system that is featured by XML-based course structure representation, JSP-based dynamic courseware presentation, and WebDAV-based collaborative courseware authoring. While the first system implementation employs a proprietary design that uses a self-defined XML DTD to represent course structure, the second and the third system implementation take an open standard based approach, which are respectively SCORM 1.1 and SCORM 1.2 conformant. In the latter two implementations, all learning resources contained in an existing Java course are re-designed according to the SCORM 1.1 and SCORM 1.2 Content Model and further annotated with corresponding SCORM metadata. In addition, the course structure is re-constructed utilizing SCORM 1.1 Content Structure Format and SCORM 1.2 Content Packaging Specification. The evolution of the collaborative courseware generating system is motivated by our efforts to improve the reusability and interoperability of learning resources.

1 Introduction

Since the summer semester 1999, the joint CS1 course “Introduction to Java Programming” (Info1 for short) has been shared between three German universities and the Free University of Bozen in Italy. During the past two years, we have been successively working on three system implementations of Info1 with the purpose of exploring efficient approaches to improving the reusability and interoperability of learning resources. While the first system implementation employs a proprietary design that uses a self-defined XML (eXtensible Markup Language) DTD (Document Type Definition) to represent course structure, the second and the third system implementation take an open standard based approach, which are respectively SCORM (Sharable Content Object Reference Model) 1.1 [1] and SCORM 1.2 [2] conformant. In the latter two implementations, all learning resources contained in Info1 are re-designed according to the SCORM 1.1 and SCORM 1.2 Content Model and further annotated with corresponding SCORM metadata. Also the course structure is re-constructed utilizing SCORM 1.1 CSF (Content Structure Format) and SCORM 1.2

CP (Content Packaging) Specification. In this paper we will present these three system implementations of Info1, showing its evolution towards open standards.

2 General Design

In figure 1 we illustrate the general infrastructure of the collaborative courseware generating system.

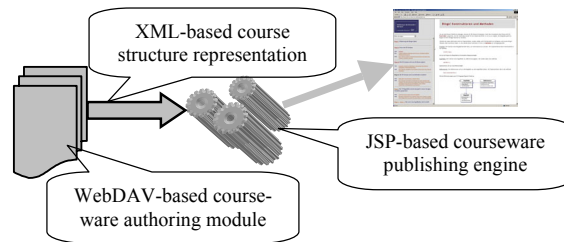


Fig. 1. General infrastructure of the collaborative courseware generating system

The system is constructed from a WebDAV (Web-based Distributed Authoring and Versioning)-based courseware authoring module and a JSP (Java Server Pages)-based courseware publishing engine. The standard data interface between both is XML.

Although the general infrastructure is commonly shared by all three system implementations of Info1, there are some essential differences between them. First of all, the three system implementations are different in how they represent the course structure using XML. This essential difference clearly marks the system's evolution towards open standards. Moreover, the different representations of the course structure also determine the reusability of our JSP-based courseware publishing engine that is responsible for dynamically presenting XML-based course structures on the Web. In figure 2 we firstly illustrate a common module of all three system implementations of Info1: the WebDAV-based courseware authoring module. It is used to support collaborative courseware authoring in three system implementations.

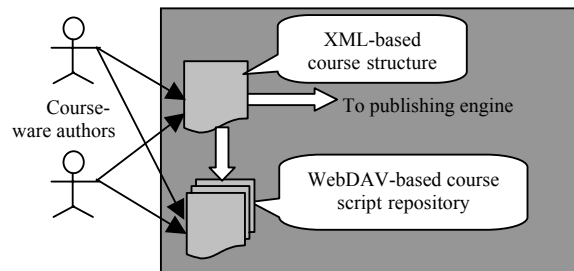


Fig. 2. WebDAV-based courseware authoring module

The courseware authoring module comprises a WebDAV-based courseware repository that is used to store course script files, and an XML file that is used to represent

the course structure. The latter also serves as the standard data interface between the courseware authoring module and courseware publishing engine in order to cleanly separate course content from courseware presentation. The WebDAV-based courseware authoring module is shared by all three system implementations of Info1, which can enable geographically-dispersed authors to collaboratively accomplish the courseware authoring process.

WebDAV [3] is an IETF specification that is originally designed to add interoperability and collaborative capabilities to the Internet. It provides sets of extensions to the HTTP protocol that allows geographically-dispersed users to collaboratively edit and manage documents directly on the remote server. The current functionalities of WebDAV include: (1) locking mechanism, used to prevent the “overwriting” of changes in a distributed, multi-user authoring environment; (2) namespace manipulation, used to manage document repository on the remote server; (3) properties manipulation, used to handle XML-based metadata of document; and (4) collections, used to create sets of related documents and to retrieve listing of their members. Utilizing WebDAV, the courseware authors can “in-place” (directly on the remote server) implement most of activities needed for collaborative courseware authoring, e.g., editing course script files stored in the courseware repository, manipulating the repository’s namespace, utilizing locking mechanism to prevent “overwriting”, or manipulating properties of a specific course script file in order to exchange ideas and opinions among lecturers. At present there are many popular WebDAV-enabled authoring tools, e.g., Microsoft Windows 2000, Windows XP, Office 2000, Office XP, FrontPage 2000, Internet Explorer 5.0, Adobe Photoshop 6.0, Acrobat 5.0, Macromedia Dreamweaver 4.0, etc., which can be directly used by the authors to complete courseware authoring process. Since all these WebDAV-enabled tools are aware of WebDAV’s methods (propfind, lock, unlock, etc.), they can “in-place” handle all course script files without the need of an explicit download and upload process. Additionally, the document locking and unlocking are also automatically managed by these WebDAV-enabled authoring tools. In fact, according to our practical experience, the WebDAV-based courseware authoring module has greatly improved the efficiency of the courseware authoring process [4].

3 The First System Implementation: Proprietary Design

The first system implementation of Info1 adopted a self-defined XML DTD to represent the course structure. In figure 3 we illustrate this XML DTD.

```
<!ELEMENT Courseware (Title, Author+,Description?, CourseUnit+)>
<!ATTLIST Courseware
  xmlns:courseware CDATA #FIXED "http://www.kbs.uni-hannover.de/Courseware" >
<ELEMENT Title (#PCDATA)>
<!ATTLIST Title
  pic CDATA #IMPLIED >
<ELEMENT Author (#PCDATA)>
<ELEMENT Description (#PCDATA)>
<ELEMENT CourseUnit (Overall,Location?, CourseElement*)+>
<!ATTLIST CourseUnit
  name CDATA #REQUIRED
```

```

url CDATA #REQUIRED>
<!ELEMENT Overall (#PCDATA)>
<!ELEMENT Location EMPTY>
<!ATTLIST Location
uni (all|Hannover|Dresden|Hildesheim|Bozen) #IMPLIED>
<!ELEMENT CourseElement (#PCDATA)>
<!ATTLIST CourseElement
name CDATA #REQUIRED
url CDATA #REQUIRED >

```

Fig. 3. Self-defined XML DTD

In the DTD definition, several self-defined XML elements, e.g., “CourseUnit”, “CourseElement” are adopted to describe the course structure. Also the metadata of the course scripts (e.g., URIs or URLs) are described in these elements in the form of “attributes”. Although principally this is a proprietary approach to representing the course structure, we can still achieve a certain reusability of courseware publishing engine based on this DTD. In fact, all courseware that are represented using above XML DTD can be directly rendered by our JSP-based courseware publishing engine without the need of any re-configuration process [4].

Figure 4 shows the infrastructure of the courseware publishing engine implemented in the first system implementation. Besides JSP, we also employ several “re-usable” technologies, e.g., JavaBeans, JSR (Java Specification Requests) 31-based Java XML data-binding [5], and JSP tag libraries in order to achieve the reusability of the courseware publishing engine. For a more detailed description of this courseware publishing engine please refer to our previous publication [4].

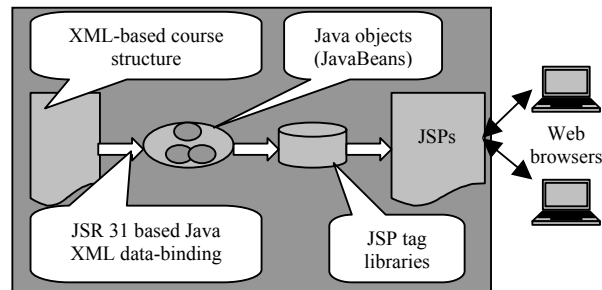


Fig. 4. Infrastructure of courseware publishing engine

4 The Second System Implementation: SCORM 1.1 Conformant Design

Although we have achieved certain reusability of the courseware publishing engine thanks to the inherent flexibility of XML, the first system implementation has two notable drawbacks. First, it is proprietary. On the one hand, the course structure represented using our self-defined XML DTD cannot be directly rendered by other courseware publishing engines. On the other hand, the courseware publishing engine bound to the self-defined XML DTD cannot be re-used to generate other courseware represented using other XML formats.

Second, the metadata of learning resources contained in Info1 are not annotated and managed in the first system implementation. This makes it very difficult to reuse and exchange learning resources between our partner universities.

Therefore, in order to achieve more interoperability, especially in order to find an efficient way to reusing and exchanging learning resources, we decided to shift to the open standard: SCORM 1.1 in the second system implementation.

The SCORM 1.1 was released by ADL (Advanced Distributed Learning) in January 2001. One of the most important features of SCORM is its good compatibility with other learning resource specifications. The SCORM 1.1 smartly references IMS Learning Resource Metadata Specification [6] (in SCORM 1.2, also IMS Content Packaging Specification [7]) and IEEE LOM (Learning Object Metadata) [8] as well as other specifications and further integrates these specifications with one another to form a more complete and easier to implement model. With regard to metadata sets, SCORM 1.1 is downwards compatible with IEEE LOM 3.5 and IMS Metadata Specification 1.1. Regarding Content Structure representation, it defines SCORM 1.1 CSF, which itself is derived from AICC CMI CSF [9]. The SCORM 1.1 also defines a Content Model that consists of three components: Raw Materials, SCO (Block), and Course. Together with its metadata specification and CSF, the Content Model can enable the reuse and exchange of learning resources at different aggregation levels. More importantly, the SCORM 1.1 also provides a RTE (Run-Time Environment) that offers a standardized way for SCO (Sharable Content Object)-based learning resources to communicate with LMS (Learning Management System) through the use of common API. During the development process, the RTE can provide us with the beneficial guidance to the system implementation.

The SCORM 1.1 conformant design of the second system implementation consists of four tasks: (1) adapting existing learning resources into SCORM 1.1 Content Model; (2) representing course structure using SCORM 1.1 CSF; (3) annotating and managing learning resource metadata; and (4) constructing SCORM 1.1 RTE.

4.1 Adapting Learning Resources into SCORM 1.1 Content Model

The learning resources contained in Info1 include not only some self-made “internal” materials, but also lots of “external” learning resources that directly exist on the Web. According to the SCORM 1.1 Content Model, these “internal” and “external” learning resources are reasonably designed as Raw Materials, SCO (Block), and Course in the second system implementation, as depicted in figure 5.

During the design process, we’ve given a special consideration to the differentiation between Raw Materials and SCOs. While each course unit of Info1 can be naturally designed as a SCO and all its underlying raw materials (e.g., figures, tables, etc.) can be naturally designed as Raw Materials, the “external” resources have to receive more attention while being adapted into the SCORM 1.1 Content Model. Because the SCO represents the lowest level of granularity of learning resources that can be tracked by a LMS using the SCORM RTE, and also SCO itself must be independent of learning context, we intentionally designed all “external” learning resources as Raw Materials in order to retain some reasonable learning context between “external” resources and SCOs (course units)[10]. Additionally, we have also organized several

SCOs into higher aggregations (Blocks), which can further facilitate the reuse and exchange of learning resources at different aggregation levels.

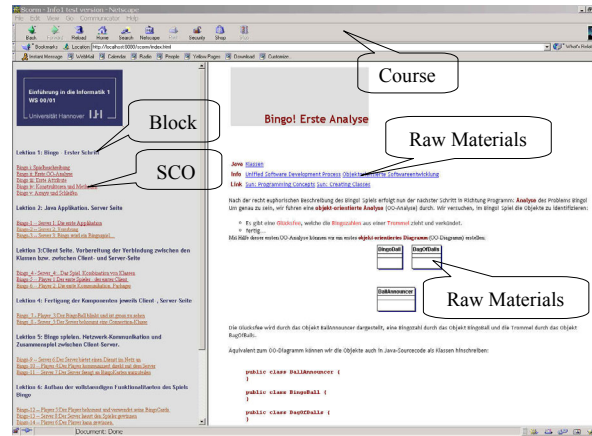


Fig. 5. SCORM 1.1 conformant Info

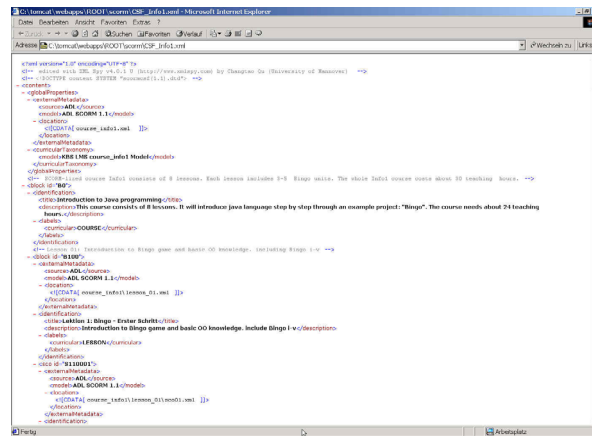


Fig. 6. SCORM 1.1 CSF-based course structure representation of Info

4.2 Representing Course Structure Using SCORM 1.1 CSF

The SCORM 1.1 employs CSF to aggregate learning resources into a cohesive unit of instruction, e.g., course, lesson, and module, etc. In comparison to the use of self-defined XML DTD in the first system implementation, representing course structure

using SCORM 1.1 CSF constitutes the key to our shift from proprietary design to open standard based development. On the one hand, the CSF-based course structure can be now directly rendered by any other SCORM 1.1 (also AICC CMI CSF) conformant courseware publishing engine, on the other hand, our courseware publishing engine (SCORM 1.1 RTE) implemented in the second system implementation can be now re-used to generate other SCORM 1.1 conformant courseware. In figure 6 we illustrate the SCORM 1.1 CSF-based course structure representation of Info1. It could be directly rendered by any SCORM 1.1 conformant courseware publishing engine.

4.3 Annotating and Managing Learning Resource Metadata

In order to facilitate the reuse of learning resources at different aggregation levels, all learning resources in Info1 are annotated with SCORM 1.1 metadata on the basis of four aggregation levels (Raw Materials, SCO, Block, and Course). During the metadata annotation process, we've paid special attention to the metadata's compatibility with other popular specifications while still remaining 100% compatibility with the SCORM. The SCORM 1.1 Metadata Information Model is broken up into nine categories: General, Lifecycle, Meta-metadata, Technical, Educational, Rights, Relation, Annotation, and Classification. Besides complying with all guidelines provided by the SCORM "best practice" for each category, we applied the ACM Computing Classification System [11] in the "Classification" category, which fits very well to describe learning resources contained in Info1 at the "ontology" or "terminology" level. Also in the "Relation" category, the relationships between four aggregation levels are described using "HasPart", "IsPartOf", etc., which nicely reflects the course structure at "metadata" level. In figure 7 we illustrate the SCORM 1.1 conformant metadata of a typical Block in Info1.

```

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE report [<!DOCTYPE report SYSTEM "http://www.adlnet.org/xml/imsreport.dtd" >>
<report xmlns="http://www.adlnet.org/xml/imsreport.dtd" >
  <meta >
    <meta:meta >
      <meta:meta:category >ADL SCORM 1.1</meta:meta:category >
      <meta:meta:language >en</meta:meta:language >
    </meta:meta >
  </meta >
  <general >
    <title >
      <string >Block: First step (Steps 1-3)</string >
    </title >
    <category >
      <string >Course Configuration Management System Path</string >
    </category >
    <entry >
      <string >Macon/SCORM_Courses/course_info1/lesson_01</string >
    </entry >
    <category >
      <string >en</string >
    </category >
    <description >
      <string ></string >
    </description >
    <keywords >
      <string ></string >
    </keywords >
    <aggregationlevel >
      <string >2</string >
    </aggregationlevel >
    <general >
      <string ></string >
    </general >
    <technical >
      <string ></string >
    </technical >
    <educational >
      <string ></string >
    </educational >
    <rights >
      <string ></string >
    </rights >
    <relation >
      <string >IsPartOf</string >
    </relation >
    <meta >
      <string ></string >
    </meta >
    <description >
      <string >KBS_Info1</string >
    </description >
    <resources >
      <string ></string >
    </resources >
    <relation >
      <string >HasPart</string >
    </relation >
    <meta >
      <string ></string >
    </meta >
    <description >
      <string >Info1_Lesson01_sco01</string >
      <string >Info1_Lesson01_sco02</string >
      <string >Info1_Lesson01_sco03</string >
    </description >
  </general >

```

Fig. 7. SCORM 1.1 conformant metadata of a block

In order to effectively manage the metadata of learning resources, we choose a native XML database: dbXML [12] to store SCORM metadata Application Profiles.

Figure 8 shows the architecture of dbXML-based, SCORM 1.1 conformant metadata repository.

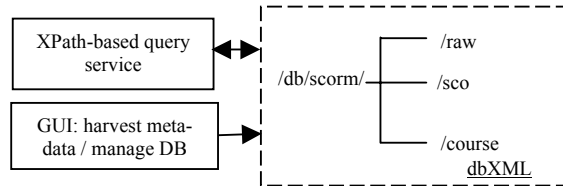


Fig. 8. Architecture of dbXML-based, SCORM 1.1 conformant metadata repository

As a so-called native XML database, dbXML provides a natural way to store, retrieve, update, search, and discover SCORM metadata. In dbXML, all metadata Application Profiles are stored in their original XML format according to three aggregation levels defined in the SCORM 1.1 Content Model. The search and update of metadata can be easily accomplished taking advantage of W3C XPath language [13] and XUpdate language from XML:DB Initiative.

4.4 Constructing SCORM 1.1 RTE

The SCORM 1.1 RTE serves actually as our new courseware publishing engine in the second system implementation. It takes SCORM 1.1 CSF as the input and then dynamically generate courseware presentation on the Web. In figure 9 we illustrate the infrastructure of SCORM 1.1 RTE.

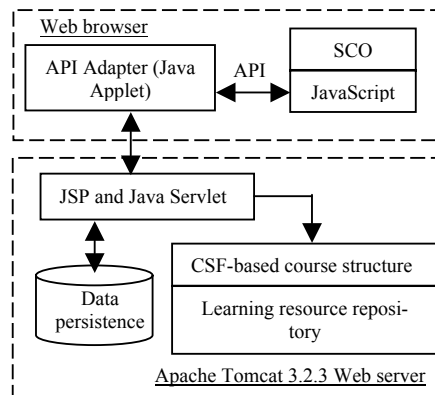


Fig. 9. Infrastructure of SCORM 1.1 RTE

The SCORM 1.1 RTE is constructed on a JSP&Servlet-enabled Web server: Apache Tomcat 3.2.3. On the server side, a JSP component is used to dynamically render SCORM CSF-based course structure into the navigation menu which is de-

picted in the left frame of figure 5. This menu appears as a series of hyperlinks whose targets contain the corresponding launch locations of SCOs. Additionally, there are also several Java Servlet components that are responsible for controlling actual sequencing of SCOs, handling communication between RTE and SCOs (e.g., getting and setting Data Model), and managing persistence of Data Model [10]. Our current SCORM 1.1 RTE implementation directly employs the CMI Data Model Java binding API provided by AICC. On the server side, the persistence management of Data Model uses Java serialized objects.

On the client side, a non-face Java Applet is implemented as the SCORM RTE API Adapter and embedded in the left frame of figure 5. This API Adapter Applet provides the communication to the RTE server-side Servlet components for Data Model persistence management. Note that on the client side, the SCOs cannot make direct communication with the RTE server to call API functions. All calls from SCOs must take the API Adapter as a broker and use client-side JavaScript. Moreover, all learning context existing within a SCO must also be managed by SCO itself using embedded client-side JavaScript.

5 The Third System Implementation: SCORM 1.2 Conformant Design

At the beginning of October 2001, we began to develop the third system implementation inspired by our desire of pursuing more openness and interoperability of the collaborative courseware generating system. The third system implementation is based on SCORM 1.2, which was released by ADL in October 2001. In comparison to SCORM 1.1, SCORM 1.2 has several important improvements. Regarding metadata specification, the SCORM 1.2 sits on a higher level than SCORM 1.1, offering downwards compatibility with IMS Metadata Specification 1.2.1 (instead of IMS 1.1 in SCORM 1.1) and IEEE LOM 6.1 (instead of LOM 3.5 in SCORM 1.1). With regard to the Content Structure representation, SCORM 1.2 deprecates SCORM 1.1 CSF and provides a CP specification which is derived from the IMS CP specification 1.1.2 [7]. As a matter of fact, the use of SCORM 1.2 CP enables a new functionality of our courseware generating system. That is, on the basis of SCORM 1.2 CP, the learning resources in Info1 can be physically packaged and unpackaged. This will greatly facilitate the exchange of learning resources between different LMSs.

In general, in order to shift the second system implementation to the third one, we have to fulfill four tasks: (1) transferring learning resources from SCORM 1.1 Content Model to SCORM 1.2 Content Model; (2) representing course structure using SCORM 1.2 CP; (3) annotating and managing learning resource metadata; and (4) constructing SCORM 1.2 RTE.

5.1 Transferring Learning Resources from SCORM 1.1 Content Model to SCORM 1.2

Although there are some nomenclature changes from SCORM 1.1 Content Model to SCORM 1.2 Content Model, the structure of the Content Model remains untouched.

We can simply transfer the learning resources from SCORM 1.1 Content Model to SCORM 1.2 using the rules defined in table 1.

Table 1. Content Model transfer from SCORM 1.1 to SCORM 1.2

SCORM 1.1 Content Model		SCORM 1.2 Content Model
Raw Materials	⇒	Assets
SCO	⇒	SCO
Block	⇒	Content Aggregation
Course	⇒	Content Aggregation

5.2 Representing Course Structure Using SCORM 1.2 CP

The SCORM 1.2 CP extends the latest IMS CP specification with several additional SCORM-specific elements particularly in the “organization” section where SCORM 1.2 Content Structure is located. By means of such sort of extension, the SCORM 1.2 CP can effectively define the structure and the intended behavior of a collection of learning resources along with the 100% downwards compatibility with the IMS CP. In comparison to our second system implementation in which the course structure is represented using SCORM 1.1 CSF, representing course structure using SCORM 1.2 CP in the third system implementation can achieve more interoperability thanks to the higher popularity of IMS CP. More importantly, because the course structure is now self-contained described in a SCORM 1.2 CP Application Profile, including all descriptions of dependency and relationships existing between learning resources, not only those “internal” resources existing physically in a package and described by URI, but also those “external” resources existing on the Web and described by URL, all learning resources in Info1 can be now exchanged between different LMSs based on SCORM 1.2 CP, either partially or as a whole. Such sort of exchange, namely, importing, exporting, aggregating, or disaggregating packages of learning resources, makes it feasible to reuse the learning content at various aggregation levels.

As an example, we illustrate the SCORM 1.2 CP Application Profile of Info1 in figure 10. Based on this CP Application Profile, Info1 can be not only physically packaged and unpackaged, but can also be dynamically presented on the Web by any SCORM 1.2 CP (also IMS CP) conformant courseware publishing engine.

5.3 Annotating and Managing Learning Resource Metadata

Because IEEE LOM, the cornerstone of SCORM 1.2 metadata specification, has experienced considerable change from version 3.5 to version 6.1, all SCORM 1.1 conformant metadata we’ve generated in the second system implementation have to be modified according to the SCORM 1.2 metadata specification in the third system implementation. Fortunately, since the SCORM Content Model remains almost unchanged from SCORM 1.1 to SCORM 1.2, we only need to concentrate on the syntax change while transferring SCORM 1.1 metadata Application Profiles from the second system implementation to the third one. However, due to the nomenclature change, and especially the change of metadata schema from SCORM 1.1 to SCORM 1.2, the

Exchange learning resources	N/A	Yes, based on SCORM 1.1 Content Model	Yes, based on SCORM 1.2 Content Model
Physically package & unpackage learning resources	N/A	N/A	Yes, based on SCORM 1.2 CP or IMS 1.1.2 CP
Courseware interactivity	N/A	Yes, based on SCORM 1.1 RTE Data Model (AICC CMI Data Model)	Yes, based on SCORM 1.2 RTE Data Model (AICC CMI Data Model)

The evolution of our XML/JSP/WebDAV based collaborative courseware generating system is actually motivated by our efforts to improve the reusability and interoperability of learning resources. From the proprietary design in the first system implementation to the SCORM based development in the second and the third system implementation, our system always evolves towards open standards and has become increasingly open and interoperable. Currently the exchange of learning resources based on our second and third system implementation is already underway between several German universities and institutions. Also the SCORM 1.1 and SCORM 1.2 conformant metadata repositories are now being integrated into a Peer-to-Peer distributed searching network: Edutella (<http://edutella.jxta.org>) with the purpose of further improving the reusability and interoperability of learning resources.

References

1. ADL Technical Team: SCORM Specification V1.1. <http://www.adlnet.org>
2. ADL Technical Team: SCORM Specification V1.2. <http://www.adlnet.org>
3. Goland, Y. Y., Whitehead, E. J., Faizi, A., Carter, S., Jensen, D.: HTTP Extensions for Distributed Authoring-WEBDAV. RFC 2518, Feb. 1999
4. Qu, Ch., Gamper, J., Nejd, W.: A Collaborative Courseware Generating System based on WebDAV, XML, and JSP. in Proc. of IEEE International Conference on Advanced Learning Technologies (IEEE ICALT 2001), Madison, WI, USA, Aug. 2001
5. Sun Microsystems, Inc.: JSR 31: XML Data Binding Specification. <http://jcp.org/jsr/detail/031.jsp>
6. IMS Global Learning Consortium, Inc.: IMS Learning Resource Metadata Specification V1.2.1. <http://www.imsproject.org/metadata/index.html>
7. IMS Global Learning Consortium, Inc.: IMS Content Packaging Specification V1.1.2. <http://www.imsproject.org/content/packaging/index.html>
8. IEEE Learning Technology Standards Committee: IEEE LOM Working Draft 6.1. <http://ltsc.ieee.org/wg12/index.html>
9. AICC: CMI 001-AICC/CMI Guidelines for Interoperability V3.5. <http://www.aicc.org/docs/tech/cmi001v3-5.pdf>
10. Qu, Ch., Nejd, W.: Towards Interoperability and Reusability of Learning Resource: a SCORM-conformant Courseware for Computer Science Education. in Proc. of IEEE International Conference on Advanced Learning Technologies (IEEE ICALT 2002), Kazan, Tatarstan, Russia, Sept. 2002
11. ACM: ACM Computing Classification System. <http://www.acm.org/class/1998>
12. dbXML Group: dbXML Native Database. <http://www.dbXML.org>
13. Clark, J., DeRose, S.: XML Path Language (XPath). <http://www.w3.org/TR/xpath>