Model of
Distributed Learning Objects Repository
for Heterogenic Internet Environment

Jerzy Kaczmarek and Agnieszka Landowska
Department of Software Engineering, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, Gdansk, Poland

ABSTRACT

In this paper, an extension of the existing structure of learning objects is described. The solution addresses the problem of the access and discovery of educational resources in the distributed Internet environment. An overview of e-learning standards, reference models and problems with educational resources delivery is presented. The paper describes a new concept of learning component that is created by applying object-oriented paradigms to learning objects. The proposed transformation adds an interface to a learning object to create unified and platform-independent data access. The Web services are proposed for the implementation of the learning components, which enables interoperability in heterogenic environments. An outline of the system, that uses learning components and Web services in e-learning, is presented. The model can be applied in the organization of learning objects distributed repositories.

INTRODUCTION

After years of rapid growth and evolution, Internet becomes the largest knowledge base in the world. An Internet user has one standard interface - a Web browser that allows to access application servers and distributed data repositories containing required information. Many researchers have addressed the problem of effective storage, seeking and sharing of data, which resulted in the development of many standards.

An educational resource is one of data types that are stored in distributed Internet repositories. Many standards have been developed to organize the discipline of e-learning. Currently e-learning is organizing resources into standardized small units called learning objects as reported in reference models (Advanced Distributed Learning [ADL], 2004) and standards (IEEE, 2004). According to standards, a learning object consists of multimedia learning content and metadata defining its structure and characteristics such as size and technology. The paper describes a method of effective organization of distributed learning objects repositories using object-oriented approach and Web services.

Address correspondence to: Agnieszka Landowska MSc, Department of Software Engineering, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, Narutowicza Str. 11/12, 80-492 Gdańsk, Poland, e-mail: nailie@eti.pg.gda.pl
CHARACTERISTICS OF LEARNING OBJECT REPOSITORY ENVIRONMENT

The purpose of learning object introduction was to provide sharability and reusability of educational resources among all interested parties. The global vision was to provide technological specifications that allow to create globally shared repositories, which allows for a common use of the resources in schools, universities, and commercial organizations. However, problems concerning learning object diversity, technological heterogeneity, and copyrights must be solved to implement the vision in practice.

The most important problem is the diversity of learning object scope, size, and technology. A learning object can have the form of a single image, a definition, a part of text, a listing, a lesson, a whole course, an e-book, and so forth. The resources are written with the use of different technologies: in the plain text, in HTML, XML, Flash, and so forth. Additionally, educational environments differ significantly in their functionality and technology, which creates a very heterogenic environment. Internet is assumed as the primary distribution and delivery environment for educational resources. There are billions of websites available in Internet together with different search technologies that are partly applicable to educational resources. The e-learning standards try to organize the environment by metadata definition and API interface standardization for Learning Management Systems (LMS) (ADL, 2004). They recommend XML as a learning object technology.

Copyright is another important problem, because data published on Internet pages may be easily replicated and used in an unauthorized way. Many repositories contain free resources, however other request payment. There is a problem who ought to pay the producers of learning objects and how to secure the copyrights. Figure 1 shows current state and problems of learning objects sharing.

Fig. 1. Involved parties and their problems in distributed educational repository environment

Three actors of resource sharing can be distinguished: the learner, the learning objects repository owner and the search engine owner. The cooperation between the parties is done as
follows: (i) the learner requests an object from the search engine, (ii) the search engine searches the metadata attributes database and provides results to the learner, (iii) the learner uses the data found to contact with the learning objects provider and to get resource from the repository. In most cases, learners are not able to view internal representation of a learning object, because it is not implemented in a technology that is understandable by a browser. Most of LMSs and Learning Content Management Systems (LCMS) provide the learner with learning objects in a form that is readable by a Web browser that is HTML, Flash, text or other. The cost of purchasing and maintaining such systems is high and therefore they are usually installed on commercial servers. In most cases, repository owners do not want to provide their educational resources for free, which may result in attempts of copyright violations. Currently the repositories are placed mainly at universities and are available only for the students through the university's LMS.

The future vision of e-learning market assumes the separation of resources production, teaching and certification (Gwozdzinska, Kaczmarek, 2003). A student will be able to get resources from a university in England, learn them with a help from a teacher from Poland and then pass examinations and get a certificate from a university in Spain. In this model the resource producers must be paid for their work. It is necessary to develop methods of searching and distribution of learning objects and solve the copyright problem to enable world-wide exchange of educational data.

EDUCATIONAL DATA SHARING IN DISTRIBUTED ENVIRONMENT

There are many organizations working on e-learning standards: Advanced Distributed Learning (ADL), Instructional Management System (IMS), Aviation Industry CBT Committee (AICC), IEEE, ISO Committee and others. They provide technical specifications and reference models that are usually accepted as standards. Subjects of standardization are: learning object structures, methods of its packaging and sequencing, meta-data, the construction of LMS, repositories interoperability, learner information, and others. The basic reference model is Sharable Content Object Reference Model (SCORM). There exists also an important specification of educational repositories construction - Digital Repositories Interoperability specification (Instructional Management System DRI, 2003).

The SCORM reference model (ADL, 2004) uses the SCO (Sharable Content Object) as the basic educational resource unit. SCORM specifies the run-time environment, in which SCO defines the content and interface of an object. An SCO package is sent to a user machine, where a standardized API interface is used to communicate with an LMS and deliver the content to the learner. The model is based on the assumption, that API is launchable at the client machine and that there is an LMS available that provides required methods. The logic is encapsulated in the learning object.

IMS DRI is another important standard applicable to repositories (IMS DRI, 2003). It specifies the basic methods that digital repository must implement and provide: submit/store, search/expose, request/deliver, and alert. The specification does not propose technology, but defines repository functionality.

There are two generic models of communication between distributed applications: data exchange and procedure call. Modularisation and procedure call communication are current trends in distributed applications programming. Most of modern technologies such as J2EE and .NET apply modularisation and procedure call.

Each e-learning standard seems to propose a model of data exchange for learning object sharing and delivery. SCORM defines the model of learning object which includes content
and application logic, however those objects are transported using data exchange model. SCO is sent to a client machine, where it is unpacked and executed. The research described in the paper applies a different approach to learning objects sharing based on method call.

THE LEARNING OBJECTS ACCESS MODEL

Although a term of learning object refers to object-orientation, it is, according to standards, a set of structured data. An original method of modifying the current model of learning objects is proposed as follows:
- the basic structure of a learning object is extended with methods and other object-oriented paradigms and a learning component is created,
- the new learning component is accessed with the use of Web services and UDDI search mechanisms.

The extension improves the interoperability by using Web services and, additionally, it simplifies data search and copyright protection. A learning component is created from learning object data structure by adding methods to make a full object in the sense of object-oriented programming. The proposed transformation adds an interface to a learning object and creates unified and platform-independent data access. Get and set operations are used to manipulate the data encapsulated within a learning object similarly to regular objects. However, it is difficult to synthesize learning object behaviour because of data type diversity. As an example, consider a learning object that is a picture and can be resized or transformed to black and white. However, learning objects can have wider functionality. A learning object can be a test made according to IMS Question and Test Interoperability Specification (IMS QTI, 2003), which allows to define questions together with data necessary to check answer correctness. The method of getting one random test question can be defined, which allows to make an exam from multiple learning objects. A detailed description of the learning component methods and structure is presented in Section 5.

Web services can easily be applied for learning component access in common object-oriented environments. Web service is a programming concept that uses WSDL for service definition, SOAP for communication and UDDI for search (Graham, Simeonov, Boubez, Davis, Daniels, 2003). If a learning component is implemented as a Web service it can be accessed by any Web services client. The solution increases significantly the interoperability and availability of components. The Web service representation of learning component functionality is registered in a UDDI registry to enable searching and binding. In the UDDI server data structures, each service is given an identifier while its definition and location are stored. A detail description of learning components implementation as Web services is presented in Section 6. The model of distributed learning components repository organization using learning components is shown in Figure 2.
The proposed model requires some changes in the repository and search engine organization. Both should be provided with Web service server and method definitions. To provide compatibility, a set of standard methods need to be added to e-learning standards.

LEARNING COMPONENT METHODS AND DATA STRUCTURE

Learning Object Data Structure According to the Standards

The IEEE standard defines a learning object as: any entity, digital or non-digital, which can be used, re-used and referenced during technology-supported learning (IEEE, 2004). The definition is very broad, which has been criticized (Firesen, 2004). The SCORM definition of a learning object is a collection of one or more assets that include a specific launchable asset that utilizes the SCORM run-time environment to communicate with LMS (ADL, 2004). The definition is narrowed to SCORM-compliant resources. The SCORM provides the data structure for learning objects. The SCO unit consists of assets, that are electronic representation of media, such as text, sound, image, and assessment objects. The basic requirement is that an asset can be rendered by a Web browser and presented to a learner. Example assets include HTML files, XML files or Flash objects. The standard states, that a set of assets is given a manifest file that provides the meta-data, assets organization and internal and external resources description. The basic idea of the learning object structure and its SCORM representation is shown in Figure 3.
The manifest file and physical files representing assets are packed into one PIF (Package Interchange File). The SCO package contains the implementation of behaviour and asset files. Therefore, the package requires to be unpacked and then to launch an API at a client machine. It also requires an LMS available to be delivered to the learner.

Adding Methods to Learning Object Definition

The data exchange model proposed in the paper differs from the model used in the standards. A method of content packaging and launching using Web services is proposed. In the first step of the process, a learning object is extended to be an active component, which requires the addition of a set of methods. A learning component is created from the learning objects contents and new methods, and can be used as a programming component. The idea of a learning component is shown in Figure 4.

Methods of a learning components include set and get operations that operate on the encapsulated data. Methods: Get_Object_Package() and Set_Object_Package() implement the exchange of the whole learning object, which is compliant with SCORM model. The other get methods are used to get a part of a learning component without getting and unpacking the whole package. Methods: Get_Object_Metadata() and Get_Object_Manifest() are used by metadata registers. The method Get_Object_Content() is used when opening the object with a
There are equivalent methods for the set operations. The basic operations implemented by the learning component are described in detail in Table 1.

**Table 1. Description of basic methods that operate on a learning object**

<table>
<thead>
<tr>
<th>Method name</th>
<th>Method description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Object_Package()</td>
<td>sends the whole package of a learning object on request, applicable for SCORM-compliant environments and for learning object replication between repositories</td>
</tr>
<tr>
<td>Set_Object_Package()</td>
<td>the corresponding method for changing the whole object definition for example putting a new version to a repository</td>
</tr>
<tr>
<td>Get_Object_Metadata()</td>
<td>sends the metadata attribute value, the name of the attribute must be specified as input parameter, can be used for searching, makes attributes (cost, version, and others) visible</td>
</tr>
<tr>
<td>Set_Object_Metadata()</td>
<td>the corresponding method for changing the metadata attribute value, the attribute name and its new value must be specified as input parameters</td>
</tr>
<tr>
<td>Get_Object_Manifest()</td>
<td>sends the whole manifest file on request, can be used by search engines to get values of all metadata attributes at one time, can be also launched by LMS in order to get organization part of the manifest</td>
</tr>
<tr>
<td>Set_Object_Manifest()</td>
<td>the corresponding method for changing the manifest file</td>
</tr>
<tr>
<td>Get_Object_Content()</td>
<td>sends the whole learning object content on request, the content returned should be openable by a Web browser: HTML, XML files, images, plain text, Flash objects, and so forth</td>
</tr>
<tr>
<td>Set_Object_Content()</td>
<td>the corresponding method to replace object content files with a new version</td>
</tr>
<tr>
<td>Get_Methods_List()</td>
<td>sends methods list of the learning object, can be used by search engine or LMS to get accustomed with object functionality</td>
</tr>
<tr>
<td>Get_Method_Description()</td>
<td>sends the method description on request, method name must be specified as input parameter, the output parameter is WSDL definition, can be used by search agents</td>
</tr>
</tbody>
</table>

The diversity of learning objects content results in the variety of their behaviour. Other methods can be defined for different learning object types for example Resize() method for a learning object that is a picture and Check_Answer() method for a test. The exemplary additional methods for different learning object types are listed in Table 2.

It is not possible to specify potential methods of a learning object, however, a mechanism that manages optional methods is necessary. Two operations implement this functionality. The Get_Method_List() operation sends the list of additional methods defined for a learning object and the Get_Method_Description() operation sends detailed information about a specific method. A detailed description of the two methods is presented in Table 1.
Table 2. Additional data access methods for different types of learning objects

<table>
<thead>
<tr>
<th>Learning object type</th>
<th>List of additional methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>Resize()</td>
</tr>
<tr>
<td></td>
<td>Get_Black_and_White()</td>
</tr>
<tr>
<td>Test</td>
<td>Get_One_Random_Question()</td>
</tr>
<tr>
<td></td>
<td>Check_Answer(test_question_number, answer_value)</td>
</tr>
<tr>
<td>Text</td>
<td>Enlarge_Font(points)</td>
</tr>
<tr>
<td></td>
<td>Generate_Print_Version()</td>
</tr>
<tr>
<td>Structured XML Document</td>
<td>Generate_HTML()</td>
</tr>
<tr>
<td></td>
<td>Generate_TEX()</td>
</tr>
<tr>
<td></td>
<td>Change_Display_Schema(XSL file)</td>
</tr>
<tr>
<td>Aggregator</td>
<td>Get_Next_Portion()</td>
</tr>
<tr>
<td></td>
<td>Get_Content_List()</td>
</tr>
<tr>
<td>DB generated</td>
<td>Generate_HTML_Version()</td>
</tr>
<tr>
<td></td>
<td>Generate_XML_Version()</td>
</tr>
</tbody>
</table>

SCORM compliant learning objects are a particular type of learning objects. Although SCO defines a set of methods (Initialize(), Terminate(), Get_Value(), Set_Value(), Commit() and few error methods), the methods are designed to communicate with a LMS. In the presented approach, universal methods are supplied that operate on learning object data. The extension of a learning object with the methods create a full object in the sense of object-oriented approach.

Object-oriented Approach to Learning Objects

The object-oriented approach is based upon the concept of an object, however there is no agreement about a precise definition of the object. Some resources state, that objects are units into which we divide the world, the molecules of our model (Rumbaugh, Blaha, Premerlani, Eddy, Lorensen, 1991). The UML specification gives the following definition: something with a unique identity that encapsulates some know-how (operations) and some values (attributes) (Object Management Group, 2003). Learning objects conform to both definitions. Learning object is a molecule of the model as it is the basic unit proposed for educational resources organization. It also encapsulates data and supplies the behaviour, as described in the previous section. Therefore, the UML definition can be applied.

Object-orientation is based on the following paradigms: abstraction, encapsulation, modularity, and hierarchy. Some terms from object-oriented design (late binding, inheritance, overriding, polymorphism, and aggregation) are also applicable to learning components.

Abstraction denotes essential characteristics of an object that distinguish from other kinds of objects (Booch, 1994). Abstraction characterizes the most important features neglecting the details. The learning object abstract class can be defined, because the structure of a learning object fulfills requirements of abstraction. The learning object construction is unique: it consists of content, organization, and meta-data. The structure is not dependent on the implementation technology and is different form other entities that can be found in the Internet. The basic interface is the same for each learning object.
The diversity of learning object types and their methods requires specialization. Therefore, a set of sub-classes representing types of learning objects must be added to the model. With the use of abstraction, generalization, and inheritance the hierarchy of learning objects can be defined, as presented in Figure 5.

![UML model of learning objects generalization hierarchy](image)

**Fig. 5.** UML model of learning objects generalization hierarchy

The learning object class is the basic class for all learning objects. Subclasses can override chosen methods to confirm individual requirements, for example Get_Object_Content() need to be rewritten in learning object subclasses. On the other hand, a part of objects functionality is the same. For example, Get_Object_Metadata() does not need to be reimplemented in subclasses.

The aggregation helps in expressing learning object binding, during course creation. Most of the standards stress learning object re-usability, which means that a learning object, once created, can be used in many different courses. The weak aggregation concept is applicable, because course creation means that object references are joined (sequenced) within another learning object. The course delivery consists of learning objects launching using references and the sequence written in the aggregation object, which corresponds to the idea of late binding in object-oriented approach. Aggregation causes problems during replication of learning objects in repositories. Aggregation learning object can be replicated with links or with all applicable learning objects that are linked within it. Object-oriented approach supply shallow and deep copying techniques to solve this problem.

Encapsulation means hiding all the secrets of an object that do not contribute to its essential characteristics, typically, the structure of an object is hidden as well as the implementation of its methods (Booch, 2004). In learning objects, content, and meta-data should be hidden from a direct access and adequate methods should be provided.

If learning objects are published in the Internet, a protection against illegal copying must be applied. An access to the whole learning object package by an unauthorized user may result in an easy replication and unauthorized use.
The check of access rights is easy to implement because object data is accessible only through object methods for example Get_Object_Content() method. Although one may attempt to reconstruct the whole object from the data acquired with object methods, the process is much harder to perform. Apart from data protection, an access to methods can be limited. Method visibility determines, whether methods are publicly or locally available.

Methods that allow an easy search of learning objects such as Get_Object_Metadata() or Get_Object_Method_List() can be public. Methods concerning data with copyrights, for example content, can be restricted, which means that their use will be limited to authorized users. Internal methods can be reserved for the repository owner. The internal methods include set operations and the Get_Object_Package() method, that can be used for replication between repositories. The proposed visibility is useful if learning objects should be highly protected. If the repository is open, all methods can be public to provide learning objects for a free and broad use.

Benefits of treating learning objects as learning components

The IEEE standard definition of a learning object is very broad and there is rarely an entity that cannot be included in the term of a learning object (Friesen, 2004). The object-oriented approach does not conform to the IEEE definition, because of the different concept of an object. The SCO model is closer to the object-orientation, as it defines the structure and the behaviour. However, the concept of a learning component proposed in the paper is much closer to a typical object in the context of object-oriented approach.

The proposed model of learning components assumes communication from LMS to a learning object, which is the second innovation. According to SCORM, all communication between the LMS and the SCO is initiated by the SCO. There is currently no supported mechanism for LMS to initiate calls to functions implemented by a SCO (ADL, 2004). However, the proposed model does not violate SCORM learning object and run-time models. SCO can be obtained by a user, using the proposed Get_Object_Package() method, and then launched according to SCORM run-time model.

The proposed approach gives LMSs the possibility to use learning object methods. Additionally the methods can be also used by search engines, other applications and almost straightforwardly by final user in the distributed Internet environment.

THE USE OF WEB SERVICES FOR LEARNING OBJECTS IMPLEMENTATION

Web services characteristics

Web service is a programming model that integrates heterogenic platforms and programming languages. The following operations are defined for a component: the description with the use of standard description language, the publication in a registry, the discovery with the use of the standard mechanisms and the use by a defined interface (Graham et al., 2003). Web services architecture assumes three roles: service provider, requester and registry. Additionally, it implements three operations: publish, find and bind. The schema of communication starts with the service provider who publishes in registry the definition of its
service. Then, the service requestor asks the registry for services and receives the reference to
the service and its interface, this allows the binding of the requestor to the provider.

A set of standards is used to implement Web services. SOAP is the communication
protocol in which messages are sent via Internet usually with HTTP protocol. WSDL is used
for services interface definition and UDDI is used for searching. The standards use the XML
document standard.

The UDDI structures are prepared to share large amount of Web services data. There are
three basic structures: BusinessEntity, which corresponds with Web service provider,
BusinessService, which describes service, and BindingTemplate, which provides service
interface detail. There is also a structure called tModel, which can be used for services
categorization. The tModel structure can also provide a standard way of service interface
definition.

The idea of using Web services in learning environments has been researched. There were
attempts of using Web services in educational environments (Liu, El Saddik, Georganas, 2003
and Xu, El Saddik, 2003). IMS Digital Repositories Interoperability - Core Functions Best
Practice Guide (IMS DRI Best Practice, 2003) mentions Web services as a mean of
communication with repositories. However, the models assume Web services use for
repositories functionality implementation. The presented solution proposes the use of Web
services for learning objects methods invocation, which has not been addressed earlier.

Sharing Learning Objects with UDDI Server

A direct use of Web services and UDDI in e-learning is impossible because a programming
platform is required, while the learner has only a Web browser.

However, it is possible to implement a model, in which a client, equipped with a Web
browser, accesses data available through Web services. The model that uses Web services and
a UDDI server is presented in Figure 6. For short, the term HTTP denotes formats readable by
a Web browser (HTML, XML, Flash, text and others) and the term SOAP denotes a Web
service call, although SOAP may use the HTTP bindings.
Two communication protocols are used in the model: SOAP and HTTP by Web services and the client respectively. On the borderline, the cooperation between HTTP and SOAP is implemented by agents: the Search Agent and the Get Agent. The agents are simple applications that translate HTTP requests to SOAP messages and then unpack and deliver SOAP answers to a Web browser. The agents are necessary because Web browsers are not currently able to understand and interpret SOAP messages. They can display XML, however only plain contents of a SOAP message can be presented, which is meaningless for the user.

The simplest implementation of the model is a UDDI server with simple Web application that provides the functionality of Search and Get Agents. The agents are presented in the model separately to indicate the possibility of separation. The UDDI standard assumes that it is the UDDI provider responsibility to provide a Web browser access to the search functionality. Therefore, Search Agent can be assumed to work on a UDDI server.

There are many solutions to implement the Get Agent. It can be installed as a SOAP reading plug-in on the client browser. The solution allows a user to access to a repository that uses SOAP communication, which may be the next step in Internet evolution. For example, Mozilla developers have already started a project to make SOAP openable by the browser. However, the solution faces the problems of persuading the user to install the plug-in and using the plug-in in older versions of browser. Another solution is to place the Get Agent on application server accessible from the Internet or to include the get functionality within working LMS systems. The Get Agent can be also included in learning object repositories.
The solution requires that a repository receives an HTTP request, binds the request to a Web service that is invoked and translated to simple HTML.

**Model Implementation**

The model of learning components access with the use of Web services is partly implemented. The solution of combining the Search and Get Agents with the UDDI server is implemented in the IBM WebSphere application server. A learning object provider is represented as the BusinessEntity class with services defined by the BusinessServices class. The BusinessService structure is used for learning component methods representation. The tModel structure is used for learning objects and their services categorization and taxonomy. In early versions of UDDI servers only one taxonomy was available, which was not suitable to catalogue educational services. In recent versions, it is possible to use different (many) taxonomies as well as to define new ones. The UDDI server is prepared to serve as a search engine of learning object services if the limits defined for public registries are not restricted.

The problem of server performance in the presence of a large number of registered learning object need to be solved. It seems that replication is the best solution, especially that it has already been anticipated by the UDDI standard. Currently, MSc students develop security methods and implement the Get Agent system that is independent from the UDDI server.

**CONCLUSIONS**

Although the amount of data available in the Internet is extensive, the shortcomings of search methods limit the access. Perhaps, the future is the data organization into packages described with metadata. The e-learning domain leads in the creation of standards and specifications describing the educational data structures called learning objects.

New technologies such as Web services, especially UDDI servers, allow to organize learning objects storage and seeking effectively. The solution proposed in the paper transforms learning objects to learning components, which modifies the former approach of treating learning objects as data structures. The concept of learning component is used to encapsulate the data within a Web component equipped with the functionality of data processing and data access. The solution is based on object-oriented approach. It allows to use the objects made by different producers in heterogenic Internet environment.

The model presented in the paper can be applied in organization of learning objects distributed repositories. Additionally, the presented approach is applicable in other domains in which the standardization of data structures exists.

**REFERENCES**


Booch G. (1994) *Object-oriented analysis and design with applications,* The Benjamin/Cummings Publishing Company, California, USA


