ELO-Tool: Taking Action in the Challenge of Assembling Learning Objects

Liliana Patricia Santacruz-Valencia
Universidad Carlos III de Madrid, Dpto. Ingeniería Telemática, Spain // liliana@it.uc3m.es

Antonio Navarro
Universidad Complutense de Madrid, Dpto. Ingeniería del Software e Inteligencia Artificial, Spain // anavarro@sip.ucm.es

Carlos Delgado Kloos
Universidad Carlos III de Madrid, Dpto. Ingeniería Telemática, Spain // cdk@it.uc3m.es

Ignacio Aedo
Universidad Carlos III de Madrid, Dpto. Informática, Spain // aedo@ia.uc3m.es

ABSTRACT

Nowadays, a wide range of initiatives are focused on providing solutions relating to the integration and reuse of different types of learning objects. This paper deals with the assembly of learning objects, one of the most difficult components for reuse. Our proposal takes into account the requirements and competencies defined for each learning object in order to allow them to be assembled in a coherent way. As a result of this approach, we have developed *ELO-Tool*, a web application that implements an assembly mechanism based on a semantic comparison of requirements and competencies through the use of ontologies. Moreover, the application generates the appropriate set of meta-data to describe the new learning objects resulting from the assembly process.

Keywords

Learning objects, Assembly, Requirements, Competencies, Ontology.

Introduction

Nowadays, with the widespread presence of virtual learning environments, the use and reuse of educational contents has become a key issue in the world of educational technologies (Wiley, 2002). In particular, reusability is closely related to usability, emphasising the process of using resources in new ways (Robson, 2006). From a content development perspective, one possible reuse scenario involves authoring teams saving elements and templates for use in multiple contexts. From the digital library point of view, the most gratifying reuse occurs when users find content and use it in a way that was not anticipated by the creator. In addition, different perspectives and requirements for reuse exist. Developers and vendors usually need source material that can be adapted to their needs. Teachers typically just refer to content or possibly combine it with some minor adaptation. Finally, students use content material as they obtain it.

An effective reuse of learning objects requires the consideration of, at least, five components (Robson, 2006). First, the location of appropriate learning components that might eventually be combined to satisfy the proposed pedagogical objectives. Such a learning component must be accompanied by high level metadescriptions (or *meta-data*). These meta-data enhance the search and discovery of useful resources. Second, the usage policies of the resources which determine how resources can be redistributed or modified. Third, the design of learning objects with a view to reusing them. Fourth, interoperability to encompass most of the technical issues involved in digital educational resources, including software and platform compatibility. And fifth, a set of compositional rules which regulate the construction of new objects from old ones.

Motivation

Therefore, the reuse of electronic resources is a complex matter that should guarantee more attractive, effective, efficient and accessible learning experiences for students (Campbell, 2003). Reuse (as a result of discoverability), and interoperability (as a result of standardization) should be encouraged, providing a flexible approach sensitive to

each learning need and context, as well as the opportunity to personalize and structure the learning content (Ip, 2005). One of the most difficult aspects of reuse is the assembly of *learning objects* (LOs) to generate new content (Santacruz-Valencia et al., 2005). In particular, the creation of didactic units (still developed manually) requires a lot of time and effort by teachers and content designers. Finally, the management of integrable pieces of content is a key issue in the development of web-based learning tools. Thus, these tools should support the management of a wide range of types of learning objects. Nevertheless, assembly and interoperability between heterogeneous types of learning objects is not always supported by learning content management systems (Santacruz-Valencia et al., 2003a), (Santacruz-Valencia et al., 2003b).

Approach

To take action on assembly and interoperability challenges, we propose *OntoGlue*, an approach to automate the assembly of heterogeneous learning objects, based upon the following aspects:

- *Semantic meta-data*: The provision of an appropriate description of learning objects is fundamental in order to permit their discovery and to enhance their reuse possibilities. The LOM standard (LOM, 2003) does not provide a sufficient description of learning objects (DiNitto, 2006) (Santacruz-Valencia et al., 2005). For this reason, we have extended this standard. The effect produced by mixing different types of meta-data was introduced in a previous paper (Santacruz-Valencia et al., 2003c). Our approach promotes the definition of the requirements and competencies associated with each learning object. These requirements and competencies are called *associated knowledge*, and are defined in terms of classes of ontologies, possibly related via mappings. Thus the semantic web approach (Berners-Lee, 2002) is brought to the e-learning domain.
- *Search*. The ELOs' semantic meta-data are the basis for a comparison mechanism, which permits a semantic discovery of learning objects. In this way, it is possible to perform a semantic search for learning objects.
- *Assembly*: Semantic meta-data also permits the assembly of learning objects. Thus, given two learning objects enriched with these meta-data, our approach allows us to decide if their assembly is coherent from the point of view of requirements and competencies (Santacruz-Valencia et al., 2005). In addition, our approach describes the meta-data of the resulting assembled learning object.
- *Reuse*: Semantic discovery, as well as the assembly mechanism, promotes the possibilities of reutilization of educational resources in different educational contexts. The presence of semantic meta-data descriptors, enhance the reutilization of learning objects, one of the most interesting challenges (Santacruz-Valencia et al., 2003c).
- *Automation*: To provide the automation of the processes for the generation of learning objects with semantic meta-data, their search, assembly and reuse according to the OntoGlue approach, we have developed *ELO-Tool*, a web application based on our previously defined conceptual model (Santacruz-Valencia et al., 2003a), (Santacruz-Valencia et al., 2003b), (Santacruz-Valencia et al., 2005).

Thus, there are several advantages associated with the OntoGlue approach. Firstly, it provides an enhanced description of the learning objects that allows them to be searched and reused taking into account requirements and competencies. Secondly, during the assembling process of two LOs, OntoGlue checks that the competencies of the first LO cover the requirements of the second LO, guaranteeing a coherent assembling process in terms of requirements and competencies. Thirdly, OntoGlue automatically calculates the meta-data of the resulting assembled LO and also helps to provide their values. Fourthly, the definition of associated knowledge in terms of classes of ontologies, possibly related by mappings, permits a semantic comparison of requirements and competencies during the assembly and search processes. Finally, the ELO-Tool permits the automation of these activities.

This paper provides a description of the web application ELO-Tool. To that end, after a brief analysis of the related work, section 2 provides a brief description of the OntoGlue approach. Section 3 provides a description of the ELO-Tool application. The data model, the architecture, and the implementation technologies used in the development of this application are described. Finally, section 4 presents the conclusions and future work.

Related work

Nowadays several initiatives are focused on providing solutions related to different aspects of learning objects technology. These initiatives can be classified into three categories:

- Those that establish a strong relation to meta-data standards in order to facilitate the discovery, retrieval, reuse and interoperability of and between contents and learning systems. *LOM* (LOM, 2003), *IMS LRMS (*IMS, 2002), *Learnativity Content Model* (LCM) (Wagner, 2002), *SCORM Content Model* (SCORM, 2004), *VC-LOM* (DiNitto, 2006) and our own approach, are examples of these initiatives. Regarding these initiatives, our approach basically enhances the LOs description including competencies and requirements. Additionally, our approach provides guidelines for the coherent assembly of learning objects from the point of view of these requirements and competencies, instead of conditional educational paths through several learning contents followed according to student achievements only (IMS, 2003).
- Those that combine pedagogical and technical perspectives, and define a set of components whose expectations for reuse are determined by their granularity or aggregation level. *RIO/RLO Model* (Barritt, 1999), *NETg Learning Object Model* (L'Allier, 1998), and *ALOCoM* (Verbet, 2006) are examples of these initiatives. Regarding these initiatives, in our approach requirements and competencies are defined in terms of descriptors, instead of in terms of LOs themselves. This makes our approach more flexible as when the competencies of an LO are taken into account, several LOs can match a requirement, while if LOs are used as requirements, the LO mentioned in a requirement is the only LO able to match it. In addition, OntoGlue defines an assembly mechanism, instead of educational workflows.
- Those that explore the dynamic assembly of learning objects based on the relative match of the learning object's content and meta-data with the learner's needs, preferences, context, and constraints. *Dynamic Assembly of Learning Objects* (DALO) (Farrell, 2004) and *Learner Intelligent Advisor* (LIA) (Capuano, 2002) are examples of these initiatives. Regarding these initiatives, our approach includes requirements and competencies explicitly defined in terms of classes belonging to one or several ontologies. In addition, our approach considers the presence of mappings between ontologies, which permits a semantic comparison of knowledge.

OntoGlue

ELOs

The OntoGlue conceptual model is based on the definition of the *Electronic Learning Object*, ELO. We define ELOs as learning resources described by meta-data and organized in a multilayer structure, where the highest elements possess information about their *associated knowledge* that facilitates their assembly and reuse (Santacruz-Valencia et al., 2005).

As we already mentioned, associated knowledge represents the set of *requirements* that an ELO needs in order to be used and a set of *competencies* that the ELO provides after its comprehension. Our approach defines three types of ELOs: information units, content units and didactic units.

The most basic ELOs are *Information Units* (IUs), which represent atomic elements and are self-contained and highly reusable. Each of them contains a single multimedia file. These elements lack associated knowledge. They need a context before they can acquire educational significance, because in themselves they do not provide knowledge to the student. The conceptual data schema of IUs coincides with the LOM conceptual data schema. Files with text, images or audio are instances of IUs. From an editorial perspective, an image within a technical book can be considered an IU.

Content Units (CUs) are the next stage in the multilayer structure. They represent educational experiences with associated knowledge. Thus, they need requirements for their understanding and provide competencies after their absorption. The CUs conceptual data schema extends the IUs conceptual data schema with the elements *requirements*, *competencies* and *files*. From an editorial perspective, a chapter within a technical book can be considered a CU.

Finally, *Didactic Units* (DUs) represent knowledge related to a specific area after the accumulation of different related educational experiences. DUs extend CUs' conceptual data schema with extra information regarding the *objectives*, *summary* and *evaluation mechanism*. From an editorial perspective, a technical book, or a collection of technical books on a subject, can be considered a DU.

As regards the combination of ELOs, not all possible combinations among their types are allowed due to their semantic nature. Table 1 presents the combinations allowed for the assembly of ELOs, in order to generate new ones.

Table 1. Combination allowed between different types of ELOs

	IU	**CU**	**DU**
IU	CU	CU	×
CU	CU	DU	DU
DU	×	DU	DU

Note that when dealing with a computer system that structures and manages information some assumption has to be made for its behaviour to be made easier. Thus, it is possible, that the assembly of two information units may not always lead to a content unit. For example, if the first content unit is a simple text, and the second content unit is a picture, a text with a picture may not be a content unit. The key question is that if physical representation of the text plus the picture is made in a single file, this new learning object may also be represented as an information unit. If the text plus the picture is not integrated in one single file and two different files have to be referenced, one information unit (i.e. a LOM object) is not sufficient to reference both files. Thus, as our approach is not concerned with the physical mixing of data, the assembly of two information units needs to deal with two different files, and a content unit is needed. If the teacher does not agree with this assembly, he/she is free to retrieve both information units, to edit them, to make a new information unit and to upload it into the system as a new information unit.

Regarding the assembly of content units, as they represent something similar to a chapter, the aggregation of content units makes up a didactic unit (i.e. a book). Again, if the user feels that the assembly of two content units should be a content unit, he/she is free to mix both content units into a single content unit and upload it into the system. In this case, the mix could be potentially dangerous if the sequential access to the contents of the first content unit is expected in order to cover the requirements of the second content unit, because during the mixing of both content units, data from both units could be merged.

In any case, in order to achieve a coherent assembly of ELOs, the comparison between the ELOs' associated knowledge is the key issue. The next section describes OntoGlue's comparison mechanism.

Comparison in OntoGlue

The presence of associated knowledge in terms of classes of ontologies permits the semantic comparison of requirements and competencies. For example, if ELO_2 has only one requirement called *add*, and ELO_1 has only one competency called *add*, it is reasonable to assume that the composition of ELO_1 with ELO_2 (represented as $ELO_1 \circ ELO_2$) is a coherent combination because the student can learn ELO_1, and will then be able to understand ELO_2.

The problem arises when syntactic mismatches appear in semantically identical competencies and requirements. In the previous example, let us suppose that the competency of ELO_1 is *addition*. Is their assembly possible now? From a semantic point of view, this assembly seems reasonable, but from a computational point of view, there is a considerable mismatch between *add* and *addition*. In this case, if *add* and *addition* were classes of different educational ontologies related via a mapping, a semantic comparison may be possible in spite of the syntactic mismatch. In addition, the presence of ontologies permits a reasonable comparison of knowledge. For example, if the competency of ELO_1 is *arithmetic operators*, and *add*, *subtract*, *multiply* and *divide* are subclasses of this competency in some educational ontology, then it would seem reasonable that learners can understand ELO_2, whose only requirement is *add*, if they have learnt the ELO_1, which provides *arithmetic operators* as competency. The OntoGlue mechanism formalizes this rationale and uses it during the learning objects assembly process.

Ontologies in OntoGlue

An ontology is a specification of a conceptualization (Gruber, 2004). A body of formally represented knowledge is based on a conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of

interest and the relationships between them (Genesereth, 1987). Thus, a conceptualization is an abstract, simplified view of the world that we wish to represent for a specific purpose.

In our approach, ontologies specify hierarchical characterizations of a body of knowledge. For example, the *Guide to the Software Engineering Body of Knowledge* (SWEBOK) (SWEBOK, 2005) is an IEEE-led project that provides an explicit characterization of the boundaries of software engineering. This guide induces a taxonomical classification of the software engineering domain. Therefore, this guide induces a software engineering ontology (Wille, 2003).

In OntoGlue, ontologies are equated with taxonomic hierarchies of classes. Therefore, only class definitions and the *subsumption* (Wikipedia, 2006) relation matter. Thus, in OntoGlue, the reasoning as regards the instances of these classes is not important. Note that in most of the learning curricula, knowledge is represented in terms of descriptors rather than specific references (IEEE/ACM, 2001). Thus, the classes of educational ontologies play the role of educational descriptors or educational controlled vocabulary. Therefore, in OntoGlue the reasoning as regards classes is limited to checking the subsumption relation in closed ontologies. Thus, OntoGlue is not concerned with the use of ontologies to reason over classes or instances. This is a characteristic that makes developing systems that implement the OntoGlue mechanism (i.e. ELO-Tool) easier. In these systems it is not necessary to include sophisticated reasoning facilities. Instead, only structural checking of static graphs (i.e. the trees of the ontologies that make up a graph due to the mappings among these trees) must be carried out.

OntoGlue Assembly

OntoGlue is used to assure a coherent combination of learning objects in terms of requirements and competencies. Therefore, in OntoGlue, the key question is, if given a requirement *r* necessary to understand an LO, and given a competency *c* that the learner has acquired after the understanding of another LO, is the knowledge of *c* sufficient to master *r*? Or in other words, does *c cover r*? If the answer is yes, both learning objects can be assembled. Otherwise, the assembly of the LO is not recommended.

As in OntoGlue, ontologies are taxonomies, and ELO's associated knowledge (requirements and competencies) are classes of ontologies, the important question is to be able to answer whether a requirement *r* is subsumed by a competency *c*. If *r* and *c* are classes of the same ontology, this is an almost trivial question. The question becomes more complicated if several ontologies, related by mappings, are considered. Therefore, OntoGlue calculates the transitive closure (Answers, 2006) of: (i) the structural relationships induced by subsumption relation; and (ii) the semantic relationships induced by mappings.

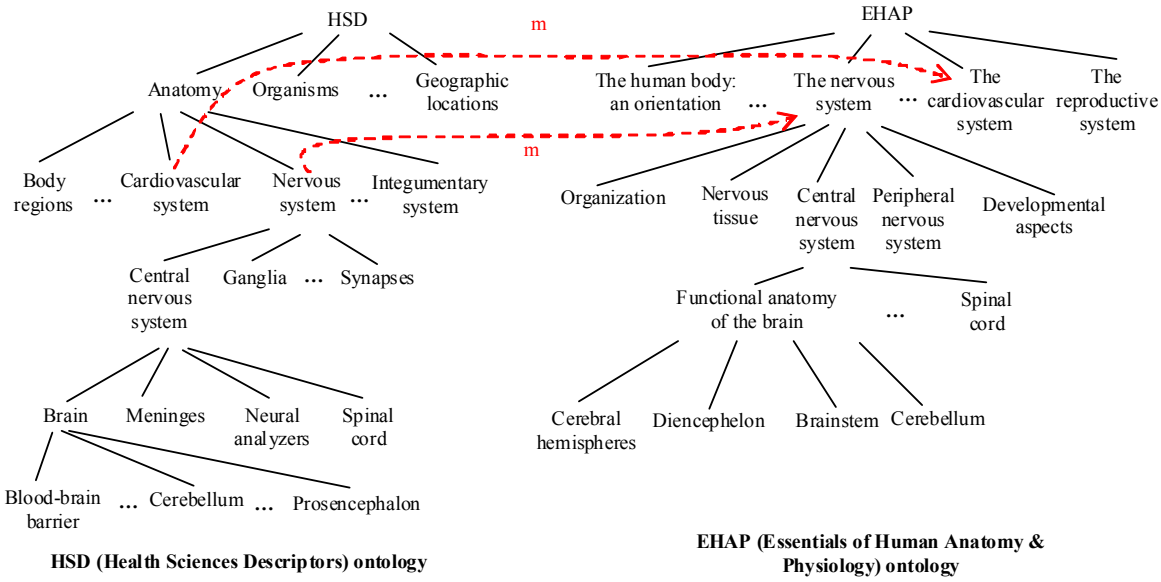

Figure 1. Two ontologies of human anatomy related by mapping *m*

Thus, we can say that *c* covers *r* if *c* = *r*, or if *r* is a subclass of *c*, or if there is a mapping *m* that *m(r)* = *c*, or if there is a mapping *m* that *m(c)* = *r*, or if there is a mapping *m* that *m(r)* is a subclass of *c*, or if there is a mapping *m* that *m(c)* is a superclass of *r*, etc.

In order to define a comparison mechanism between classes of ontologies, OntoGlue defines two key concepts: covered knowledge and sufficient knowledge. Given a class *c*, the *covered knowledge* by this class consists of all its subclasses, either in the same ontology, or in ontologies reached by *c* or its subclasses via mappings. Thus, the covered knowledge of a class *c* is the transitive closure of the relations *subclass of* class *c* and *image of* class *c* via a certain mapping. For example, let us consider the ontologies of Figure 1.

In this figure there are two ontologies of anatomy (related by mapping *m*): *HSD* (Health Sciences Descriptors) (HSD, 2006) and *EHAP* (Essentials of Human Anatomy & Physiology). The first one is trilingual structured vocabulary (English-Spanish-Portuguese) with the structure of a taxonomy made by BIREME, a specialized center of the World Health Organization (BIREME, 2006). The second one is the taxonomy induced by a well-known book about human anatomy (Marieb, 1999). If class *Nervous system* of the ontology *HSD* is considered, the covered knowledge of this class is made up of, among others, the classes *Nervous system*, *Central nervous system, Brain, ..., Cerebellum... Meninges,..., Ganglia,* etc. in the ontology *HSD*, and by classes *The nervous system, Organization,..., Central nervous system, Functional anatomy of the brain, Cerebral hemispheres,.., Cerebellum,..., Peripheral nervous system,* etc. in the ontology *EHAP*. The term covered knowledge is used because, according to Figure 1, if a learner knows about the nervous system, then he/she knows about the central nervous system, brain, ..., cerebellum, ..., the functional anatomy of the brain, cerebral hemispheres, etc. Finally, in this example, only two ontologies are depicted, but in OntoGlue, the transitive closure is made on all the ontologies related via mappings.

Given a class *r*, the *sufficient knowledge* of this class consists of all its superclasses, either in the same ontology, or in ontologies reached by *r* or its superclasses via mappings. Thus, the sufficient knowledge of a class *r* is the transitive closure of the relations *superclass of* class *c* and *image of* class *c* via a certain mapping.

Let us consider the ontologies depicted in Figure 2. In addition to the ontology *EHAP* there is a new ontology named *HEHAP* (Hole's Essentials of Human Anatomy & Physiology), based on the ontology induced by another well-known book of literature (Shier et al., 2006). If the class *Cerebellum* of the ontology *HEHAP* is considered, the sufficient knowledge of this class is made up of the classes *Cerebellum, Brain, Nervous system, and HEHAP* in the ontology *HEHAP*, and by classes *Functional anatomy of the brain*, *Central nervous system, The nervous system and EHAP* in the ontology *EHAP*. The term sufficient knowledge is used because, according to Figure 2, if a learner knows about the brain, nervous system, functional anatomy of the brain or the central nervous system, this is sufficient in order to know about the cerebellum. Again, in this example, only two ontologies are depicted, but in OntoGlue, the transitive closure is made on all the ontologies related via mappings.

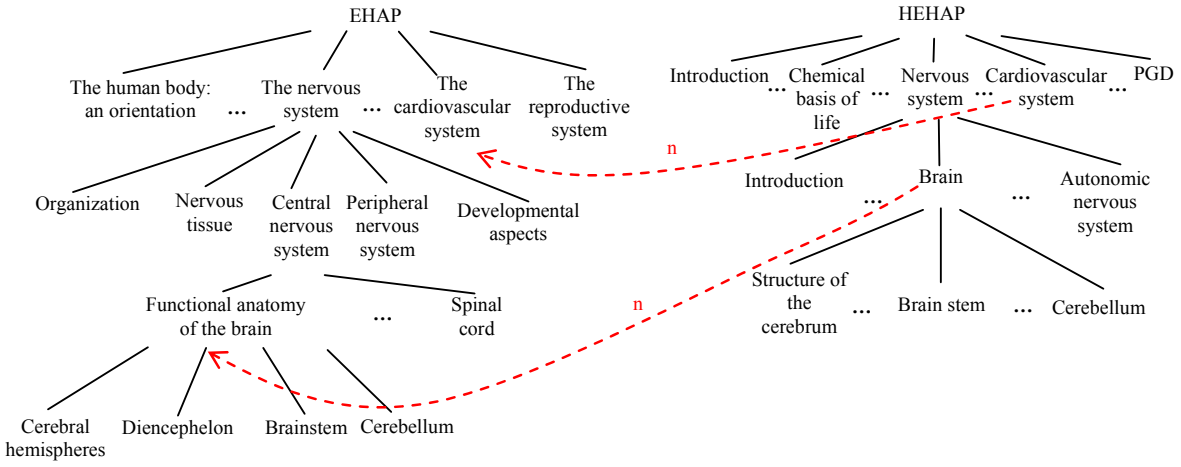

Figure 2. Two ontologies of human anatomy related by mapping *n*.

Once these concepts have been defined, it is possible to know whether a competency *c* covers a requirement *r*. Covered knowledge of class *c* and sufficient knowledge of class *r* are calculated. If the intersection of these sets is not empty, the competency *c* covers requirement *r*. For example, the competency *Nervous system* in the ontology *HSD* (Figure 1) covers the requirement *Cerebellum* in the ontology *HEHAP* (Figure 2), because, among others, the class *Central nervous system* of the ontology *EHAP* is within the covered knowledge of the competency and within the sufficient knowledge of the requirement. In other words, considering the ontologies depicted in Figure 1 and Figure 2, if a learner knows about the nervous system, then he/she knows about the cerebellum, although they are terms of different vocabularies.

A Brief Example of ELO Assembly

Let us suppose that a doctor has got several clinical cases. This doctor wishes to make a didactic unit about clinical cases, but he/she wants to include an introductory material that permits the understanding of these clinical cases. Let us suppose that the first clinical case is about cerebellar infarction and vascular lesions (e.g. (Min, 1999)). The requirements of this clinical case (which is represented using a content unit called CU_{civl}) are *Cardiovascular system* and *Cerebellum* (in terms of classes of the ontology HEHAP). Then, the doctor uses an OntoGlue-aware learning content management system to search for an ELO able to cover these requirements. Among others, let us suppose that this learning content management system retrieves the didactic unit DU_{hap} about human anatomy and physiology. This didactic unit provides the competencies *Cardiovascular system* and *Nervous system* (in terms of the ontology HSD), among others. Table 2 characterizes the requirements and competencies of both ELOs, DU_{hap} and DU_{civl}.

Table 2. Candidate ELOs

ELO	Description	Requirements	Competencies
$\mathbf{DU_{hap}}$	Human anatomy and physiology	– Secondary education	– Body regions ..... – Cardiovascular system – Nervous system ..... – Integumentary system
$\mathbf{CU_{civl}}$	Cerebellar infarction and vascular lesions	– Cardiovascular system – Cerebellum	– Cerebellar infarction and vascular lesions

The main question is: can both ELOs be assembled? Yes, because the competencies of DU_{hap} cover the requirements of CU_{civl}. Note that the class *The cardiovascular system* of the ontology *EHAP*, is both included in the covered knowledge of the class *Cardiovascular system* of the ontology HSD (Figure 1), and in the sufficient knowledge of the class *Cardiovascular system* of the ontology *HEHAP* (Figure 2), because in these cases there is a direct mapping (*m* and *n*) between these classes. Note that without these mappings, although the requirement and the competency are syntactically coincident in both cases, the requirement and the competency cannot be directly compared, because they are terms of different ontologies. In addition, the requirement *Cerebellum* of CU_{civl} is covered by the competency *Nervous system* of DU_{hap} as it was described in the previous section. Table 3 depicts the resulting assembled ELO. Note that, according to OntoGlue, this assembled ELO has the requirements of the first ELO, and the competencies of both ELOs.

Table 3. Assembled ELO.

ELO	Description	Requirements	Competencies
$\mathbf{DU_{hap} \circ CU_{civl}}$	Human anatomy and physiology with clinical cases (I).	– Secondary education	– Body regions ..... – Cardiovascular system – Nervous system ..... – Integumentary system – Cerebellar infarction and vascular lesions

Note that this electronic example has a direct real translation. It is difficult for a person without any knowledge of anatomy and physiology to understand a technical clinical case (i.e. a case about vascular problems in the cerebellum). If some publishing house adds these clinical cases to a book of general anatomy and physiology, it is easier for a person without any knowledge of anatomy and physiology to understand these clinical cases (as long as this person reads the chapters on human anatomy and physiology). Of course, if several clinical cases are going to be considered, all the requirements of these clinical cases should be considered when selecting the learning object responsible for covering them. In this example, for the sake of simplicity, only the requirements of one clinical case have been considered. In addition, it should be highlighted that this approach is valid for both electronic and printed learning objects.

ELO-Tool

The ELO-Tool is a web application that implements the OntoGlue approach. The following sections describe this tool from different viewpoints: its data model, its architecture, and the technologies used during its implementation.

Data Model

As we already mentioned, the ELO IU corresponds with the IEEE Learning Object Meta-data (LOM) standard (LOM, 2003). Extensions to this standard are then made in order to define CUs and DUs.

In e-learning, a concrete representation of meta-data is achieved by means of the definition of a *binding*: one set of requirements used to construct one XML representation (W3C, 2003a) of a particular model of information (W3C, 2006). In ELO-Tool we have developed a specific binding based on XML *Schemas* (W3C, 2003b), since the available ones do not adapt to the requirements of our proposed conceptual model (Santacruz-Valencia et al., 2005).

LOM Extension

The LOM standard (LOM, 2003) provides a semantic model to describe the properties of the objects themselves, rather than the form in which they can be used to support learning. LOM provides legal values, an informal semantics of meta-data elements and their dependency with regard to other elements. Its structure is such that it allows us to introduce *extensions*. It does not specify implementations or particular representations thus any LOM-compliant system designer can use any interface that he/she wishes and store the meta-data in the form of his/her choice. Therefore, LOM specifies only the meta-data and its semantics, so meta-data interchange between different systems is possible.

In some cases the predefined LOM elements are adapted to describe the learning resources, in other cases, they need to be extended using the element *9:Classification* of the standard. Nevertheless, it is often necessary to extend restricted vocabularies and in other cases to add new elements to the structure of the standard. This last option is the one chosen in our approach.

The extensions to the LOM standard are enumerated in the same way as the LOM elements, continuing with the corresponding enumeration inside the category *5:Educational* of the standard. This is done in order to facilitate the extensions location inside the meta-data scheme and does not imply that the added elements could only occur in the enumerated positions.

LOM XML Binding

In order to extend the LOM standard, first a XML binding was created, since the available binding does not adapt to our purposes. Figure 3 represents a UML class diagram (Arlow, 2005) for the LOM binding and the proposed extension.

An XML schema type has been associated to each of the nine LOM standard categories including the `TypeLom` as the type associated to the schema root element `Lom`. Thus, as IUs are based on LOM standard, the `TypeLom` also characterizes the type of IU. In addition, `TypeLom` is composed of: `TypeGeneral`, `TypeLifeCycle`, `TypeMeta-metadata`, `TypeTechnical`, `TypeEducational`, `TypeRights`, `TypeRelation`, `TypeAnnotation`, `TypeClassification`.

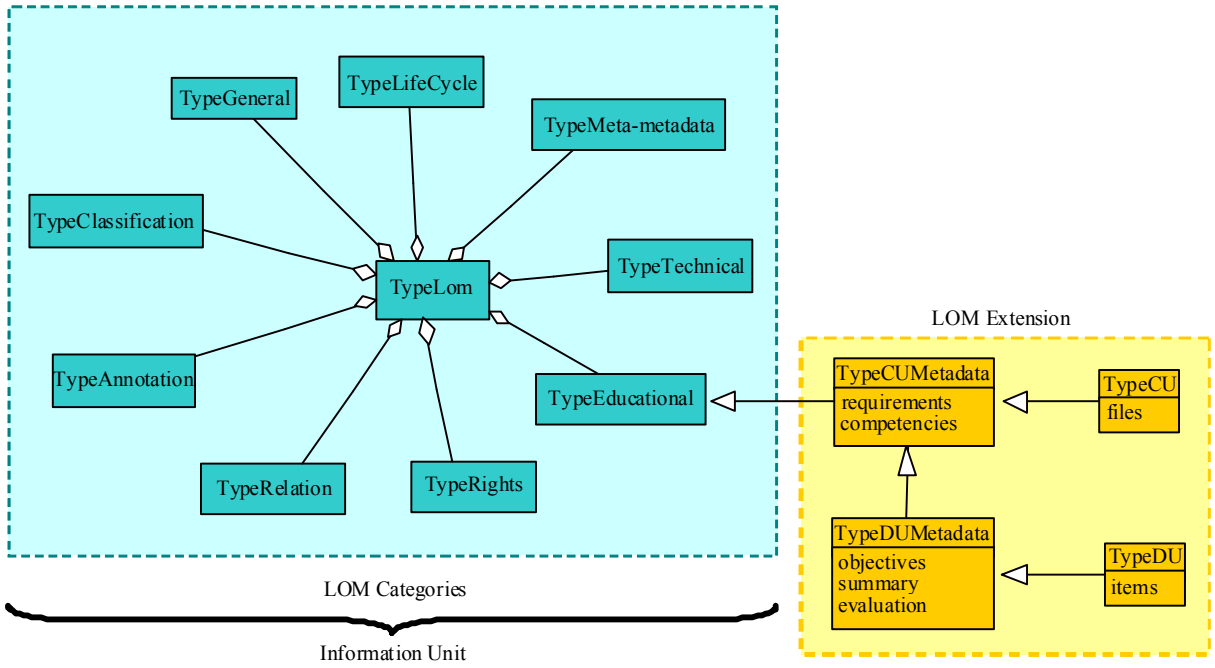

Figure 3. UML class diagram for the LOM binding and its extensions

The extensions depicted in Figure 3 allow us to describe the types of CUs and DUs. CUs are described using the types `TypeCUMetadata` and `TypeCU`. `TypeCUMetadata` extends the type `TypeEducational` with two elements, `requirements` and `competencies`. `TypeCU` allows us to identify the `files` that contain the resources used to create the CU. Moreover, as shown in Figure 3, we separate the `TypeCUMetadata` from the `TypeCU` because the `TypeDUMetadata` extends the `TypeCUMetadata`, to allow the description of the UDs with their associated `requirements`, `competencies`, `objectives`, `summary` and `evaluation`. `TypeDU` allows us to describe the type of ELOs used to create the DU. Note that according to these types, DUs use high level locators (i.e `items`), that make references to CUs or DUS, to characterize their components, while CUs use low-level locators (i.e. `files`), that make references to IUs, to characterize their components.

Other Data Models

With regard to ontologies, the *OWL-Lite XML Schema* (W3C, 2004a) was used in order to represent the ontologies used in the ELO-Tool.

With regard to the remaining conceptual data schemas used in the ELO-Tool as input format to be translated into ELO format (e.g. *Dublin Core Metadata Initiative* (DCMI, 2004)), appropriate XML schemas were defined.

System Architecture

As previously mentioned, the ELO-Tool supports the generation, assembly, search and reuse of learning objects. This section describes the system architecture, from the point of view of its functional modules. The ELO-Tool

(Santacruz-Valencia et al., 2003b) consists of four subsystems: *ELOs management*, *Ontologies management*, *LOs translation* and *ELOs assembly*. These modules interact with three repositories: *ELOs repository*, *Ontologies repository* and *XML schemas repository*. Figure 4 depicts this architecture. In this figure the dependencies between the four subsystems and the XML schemas repository is made clear via the dependencies between the ELOs and ontologies repositories with this XML schema repository.

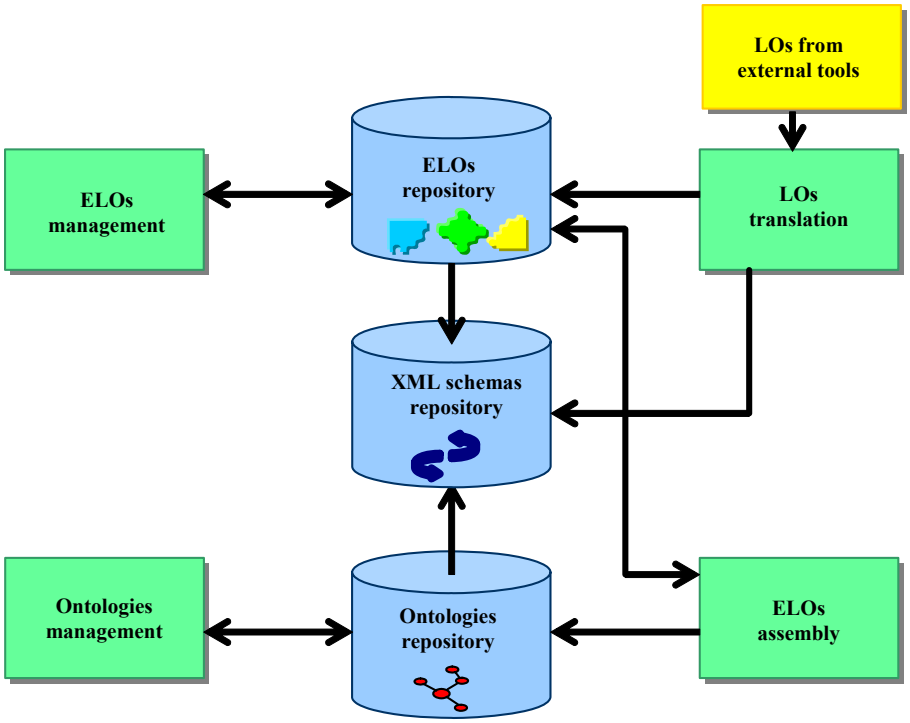

Figure 4. Subsystems of the ELO-Tool application

- The *ELOs management* subsystem includes the following functionalities:
  - *ELO creation*. The tool permits the creation of all types of ELOs (IUs, CUs, and DUs). The created ELO is stored in the ELO repository. Figure 5 depicts the user interface of the application for the creation of an IU.
  - *ELO load*. The tool permits the loading of ELOs developed with external tools (i.e. an XML editor). The validation of the ELO is performed during loading. If valid, the ELO is stored in the ELOs repository.
  - *ELO removal*. An ELO present in the ELO repository is removed from it.
  - *ELOs access*. The ELOs present in the ELO repository are depicted to the user.
  - *ELO search*. ELOs are selected according to the search criteria. This criterion includes meta-data from the LOM categories *General*, *Technical* and *Educational*, as well as requirements and competencies. Note that OntoGlue enables a semantic search. If a user needs to find LOs that teach a given a set of requirements, ELOs whose competencies cover the given requirements can be provided. Thus, ELO-Tool can find a didactic unit about human anatomy and physiology, if the teacher needs a LO able to cover the requirements regarding the cardiovascular system and cerebellum. On the other hand, if the user needs to find LOs that can be taught after mastering a given set of competencies, ELOs whose requirements are covered by these given competencies can be provided. Thus, ELO-Tool can find a content unit on vascular problems in the cerebellum, if the teacher looks for a LO to teach after mastering the didactic unit on human anatomy and physiology. Figure 6 depicts the user interface for the search for ELOs.

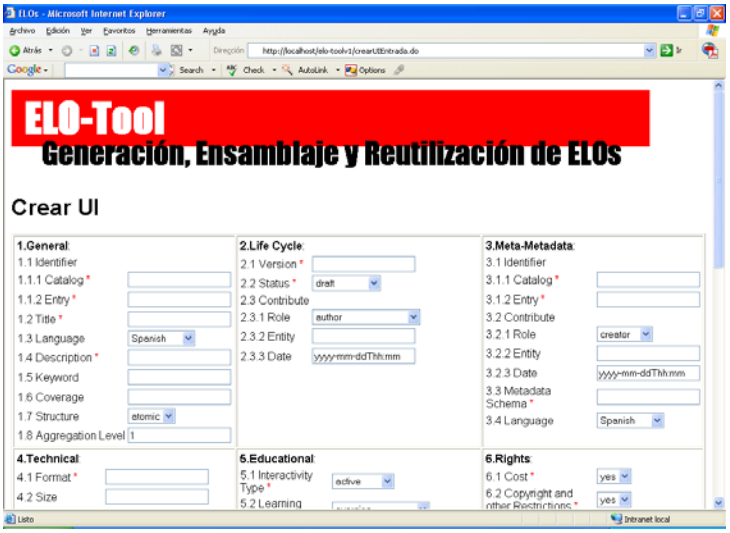

Figure 5. IU creation in ELO-Tool

- The *Ontologies management* subsystem includes the following functionalities:
  - *Ontology load*. The tool permits the loading of ontologies developed with external tools (i.e. an XML editor). The validation of the ontology is performed during loading. If valid, the ontology is stored in the ontology repository.
  - *Ontology removal*. An ontology present in the Ontologies repository is removed from it. The removal of an ontology does not check the presence of ELOs whose associated knowledge makes reference to this ontology. Note that the validation of this associated knowledge is made only during assembly and search processes. Thus, in the repository of ELOs, the presence of ELOs with associated knowledge defined in terms of ontologies not present in the application, is permitted.
  - *Ontologies access*. The ontologies present in the Ontologies repository are depicted to the user.

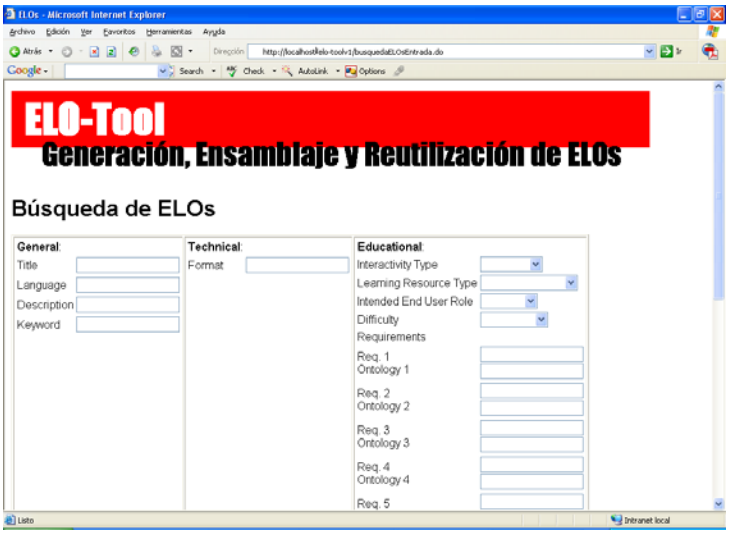

Figure 6. ELOs search in ELO-Tool

- The *LOs translation* subsystem is used to translate the learning objects, generated using external tools and described with different conceptual data schemas (e.g. IMS (IMS, 2002), LOM, DCMI and ARIADNE (ARIADNE, 2000)), to the ELO format. Then, the translated ELOs are stored in the ELO repository. Thus, they can be used later in reuse or assembly processes. Because there is no associated knowledge in the conceptual data schema of the input LOs, they are translated into ELO IU conceptual data schema.

- The *ELOs assembly* subsystem implements the assembly mechanism included in the OntoGlue approach. Figure 7 depicts the screenshot depicted to the user after the assembly process of two ELOs. In this example, both ELOs can be assembled. Therefore, some meta-data of the resulting ELO have been automatically defined taking into account the assembled ELOs. The remaining meta-data have to be provided manually by the user. Note that the ELO-Tool does not manage the merging of the data referenced by the LOs. Only an arrangement in sequence of the resources is made. If data requires to be merged, the use of an external tool is necessary. As regards meta-data of the assembled ELO, they are defined using several rules described in terms of *XPath* (W3C, 2004b) expressions. There are three types of rules: (i) those that automatically calculate the values of the meta-data of the assembled ELO (e.g. the requirements and competencies); (ii) those that require the provision of the meta-data values of the assembled ELO by the user (e.g. the educational description); and (iii) those that provide a tentative value for the meta-data of the assembled ELO that can be updated and revised by the user (e.g. the title). Thus, the resulting ELO is *elaborated* (Mellor, 2004). This approach is common in those environments that rely on automatic transformations to generate new data from existing data (e.g. *Model Driven Architecture* (Mellor, 2004)).
  - The *ELOs repository* stores the ELOs created by or loaded in the application.
  - The *Ontology repository* stores the ontologies loaded in the application.
  - The *XML schema repository* stores the XML schemas of the ELOs, the different conceptual data schemas that can be translated into ELO schema, and the OWL-Lite Schema.

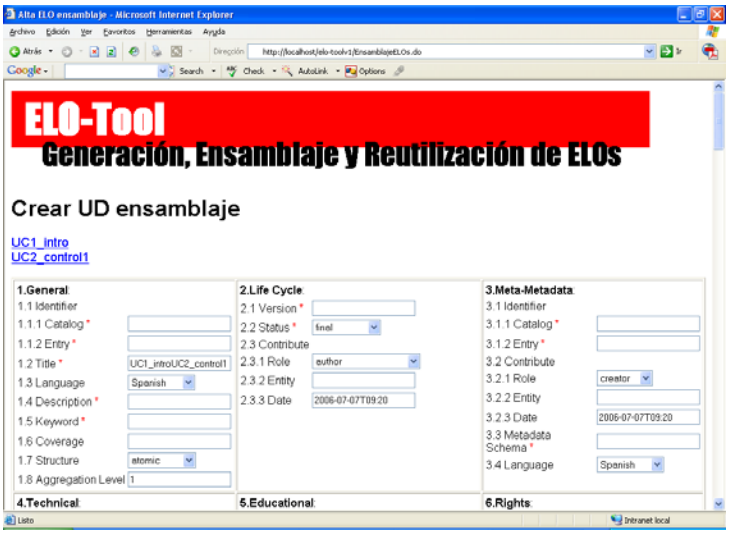

Figure 7. Result of an assembly process

Implementation Technologies

The ELO-Tool is implemented using Java technologies (Brown, 2001). This implementation makes extensive use of different *Application Program Interfaces* (APIs) which have facilitated the development of the tool. This section enumerates these technologies:

- *Jakarta Struts*. The ELO-Tool was designed according to a *Model-View-Controller Architecture* (MVC) (Brown, 2001). Jakarta Struts (Goodwill, 2004) was used to implement this architecture. Struts framework allows the creation of web applications according to MVC architecture combining JSPs, customized tags and Java Servlets (Goodwill, 2004). Thus, this framework allows us to separate the user interfaces aesthetics from the system logic, achieving modularity and easy maintenance of the application.
- *OWL Lite*. The OWL (*Web Ontology Language*) (W3C, 2004a) language, is a markup language for the publication of ontologies on the web. It allows us to define the classes and instances which belong to an ontology, as well as the mappings established between ontologies. In conjunction with the standards for the semantic web and technologies like RDF (W3C, 2000), this technology provides a framework which allows us to share information in a dynamic and flexible way. The OWL language is divided into three sublanguages: *OWL-Lite* (the most simple), *OWL-DL* (fairly complex) and *OWL-Full* (the most complex). In the ELO-Tool, we

have used the *OWL-Lite* language, to define the ontologies, and to establish hierarchical relations between their classes via mappings.

- *Jena API*. Jena (Jena, 2004) is a framework to build applications for the semantic web which allows the management of OWL documents. Since one of the functionalities of the ELO-Tool is the management of ontologies, Jena represents a tool which facilitates this task.
- *Xerces-DOM*. DOM (*Document Object Model*) (DOM, 1998), is a W3C specification, which represents an XML document as a hierarchical tree in the memory. The information contained in the tree can be handled using the API DOM. There are several implementations of this API. We have selected *Xerces* (Xerces, 2006) because it is one of most commonly used and stable API. Xerces has been used extensively throughout the application to manage the XML documents (i.e. LOs, ELOs and ontologies).
- *XSLT*. XSLT (*XSL Transformations*) (W3C, 1999), is a W3C specification, which allows us to transform document instances conforming to one meta-data schema into document instances conforming to another meta-data schema. Thus, XSLT has been used to translate meta-data schemas from other learning objects initiatives (e.g. Dublin Core Metadata Initiative) into ELO format (i.e. information units).
- *MySQL*. We selected the relational data base management system MySQL (MySQL, 2006) as technology for the underlying support of the system's entities and their concurrent access. The possibility of having data tables favors the speed of access and provides maximum flexibility for their management. Given that the ELO-Tool is a web tool, MySQL is a suitable choice as it is an easy to use technology as well as fast and safe with high connectivity.

Conclusions and Future Work

At present, e-learning has acquired great importance in both computing and teaching disciplines. In particular, learning objects are one of the key elements in e-learning. However, in spite of the presence of a plethora of learning content management systems, there is no conceptual framework or system able to provide a consistent assembly mechanism for learning objects.

This paper describes the OntoGlue mechanism that enhances the LOM conceptual data schema with associated knowledge (i.e. requirements and competencies). Thus the ELOs are defined. This associated knowledge permits a coherent assembly mechanism in terms of the requirements and competencies of the assembled learning objects. Furthermore, OntoGlue describes the resultant meta-data schema of the assembled learning objects, as well as helping to provide their values.

In order to permit a semantic comparison of associated knowledge during assembly and search processes, OntoGlue also uses classes of ontologies, possibly related by mappings, as descriptors of requirements and competencies. Thus, the semantic web approach is brought to the e-learning domain.

The OntoGlue conceptual framework was implemented in the ELO-Tool learning content management system. Thus, an XML-binding was provided in order to characterize the ELOs conceptual data schema in terms of an XML data model. In addition, the assembly mechanism was properly implemented using several technologies.

The ELO-Tool is mainly focused on the generation, assembly and search for ELOs, while ontology management is restricted to loading and removal. Thus the development of the ELO-Tool has been simplified substantially. On the other hand, this decision makes the use of external tools necessary for the generation and maintenance of ontologies.

As regards future work, although the development of the ELO-Tool has allowed us to verify the functionality and technical viability of our approaches, more extensive field trials are necessary to improve the characteristics of our system. We have also identified other interesting lines of research. Firstly, the study and definition of agents for the dynamic composition of customized content. Intelligent systems permit their users to provide rules that not only determine the content dynamically, but also the objectives, adapting the content selection to every stage and reinforcing the interaction of the user with the system. Secondly, the study and definition of ontology management environments with visual interfaces that allow us to create, modify and establish relations between them. Thirdly, the study and definition of e-learning environments that permit the management of learning objects with a different quality of service across wireless networks. Finally, we are involved in an exhaustive evaluation of learning objects in the ELO-Tool environment, in terms of their behaviour throughout their life cycle.

Acknowledgements

El Ministerio de Educación y Ciencia de España (MOSAIC TSI2005-08225-C07-01, MetaLearn TIN2004-08367-C02-02 and OdA Virtual TIN2005-08788-C04-01), La Dirección General de Universidades e Investigación de la Consejería de Educación de la Comunidad de Madrid and La Universidad Complutense de Madrid (Grupo de investigación consolidado 910494) have supported this work.

References

Answers (2006). Answers.com, retrieved January 15, 2008 from http://www.answers.com/Transitive%20Closure.

ARIADNE (2000). Alliance of Remote Instructional Authoring and Distribution Networks for Europe, retrieved June 30, 2007 from http://www.ariadne-eu.org/.

Arlow, J., & Neustadt, I. (2005). *UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design. Second Edition*, Addison-Wesley.

Barritt C., Lewis D., & Wieseler W. (1999). *CISCO Systems Reusable Information Strategy Version 3.0*, retrieved July 1, 2007 from http://www.cisco.com.

Berners-Lee T. (2002). *The Semantic Web*, retrieved January 15, 2008 from http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21.

BIREME (2006). *Bireme Home page*, retrieved January 15, 2008 from http://www.bvsalud.org/local/Site/bireme/I/homepage.htm.

Brown, S., Burdick, R., Falkner, J., Galbraith, B., Johnson, R., Kim, L., Kochmer, C., Kristmundsson, T., & Li, S. (2001). *Professional JSP Second Edition*, Wrox Press.

Capuano N., Gaeta M., Micarelli A., & Sangineto E. (2002). An Integrated Architecture for Automatic Course Generation. *Proc. of the IEEE International Conference on Advanced Learning Technologies ICALT 2002,* 322-326.

Campbell, L. M. (2003). Engaging with the learning object economy. In A. Littlejohn (Ed.), *Reusing online resources: A sustainable approach to e-Learning*, London: Kogan Page.

DiNitto E., Mainetti, L., Monga, M., Sbattella, L., & Tedesco, R. (2006). Supporting Interoperability and Reusability of Learning Objects: The Virtual Campus Approach. *Educational Technology & Society, 9* (2), 33-50.

DCMI (2004). *Dublin Core Metadata Initiative*, retrieved January 15, 2008 from http://dublincore.org.

DOM (1998) *Document Object Model (DOM)*, retrieved January 15, 2008 from http://www.w3.org/DOM/DOMTR

Farrell R., Liburd S., & Thomas J. (2004). Dynamic assembly of learning objects. *Proc. of the 13th international World Wide Web conference*, New York, NY: ACM Press, 162-169.

Genesereth, M. R., & Nilsson, N., (1987). *Logical Foundations of Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann.

Goodwill, J., & Hightower, R. (2004). *Professional Jakarta Struts*, Indianapolis, Indiana: Wiley Publishing.

Gruber, T. (2004). What is an Ontology? Retrieved January 15, 2008 from http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

HSD (2006). *Health Science Descriptors - Tree structure*, retrieved January 15, 2008 from http://decs.bvs.br/I/decs2006i.htm.

IEEE/ACM (2001). *IEEE/ACM Joint Task Force on Computing Curricula - Computing Curricula 2001 - Computer Science,* retrieved January 15, 2008 from http://www.computer.org/portal/cms_docs_ieeecs/ieeecs/education/cc2001/cc2001.pdf.

IMS (2002). *IMS Global Learning Consortium Inc. LRMS Version 1.2,* retrieved January 15, 2008 from http://www.imsproject.org/metadata/index.html.

IMS (2003). *Simple Sequencing Specification,* retrieved January 15, 2008 from http://www.imsglobal.org/simplesequencing/index.html.

Ip A. (2005). *Let's take some action*, retrieved January 15, 2008 from http://elearningrandomwalk.blogspot.com/2005/12/lets-take-some-action.html.

Jena (2004). *A Semantic Web Framework for Java*, retrieved January 15, 2008 from http://jena.sourceforge.net/.

Koper R. (2004). Use of the Semantic Web to Solve Some Basic Problems in Education: Increase Flexible, Distributed Lifelong Learning, Decrease Teachers' Workload. *Journal of Interactive Media in Education*, 2004 (6), retrieved January 15, 2008 from http://www-jime.open.ac.uk/2004/6.

L'Allier J. (1998). *NETg precision skilling: The linking of occupational skills descriptors to training interventions*, retrieved January 15, 2008 from http://www.im.com.tr/skilling.htm.

LOM (2003). LOM (Learning Object Metadata), retrieved January 15, 2008 from http://ltsc.ieee.org/wg12/20020612-Final-LOM-Draft.html.

Marieb, E. N. (1999) *Essentials of Human Anatomy & Physiology*. *6th edition*, Benjamin-Cummings.

Mellor, S. J, Scott, K. ,Uhl, A., & Weise, D. (2004). *MDA Distilled: Principles of Model-Driven Architecture*, Addison-Wesley.

Min, W. K., Kim, Y. S., Kim, J. Y., Park, S. P, & Suh, C.K. (1999) Atherothrombotic Cerebellar Infarction Vascular Lesion-MRI correlation of 31 cases. *Stroke, 30* (11), 2376-2381.

MySQL (2006). MySQL, retrieved January 15, 2008 from http://dev.mysql.com/doc/.

Robson, R. (2006). Reusability and Reusable Design. In R. Reiser and J.V. Dempsey (Eds.), *Trends and Issues in Instructional Design and Technology (2nd Ed.)*, Prentice-Hall.

Santacruz-Valencia L. P., Aedo I., & Delgado Kloss, C. (2003a). A Framework for the Creation, Integration and Reuse of Learning Objects. *Learning Technology Newsletter*, 5 (1), retrieved January 15, 2008, from http://lttf.ieee.org/learn_tech/issues/january2003/index.html.

Santacruz-Valencia L. P., Aedo I. & Delgado Kloss, C. (2003b). Designing Le@rning Objects with the ELO- Tool. *Proc. of the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03)*, 372-373.

Santacruz-Valencia, L. P., Aedo I., & Delgado Kloss, C. (2003c). A Proposal for the Composition of Learning Objects using Didactical Meta-data. *Proc. of Second International Conference on Multimedia Information and Communication Technologies in Education (m-ICTE 2003)*, 1, 415-419.

Santacruz-Valencia L. P., Navarro A., Aedo I, & Delgado Kloss, C. (2005). An Ontology-based Mechanism for Assembling Learning Objects. *Proc. of E-Learning on Telecommunications (ELETE 2005)*, IEEE Computer Society, 472-477.

SCORM (2004). *SCORM CAM (Content Aggregation Model)*, retrieved January 15, 2008 from http://www.adlnet.org/.

Shier, D.., Butler, J.L., Lewis, R. (2006). *Hole's Essentials of Human Anatomy and Physiology, 11th edition*, McGraw-Hill.

SWEBOK (2005). *SWEBOK Guide of the Software Engineering Body of Knowledge*, retrieved January 15, 2008 from http://www.swebok.org.

Verbert, K., Jovanovic, J., Duval, E., Gasevic, D., & Meire, M. (2006). Ontology-Based Learning Content Repurposing: The ALOCoM Framework. *International Journal on E-Learning, 5* (1), 67-74.

Wagner, E. (2002). Step to Creating a Content Strategy for your Organization. *eLearning Developers' Journal. eLearning Guild*, retrieved January 15, 2008 from http://www.elearningguild.com/pdf/2/102902MGT-H.pdf.

Wikipedia (2006). *Ontology (Computer Science)*, retrieved January 15, 2008 from http://en.wikipedia.org/wiki/Ontology_(computer_science).

Wiley D., A. (2002). *The Instructional Use of Learning Objects*, retrieved January 15, 2008 from http://www.reusability.org/read/.

Wille C., Abran A., Desharnais J., M., & Dumke R., R. (2003). *The Quality Concepts and Subconcepts in SWEBOK: An Ontology Challenge*, retrieved January 15, 2008 from http://profs.logti.etsmtl.ca/jmdeshar/SiteWQ/Publications/Swebok_Quality_Ontology.pdf.

W3C (1999). *XSL Transformations*, retrieved January 15, 2008 from http://www.w3.org/TR/XSL.

W3C (2000). *RDF (Resource Description Framework)*, retrieved January 15, 2008 from http://www.w3.org/RDF/.

W3C (2003a). *XML (eXtensible Markup Language)*, retrieved January 15, 2008 from http://www.w3.org/XML/.

W3C (2003b). *XML Schema*, retrieved January 15, 2008 from www.w3.org/XML/Schema.

W3C (2004a). *OWL Web Ontology Language*, retrieved January 15, 2008 from http://www.w3.org/TR/OWL-features.

W3C (2004b). *XPath (XML Path Language)*, retrieved January 15, 2008 from http://www.w3c.org/TR/xpath.

W3C (2006). *World Wide Web Consortium*, retrieved January 15, 2008 from http://w3.org.

Xerces (2006). *Xerces DOM*, retrieved January 15, 2008 from http://xerces.apache.org/xerces2-j/dom.html.