

Versioning of Learning Objects

Christopher Brooks, John Cooke, Julita Vassileva
ARIES Laboratory, Computer Science Department
University of Saskatchewan
Saskatoon, SK, S7N 5A9 Canada
cab938@mail.usask.ca, {cooke, jiv}@cs.usask.ca

Abstract

This paper outlines the issues associated with creating derivative works based on learning objects in a general manner, and discusses the support that exists within current metadata specifications.

1. Introduction

Learning objects are reusable pieces of educational material intended to be strung together to form larger educational units such as activities, lessons, or whole courses. These materials are stored in learning object repositories which can be distributed in nature (e.g. [5]). Objects are then retrieved and integrated into learning management systems to be delivered to the learner. Compared to traditional educational materials (e.g. film, books), electronic learning objects are easily modified. The distributed nature and highly mutable nature of learning objects make it very difficult to keep consistent version information. This paper discusses some of the issues involved, and introduces a metadata model with which to address them.

In the context of electronic documents the act of creating and maintaining versions is typically known as *versioning*. Versioning is a well studied area of computer science, and has been successfully applied in the disciplines of software configuration management, knowledge representation, and hypermedia. A core component of versioning is the version model which identifies the *kinds of artifacts* to be versioned, the *properties associated with those artifacts*, and the *changes that can be applied* to artifacts to result in a version change [2].

In practice, most version control systems also capture the *nature of changes* as they are applied to artifacts under version control. These changes can be captured both as a collection of syntactical operations (typically called a *delta*), and as a collection of semantic operations. Semantic operations can be captured both in a computer understandable manner as well as a human readable manner (typically called a *change log*).

Learning objects are an ordered aggregation of content, which are often referred to as reusable information objects [1]. The content can be made up of

any kind of media ranging from printed word to interactive Java Applets, though for practical purposes traditional forms of content that can not be easily integrated into an online course are not considered. There are no specific limitations on the granularity of the content, though generally content ranges from a single demonstration to a chapter or unit of material. Learning objects are annotated with metadata, which are a set of key-value tuples that describes the content of the learning object as a whole. Each key is associated with a value that may be constrained by some vocabulary. Further, models often capture the relationships between keys. Finally, depending on the metadata model used, metadata can be realized in a number of different data file formats. Web-enabled markup languages such as the HyperText Markup Language (HTML), the Extensible Markup Language (XML), and the Resource Description Framework (RDF) are most frequently used.

2. Change in learning objects

Learning objects are meant to be shared, repurposed with the help of an instructional designer, and reused in different learning environments. With no set granularity size for a learning object, it is highly unlikely that all learning objects will be used “out of the box” for a given course. Instead, they will be tailored towards an organization’s needs by human authors. Using the description of a learning object given above, this tailoring consists of four actions:

- Adding content
- Subtracting content
- Modifying content
- Reordering content

Each of these actions is made up of a number of more discrete syntactic operations that may take place when content is being manipulated. For instance, an author may add lines to a text file, insert elements into an XML file, or remove slides from a PowerPoint presentation.

The semantic nature of each change applied to a learning object is of principle interest to entities interested in comparing versions of objects. Foremost in this category are instructional designers and intelligent software agents who create courses for learners or groups of learners. Given some learning object *lo1*, there are at

least four states that the application of some delta $d1$ could cause when creating a derivative learning object $lo2$:

1. $lo1$ and $lo2$ are functionally equivalent: The meaning of the content or metadata is unchanged between the two versions.
2. $lo1$ is a subset of $lo2$: The new learning object contains all of the information from the one it was derived from, as well as new information.
3. $lo1$ is a superset of $lo2$: The new learning object contains only material from the one it was derived from, but does not contain all of the material from the one it was derived from.
4. There exists some non-null intersection between $lo1$ and $lo2$. There is some overlap between the two learning objects, but one is not a proper subset or superset of the other.

One change within a learning object can influence the understanding of that learning object in a number of different ways. Consider an online tutorial about database systems built for students in a final year undergraduate level course. An instructor for a graduate course on database systems may well adopt this learning object but take out any information not explicitly on relational database systems, and use it as an introduction. This reworking of the learning object has broadened its audience while restricting the topics that it is relevant to. Thus, the nature of change must be captured with respect to different perspectives.

3. A Metadata Model for Versioning

Support for the versioning of learning objects comes through metadata specifications. The most mature and comprehensive metadata specifications for learning objects are the LOM [4] and the Dublin Core Metadata Initiative [3]. Both of these specifications contain support for capturing the fact that a relationship exists between two learning objects, the status of this relationship in a human readable fashion and whether a version is a variant or a revision. However, neither of these specifications provides a method for capturing the syntactic changes that occur when derivative learning objects are created. They also neglect to codify the nature of change between learning objects and link this nature to a given perspective.

It is useful to describe a metadata model for versioning that maps a given delta more generally to the metadata that describes a learning object. This has two advantages; it eliminates the need for an instructional designer or software agent to have access to previous versions of a learning object when reasoning about its fitness for purpose, and it allows an instructional designer or software agent to understand the nature of changes without knowing the details of the vocabulary being used to describe the learning object. Further, any fully featured version control system must capture the syntax so that

functions such as the merging and rolling-back of learning objects can be supported.

4. Conclusions and Future Work

While current metadata specifications have some support for the versioning of learning objects, they are lacking in a number of different areas. They are unable to capture syntactical changes, making it impossible to provide common version control features (e.g. roll-backs), and they restrict the semantic understanding of versioning change to the comparison of metadata records. This requires that software agents and instructional designers who wish to reason about different versions of learning objects are required to have access to both of the learning objects and be able to understand the vocabularies used to describe those objects.

Instead, we are developing a model and framework for capturing both the syntactical and the semantic changes that occur when learning objects are versioned. This model builds off of the current e-learning metadata specifications, and represents information in a manner friendlier to software agents. Specifically, it allows for agents to better reason about versioning changes between learning objects even if the vocabularies being used to describe the objects are not known. In addition, it allows enabled learning object repositories to provide a higher level of versioning services (e.g. roll-backs, branching, etc).

We are investigating the use of human-in-the-loop techniques to capture semantic changes within a learning object authoring environment. As an author modifies a learning object, he or she can provide good cues about how that object has been changed. Further, we have developed a simple model of the syntactical elements that make up a learning object. This representation is required in order to develop a common vocabulary for codifying syntactical change operations.

References

- [1] Barritt, C. and Lewis, D., *Reusable Learning Object Strategy*, Cisco Systems, Inc., 2002.
- [2] Conradi, R. and Westfechtel, B., "Version Models for Software Configuration Management," *ACM Computing Surveys*, vol. 30, no. 2, June 1998 pp. 232-282.
- [3] Dublin Core Metadata Initiative, *Dublin Core Metadata Initiative (DCMI)*, <http://dublincore.org/> (current January 8, 2003).
- [4] *IEEE P1484.12.1-2002, Draft Standard for Learning Object Metadata*, IEEE, Inc., 2002.
- [5] Richards, G. and Hatala, M., "POOL, POND and SPLASH - A Peer to Peer Architecture for Learning Object Repositories," *Proceedings of Internet 2 Conference, Workshop on Collaborative Computing in Higher Education: Peer-to-Peer and Beyond*, 2002.