Design of an Extensible Metadata Editor Based on RDF

José Paulo Leal[1]
Ricardo Queirós[2]

[1] CRACS/INESC-Porto & DCC/FCUP, University of Porto, Portugal
[2] CRACS/INESC-Porto & DI/ESEIG/IPP, Porto, Portugal

zp@dcc.fc.up.pt
ricardo.queiros@eu.ipp.pt

Abstract: The content of a Learning Object is frequently characterized by metadata from several standards, such as LOM, SCORM and QTI. Specialized domains require new application profiles that further complicate the task of editing the metadata of learning object since their data models are not supported by existing authoring tools. To cope with this problem we designed a metadata editor supporting multiple metadata languages, each with its own data model. It is assumed that the supported languages have an XML binding and we use RDF to create a common metadata representation, independent from the syntax of each metadata languages. The combined data model supported by the editor is defined as an ontology. Thus, the process of extending the editor to support a new metadata language is twofold: firstly, the conversion from the XML binding of the metadata language to RDF and vice-versa; secondly, the extension of the ontology to cover the new metadata model. In this paper we describe the general architecture of the editor, we explain how a typical metadata language for learning objects is represented as an ontology, and how this formalization captures all the data required to generate the graphical user interface of the editor.

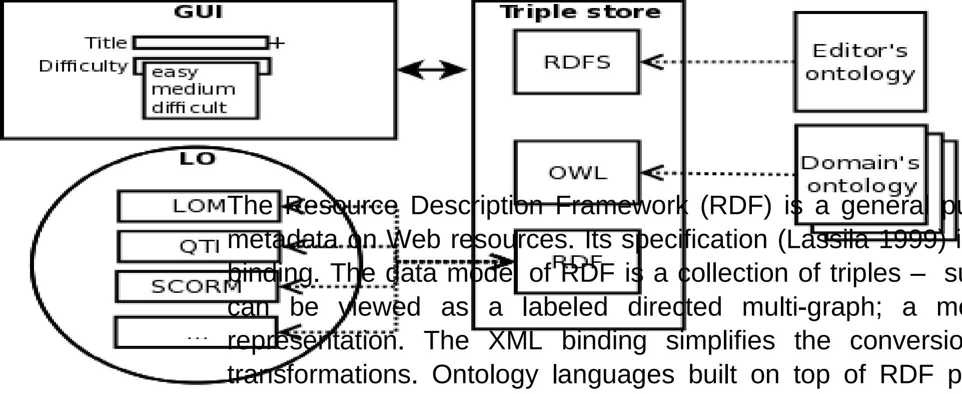
Keywords: e-Learning; ontology; RDFS; OWL.

1.Motivation and Related work

The purpose of standards in e-Learning is to ensure interoperability and reusability. To achieve these goals the organizations concerned with the development of e-Learning rely on the standardization of learning objects (Friesen, 2005). The most widely used standard for learning objects is the IMS Content Packaging (IMS CP, 2010). This content packaging specification uses an Extensible Markup Language (XML) manifest file wrapped with other resources inside a zip archive. The manifest includes the IEEE Learning Object Metadata (LOM) standard to describe the learning resources included in the package.

The IEEE/LOM standard is being used in several e-Learning projects all over the world (Godby, 2004). It is composed of a set of 77 optional elements, distributed among nine categories. Since LOM elements are optional and in some cases too generic, many projects define their own application profile to meet the needs of a specialized domain (Godby, 2004). In this context the term *application profile* refers to "the adaptation, constraint, and/or augmentation of a metadata scheme to suit the needs of a particular community" (IMS Application Profile). A well know e-Learning application profile is SCORM (SCORM) that extends IMS CP with more sophisticated sequencing and Contents-to-LMS communication.

The growing popularity of learning objects lead to the development of specialized editors supporting e-Learning metadata (Norm Friesen, 2005) such as RELOAD, Aloah II, eRIB, Paloma, LOMPad and SHAME. RELOAD (RELOAD 2010) is arguably the most developed of these project and is available both as a standalone Java application and as an Eclipse plugin. It supports a broad range of metadata formats, including IMS metadata, IEEE/LOM, IMS Content Packaging 1.1.4, SCORM 1.2, and SCORM 2004, but cannot be extended to support new languages. The majority of the editors cover a variable set of application profiles, including those supported by RELOAD and also NORMETIC and CanCore. The SHAME project - Standardized Hyper Adaptible Metadata Editor – stands out from the rest since it is actually an RDF editor with support forLOM.



The Resource Description Framework (RDF) is a general purpose language for representing metadata on Web resources. Its specification (Lassila 1999) includes a data model and a XML binding. The data model of RDF is a collection of triples – subject, predicate and object - that can be viewed as a labeled directed multi-graph; a model well suited for knowledge representation. The XML binding simplifies the conversion to other formats using XML transformations. Ontology languages built on top of RDF provide as formal way to encode knowledge about specific domains, including reasoning rules to process that knowledge (Corcho et al, 2000). In particular, RDF Schema (Brickley and Guha, 2004) provides a simple ontology language for RDF metadata that can be complemented with the more expressive constructs of OWL (Nilsson et al., 2003). Both RDF Schema and OWL are represented as RDF triples that can be persisted in specialized databases known as *triplestores*. In summary, the generality of RDF and the expressiveness its ontology languages play an instrumental role in bringing uniformity to metadata editing.

The generality of RDF is exploited by the previously mentioned SHAME project. To support learning object metadata they created a LOM/RDF binding (Nilsson et al., 2003) and similar bindings could be created other metadata formats. In their approach the LOM/RDF binding is complemented with annotation profiles to generate a LOM metadata editor. Annotation profiles define how RDF metadata is read and modified, what input is allowed (e.g. multiplicity and vocabularies), and how it is presented (e.g. order, grouping and labels). OLR-3 (Kunze et al., 2002) uses also RDF to edit metadata from multiple formats and is also schema driven. Other RDF editing tools, such as RdfEditor, Rhodonite, and RDFe, are restricted to the RDF edition and do not support any metadata domain definition.

Our approach to metadata editing shares with SHAME the use of RDF as a basis for metadata editing. Nevertheless, instead of creating a new formalism for binding a data model to the editor, such as their annotation profile, we use ontologies. The expressiveness of ontologies is enough to group and label properties, define their multiplicity, create controlled vocabularies and other definitions required to describe a e-Learning metadata domain. Moreover, ontologies simplify the binding of a metadata formalism to RDF. For instance, in the LOM/RDF binding (Nilsson et al., 2003) the authors of SHAME require the use of containers (e.g. RDF Bag) to deal with properties with multiple values, such as translations to different languages. Since RDF properties can be repeated it is easier and more natural to control their repetition in an ontology using the Owl cardinality restrictions.

2.Design of the editor

There are several learning objects standards supported by e-Learning systems. The most widely used is the IMS CP which uses LOM to describe the learning resources included in a package. Since IMS CP was designed to be straightforward to extend, meeting the needs of different user communities, several application profiles were created. This resulted in the coexistence of several metadata formats and its syntactical differences inside of a single learning object, making it hard for tools to visualize, edit or even query a learning object metadata in a uniform way.

In this context, the creation of an editor covering several metadata languages, easily configurable to support new languages for domain-specific metadata, assumes an important role. The easiest way to support multiple languages is to reduce them to a common formalism. RDF is a natural candidate since it has a semantics suited for knowledge representation and has an XML binding. By operating directly over the metadata semantics we are able to avoid syntactical peculiarities of each formalism. The XML binding ensure a simple conversion from and to every metadata format, since XML is a formalism supported virtually by all metadata formats.

Figure 1: Overall architecture of editor

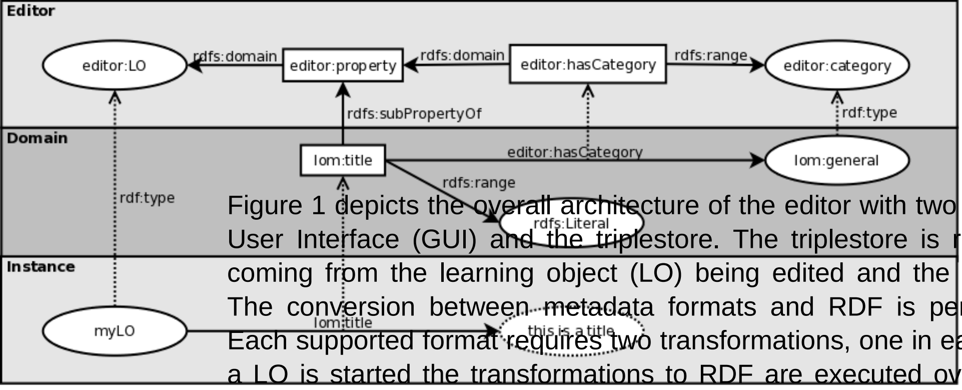


Figure 1 depicts the overall architecture of the editor with two main components. The Graphical User Interface (GUI) and the triplestore. The triplestore is responsible for holding RDF data coming from the learning object (LO) being edited and the ontologies configuring the editor. The conversion between metadata formats and RDF is performed by XSL transformations. Each supported format requires two transformations, one in each direction. When the edition of a LO is started the transformations to RDF are executed over the metadata contained in the manifest. The RDF triples collected this way is loaded into the triplestore, complementing the triples loaded from the ontologies. The GUI component queries the triplestore to generate a layout with controls defined by the ontology, populated with data coming from the LO. The interaction with the user updates data in the triplestore. To save the new version of the LO the data in the triplestore is exported to RDF and converted back to the original matadata formats

For each metadata domain is necessary to extend the ontology controlling the generation of the GUI. This ontology is structured in two layers:

Editor: Describes generic concepts common to all metadata. These concepts support the definition on concepts on the following layer.

Domain: Describes the properties in a metadata domain using the concepts in the top layer.

Figure 2: Semantic model

The graph in Figure 2 shows a fragment of the triplestore content, aligning the layered ontology with the RDF metadata for a single LOM element (lom:title) describing a particular LO (myLO). Each layer corresponds to a named graph loaded from different source, as introduced in Figure 1.

The top layer of ontology, labelled as *Editor*, describes the common concepts that will guide the creation of the editor and uses solely RDF Schema constructs. The `editor:LO` class that types LO instances. This class is the domain of the generic property `editor:property` that domain properties must extend. Properties can be grouped in categories using the `editor:hasProperty` that has the previous property as domain an the `editor:category` as range. With these definitions we can model the structure GUI of editor. For instance, the categories could be rendered as tabs and the properties as form controls.

The domain layer defines the properties of each supported metadata format and requires OWL DL constructs. Figure 2 shows how a LOM property – *title* – is defined by extending the `Editor:Property` introduced in the Editor layer. This is a simple property with literal values and repetition of values (consider the translations of title to different languages). More complex property definitions may require either a controlled vocabulary or limits to repeated values. These constraints are implemented with OWL enumerations and restrictions. An enumerated class description is defined with the `owl:oneOf` property. The OWL property value is a list of individuals (e.g. `lom:`*`easy`*, `lom:medium` and `lom:difficult`). To control repetition we use the cardinality restriction to limit the number of values that a property can take (e.g. 1). The title property is assigned to the `lom:General` category, introduced in this layer as resources of type `Editor:Category`.

The instance layer contains the learning object metadata. In this example, we defined a `Editor:Lo` resource – LO1 - with the property `lom:title`, defined in the LOM domain layer, with the value “this is a title”.

3.Conclusion and Future work

Editing metadata is in itself a challenging problem. Editing LO metadata is even more complex since a single LO typically contains metadata from several data models. We propose the use of RDF as a common metadata formalisms in which all LO metadata serialized in XML can easily be converted. We propose a method for binding LO metadata (LOM, QTI and other) to RDF that is simpler than the method available in the literature. We make use of ontological languages to describe metadata domains and relate them to editor's constructs.

At this stage we have a design for a metadata editor following the approach presented in the previous section, and we selected the tools for its implementation. The design requires an RDF *triplestore* and a GUI *toolkit*. To process with RDF triples, including LO metadata and the ontology, we selected Jena - an open source Java framework for building Semantic Web applications. The Jena triplestore can be queried and updated using SPARQL. To generate and manage the GUI we selected the Google Windowing Toolkit (GTW). These tools are both open source and have an API for Java, the language selected for implementing of the editor.

We are particularly interested in editing learning objects containing programming problems as a means to promote interoperability among systems using automatic evaluation. Thus, we plan to integrate this editor in crimsonHex (Leal and Queirós, 2009), an extensible and interoperable repository of learning objects.

References

ADL SCORM, [online], SCORM, http://www.adlnet.gov/Technologies/scorm

Dan Brickley and R.V. Guha (2004) “RDF Vocabulary Description Language 1.0: RDF Schema”, [online], RDF, http://www.w3.org/TR/rdf-schema/

Friesen, N. (2005) - Learning Object Metadata Editors, [online], Cancore, http://cancore.athabascau.ca/editors.htm

Godby, C.J. “What Do Application Profiles Reveal about the Learning Object Metadata Standard?” Ariadne Article in eLearning Standards, 2004.

IMS Application Profile Guidelines Overview, Part 1 - Management Overview, Version 1.0., [online], IMS Global, http://www.imsglobal.org/ap/apv1p0/imsap_oviewv1p0.html.

Kunze, T. and Brase, J. and Nejdl, W. (2003) “Editing learning object metadata: Schema driven input of rdf metadata with the olr3-editor”, Proceedings of the Semantic Authoring, Annotation & Knowledge Markup Workshop (SAAKM 2002) at 15th European Conf. on Artificial Intelligence, July 22-26, Lyon, France

Leal, José Paulo and Queirós, Ricardo (2009) "CrimsonHex: a Service Oriented Repository of Specialised Learning Objects", in "ICEIS 09 - 11th International Conference on Enterprise Information Systems, Milan, Italy" , Springer-Verlag, LNBIP, vol. 24, pp.102-113, May 2009.

Lassila, O. and Swick, R, (1999) “Resource Description Framework (RDF) Model and Syntax Specification", [online], W3C, http://www.w3.org/TR/PR-rdf-syntax/

Nilsson, M., Palmer, M., Brase, J. (2003) “The LOM RDF binding - principles and implementation” In Proceedings of the 3rd Annual ARIADNE Conference, Leuven, Belgium.

Smith, Michael K.; Chris Welty, Deborah L. McGuinness (2004). "OWL Web Ontology Language Guide". W3C. Retrieved 2008-07-15.

Oscar Corcho, Asuncion Gomez-Perez (2000), A Roadmap to Ontology Specification Languages, LNCS, ISSU 1937, pages 80-96.

Friesen, N. (2005) “Interoperability and Learning Objects: An Overview of E-Learning Standardization". Interdisciplinary Journal of Knowledge and Learning Objects. 2005.

IMS-CP – IMS Content Packaging, Information Model, Best Practice and Implementation Guide, Version 1.1.3 Final Specification IMS Global Learning Consortium Inc., [online], IMS Global, http://www.imsglobal.org/content/packaging.

RELOAD (2010) - Reusable elearning object authoring and delivery, [online], RELOAD, http://www.reload.ac.uk/