A framework for the implementation of Application Profiles in XML Schemas

Nicholaos Mourkoussis, Manjula Patel*, Martin White
N.Mourkoussis@sussex.ac.uk, M.Patel@ukoln.ac.uk, M.White@sussex.ac.uk
Centre for VLSI and Computer Graphics, University of Sussex, UK
*UKOLN, University of Bath, UK

Abstract

The concept of an application profile (AP) has been developed to allow implementers to draw on metadata terms from existing vocabularies and customise them for a local application. APs play an important role in enhancing interoperability between diverse applications. From our experience of encoding an AP targeted to the digital heritage domain, we have devised a generalized XML Schemas Definition (XSD) framework capable of satisfying the functional and modelling characteristics of APs with either flat or nested structures. This paper presents the framework and its technical implementation, its potential impact on the development of dynamic machine-processible APs, and its current limitations. The framework presented has a layered structure to explicitly separate the authoritative, the non-authoritative, and the application profile schemas. We believe this framework to be an important step in encoding APs that can be dynamically updated with information relating to the terms they reuse, directly from schemas on remote locations (e.g. the web), enabling the automatic creation and validation of AP instance records.

Keywords:
Application Profiles, XML Schemas, Metadata, Digital Heritage Application Profiles, Digital Libraries.

1 Motivation and related work

The notion of application profiles (APs) first emerged from the DESIRE project (Heery et al. 2000b). Since then they have attracted considerable attention in the digital libraries world, both in terms of their characteristics as well as their implementation in machine readable formats, such as XML Schema Definition (XSD) (Fallside 2001; Thompson et al. 2001; Biron and Malhotra 2001) and RDF Schema (RDFS) (Brickley and Guha 2004). In broad terms, APs are a type of element set that draws on metadata terms from existing vocabularies and customises them for a specific application (Heery and Patel 2000).

The collecting together of distinct vocabularies is by no means a new concept. In particular the Warwick Framework (Dempsey and Weibel 1996) proposed the packaging of metadata for its transfer between applications. However, whilst the Warwick Framework works with whole sets of metadata, APs work at a more granular level in order to allow the take-up of individual metadata terms as required. More recently, the aggregation of multiple vocabularies in order to provide a comprehensive set of metadata terms has been revived in the form of the Metadata

Encoding and Transmission Standard (METS 2004) which includes descriptive, administrative and structural metadata for complex digital resources.

Within the DESIRE and SCHEMAS (SCHEMAS 2000) projects the concept of APs was used as a means of disclosing terms that had been used in particular applications, to facilitate the reuse of terms (rather than reinvention) and thereby enhance the potential for interoperability between disparate systems. Additionally, the SCHEMAS project made significant early advances with regard to the formulation of APs in a machine processible format using RDFS (Baker et al. 2001). This work continued to mature in the MEG Registry project (Heery et al. 2002). Machine processible encoding of APs is a necessary requirement for automated data mining and querying across vocabularies; it helps in the process of making apparent emerging trends and patterns in metadata vocabulary and term usage, in turn aiding the process of consensus building.

Existing literature suggests that there are two main contenders for encoding an AP so that it is machine processible: XSD and RDFS. The XSD language provides a structured expression that supports validation of instance metadata. Using XSD, implementers are able to define explicit structural, cardinality and datatype constraints enabling the automatic validation of metadata records. On the other hand RDF schema (Baker *et al.* 2001) expresses relationships between terms, providing a data model for expressing the semantics of terms. As for APs, there is a need for XML schemas as a basis for the automatic validation of metadata records. However, semantic interoperability would seem also to require the use of RDF schemas for a more semantically rich AP. In this paper we acknowledge that a significant number of applications use XML technologies either because semantic web technologies, such as RDFS, have not been mature enough or simply because they do not require the functionality that such technologies offer. We therefore describe a framework for implementing an AP using XSD alone. Furthermore, we introduce a dedicated XSD encoding called 'Semantic information'. This encoding defines an extensible set of term usage attributes (CEN 2003; CEN 2005a; CEN 2005b). In this way we believe that we are partly able to address the requirement of providing support for semantic information using XSD alone. A detailed explanation of our approach is further discussed throughout the rest of the paper.

Hunter et al. (2001) attempted to combine XSD and RDF to satisfy an AP's encoding requirements. They suggested a web-based metadata architecture, which combines the best features of both XSD and RDFS. They suggested that the XSD encoding, for each element, should contain only local usage constraints and no semantic definitions such as the semantic descriptions inside the annotation and documentation tags. Additionally, they suggested that the RDF schema representation of each element should only contain semantic definitions. Therefore, they proposed two alternative methodologies to realize the combination of the two schema languages. In the first approach they incorporated the RDFS definitions into the XSD file, using annotation tags. The second approach involves XSD referencing to an external or remote RDFS namespace. However, both methods seem to require either the extension of existing parsers, or mechanisms for smoother integration of these two schema languages. More importantly, they propose a hybrid solution requiring both XSD and RDFS whereas our solution makes use of XSD alone. Furthermore, our framework could be modified to interact with a hybrid XML/RDF solution. This would probably lead to a similar approach to that adopted by Hunter et al. (2001), where they incorporated the RDFS definitions into the XSD file, using annotation tags as discussed above.

In the absence of guidelines, implementers of APs are likely to use a wide range of presentation formats and proprietary encodings to create their APs. Work of the European Committee for Standardization (CEN) reported that APs, in practice, are created for a wide range of purposes, including:

1. Documenting the semantics and constraints used for a set of metadata records.
2. Helping communities of implementers harmonize metadata practice among themselves.
3. Identifying emerging semantics as possible candidates for formal standardization.
4. Serving as guides for semantic crosswalks and format conversions.
5. Serving as specifications for formal encoding structures.
6. Interpreting or presenting legacy or proprietary metadata in terms of widely understood standards.
7. Documenting the rules and criteria according to which a set of metadata records was created.

The CEN work concludes with a series of reports that provides several guidelines on: how information should be structured and presented in Dublin Core APs (CEN 2003); how to name and maintain element declarations and APs (CEN 2005b); and finally suggests a machine readable representation of DCAP using the convention of RDFS (CEN 2005a).

Nagamori and Sugimoto (2004) proposed a metadata schema framework and functional extensions to the DCMI metadata schema registry to provide services related to metadata schemas and software tools for metadata schemas. In particular, these extensions include a cross-schema search function which associates metadata terms across multiple metadata element sets, an element extraction function which extracts common elements among multiple APs, and a software generator which produces software tools such as a metadata editor, a metadata search tool and a metadata database management tool. However, their approach is logically implemented with a combination of technologies such as (Relax NG, RDF, RDFS, DAML+OIL, OWL etc.). Additionally, this layered model was primarily introduced to separate syntactic and semantic features of metadata schema descriptions in order to clarify relationships among constructs of metadata schemas and to help cross-schema mappings for metadata interoperability.

In contrast to that work, our paper discusses a framework for the encoding of APs capable of being dynamically updated with information with respect to the terms they use directly from schemas on remote locations. The derived AP schemas could in turn be used for the automatic validation of XML instance metadata records in application repositories.

In this paper we acknowledge that a significant number of APs may rely for their implementation only on XSD. We therefore took into consideration the existing work of peers, mentioned previously, and implemented a generic XSD framework for the realization of APs in machine-readable format. This paper presents that framework and discusses its technical implementation and current limitations. The framework presented has a layered structure to explicitly separate the authoritative, the non-authoritative, and the application specific parts. The remainder of this paper is organized as follows. In Section 2, we outline the characteristics of APs and discuss the fundamental concepts upon which this work is based. In Section 3, we describe the generalized XSD framework for the implementation of APs. Section 4 discusses the implementation of two APs (i.e. AMS and IPL-ASIA) based upon the proposed XSD framework. Finally, section 5 discusses the framework's potential and current limitations.

2 Fundamental concepts

This section describes the fundamental concepts upon which the implementation of the XSD framework is based.

2.1 APs and the layered structure

The AP model differentiates between *element sets* and *application profiles*, as a means of distinguishing where and how terms are defined as opposed to how they are used and adapted in practice (Heery et al. 2002). Terms are defined in element sets as a means of unambiguously identifying them, in particular for re-use. APs draw on terms defined in one or more element sets and adapt them for use in particular applications. For example, an AP can refine the semantics of a standard term by making it narrower or more specific. APs are not allowed to define new terms; consequently other APs are disallowed from drawing terms from already specified APs in order to avoid semantic drift. Semantic drift occurs when a term is successively modified to the extent that its semantics no longer correspond to the term's original definition. Structuring of metadata vocabularies into element sets and APs encourages a layered organisation, making apparent which parts are available for re-purposing and which parts have been adapted from elsewhere. This is a crucial function for metadata interoperability since implementers can extend authoritative schemas in accordance to their requirements and at the same time they can enhance metadata interoperability.

Summarizing, APs are endowed with the following specific capabilities, they can:

- mix-and-match terms from multiple element sets
- specify dependencies (e.g. mandate encoding schemes)
- adapt existing definitions for local purposes
- declare rules for content (e.g. usage guidelines)
- specify whether an element is mandatory, optional or repeatable

2.2 Uniform Resource Identifier

The Uniform Resource Identifier (URI) (Berners-Lee et al. 1998) provides for unique identification of resources on the Web, it can be used to identify resources such as images, places, music, documents and people. More importantly here, it is used to uniquely identify the individual concepts, terms and relationships that constitute a vocabulary, so that it is possible to distinguish between entities with the same name label. The use of URIs for resource identification thus provides for a compelling and powerful decentralised architecture in which metadata vocabularies can be developed without the need for centralised coordination, but can nonetheless be referenced when necessary.

2.3 XML Namespace Mechanism

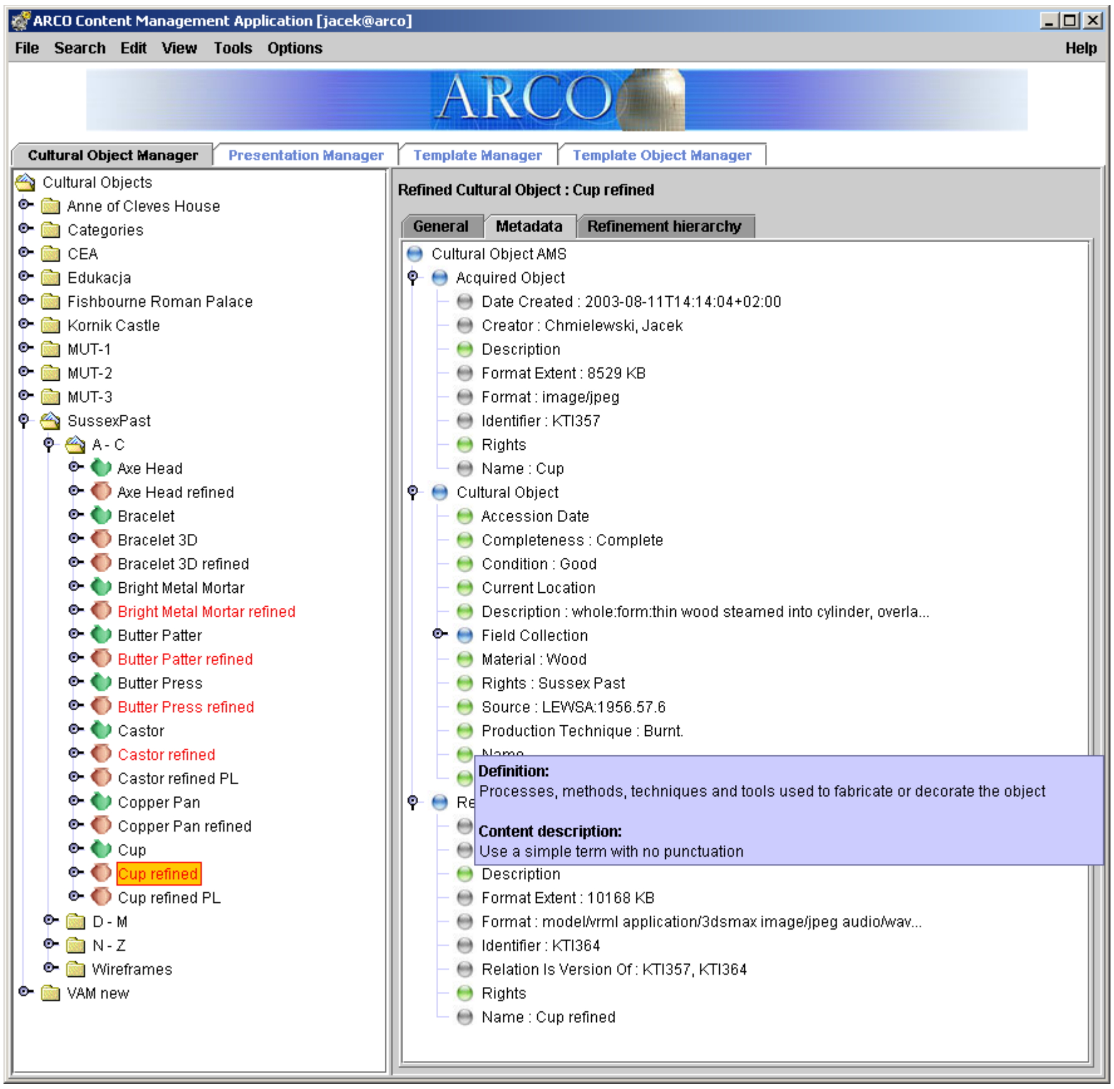
XML Namespaces were introduced into XML Schemas (Bray et al. 1999) as a way of unambiguously identifying the elements and attributes in an XML document. The XML Namespace mechanism allows elements and attributes in an XML document to be referenced in other XML documents. The XML Namespace mechanism together with the use of URIs allows metadata terms from multiple vocabularies to be referenced and thereby collected together to form an AP. Furthermore, DCMI namespace policy (Powell et al. 2001) dictates that changes of semantics in a term (such as changes in definition) are not directly applicable on the same namespace. Therefore, new semantics require a change of either the name or namespace for the term or terms in question. In order to satisfy the latter requirement in the proposed XSD framework (refer to section 3) we draw the term's datatype from the authoritative schemas into a non-authoritative XML schema and adapt its semantics at the non-authoritative layer under the new schema namespace.

Furthermore, whilst we have focused on the technicalities of implementing APs in XSD, it should be remembered that the mixing and matching together of individual terms from existing vocabularies may cause problems due to their being pulled out of a larger context. We believe this to be a modelling issue, which needs to be addressed during the design of the application's information model.

2.4 Principle of appropriate identification and metadata documentation

According to the principle of appropriate identification (CEN 2003), metadata terms should be identified and documented as precisely as possible. The preferred method for a term's identification is to cite its assigned URI. In practise though, not all authoritative metadata terms have a URI assigned. This makes the use of further descriptive attributes important, collectively known as term usage (CEN 2003). Term usage attributes include both required (i.e. Term URI, Defined By, Name, Label) and optional definitional attributes (i.e. Definition, Comments, Type of term), relational attributes (i.e. Refines, Refined By, Encoding Scheme For, Has Encoding Scheme, Similar To), and constraints (i.e. Obligation, Condition, Datatype, Occurrence).

Correct use of term usage attributes has the potential to support both users and tools in understanding and correctly using an XSD encoded AP. For instance, it is widely considered easier for people to understand descriptive rather than XSD encodings. Furthermore, implementers of AP and metadata schemas software editors may use this supplementary term usage information to automatically generate tool tips and other helper functions for their users. Figure 1 presents an example of a metadata editor that makes use of term usage attributes encoded inline with each term XSD encoding, to generate tool tips and other helpful information for users. This particular example refers to the ARCO (White et al. 2003) Metadata Schema AP, collectively known as AMS (Mourkoussis et al. 2003; Patel et al. 2005). The scope as well as the encoding of the AMS AP, according to the proposed framework will be discussed in Section 4 of this paper.


ARCO Content Management Application [jacek@arco]

File Search Edit View Tools Options Help

Cultural Object Manager | Presentation Manager | Template Manager | Template Object Manager

```
Cultural Objects
  Anne of Cleves House
  Categories
  CEA
  Edukacja
  Fishbourne Roman Palace
  Kornik Castle
  MUT-1
  MUT-2
  MUT-3
  SussexPast
    A - C
      Axe Head
      Axe Head refined
      Bracelet
      Bracelet 3D
      Bracelet 3D refined
      Bright Metal Mortar
      Bright Metal Mortar refined
      Butter Patter
      Butter Patter refined
      Butter Press
      Butter Press refined
      Castor
      Castor refined
      Castor refined PL
      Copper Pan
      Copper Pan refined
      Cup
      Cup refined
      Cup refined PL
    D - M
    N - Z
    Wireframes
  VAM new
```

Refined Cultural Object : Cup refined

General | Metadata | Refinement hierarchy

```
Cultural Object AMS
Acquired Object
  Date Created : 2003-08-11T14:14:04+02:00
  Creator : Chmielewski, Jacek
  Description
  Format Extent : 8529 KB
  Format : image/jpeg
  Identifier : KTI357
  Rights
  Name : Cup
Cultural Object
  Accession Date
  Completeness : Complete
  Condition : Good
  Current Location
  Description : whole:form:thin wood steamed into cylinder, overla...
  Field Collection
  Material : Wood
  Rights : Sussex Past
  Source : LEWSA:1956.57.6
  Production Technique : Burnt.
  Name
Re
  Description
  Format Extent : 10168 KB
  Format : model/vrml application/3dsmax image/jpeg audio/wav...
  Identifier : KTI364
  Relation Is Version Of : KTI357, KTI364
  Rights
  Name : Cup refined
```

Definition:
Processes, methods, techniques and tools used to fabricate or decorate the object

Content description:
Use a simple term with no punctuation

Figure 1: An example of tool tips generated dynamically from term usage attributes encoded with the AP (Patel et. al, 2005)

2.5 Descriptive header

A Descriptive Header (CEN 2003) places the AP into an interpretive context by specifying, at a minimum, a Title, Creator, Date, Identifier, and Description for the AP.

3 The Proposed XSD Framework

This section discusses the proposed XSD framework for encoding APs that can be dynamically updated with information on the terms they use directly from schemas at remote locations such as the web. First, we discuss the terminology used, then the building blocks (i.e. layers) of the framework, and finally guidelines for making the technical implementation of this framework possible.

3.1 Terminology

Table 1 presents and provides definitions for the terms used in the proposed XSD framework.

Table 1: Explanation of terms

Term	Description
Authoritative schemas:	Schema encodings, which are made available by an authoritative body such as DCMI (DCMI XML schemas 2004) or CIMI (CIMI XML Schema 2002).
Non-authoritative schemas:	An implementer provides these encodings locally when an XSD version of an authoritative element set is not available from the associated authoritative body. This term also caters for an application specific *element set*, which serves to introduce new terms, which cannot be satisfied by an existing standard.
AP schemas:	These encodings draw terms from authoritative and non-authoritative schema encodings.
Schemas container	In general an XSD encoding of a schema container makes possible the use of terms from schemas with different target namespaces. It also makes possible filtering out of the original namespaces of adopted terms. In this framework we make use of two schema containers: 'non-authoritative schema container' and 'application schema container'
Non-authoritative schemas container	The container schema that allows schemas in the 'non-authoritative' level to access data types or terms from the 'Authoritative' level schemas for customizing them according to the requirements of the AP.
Application schemas container	The container schema that allows the 'APs' schemas to reference terms of both the 'authoritative' and 'non-authoritative' schemas.
Term URI	A Uniform Resource Identifier used to identify the term (CEN 2003).
Name	A unique token assigned to the term (CEN 2003).
Label	A human-readable label assigned to the term (CEN 2003).
Defined By	An identifier of a namespace, pointer to a schema, or bibliographic reference for a document within which the term is defined (CEN 2003).
Definition	A statement that represents the concept and essential nature of the term (CEN 2003).
Comments	Additional information about the term or its application (CEN 2003).
Type of term	A grammatical category of the term (e.g. "Element", "Element Refinement", or "Encoding Scheme") (CEN 2003).
Refines	The described term semantically refines the referenced term (CEN 2003).
Refined By	The described term is semantically refined by the referenced term (CEN 2003)
Encoding Scheme For	The described term, an Encoding Scheme, qualifies the referenced term (CEN 2003).
Has Encoding Scheme	The described term is qualified by the referenced Encoding Scheme (CEN 2003).
Similar To	The described term has a meaning the same as, or similar to, that of the referenced term (CEN 2003).
Obligation	Indicates whether the element is required to always or sometimes

	be present (i.e., contain a value). Examples include "Mandatory", "Conditional", and "Optional" (CEN 2003).
Condition	Describes the condition or conditions according to which a value shall be present (CEN 2003).
Datatype	Indicates the type of data that can be represented in the value of the element (CEN 2003).
Occurrence	Indicates any limit to the repeatability of the element (CEN 2003).

3.2 The layered XSD framework

The XSD framework of our approach consists of five layers (see Figure 2) of XML schema encodings:

1. **Authoritative schemas** layer includes the XSD encodings, provided by the associated authoritative body, from which the AP will adopt terms.
2. **Non-authoritative schemas container** layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the import mechanism and provides access to terms and datatypes of XSD encodings of the 'authoritative', which might be later, used by the encodings of the 'non-authoritative' schemas layer.
3. **Non-authoritative schemas** layer includes the XSD encodings of non-authoritative XSD implementations of element sets in the event of such not being provided by the corresponding authority; application specific element sets; and finally XSD encodings of derived datatypes from the original datatypes of 'authoritative' schemas in order to create new terms with specified dependencies. At this level implementers may also adapt existing definitions for local purposes as well as define rules for content and other term usage attributes. This level also includes the XSD that defines the term usage attributes.
4. **Application schemas container** layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the <import> mechanism and provides access to terms existing in the 'authoritative' and 'non-authoritative' schemas, which will be later used by the AP schemas.
5. **Application profile schemas** layer consists of a number of XSD encodings that satisfy the data model of each AP. These encodings adopt terms from the XSD encodings of the 'authoritative' and 'non-authoritative' schemas layers through the use of the 'APs container' XSD encoding. At this level implementers may customise the cardinality constraints of the terms.

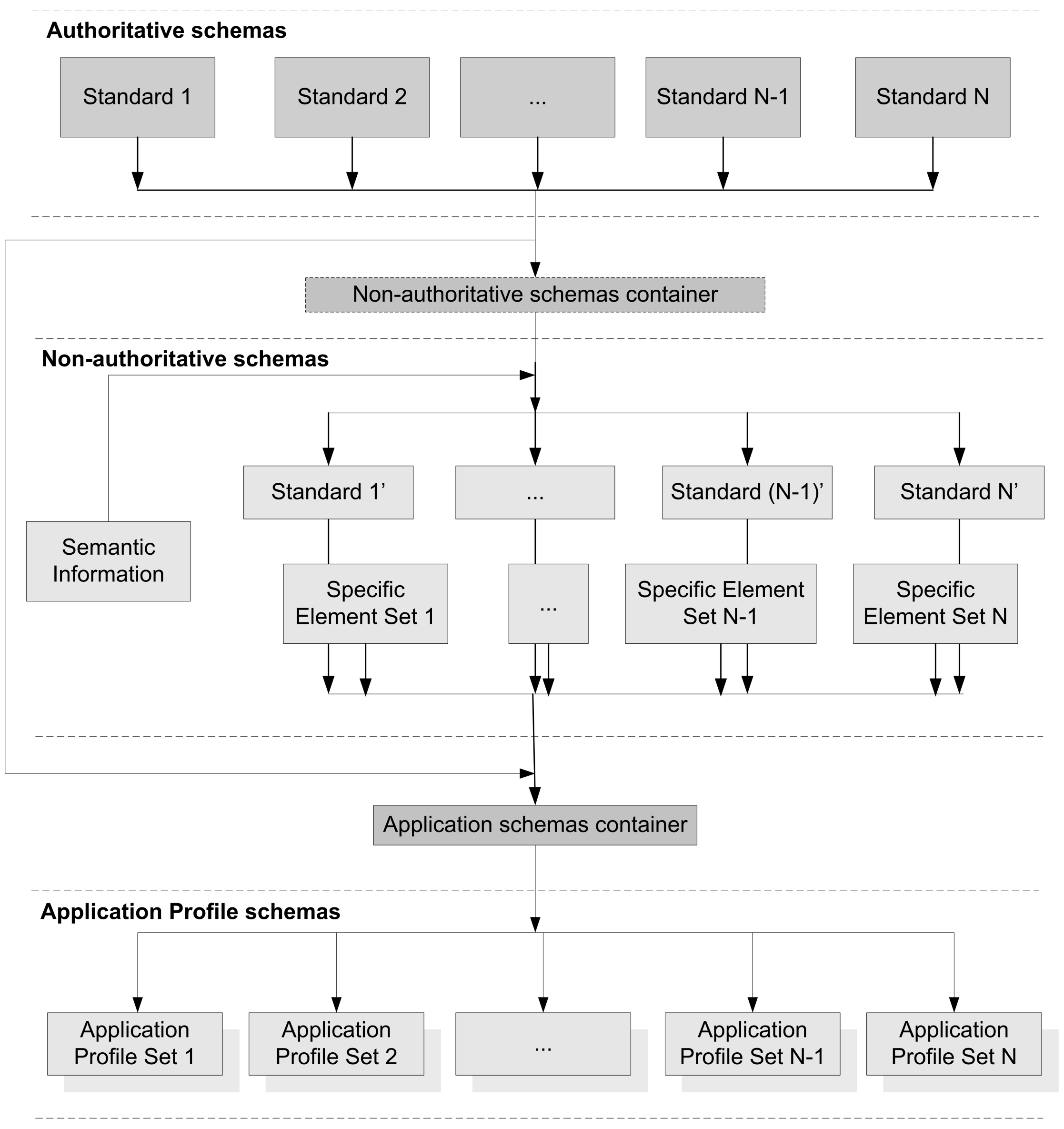

Authoritative schemas
Standard 1
Standard 2
...
Standard N-1
Standard N
Non-authoritative schemas container
Non-authoritative schemas
Standard 1'
...
Standard (N-1)'
Standard N'
Semantic Information
Specific Element Set 1
...
Specific Element Set N-1
Specific Element Set N
Application schemas container
Application Profile schemas
Application Profile Set 1
Application Profile Set 2
...
Application Profile Set N-1
Application Profile Set N

Figure 2: Generic XSD framework for the implementation of APs

3.3 General guidelines

A set of general guidelines are proposed that may be used for implementing the proposed XSD Framework:

1. This framework is based only on XSD encodings.
2. The framework allows implementers to use any number of XSD encodings in order to satisfy requirements of an AP both in terms of metadata terms and data models.
3. Implementers are advised to use the available encodings from authoritative initiatives, if these are provided.
4. Implementers are advised to follow any implementation guidelines and recommendations, if available, when they encode 'non-authoritative' versions of standards.
5. Implementers must not create new terms in the APs layer. If the metadata requirements of an AP cannot be satisfied by terms existing in the

‘authoritative’ schemas, then the new terms must be defined at the ‘non-authoritative’ schemas layer of the framework.
6. Implementers must make use of the concept of target XML namespaces to distinguish between ‘authoritative’, ‘non-authoritative’ and ‘AP’ schemas.

3.4 Modelling and implementation guidelines

A set of modelling and implementation guidelines are proposed to provide more implementation detail:

1. Implementers who need to be able to mix and match terms should make use of the two ‘container schemas’. The ‘container schemas’ should be encoded without a target namespace to enable the use of distinct namespaces between ‘authoritative’, ‘non-authoritative’, and ‘AP’ encodings. Finally ‘schemas containers’ must always make use of the <import> XSD mechanism, whereas encodings that wish to access components (i.e. terms or datatypes) made available by the container, should make use of the <include> mechanism. In that way AP developers will be able to draw elements and datatypes from authoritative namespaces, customize them if that is required in the non-authoritative namespace and adopt them in AP layer, without affecting the terms in the original namespace.
2. Implementers might wish to adopt a term without modifying either its semantic, structural or dependencies properties. In that case terms should be drawn directly from the ‘authoritative’ schemas layer.
3. Implementers are either allowed to tailor an adopted term in the ‘AP’ schemas layer, and/or tailor an adopted datatype (cf. Figure 5, Figure 10, related text and provided source code) so as to create a derived datatype and then create a new term in the ‘non-authoritative’ schemas layer.
4. Implementers might wish to encode semantically ‘rich’ or application-specific information relating to terms adopted in their AP. Term usage attributes and their definitions are available at the ‘non-authoritative schemas’ layer, through a set of terms titled ‘semantic information’ as illustrated in Figure 2. The encoding of each term’s attributes should occur under the <appInfo> XSD element of the each adapted term (cf. Figure 6, Figure 11, related text and provided source code). The list of term attributes currently includes all the suggested term usage information as reported in CEN (2003) and CEN (2005a). However, the system caters for extensibility so if implementers wish to update the list of term usage attributes, in order to satisfy a specific requirement of their particular application, they simply have to edit that particular schema.
5. Implementers might wish to change the cardinality constraints of an adopted term. In that case they should first adopt the desired term and then alter its cardinality constraints by modifying its min-max occurrences.
6. If implementers wish to specify dependencies in an adopted term, they should first adopt the datatype of the original term from the ‘authoritative’ schemas at the ‘non-authoritative’ schema layer and then tailor it. This can occur by either the use of several XSD mechanisms (i.e. <list>, <union>, <extension>, <restriction> and <redefine>), or in the case that ‘authoritative’ schemas are encoded in such a way that they do not allow their terms’ datatypes to be tailored, implementers are advised to define a specific <attribute> for the

element, named value, in order to carry the term's value encoded with the appropriate dependencies.

7. If implementers wish to adapt existing terms and specify dependencies such as element refinements, or create nested APs, they should follow a similar approach to proposal 6 (above). That is, they should first adopt the datatype of the original term from the 'authoritative' schemas at the 'non-authoritative' schema layer and then tailor it. This can occur by the use <extension> XSD mechanism and then create the desired element refinements. Furthermore, if implementers wish to create element refinements by adopting existing terms from other non-authoritative schemas encodings, they are advised to use the <import> XSD mechanism and import the desired XSD. In this way they will be able to draw on global elements from the same framework layer and use them as element refinements.

4 Experimental studies validating the framework

To evaluate the proposed XSD framework and identify its potential and limitations, two APs were selected to be encoded; the ARCO Metadata Schema— AMS (Mourkoussis et al. 2003; Patel et al. 2005), and the Internet Public Library Asia— IPL-Asia (Lee at al. 2003). The AP requirements were chosen as criteria to validate in part the proposed framework. These criteria were first listed at section 2.1, however they are repeated below for convenience:

- mix-and-match terms from multiple element sets
- adapt existing definitions for local purposes
- declare rules for content (e.g. usage guidelines)
- specify whether an element is mandatory, optional or repeatable
- specify dependencies (e.g. mandate encoding schemes)

In addition to the above, two more criteria were selected. First, the framework should be able to create both flat and structured APs. Second, it should allow for terms to be unbundled and adopted from both flat and structured metadata schemas. The following two sub-sections discuss the encoding of the two APs (i.e. AMS, IPL-ASIA) using the proposed XSD framework. Successful implementation of those two APs with the proposed framework would meet all the above criteria. Schematic illustrations and code extracts from existing implementations are provided to demonstrate the completeness of each implemented AP. Lessons learned and other concluding remarks will be further discussed in section 5.

4.1 Experimental Study 1: AMS AP

AMS is an AP designed and implemented to satisfy the metadata requirements of a European Union funded project titled Augmented Representation of Cultural Objects (ARCO). The ARCO system is modular providing museums with a set of tools to enable their staff to create, manage, and present virtual exhibitions (White et al. 2003; Patel et al. 2005). AMS is an integral part of the architecture of the system. It is an AP, which underpins both the heritage and technical aspects of the project. It provides terms that describe cultural artefacts, their digital surrogates, as well as technical data relating to virtual exhibitions (Mourkoussis et al. 2003; Patel et al. 2005). In the interests of interoperability and to avoid reinventing metadata elements, the terms in the AMS comprise those from standards (i.e. DCMES and SPECTRUM (Grant 1997)), but are also supplemented with ARCO specific metadata terms. A full

AMS specification and metadata requirements analysis is out of the scope of this paper; however it can be found on the ARCO project website (ARCO Consortium 2004).

Figure 3 illustrates the encoding of AMS using the proposed XSD framework (ARCO Consortium 2006). In brief, as described above, there are five layers of XSD encodings:

1. **Authoritative schemas** layer includes the XSD encodings, provided by Dublin Core (DCMI XML schemas 2004 (later versions now available)) and SPECTRUM.
2. **Non-authoritative schemas containe**r layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the <import> mechanism and provides access to terms and datatypes of XSD encodings of the 'authoritative' that might be later used by the encodings of the 'non-authoritative' layer.
3. **Non-authoritative schemas** layer includes the XSD encodings of non-authoritative XSD implementations of element sets in the event of such not being provided by the corresponding authority; application specific element sets (e.g. *'ARCO specific terms'*); and finally XSD encodings of terms with specified dependencies derived from the original datatypes of 'authoritative' schemas (e.g. 'DC Cultural Object'). The 'semantic information' XSD defines the term usage attributes. At this layer existing definitions are adapted for local purposes as well as rules for content and other term usage attributes are defined. To satisfy this requirements the non-authoritative schemas <import> the 'semantic information' XSD and encode term usage attributes as suggested in proposal 4 in section 3.4.
4. **Application schemas container** layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the <import> mechanism and provides access to terms existing in the 'authoritative' and 'non-authoritative' schemas that will be later used by the 'AP' schemas.
5. **AP schemas** layer consists of a number of XSD encodings that satisfy the data model of the AMS AP (e.g. *'Cultural Object'*). These encodings adopt terms from the XSD encodings of the 'authoritative' and 'non-authoritative' schemas layers through the use of the 'APs container' XSD encoding. At this level the cardinality constraint the adopted terms are customized to satisfy the application's requirement.

In more detail, the mix-and-match of terms from multiple element sets requirement became possible since terms from different 'authoritative and non-authoritative' schemas can be adopted from both authoritative and non-authoritative schemas through the containers XSD (i.e. Non-authoritative schemas container, Application schemas container). To achieve this, the 'schemas container' makes use of the <import> (Fallside 2001) mechanism provided in the XSD language.

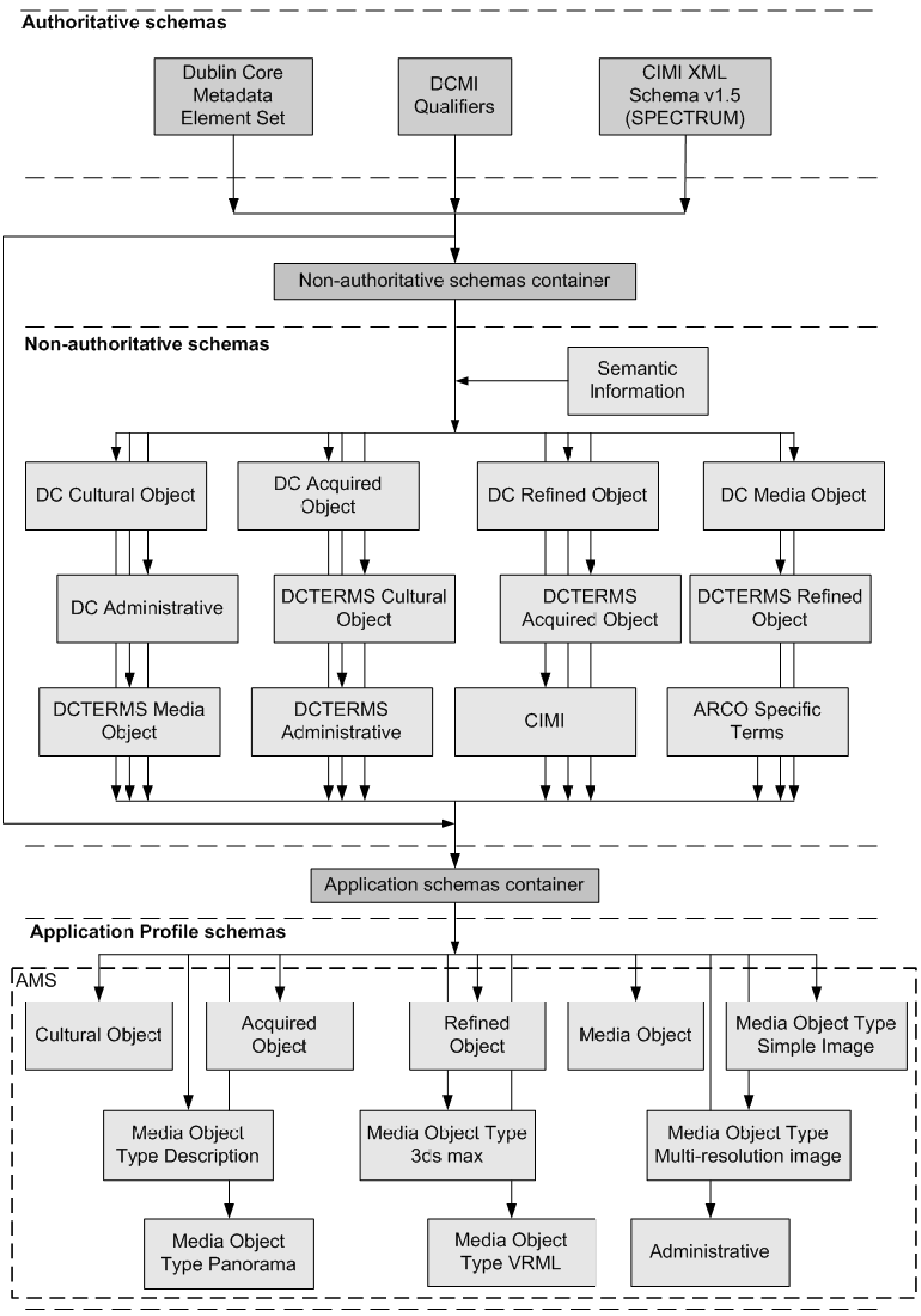


Figure 3: Schematic diagram demonstrating AMS encoding with the proposed XSD framework

In turn the ‘AP’ schemas use the <include> (Fallside 2001) mechanism of the XSD language in order to enable access to the ‘schema container’ components. This idea draws on guideline notes on the W3C XML Schemas for Qualified Dublin Core (DCMI XML schemas 2004 (later versions now available)). The <import> mechanism permits terms from different ‘authoritative and non-authoritative’ schemas (i.e. different target namespaces) to be used together in the ‘schema container’, and hence it enables the schema validation of instance content defined across multiple namespaces. The effect of the <include> mechanism is to provide access to the terms of the XML schema container, and make them available as part of the 'AP' schemas. However, the <include> mechanism requires that the target namespace of the included terms must be the same as the target namespace of the including schema (i.e. 'APs' schema). For this reason we defined the 'schema container' with no target namespace and the effect was that the components included in the AP schemas inherited the target namespace of the AP schemas.

Figure 4 provides a source code extract demonstrating that encoding AMS with the proposed framework we were able to satisfy the 'mix-and-match terms from multiple element sets' AP requirement. The provided extract depicts the 'cultural object' AMS XSD encoding, in which terms are adopted by the adapted Dublin Core and SPECTRUM standards, as well as from the ARCO Specific terms schema.

```
<xs:schema targetNamespace="http://arco-web.org/schemas/cultural/" xmlns:termsclt=
. . .
. . .
      <xs:include schemaLocation="containerapplication.xsd"/>
      <xs:element name="cultural">
              <xs:complexType>
                      <xs:sequence>
. . .
. . .
                              <xs:element ref="dcclt:contributor" minOccurs="0"
maxOccurs="unbounded"/>
                              <xs:element ref="termsclt:type"/>
                              <xs:element ref="termsclt:description"/>
                              <xs:element ref="dcclt:rights"/>
                              <xs:element ref="cimi:objectProductionDate"
minOccurs="0"/>
                              <xs:element ref="cimi:objectProductionPlace"
minOccurs="0" maxOccurs="unbounded"/>
                              <xs:element ref="cimi:completeness" minOccurs="0"/>
                              <xs:element ref="cimi:condition" minOccurs="0"/>
. . .
. . .
                              <xs:element ref="arco:component" minOccurs="0"/>
                              <xs:element ref="cimi:acquisitionSource"
minOccurs="0"/>
                              <xs:element ref="cimi:accessionDate"/>
                              <xs:element ref="cimi:currentLocation"/>
                              <xs:element ref="cimi:fieldCollection"/>
                      </xs:sequence>
              </xs:complexType>
      </xs:element>
</xs:schema>
```

Figure 4: Source code extract validating the realization •of the 'mix-and-match terms from multiple element sets' AP requirement

The second requirement concerns specifying dependencies for an adopted term. The XSD language does not allow direct customization of adopted (i.e. imported) terms without affecting the original term. However, it is possible to tailor the datatype (Biron 2001) of the term under question since XSD provides several mechanisms (i.e. <list>, <union>, <extension>, <restriction>, <redefine>) (Fallside 2001) to derive a new datatype based on the existing datatype of the term and then create a new term based on the derived type. To satisfy this requirement we had to draw the original datatype from the authoritative layer to the non-authoritative layer, derive a new datatype based on it, and then create a new term based on the derived datatype. The idea of this approach is based on Hunter's work (Hunter 2001).

Figure 5 illustrates how the *date* SPECTRUM term has been adapted to satisfy AMS requirements. Provided by the authoritative encoding of SPECTRUM (i.e. CIMI XML Schema v 1.5) cs:date is a nested complex datatype. For the AMS AP requirements we needed only a part of this complex data type. In particular, we wanted only the *text* term. We created a new datatype called productionDate that was derived by applying restrictions to the authoritative *'cs:date'*. In particular, we modified the cardinality constraints of the not-wanted terms to be of both minimum and maximum occurrence equal to zero. Then we created a term called *ObjectProductionDate* based on the latter derived datatype.

```
<xs:complexType name="productionDate">
              <xs:complexContent>
                     <xs:restriction base="cs:date">
                            <xs:sequence>
                                   <xs:element name="association" type="xs:string"
minOccurs="0" maxOccurs="0">
                                          <xs:annotation>
                                                 . . .
                                                 . . .
                                          </xs:annotation>
                                   </xs:element>
                                   <xs:element name="earliest" minOccurs="0"
maxOccurs="0">
                                          <xs:complexType>
                                                 <xs:sequence>
                                                        <xs:element
name="certainty" type="xs:string" minOccurs="0">
                                                               <xs:annotation>
                                                                      . . .
                                                                      . . .

                                                               </xs:annotation>
                                                        </xs:element>
                                                               . . .
                                                               . . .
                                                               . . .
                                   <xs:element name="text" type="xs:string">
                                          <xs:annotation>
                                                 . . .
                                                 . . .
                                          </xs:annotation>
                                   </xs:element>
                            </xs:sequence>
                     </xs:restriction>
              </xs:complexContent>
       </xs:complexType>
       <xs:element name="objectProductionDate" type="cimi:productionDate">
              <xs:annotation>
                     . . .
                     . . .
              </xs:annotation>
       </xs:element>
```

Figure 5: Source code extract validating the realization of the 'specifying dependencies' AP requirement

The third (i.e. adapt existing definitions for local purposes) and fourth (i.e. declare rules for content) AP requirements were achieved by the use of the <appInfo> (Fallside 2001) mechanism of the XSD language. As defined in the XML Schema W3C recommendation, the <appInfo> component is used to provide information for tools, stylesheets and other applications. In our case we used <appInfo> in order to provide 'rich' information, such as definitions and declare rules for content, for each adopted term. For this purpose we updated our own convention (Mourkoussis et al. 2003) for the encoding of semantically 'rich' and additional application specific information with the guidelines for term usage provided by the CEN workshop

Agreements reports (CEN 2003, CEN 2005a). This includes identification, definitional, relational and constraints attributes. Figure 6 provides an example of such an encoding. The definitions of the term usage terms are encoded in a separate XSD, named *'Semantic Information'* (refer to Figure 3). This allows the list to be extensible so as to be able to satisfy application specific requirements of tools that will process the elements for example as a tool-tip or mappings between profiles. Figure 6 provides a source code extract to demonstrate the use of <appInfo> and term usage attributes in order to adapt existing definitions for local purposes, declare rules for content, and provide other semantic information.

```
<xs:element name="source" type="dc:SimpleLiteral">
              <xs:annotation>
                      <xs:appinfo>
              <app:termURI>http://purl.org/dc/elements/1.1/source</app:termURI>
                              <app:name>source</app:name>
                              <app:label>Source</app:label>
                              <app:definedBy>DCMI</app:definedBy>
                              <app:definition>A unique identifier for the physical
artefact for which the CO is a surrogate</app:definition>
                              <app:comments>The identifier will be unique for the
museum, e.g. LEWSA (institution).1973 (year when object was acquired for the
museum).3 (a unique number for the donor). 5 (individual numbers for the objects
within the donor's bequest) e.g. LEWSA:1964.1.158</app:comments>
                              <app:typeOfTerm>Element Refinement</app:typeOfTerm>
                              <app:refines>culturalObject</app:refines>
                              <app:obligation>Mandatory</app:obligation>
                              <app:datatype>String</app:datatype>
                              <app:occurrence>1</app:occurrence>
                      </xs:appinfo>
              </xs:annotation>
       </xs:element>
```

Figure 6: Source code extract validating the realization of the 'adapt existing definitions for local purposes' and 'declare rules for content' AP requirements

The fifth and final (i.e. alter the cardinality constraints) requirement was achieved with the use of <minOccurs> (Fallside 2001) and <maxoccurs> (Fallside 2001) cardinality constraints provided by the XSD language. Figure 7 illustrates in part the AMS 'acquired object' XSD. In this particular example cardinality constraints are applied to the adopted terms.

```
<xs:schema targetNamespace="http://arco-web.org/schemas/acquired/"
. . .
elementFormDefault="qualified" attributeFormDefault="unqualified">
       <xs:include schemaLocation="containerapplication.xsd"/>
       <xs:element name="acquired">
              <xs:complexType>
                      <xs:sequence>
                              . . .
                              <xs:element ref="dcacq:title"/>
                              <xs:element ref="dcacq:publisher" minOccurs="0"/>
                              <xs:element ref="dcacq:creator"/>
                              <xs:element ref="dcacq:contributor" minOccurs="0"
maxOccurs="unbounded"/>
                              . . .
                      </xs:sequence>
              </xs:complexType>
       </xs:element>
</xs:schema>
```

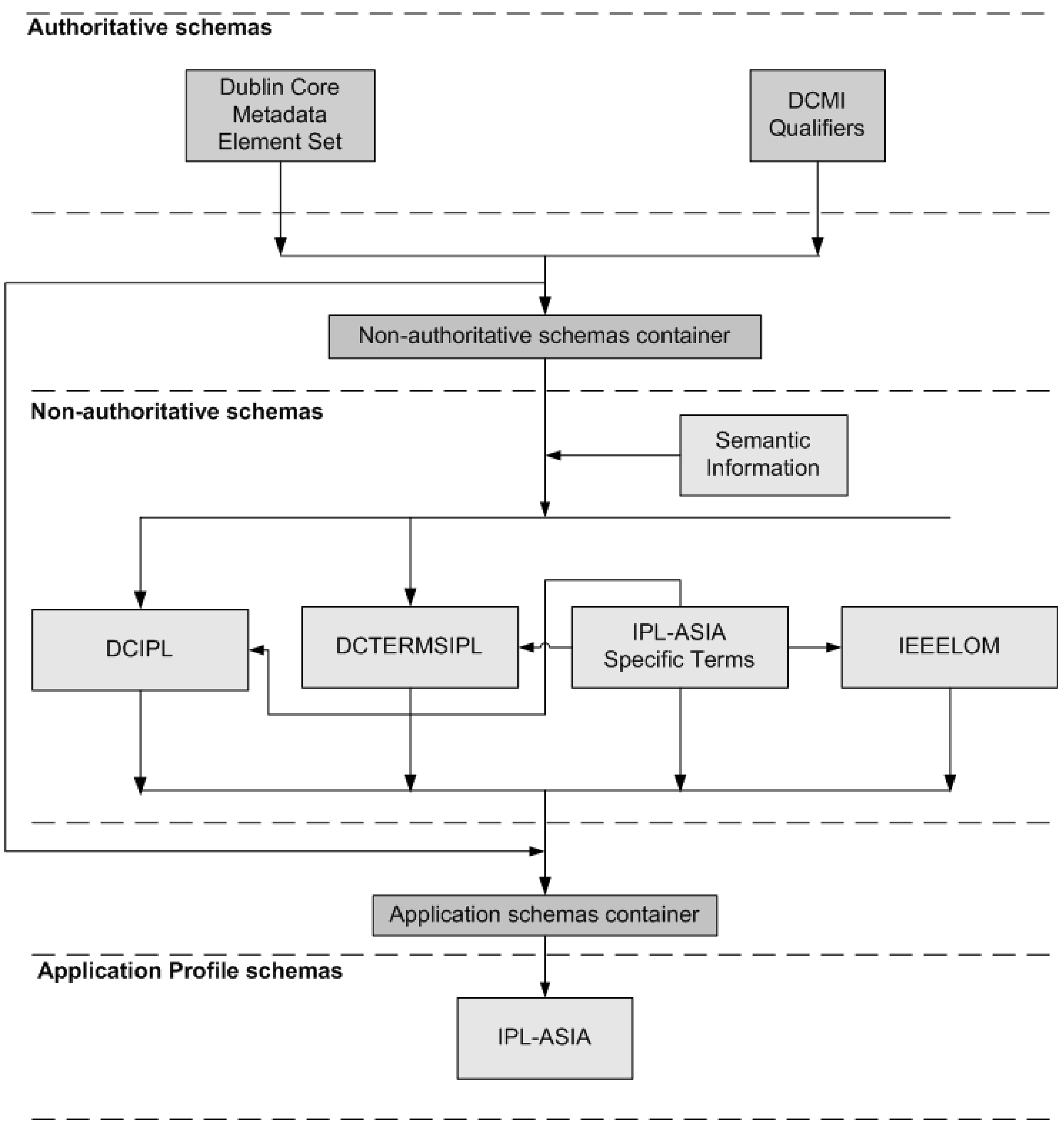
Figure 7: Source code extract validating the realization of the 'alter the cardinality constraints' AP requirement

4.2 Experimental Study 2: IPL – ASIA AP

IPL-ASIA is an AP designed and implemented to satisfy the metadata requirements of a multilingual subject gateway, which is partly a collaborative project with the Internet Public Library (Lee et al. 2003). The AP is focused on resource collections published in Chinese, Japanese and Korean (CJK) languages. The underlying principle is that the AP will be able to document each information resource in four languages (CJK plus English). In the interests of interoperability and to avoid reinventing metadata elements, the terms in the IPL-ASIA comprise those from standards (i.e. DCMES and IEEE LOM (IEEE LTSC 2002)), but are also supplemented with IPL-ASIA specific metadata terms.

Figure 8 illustrates the encoding of IPL-ASIA using the proposed XSD framework. As described above there are five layers of XSD encodings:

1. **Authoritative schemas** layer includes the XSD encodings, provided by Dublin Core (DCMI XML schemas 2004 (later versions now available)).
2. **Non-authoritative schemas container** layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the <import> mechanism and provides access to terms and datatypes of XSD encodings of the 'authoritative', which might be later, used by the encodings of the 'non-authoritative' layer.
3. **Non-authoritative schemas** layer includes the XSD encodings of non-authoritative XSD implementations of element sets (i.e. *'IEEE LOM'*); application specific element sets (i.e. *'IA specific terms'*); and finally XSD encodings of terms with specified dependencies derived from the original datatypes of 'authoritative' schemas (i.e. *'DC IPL'*, *'DCTERMS IPL'*). The 'semantic information' XSD defines the term usage attributes. At this layer existing definitions are adapted for local purposes as well as rules for content and other term usage attributes are defined. To satisfy this requirement the non-authoritative schemas <import> the 'semantic information' XSD and encode term usage attributes as suggested in proposal 4 in section 3.4. Furthermore, at this current 'IA Specific terms' are structured as element refinements of adapted DCMI terms. In order for the latter to be achieved, the *'DCIPL'* and '*DCTERRMS IPL*' schemas <import> the *'IA Specific terms'* schema definitions and the desired element refinements are implemented.
4. **APs container** layer consists of one XSD encoding with no target namespace assigned to it. This encoding makes use of the <import> mechanism and provides access to terms existing in the 'authoritative' and 'non-authoritative' schemas, which will be later used by the 'AP' schemas.
5. **AP schemas** layer consists of IPL-ASIA AP XSD encoding. At this level cardinality constraints of adopted terms are customized to satisfy the application's requirement.



Authoritative schemas
Dublin Core Metadata Element Set
DCMI Qualifiers
Non-authoritative schemas container
Non-authoritative schemas
Semantic Information
DCIPL
DCTERMSIPL
IPL-ASIA Specific Terms
IEEELOM
Application schemas container
Application Profile schemas
IPL-ASIA

Figure 8: Schematic diagram demonstrating IPL-ASIA encoding with the proposed XSD framework

In more detail, the mix-and-match of terms from multiple element sets requirement became possible since terms from different 'authoritative and non-authoritative' schemas can be adopted through the containers XSDs (i.e. Non-authoritative schemas container, Application schemas container). The mechanism used here is the same as the one used in the AMS AP (in order to avoid text duplication please refer to section 4.1 for further information). Figure 9 provides a source code extract demonstrating that encoding IPL-ASIA with the proposed framework we were able to satisfy the 'mix-and-match terms from adapted Dublin Core (i.e. *'DCIPL'* and *'DCTERMS IPL'*) *'IEEELOM'* and *'IA Specific terms'* XSDs.

```
<xs:schema targetNamespace="http://IPL-ASIA-DUMMY.org"
. . .
elementFormDefault="qualified" attributeFormDefault="unqualified">
      <xs:include schemaLocation="containerapplication.xsd"/>
      <xs:element name="IPL-ASIA">
             <xs:complexType>
                    <xs:sequence>
                           <xs:element ref="dcipl:Title"/>
```

```
                              <xs:element ref="dcipl:Creator" minOccurs="0"/>
                                      . . .
                                      . . .
                              <xs:element ref="dcipl:Date"/>
                              <xs:element ref="lom:Metametadata"/>
                              <xs:element ref="dcTermsipl:Audience"/>
                              <xs:element ref="dcipl:Subject"/>
                                      . . .
                      </xs:sequence>
              </xs:complexType>
      </xs:element>
</xs:schema>
```

Figure 9: Source code extract validating the realization of the 'mix-and-match terms from multiple element sets' AP requirement

The second requirement concerns specifying dependencies for an adopted term. Similar to section 4.1 in order to satisfy this requirement we had to draw the original datatype from the authoritative layer to the non-authoritative layer, derive a new datatype based on it, and then create a new term based on the derived datatype. The code extract in Figure 10 illustrates how we specified dependencies on the Dublin Core adapted term *'title'*. We created a new term called *'Title'* by adapting the Dublin core *'SimpleLiteral'* datatype. We allowed for the dataype to be extended and we created two children elements (i.e. adopting the *'Main-Title'* and *'Sub-Title'* terms form the *'IA Specific terms'* XSD). In this way we managed to declare element refinements to the adopted *'title'* term and create nested APs. In order to allow the *'DCIPL'* term title to have as element refinements those two *'IA Specific terms'*, we had to use <import>, as illustrated in the source code below.

```
<xs:schema targetNamespace="http://www.ipl-asia "
. . .
elementFormDefault="qualified" attributeFormDefault="unqualified">
      . . .
      <xs:include schemaLocation="containernonauthorative.xsd"/>
      <xs:import namespace="http://www.ipl-asia-dummy.org/schemas/ia"
schemaLocation="IA.xsd"/>
      <xs:import namespace="http://www.arco-
web.org/schemas/version14/arco/application/"
schemaLocation="SemanticInformation.xsd"/>
      <xs:element name="Title">
              <xs:annotation>
                      . . .
              </xs:annotation>
              <xs:complexType>
                      <xs:complexContent>
                              <xs:extension base="dc:SimpleLiteral">
                                      <xs:sequence>
                                              <xs:element ref="ia:Main-Title"/>
                                              <xs:element ref="ia:Sub-Title"
minOccurs="0"/>
                                      </xs:sequence>
                              </xs:extension>
                      </xs:complexContent>
              </xs:complexType>
      </xs:element>
      . . .
      . . .
</xs:schema>
```

Figure 10: Source code extract validating the realization of the 'specifying dependencies' AP requirement

The third (i.e. adapt existing definitions for local purposes) and fourth (i.e. declare rules for content) AP requirements were achieved, similarly to section 4.1, by the use of the <appInfo> (Fallside 2001) mechanism of the XSD language. Figure 11 provides a source code extract to demonstrate the use of <appInfo> and term usage attributes in order to adapt existing definitions for local purposes, declare rules for content, and provide other semantic information for the *'description'* term.

```
. . .
<xs:element name="Description">
              <xs:annotation>
                      <xs:appinfo>

      <app:termURI>http://purl.org/dc/elements/1.1/description</app:termURI>
                              <app:name>Description</app:name>
                              <app:label>Description</app:label>
                              <app:definedBy>DCMI</app:definedBy>
                              <app:definition>An account of the content of the
resource.</app:definition>
                              <app:typeOfTerm>Element Refinemnt</app:typeOfTerm>
                              <app:refines>IPL-ASIA</app:refines>
                              <app:refinedBy>ia:Long</app:refinedBy>
                              <app:refinedBy>ia:Short</app:refinedBy>
                              <app:obligation>Mandatory</app:obligation>
                              <app:datatype>String</app:datatype>
                              <app:occurrence>1</app:occurrence>
                      </xs:appinfo>
              </xs:annotation>
              <xs:complexType>
                      . . .
                      . . .
              </xs:complexType>
      </xs:element>
. . .
```

Figure 11: Source code extract validating the realization of the 'adapt existing definitions for local purposes' and 'declare rules for content' AP requirements

The fifth and final (i.e. alter the cardinality constraints) requirement was achieved with the use of <minOccurs> (Fallside 2001) and <maxoccurs> (Fallside 2001) cardinality constraints provided by the XSD language. Figure 12 illustrates in part the *'IPL-ASIA'* XSD. In this particular example cardinality constraints are applied to the adopted terms.

```
<xs:schema targetNamespace="http://IPL-ASIA-DUMMY.org"
. . .
elementFormDefault="qualified" attributeFormDefault="unqualified">
      <xs:include schemaLocation="containerapplication.xsd"/>
      <xs:element name="IPL-ASIA">
              <xs:complexType>
                      <xs:sequence>
                              . . .
                              . . .
                              <xs:element ref="dcipl:Subject"/>
                              <xs:element ref="dcipl:Coverage" minOccurs="0"/>
                              <xs:element ref="dcipl:Relation" minOccurs="0"/>
                              <xs:element ref="dcipl:Source" minOccurs="0"/>
                              <xs:element ref="dcipl:Rights" minOccurs="0"/>
                              <xs:element ref="dcipl:Contributor" minOccurs="0"/>
                      </xs:sequence>
              </xs:complexType>
      </xs:element>
</xs:schema>
```

Figure 12: Source code extract validating the realization of the 'alter the cardinality constraints' AP requirement

5 Conclusion

The framework for the implementation of APs in XSD that we propose in this paper is the outcome of a case study based on our attempt to encode the ARCO metadata element set (AMS) as an AP. We believe this work has led to the successful identification of a series of XSD mechanisms that implementers can employ to satisfy the modelling and implementation requirements of APs. The applicability of the framework was further evaluated by employing it in the implementation of the IPL-ASIA AP. These two case studies, validated the proposed framework as capable of encoding both flat and nested APs, satisfying a series of criteria as described in section 4.

During this work, we came across a series of issues worthy of discussion. First, we noticed that there is no mechanism in the XML Schemas language to override a previously defined term. For example, assume that there is a term named 'A' in a metadata standard that satisfies the semantic requirements of a term that our AP wishes to inherit. Also, let us assume that in the original schema the term 'A' is defined as an 'integer', while in our AP we wish to define it as a 'string'. XML Schema provides no mechanism for performing such an operation. For this reason we proposed to draw the original datatype of the desired term in a non-authoritative version of the schema and customize the drawn datatype using XSD mechanisms such as <list>, <union>, <extension>, <restriction>, <redefine>. In turn we created a new term at the non-authoritative layer based on the adapted datatype.

Second, we noticed that there are no dedicated mechanisms in the XML schemas language to encode 'rich' semantic information. Information such as definitions, possible uses of a term, content guidelines, and URIs that facilitate the development of tools that edit, present and translate metadata values. For this reason we proposed a separate schema named 'Semantic Information' that we made available at the non-authoritative schemas layer (refer to Figure 2, Figure 3, and Figure 8). This schema provides an encoding of the term usage attributes suggested in the CEN workshop agreements reports (CEN 2003, CEN 2005a). Additionally, we suggested that implementers should use the <appInfo> element provided by the XSD language to encode values of the term usage attributes for each adopted term inline with the adapted term definition. Figure 6 and Figure 11 demonstrate this proposed solution. This approach allows the list of semantic attributes to be extensible allowing us to satisfy application specific requirements of tools that will process the elements. In particular, in our first case study, we extensively used an additional term usage attribute, collectively known as *'autoGenerated'* in order to identify which metadata terms will be directly generated by the system and which API functions should be called by the underlying application (i.e. ARCO) for the latter to occur.

Third, we noticed that metadata standards have structural differences in their hierarchy and data models that make the attempt to provide common guidelines on how to use them in APs non-trivial. This was also confirmed in the SCHEMAS, MEG Registry and CORES projects. For example implementation of the DCMES in XSD defines a 'complex type' named *'SimpleLiteral'* that is defined in such a way as to accept any value but does not allow for restricting the content of a term by the use of 'regular expressions' and 'enumerations'. To overcome this, we suggested the use of a dedicated attribute called 'value' that can be used to satisfy this type of term dependency. Another example is the CIMI XML schema that defines nested types of terms. To overcome this, we had to draw the datatype of the term to be adopted and extend its definition by setting the <min> and <max> occurrences of the unwanted elements to zero. In that way, we were able to unbundle terms from nested structures for reuse in the AP schemas. The two case studies used to validate the proposed framework indicated that the framework is capable of supporting the implementation of both nested (e.g. AP-ASIA) and flat (e.g. AMS) APs, as well as allowing APs to adopt metadata terms from both flat (e.g. DC) and structured (e.g. CIMI XML Schema) metadata schemas.

Finally, we found that the XML Namespace mechanism is ideal for drawing on multiple element sets to distinguish between authoritative, non-authoritative, and the local AP element sets.

Whilst, we have described a framework, which supports the technical implementation of APs in XSD, it should be noted that the mixing and matching together of individual terms from existing vocabularies requires careful consideration and judgement. We believe this to be a modelling issue, which needs to be addressed during the design of the application's information model.

To conclude, in this paper, we have discussed a generic framework by which implementers may encode APs in XSD. The framework presented has a layered structure to explicitly separate the authoritative, the non-authoritative, and the local application parts. This framework is an important step in encoding APs that can be dynamically updated with information on the terms they use directly from external schemas. Furthermore, such information can be integrated with semantic information into a "one-stop" encoding for the convenience of tool implementers and users.

Acknowledgements

This work has been partly funded by the EU IST Framework 5 Programme under Key Action III Multimedia Content and Tools, Action Line III.1.6 "Virtual Representations of Cultural and Scientific Objects".

References

ARCO (2004) "Augmented Representation of Cultural Objects Project homepage". ARCO Consortium last accessed October 9th 2006, http://www.arco-web.org/

ARCO Consortium (2004) "The ARCO Data Model and Element Set specification" ARCO Consortium, last accessed October 9th 2006, http://www.arco-web.org/ams/

ARCO Consortium (2006) "ARCO Metadata Application Profile Encoding using XSD", last accessed October 17th 2006, http://journals.tdl.org/jodi/rt/suppFiles/jodi-178/0

Baker, T., Dekkers, M., Heery, R., Patel, M., Salokhe, G. (2001) "What terms does your metadata use? Application profiles as machine-understandable narratives." *Journal of Digital Information* 2(2), last accessed October 9th 2006, http://jodi.ecs.soton.ac.uk/Articles/v02/i02/Baker

Berners-Lee, T., Fielding, R., Irvine, U.C., Masinter, L. (eds.) (1998) "RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax". IETF (Internet Engineering Task Force), last accessed October 5th 2006, http://www.faqs.org/rfcs/rfc2396.html

Biron, P. V., Malhotra, A. (eds) (2001) "XML Schema Part 2: Datatypes". W3C Recommendation, last accessed October 9th 2006, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

Bray, T., Hollander, D., Layman, A. (eds.) (1999) "Namespaces in XML". World Wide Web Consortium, last accessed October 9th 2006, http://www.w3.org/TR/REC-xml-names

Brickley, D. and Guha, R.V. (eds) (2004) "RDF Vocabulary Description Language 1.0: RDF Schema". W3C Recommendation, last accessed October 9th 2006, http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

CEN - European Committee for Standardization (2005b) "CWA 15249 - Guidance for naming, versioning, evolution and maintenance of element declarations and Application Profiles" CEN Workshop Agreement, last accessed October 9th 2006, http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/cwa/cwa15249.asp

CEN - European Committee for Standardization (2005a) "CWA 15248 - Guidelines for machine-processable representation of Dublin Core Application Profiles" CEN Workshop Agreement, last accessed October 9th 2006, http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/cwa/cwa+15248.asp

CEN - European Committee for Standardization (2003) "CWA 14855 - Dublin Core Application Profile Guidelines" CEN Workshop Agreement, last accessed October 9th 2006, http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/cwa/cwa14855.asp

CIMI XML Schema (2002) "CIMI XML Schema for SPECTRUM V1.5". CIMI Consortium, last updated October 9th 2006, http://xml.coverpages.org/CIMIv15-Schema.html

DCMI (2004) "Dublin Core Metadata Initiative homepage". Last accessed October 9th 2006, http://dublincore.org/

DCMI XML schemas (2004) "DCMI term declarations represented in XML schema language". DCMI, last accessed October 9th 2006, (Note: more recent versions are now available), http://dublincore.org/schemas/xmls/

Dempsey, L. and Weibel, S. L. (1996) "The Warwick Metadata Workshop: A Framework for the Deployment of Resource Description", *D-Lib Magazine*, last accessed October 9th 2006, http://www.dlib.org/dlib/july96/07weibel.html

Fallside, D. C. (ed.) (2001) "XML Schema Part 0: Primer". W3C Recommendation, last accessed October 9th 2006, http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

Gilliland-Swetland, A. J. (1998) "Defining Metadata". In *Introduction to Metadata Pathways to Digital Information*, edited by Murtha Baca (Unites States of America: Getty Information Institute), pp. 1- 8

Grant, A. (1997) SPECTRUM: The UK Museum Documentation Standard, Second Edition (*UK: The Museum Documentation Association*)

Heery, R., Johnston, P., Beckett, D., Steer, D. (2002) "The MEG Registry and SCART: complementary tools for creation, discovery and re-use of metadata schemas". In *Proceedings of the International Conference on Dublin Core. Metadata for e-Communities: supporting diversity and convergence*, (Florence: Firenze University Press), pp. 125-132, last accessed October 9th 2006, http://www.bncf.net/dc2002/program/ft/paper14.pdf

Heery, R., Gardner, T., Day, M., Patel, M. (2000b) "DESIRE Metadata Registry Framework Deliverable 3.5". DESIRE Consortium, last accessed October 9th 2006, http://www.desire.org/html/research/deliverables/D3.5/

Heery, R, and Patel, M. (2000) "Application profiles: mixing and matching metadata schemas". *Ariadne*, No. 25, September, last accessed October 9th 2006, http://www.ariadne.ac.uk/issue25/app-profiles/intro.html

Hunter, J. and Lagoze, C. (2001) "Combining RDF and XML Schemas to Enhance Interoperability Between Metadata Application Profiles", In *Proceedings of Proceedings of the tenth Web conference* (HongKong: ACM), last accessed October 9th, 2006 http://www10.org/cdrom/papers/572/

IEEE Learning Technology Standards Committee, WG12 (2002) "Final 1484.12.1 LOM Draft Standard Document", last accessed October 13th 2006, http://ltsc.ieee.org/wg12/20020612-Final-LOM-Draft.html

Johnston, P., Cole, T., Habing, T., Hunter, J., Lagoze, C., Powell, A. (2003) "Notes on the W3C XML Schemas for Qualified Dublin Core". DCMI, last accessed October 9th 2006, http://dublincore.org/schemas/xmls/qdc/2003/04/02/notes/

Lee, W., Sugimoto, S., Nagamori, M., Sakaguchi, T., Tabata, K. (2003) "A Subject gateway in Multiple Languages: a Prototype Development and Lessons Learned", In *Proceedings of International Conference on Dublin Core. Supporting communities of discourse and practice — Metadata research and* application (Seattle, Washington USA: DCMI Press), pp. 59-66, last accessed October 9th 2006, http://www.siderean.com/dc2003/203_Paper76.pdf

Mourkoussis, N., White, M., Patel, M., Chmielewski, J., Walczak, K. (2003) "AMS — Metadata for Cultural Exhibitions using Virtual Reality", In *Proceedings of International Conference on Dublin Core. Supporting communities of discourse and practice — Metadata research and application* (Seattle, Washington USA: DCMI Press), pp. 193-202, last accessed October 9th 2006, http://www.siderean.com/dc2003/601_Paper20.pdf

METS (2004) "Metadata Encoding and Transmission Standard homepage". Last accessed October 9th 2006, http://www.loc.gov/standards/mets/

Nagamori, M., Sugimoto, S. (2004) "A Framework of Metadata Schema for Functional Extension of Metadata Schema Registry" *In Proceedings of the International conference on Dublin Core and metadata applications. Metadata* *across languages and cultures* (Shanghai, China: DCMI Press), pp. 3-11, last accessed October 9th2006, http://www.slais.ubc.ca/PEOPLE/faculty/tennis-p/dcpapers2004/Paper_08.pdf

Patel, M., White, M., Mourkoussis, N., Walczak, K., Wojciechowski R., Chmielewski, J. (2005) "Metadata Requirements for Digital Museum Environments", *International Journal of Digital Libraries,* 5(3), S*pecial issue on Digital Museums*, 179-192, last accessed October 9th 2006, http://www.springerlink.com/content/ly1qg5x3p3n98nyy/

Powell, A. and Johnston, P. (2003) "Guidelines for implementing Dublin Core in XML". DCMI, Last accessed October 9th 2006, http://dublincore.org/documents/dc-xml-guidelines/

Powell, A., Wagner, H., Weibel, S., Baker, T., Matola, T., Miller, E. (2001) "Namespace Policy for the Dublin Core Metadata Initiative" DCMI, Last accessed October 9th 2006, http://dublincore.org/documents/dcmi-namespace/

SCHEMAS (2000) "Forum for Metadata Schema and implementers". The SCHEMAS project, last accessed October 9th 2006, http://www.schemas-forum.org/

Thompson, H. S., Beech, D., Maloney, M., Mendelsohn, N. (eds) (2001) "XML Schema Part 1: Structures". W3C Recommendation, last accessed October 9th 2006, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

White, M., Walczak, K., Mourkoussis, N. (2003) "ARCO: Augmented Representation of Cultural Objects", *Advanced Imaging,* June, pp.14-15, 46

Author details

Nicholaos holds a PhD in Computer Science and holds a MEng (Hons) in Computer Systems Engineering from the University of Sussex. Nicholaos worked as a research officer in the EU funded project titled “Augmented Representations of Cultural Objects” and he is currently working as a research fellow in the EPSRC funded project titled “Quantifying fidelity for virtual environment simulations employing schema assumptions”. His research interests relate to: selective graphics rendering; cognition and human memory; positive transfer of training in VR simulations; perception and human factors; digital libraries; metadata for digital museum environments; XML frameworks for metadata interoperability; and software architectures for digital cultural heritage and education.

Manjula holds a PhD in Computer Science (computer graphics and animation) from the University of Bath, as well as an MSc in Systems Design from the University of Manchester and a BSc (Hons) in Computational Science and Economics from the University of Leeds. Prior to working at UKOLN, she worked as a post-doctoral research officer, lecturer and computer manager in the School of Mathematical Sciences at the University of Bath. Manjula has been working as part of the research and development team at UKOLN since August 1998. She has been involved in numerous EU, EPSRC and JISC research projects. Her current research interests relate to: digital libraries; metadata and ontologies; knowledge representation and the semantic web; digital repositories and preservation; autonomous software agents; virtual and augmented reality representation of cultural artefacts and their use in various domains such as digital museums and learning environments.

Martin holds a PhD in Computer Science (3D Computer Graphics), and a BSc (Hons) in Computer Systems Engineering from the University of Sussex. Prior to Sussex Martin worked as an Electronic Engineer in the UK defence and communications industry. While at Sussex Martin has worked on many EU research projects as a research fellow and project manager, and was the project coordinator for the ARCO project. He has authored or co-authored 130 scientific papers on graphics, virtual and augmented reality and e-Learning. Martin's research interests include software and hardware for computer graphics, virtual and augmented reality, building haptic interfaces for virtual environments, digital preservation and virtual reconstruction related to virtual and augmented reality representations of cultural heritage use in museum and education domains.