An Overview of the MPEG-7 Description Definition Language (DDL) Proposals

Jane Hunter, Frank Nack
jane@dstc.edu.au, nack@darmstadt.gmd.de

Abstract

This paper describes the DDL proposals submitted in response to the MPEG-7 Call for Proposals and the results of their evaluation at the MPEG-7 AHG Test and Evaluation Meeting in Lancaster in February 1999. It also describes the proposal from DSTC which was considered to provide the best starting point for a DDL and the features from other proposals which were considered to be of value for future consideration and possible integration. It concludes with an overview of the current state of the MPEG-7 DDL work.

Keywords:MPEG-7, DDL, Schema, RDF, XML

---

1. Introduction

The Description Definition Language (DDL) is the language that enables the creation of new MPEG-7 Description Schemes (DSs) and Descriptors (Ds) and the extension and modification of existing DSs. The DDL has to be able to express spatial, temporal, structural, and conceptual relationships between the elements of a DS, and between DSs. It must provide a rich model for links and references between one or more descriptions and the data that it describes. It also has to be capable of validating both primitive and composite descriptor data types. In addition it must be platform and application independent and human- and machine-readable.

At the MPEG-7 AHG Test and Evaluation Meeting in Lancaster in February 1999, the MPEG-7 Description Definition Language (DDL) evaluation team compared and evaluated ten DDL proposals. A summary report was produced which included a list of recommendations and outlined the technologies which should be included, or further investigated for possible inclusion, within the DDL for the MPEG-7 Experimentation Model (XM) [1].

The DDL Evaluation team came to the conclusion that none of the proposals sufficiently met the DDL requirements, as described in the MPEG-7 Requirements Document [2], to constitute a final or complete solution. However the proposals did help to clarify the critical issues, in particular, the issue of object-oriented semantic expression versus structural validation. Most of the proposals focussed on one of these approaches and attempted to extend it towards the other but it became clear that it is difficult to adequately reconcile these disparate requirements.

However, there was a consensus that XML should be used as the *syntax* for the MPEG-7 DDL. There was also a consensus that there is a need for both the validation of structural, relational and data typing constraints and the object-oriented expression of semantics. There has been some concern expressed that declarative languages primarily support structural descriptions but provide little support for describing semantic content. As a consequence of this, the development of the DDL will be a collaborative effort to design and implement an XML-based language which attempts to bridge these two approaches through the addition of richer constraints such as data typing and inheritance.

P547 [3], the proposal from DSTC, was selected as the candidate upon which the language design will be based but with substantial integration of ideas and components from other proposals and contributors. In addition, the strategy was to continue tracking and influencing related efforts in the W3C community, in particular the XML Schema [4], Xlink [5], XPath [6]and XPointer [7] Working Groups.

A modified version of proposal P547 was submitted to the Seoul meeting [8]. It represented a first attempt to incorporate the changes which had been suggested by the DDL Evaluation Team in Lancaster. It also attempted to clarify certain issues such as the validation mechanisms to be provided by the parser. In particular, the following recommended enhancements and open issues were addressed:

- Allowing property types to be classes;
- Extending data types to include certain suggested spatial types;
- Clarifying how the constraint validation mechanisms should be implemented within the parser.

It was hoped that these changes would result in a schema which more closely satisfied the requirements of the DDL as described in the MPEG-7 Requirements Document and which could be used as a starting point for future collaborative development.

Below we provide a high level description and classification of the proposals submitted to the MPEG-7 DDL evaluation team. We then describe the preferred candidate, P547, from DSTC and the recommended changes which were incorporated.

2. MPEG-7 DDL Requirements

The MPEG-7 DDL must satisfy the following requirements outlined in the Requirements Document [2]:

- **Compositional capabilities**
  The DDL shall allow new Description Schemes (DSs) and Descriptors (Ds) to be created and existing DSs to be modified or extended.

- **Unique identification**
  The DDL shall allow a unique identification of Ds and DSs.
  Example: An example of unique identification is a namespace, which enables the unique qualification of element names and relationships and avoids name collisions on elements which have the same name but are defined in different vocabularies/domains.

- **Primitive data types**
  The DDL shall provide a set of primitive data types e.g. text, integer, real, date, time/time index, version etc.

- **Composite data types**
  The DDL shall be able to describe composite data types such as histograms, graphs, RGB values, enumerated types etc.

- **Multiple media types**
  The DDL shall provide a mechanism to relate Ds to data of multiple media types of inherent structure, particularly audio, video, audio-visual presentations, the interface to textual description, and any combinations of these.

- **Various types of DS instantiations**
  The DDL shall allow various types of DS instantiation: full, partial, full-mandatory, partial-mandatory.

- **Relationships within a DS and between DSs**
  The DDL shall be able to express spatial, temporal, structural, and conceptual relationships between the elements of a DS, and between DSs.

- **Relationship between description and data**
  The DDL shall supply a rich model for links and references between one or more descriptions and the data that it describes.

- **Link to ontologies**
  The DDL shall supply a linking mechanism between a description and several ontologies.

- **Platform independence**
  The DDL shall be platform and application independent.
- **Grammar**
  The DDL shall follow an unambiguous and easily parsed grammar.
- **Validation of constraints**
  A DDL parser shall be capable of validating the following:
  a. Values of properties
  b. Structures
  c. Related classes
  d. Values of properties of related classes
- **Intellectual property management**
  The DDL shall provide a mechanism for the expression of Intellectual Property Management and Protection (IPMP) for description schemes and descriptors.
- **Human readability**
  The DDL shall allow that Ds and DSs can be read by humans.

3. Overview of the Proposals

The mandate of the MPEG-7 Description Definition Language (DDL) evaluation team was to evaluate the ten proposals (against the DDL requirements). The outcome was a summary report and a list of recommendations outlining the technologies which should be included, or further investigated for possible inclusion, within the DDL for the MPEG-7 Experimentation Model (XM).

The ten proposals and their respective proposers were:

P109 Ricoh Company Ltd. [9]
P124 DARPA Consortium [10]
P184 DICEMAN Consortium [11]
P354 INT [12]
P484 AT&T, IBM, Columbia University [13]
P486 AVID [14]
P487 CISRA [15]
P547 DSTC [3]
P625 UPMC [16]
P644 Philips Research [17]

In addition there was one late proposal, M4386, by Nanyang Technical University at the Seoul MPEG meeting in March [18].

Since the proposals were designed using very different approaches, based on different Ds and DSs and different assumptions on the scope of the DDL, their comparison for evaluation was extremely difficult. The first task of the DDL evaluation team, after the initial individual evaluations, was to attempt to categorize the proposals to enable comparisons and rankings between them. Figure 1 illustrates the three broad categories in which the proposals could be subdivided: semantics-based, grammar-based and presentation-based approaches. Some proposals were designed purely using one of these approaches, while others adopted one approach primarily but tried to provide capabilities associated with the other approaches through extensions.

The semantics-based approaches started with an object-oriented or knowledge-base schema language, such as Open Knowledge Base Connectivity (OKBC) [10] or Resource Description Framework (RDF) [21] (a W3C XML schema intended for the expression of metadata on web resources) and used a syntax to express this. The grammar-based approaches all started with Extensible Markup Language (XML) [22] and tried to add varying

degrees of semantics through richer constraint mechanisms such as datatyping and inheritance. Presentation aspects were enhanced by the specification of spatial and temporal layout controls. The sole presentation-based approach built on the Synchronized Multimedia Integration Language (SMIL) [27].

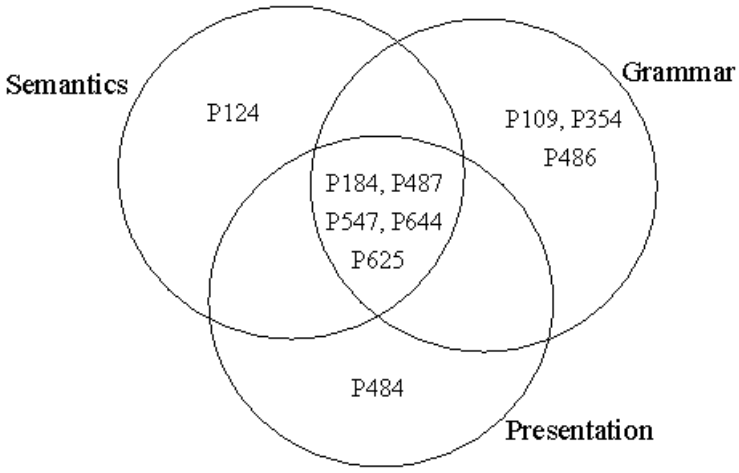


Figure 1: The Three Categories of DDL Approaches

Alternatively the proposals could be categorized according to their target applications:

- The semantics-focussed proposals were mainly interested in search and retrieve applications using knowledge representation and AI concepts;
- The structure- and validation-focussed proposals were mainly interested in controlling and automating the generation of validated content descriptions on a large scale production basis;
- The hypertext-focussed proposals were mainly interested in using links within and between descriptions and content as a means of navigating through multimedia documents.

One recurring difficulty encountered during the evaluations was determining which aspects of the proposed solutions were actually DS or D solutions and which were purely DDL solutions. This was particularly true of the temporal and spatial control specifications.

The evaluation process also made it clear that the choice of a DDL involves determining the optimum balance between simplicity and usability (on the one hand) and the extensibility needed to satisfy the MPEG-7 requirements (on the other hand). Although a pure XML approach, as proposed by several organizations, is readily available, simple and cost-effective, it fails to satisfy certain fundamental DDL requirements such as data typing or coordinate systems. Conversely, certain other proposals offer complete extensibility but at the price of high complexity in DS design and parsing.

4. Summary of the Evaluations

In this section, we describe the results of the DDL proposal evaluations which were carried out by the DDL Evaluation Team in Lancaster [1].

As explained above, the proposals were divided into two main categories – those based on XML and the structured document management approach and those based on object-oriented programming concepts and knowledge base representations. Figure 2 below, taken from [1] illustrates a taxonomy of the capabilities of the different proposals.

Only one proposal, P484 [13], used a primarily presentation-based approach. It proposed the application of SMIL and VRML to a description scheme based on descriptors corresponding to MPEG-4 BIFS. Although this

proposal was the only one to consider MPEG-4, which was acknowledged as a valuable contribution, it's dependence on SMIL seemed better suited to the specification of multimedia presentations than the content description of multimedia objects. This proposal was also hazy on details of how the proposed components were integrated and implemented.

Of the XML-based submissions, three proposed pure XML DTDs (P109 [9], P354 [12], P486 [14]). Their main objective was to show that the mechanisms of document type definitions (DTDs) and validation based on grammar, fulfilled the needs of their particular applications. Although this approach fails to satisfy certain key MPEG-7 DDL requirements such as data typing or coordinate systems, it does enable rapid development and low deployment costs.

Three submissions proposed XML with extensions built on top (P184 [11], P487 [15], P644 [17]). They all proposed extending XML with a set of data types. P184 and P487 rely on XML DTDs to perform validation. P644 uses templates without validation. Each was targeted at a different application and provided additional facilities to execute validation for their particular application. Although these proposals mainly concentrated on representing document structure, certain extensions were proposed to support the representation of varying degrees of semantic information. For example, P487 provides a mechanism for subclassing based on pre-processing of the XML file, as performed with HyTime architectures.

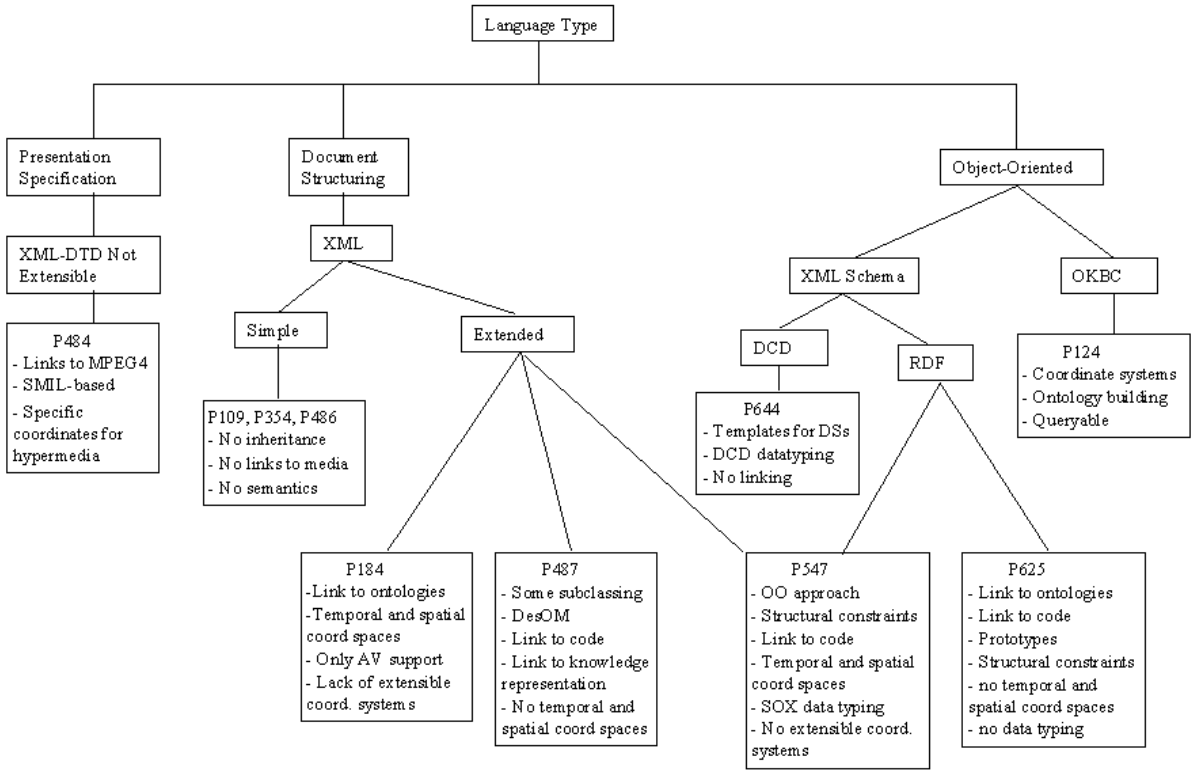


Figure 2: Taxonomy of Proposal Capabilities

P184 [11] proposed extensions to XML which included data typing, a specific temporal coordinate system and the validation of spatial, temporal and structural relations. This proposal also provided strong linking mechanisms and demonstrated a prototype description parser based on the proposed schema. This proposal did not provide any concept of inheritance.

P487 [15] proposed data types based on SOX, links to knowledge representation, mechanisms for inheritance and subclassing of content models and a Description Object Model (DesOM, based on XML DOM). This model provided a standard API so that when a description is parsed, an object model is created in memory which conforms to the DesOM. This facilitates fast, flexible development of interchangeable tools.

P644 [17] proposed Document Content Description (DCD) data types. DCD is a W3C XML schema submission [23]. P644 also used RDF collection types and a set of pre-defined built-in relationships – Order, Equivalence, Peer, Physical. An example illustrating the application of this DDL on medical images was presented.

Three of the proposals were based on object-oriented semantic approaches. These were P124 [10], P547 [3] and P625 [16].

P124 [10] was based on OKBC (Open Knowledge Base Connectivity), an exchange format for knowledge representation languages which is an object-oriented extension of KIF (Knowledge Interchange Format). This submission was the only one to address the problem of semantic expression through ontologies. It also provided powerful ways of describing relations between concepts and a mechanism for specifying any type of coordinate system. Relying on predicate logic, this DDL proposal was much more expressive than the other submissions. However this feature was also seen as a drawback. Because only a very limited set of core classes are provided, the burden falls on the DS creators to define all of the primitive structures. Whilst this may enable the creation of very complex schemes it also places a heavy burden on parsers which have to interpret them.

P547 [3] used the object-oriented, semantic concepts of RDF but expressed them within an XML schema. It extended XML to provide classes, properties and relations and inheritance between classes and properties. It also provided the ability to specify class-specific constraints both on the structures, relations and property values of related classes. It provided data typing through the addition of SOX primitive and extensible data types and cardinality on class structures and properties. The Evaluation Team decided that, although there were some uncertainties over the class representation and the complexity of validation, this proposal provided a good starting point for the DDL.

P625 [16] was also based on RDF. This submission proposed a certain number of extensions to RDF in order to allow structure validation and control, links to ontologies (expressed in Knowledge Representation formats), links to procedural code (in order to extract a descriptor for instance), expression of properties about properties and propagation of properties through links. Moreover, P625 provided an example of the use of such a DDL for a push application taking advantage of users' personalized ontologies to deliver content. However, due to the use of RDF schemas, this proposal suffered from limitations on the class representation.

5. Recommendations of the DDL Evaluation Team [1]

After evaluating all of the proposals, the evaluators came to the conclusion that none of them sufficiently met the DDL requirements to constitute a complete or final solution. The proposals themselves exposed the critical issues, in particular the contrast between object-oriented expression of semantics and validation of structure. Most of the proposals focussed on one of these approaches and tried to extend it towards the other but it became clear that no existing language implementation can adequately reconcile these disparate requirements.

There was a consensus that XML should be used as the *syntax* for the MPEG-7 DDL. There was also a consensus that there is a need for both the validation of structural, relational and data typing constraints and the object-oriented expression of semantics. There has been some concern expressed that declarative languages primarily support structural descriptions but provide little support for describing semantic content. As a consequence of this, the development of the DDL will be a collaborative effort to design and implement an XML-based language which attempts to bridge these two approaches through the addition of richer constraint mechanisms such as data typing and inheritance.

P547 [3], the proposal from DSTC, was selected as the candidate upon which the language design would be based but with substantial integration of ideas and components from other proposals and contributors. In

addition, the strategy was to continue tracking and influencing related efforts in the W3C community, in particular the XML Schema [4], Xlink [5], XPath [6]and XPointer [7] Working Groups.

The following recommendations were made by the DDL Evaluation Team in their output report [1]:

- XML syntax should be used as the DDL syntax. This is because most of the proposals use XML and the MPEG-7 DDL AHG consider it a good basis for achieving the goals of the MPEG-7 DDL. However, XML and XML DTDs have proved lacking in some areas and it was recommended that the MPEG-7 DDL extends XML. As a consequence, a further recommendation was a liaison with the W3C XML Schema Working Group which is already considering such extensions.

- The DDL should support data typing and the creation of new data types. Most of the proposals stated that the MPEG-7 DDL needs data typing mechanisms. The DDL has to define a basic set of primitive data types, such as the ones provided by DCD [23] or SOX [24] data types as in P644, P484, P547, P487, P124 and P184.

- The DDL should support constraints on primitive data types, such as the ones provided in proposals P547, P184, P484, P625.

- The DDL should provide linking mechanisms. The W3C initiatives XLink and XPointer were identified as a basis for linking mechanisms since they can provide:
  - Links between elements of a description;
  - Links between descriptions;
  - Links to a particular element of another description;
  - Links to texts (SGML/XML);

  However there are certain kinds of links which XLink and XPointer do not currently provide but which MPEG-7 requires:
  - Links to multimedia objects;
  - Links to fragments of multimedia objects (P484 gives a solution for MPEG-4 content but this solution requires further examination, P184, P547);
  - Links to knowledge representation (P184, P625, P124, P487);
  - Links to procedural code (P547, P625, P487).

  A liaison with the W3C XLink, XPointer and SMIL working groups was recommended with a goal of extending their capabilities to meet the needs of MPEG-7.

- Given the broad requirements for the expression of MPEG-7 metadata, the DDL should support object-oriented expression of descriptors and description schemes as proposed by submissions P124, P644, P547, P625, P343, P487. Although RDF provides semantics, object-oriented concepts and data modeling, it has a number of limitations. These include its property-centricity, lack of data typing, and the lack of cardinality and class-specific constraints.

- The DDL should provide the ability to define classes and associated *properties* i.e. slots, attributes, members. These properties can be either fundamental data types or instances of other classes.

- The DDL should support inheritance and sub-classing. Inheritance specifies that all properties of a class are included in its subclasses. Sub-classing specifies that properties and constraints that hold for a class also hold for all sub-classes of the class.

- The DDL should support constraints on classes in terms of logical and algebraic expressions over their properties. The logical DDL constraints will have the expressive power of first order predicate logic. The algebraic constraint expressions are defined over properties and functions of properties. The DDL should support the definition of new functions as algebraic expressions of properties or functions of properties. The algebraic operations will include the standard arithmetic and logical connectives. In the first version of the

DDL, only algebraic constraints will be supported in the analysis of schema validity, but future versions should provide for specification of general first-order predicate logic expressions as a means for defining the semantics of Ds and DSs.

- The DDL should provide for relations between classes. The constraint mechanisms for relations should provide a restriction on the membership of the relation i.e. constraints on the domain and range.
- The DDL should provide a means for expressing constraints on the instantiation of DSs. We identified the need for validation of such constraints, as outlined in a number of submissions (P184, P427, P109, P354, P486, P547, P625, P487). The base set of validation constraints should include the validation of:
  - Values of properties/attributes
  - Structures
  - Related classes
  - Values of attributes of related classes
- None of the proposals addressed the real time requirements. The DDL is defined as a language to specify DSs and Ds but it appeared that the real time specification of DSs was meaningless to all proposers. We recommend further discussion and clarification of this DDL requirement.
- Coordinate systems are specifications of axis, dimensions and units, used in descriptions to define n-dimensional references (e.g. temporal, spatial references). As different applications and different contexts require different notions of dimension, the DDL should provide a way to express new coordinate systems. The mechanism for defining coordinate systems should be based on the one provided by P124. The DDL should provide a mechanism for defining transformations between coordinate systems and for specifying a set of core affine transformations. The DDL should also provide a standard set of coordinate systems which can satisfy most applications. Any MPEG-7 user should be able to inherit these coordinate systems for his/her own purpose. The application-dependent extension of this core set by new coordinate systems for specific uses is left to DS designers.
- It is recommended that the following components which were described in the specified proposals be considered for possible integration within the XM:
  - P484 - MPEG-4 BIF links;
  - P644 - DCD data typing;
  - P124 - Object model concepts, coordinate spaces, specification of semantics;
  - P184 - XLink, XPointers, locators, coordinate spaces;
  - P547 – XML Schema, RDF concepts, inheritance, data typing, cardinality, constraints, links to procedure code;
  - P487 – Inheritance and subclassing, DesOM (not normative), reference to procedure code and reference to knowledge base;
  - P486 – Object class, unique identifier scheme (SMPTE UMIDS);
  - P625 – Object-oriented concepts of RDF, structure validation constraints, links to ontologies and procedural code.

6. Overview of DSTC's Proposal P547

In this section, the first author of this paper (who is also the author of Proposal P547) provides the reasoning behind the design and a summary of the features of DDL proposal P547 [3].

An analysis of available schemas (RDF, XML DTDs, DCD, SOX) [19] determined that none of these existing schema languages is ideal for describing multimedia content. Proposal P547 attempts to take the optimum capabilities of each of these schemas and combine and extend them to satisfy the requirements of the MPEG-7 DDL. This proposed schema language is based on a model consisting of classes, properties and relations between classes. It uses the classes and properties model of RDF but adds attributes, relations, timing and spatial controls and data typing capabilities. This schema provides the ability to define generic relationships and apply them with constraints on the attribute values of the related classes. It is also capable of using these definitions to

validate input descriptions. Below is a more detailed description of the different features provided within the MPEG-7 schema language proposed in P547.

6.1 Namespace Declarations

The XML namespace facility [20] enables the inclusion of multiple namespaces. This enables the same feature to have different descriptors which correspond to different domains or description schemes. It also enables the MPEG-7 namespace to borrow descriptors from other namespaces. The ability to mix classification vocabularies within one XML-based encoding allows content description authors to deliver richer domain-specific content descriptions thus increasing the accessibility and re-usability of audiovisual content on the Web. The proposed MPEG-7 schema can also be included in other schemas or descriptions using this same facility. This is a key requirement of the MPEG-7 DDL.

```
<?xml version="1.0"?>
<mpeg7
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:dc="http://purl.org/metadata/dublin_core#"
  xmlns:dcq="http://purl.org/metadata/dublin_core_qualifiers#"
  xmlns:smil ="http://www.w3c.org/TR/WD-smil#">
</mpeg7>
```

6.2 The CLASS Type Declaration and Class Hierarchies

RDF is based on a class and property data model. A collection of classes and the definition of their properties and corresponding semantics represent an RDF schema. Classes are organized in a hierarchy, and offer extensibility through subclass refinement. This way, in order to create a schema slightly different from an existing one, one can just provide incremental modifications to the base schema. In other schemas and XML DTDs, classes and properties are not really differentiated and are all considered to be "elements".

MPEG-7 classes are equivalent to RDF classes but within each class definition, the class's properties and relations are also declared. This is more in line with XML DTDs and provides much easier readability than RDF schemas which are property-centric rather than class-centric. Using this approach, it is easier to see which properties are associated with each class.

The subClassOf property indicates the subset/superset relation between classes. It is based on the rdfs:subClassOf property. It is transitive, so that if class A is a sub-class of some broader class B, and B is a sub-class of C, then A is also implicitly a sub-class of C. Consequently, resources that are instances of class A will also be instances of C. A class can never be declared to be a sub-class of itself, nor of any of its own sub-classes.

Properties are inherited from classes to subclasses. But additional properties can be added to extend subclasses. To simplify the validation, we propose that inheritance be restricted to single inheritance only i.e. each class can have a maximum of one superclass.

```
<class id="MM_Document">
   <property type="#dc_attribs"/>
</class>
<class id="Video_Document">
   <subClassOf type="#MM_Document"/>
   <property type="duration"/>
</class>
```

6.3 The Property Type Declaration

Within RDF schemas, property type definitions specify the allowable domain(s) and range. In P547, the domain is defined by putting the property types in the relevant class definitions and replacing the range with either data type or class type constraints in the property definition. This makes the schema easier to build, read, understand and translate to an XML DTD than is possible using the RDF Schema property-centric approach.

Properties are defined using *propertyType*. Properties can be defined as combinations of other properties, a data type or a class type. The only allowable attributes of *propertyType* are *id*, *datatype* and *classtype*. If a datatype is specified, then the attributes *min, max, minexclusive, maxexclusive* and *default* can also be specified, if they are relevant to the datatype. Details of allowable data types are given in Section 3.7 of [8].

Properties can also be declared to be inherited from or a specialization of other existing properties using *subPropertyOf*. This is equivalent to RDF subPropertyOf. If some property P1 is a subPropertyOf another more general property P2, and if a resource A has a property P2 with value B, this implies that the resource A also has a property P1 with value B.

The actual association of properties with classes or within other properties, is specified using the *property* element. Permissible attributes of *property* are : *type* (mandatory) and *occurs* (optional). The *occurs* attribute specifies the occurrence of properties and can have the values: *required* (one only), *optional* (zero or one), *zeroormore*, *oneormore*, *(n,m)*. It is also possible for properties to be combined into groups (of order *Seq*, *Alt*, *Bag* or *Par*) within a propertyType.

```
<propertyType id="timeStamp">
   <Alt>
        <property type="#SMPTE"/>
        <property type="#frame_num"/>
        <property type="#secs"/>
   </Alt>
</propertyType>

<propertyType id="startTime">
   <subPropertyOf type="#timeStamp"/>
</propertyType>

<propertyType id="endTime">
   <subPropertyOf type="#timeStamp"/>
</propertyType>

<propertyType id="SMPTE" datatype="dateTime"/>
<propertyType id="frameNum" datatype="int"/>
<propertyType id="secs" datatype="float"/>

<propertyType id="keyFrame" classtype="#Frame"/>

<propertyType id="dc_attribs">
   <property type="dc:title" occurs="zeroormore"/>
   <property type="dc:creator" occurs="zeroormore"/>
   <property type="dc:subject" occurs="zeroormore"/>
   <property type="dc:description" occurs="zeroormore"/>
   <property type="dc:publisher" occurs="zeroormore"/>
   <property type="dc:contributor" occurs="zeroormore"/>
   <property type="dc:date" occurs="zeroormore"/>
   <property type="dc:type" occurs="zeroormore"/>
   <property type="dc:format" occurs="zeroormore"/>
   <property type="dc:identifier" occurs="zeroormore"/>
   <property type="dc:source" occurs="zeroormore"/>
   <property type="dc:language" occurs="zeroormore"/>
   <property type="dc:relation" occurs="zeroormore"/>
   <property type="dc:coverage" occurs="zeroormore"/>
   <property type="dc:rights" occurs="zeroormore"/>
 </propertyType>
```

6.4 The Relationship Type Declaration

One of the major advantages of RDF Schema is that its additional semantics enable relations other than "contains" to be defined. The other schemas only provide the inherent "contains" relation. However, this advantage is reduced by the need to specify a single range constraint. Since each property can only have a single range, multilayered structures can't be described using a single generic "contains" property. This requires multiple specific "contains" properties i.e. "contains_sequences", "contains_scenes", "contains_shots", "contains_frames", each with their own specific range constraint. One alternative is to implement code outside of the schema which understands specific descriptor semantics (e.g. DC.Relation.HasParts) and can perform the validation. Another alternative which has been proposed within the RDF comments mail archive [30] is to enable class-specific constraints on properties. Class-specific constraints allow the range to be specified with the property or relation inside the class definition. This proposal adopts this latter approach.

Relationships between classes can be defined using the *relationType* declaration. A *relationType* declaration must contain the name of the relation, a domain (the class to which the relation is applied) and a range (the related class). Only binary relationships are supported. In addition, it is possible to specify constraints on the properties of the related classes using the *constraint* specification. Within RDF, one cannot map relationship-type properties between classes to constraints on the property values of the classes involved. For example, if a sequence "contains" a scene, then the start and end times of the scene, must lie within the start and end times of the sequence. Mechanisms for specifying such constraints are not supported by RDF Schema.

Class-specific constraints are defined by inserting *relation* elements within class definitions. The domain corresponds to the container class and the range and constraint attributes can be used to override the range and constraint values in the relation definition. The range and domain for class-specific constraints must be sub-classes of the range and domain in the relationType definition.

The presence of an *order* attribute within the range specification implies that the range can be a group of objects of a particular class and that the group will have the specified ordering. If the *occurs* attribute specifies groups but no ordering is specified, then any of the ordering types are permissable. If the range consists of a group, then the particular attributes are specified using an array type syntax e.g. range[1] is the first element of the group, range[n] is the last element and range[i] represents all of the elements.

The *constraint* element is used to specify constraints on the property values of the related classes. It has two attributes: *type* and *value*. The *type* can be either *boolean* or *program*. The *value* is either a logical expression using range and domain properties or a call to a remote program e.g. a CORBA IDL wrapping around some external function. The program approach would be used to perform more sophisticated content checking and validation.

In addition, the relation can be defined as either uni-directional or bi-directional via attributes. If the relation is uni-directional, then an optional inverse relation name can be specified. If the relation is bi-directional, then the inverse relation is the same as the original relation.

```
<relationType id="contains" direction="uni" inverse="#contained_by">
    <domain type="#MM_Document" />
    <range type="#MM_Document" occurs="zerormore" order="Seq"/>
    <constraint type="boolean" value=
                    "((range[1].startTime>=domain.startTime)&&
                      (range[n].endTime<=domain.endTime))"/>
</relationType>

<relationType id="overlaps" direction="bi">
    <domain type="#object"/>
    <range type="#object"  occurs="zerormore" order="Bag"/>
    <constraint type= "boolean" value=
      "((domain.X.min<=(range[i].X.min +(range[i].X.max-range[i].X.min)/2)&&
        (domain.X.max>=(range[i].X.min +(range[i].X.max-range[i].X.min)/2)&&
        (domain.Y.min<=(range[i].Y.min +(range[i].Y.max-range[i].Y.min)/2)&&
        (domain.Y.max>=(range[i].Y.min +(range[i].Y.max-range[i].Y.min)/2))"/>
</relationType>

<relationType id="neighbours" direction="bi">
    <domain type="#object" />
    <range type="#object" occurs="zerormore" order="Seq"/>
    <constraint type="program" value="checkNeighbours(domain, range)"/>
</relationType>

<class id="scene">
    <subClassOf type="#MM_Document"/>
    <property type="#dc_attribs"/>
    <relation type="contains" range="#shot"/>
</class>

<class id="object">
    <subClassOf type="#MM_Document"/>
    <property type="#dc_attribs"/>
    <relation type="overlaps" range="#object"/>
</class>
```

6.5 Order and Occurs

The RDF Container syntax is used to specify sets or sequences of classes or properties. Alternatives or ordering among the elements is specified through the *order* attribute. RDF Container syntax provides Seq, Bag and Alt container types. P547 proposes the addition of a *Par* container type for a collection of "parallel" elements. Hence the proposed values for the order attribute associated with groups are:

- Seq - an ordered list of elements or a collection of objects which occur in sequence;
- Bag - an unordered list of elements;
- Alt - a list of alternative choices from which one is possible;
- Par - a collection of objects which occur in parallel or concurrently.

Valid values for the *occurs* attribute are: *required* (occurs exactly once), *optional* (occurs zero or one times), *zeroormore*, *oneormore* and (*n,m*) (A minimum of *n* and a maximum of *m* occurences. *n* must be a positive integer or zero, *m* must be an integer greater than *n*, or "*" which indicates *m* is unbounded.) The occurs attribute can be applied to *property*, *relation*, *group* and *range* elements.

6.6 Data Typing

If the property definition includes the *dataType* attribute, then it is a leaf node which includes the data type definition for this property.

```
<propertyType id="SMPTE" datatype="dateTime"/>
```

```
<propertyType id="frameNum" datatype="int" min="1" max="2000"/>
<propertyType id="secs" datatype="float"/>
```

In addition, it is possible to specify constraints on the content of particular properties using *Min*, *Max*, *MinExclusive*, *MaxExclusive*. *Max* and *Min* allow values upto and including the bound while *MaxExclusive* and *MinExclusive* allow values less than and greater than the bound, respectively, The semantics of upper and lower bounding are highly dependent on the element's Datatype; for some datatypes (e.g. uri), this property has no meaning.

```
<propertyType id="MonthofYear" datatype="int" min="1" max="12"/>
```

It is also possible to specify a default value using the default attribute on the property. For example:

```
<propertyType id="MonthofYear" datatype="int" min="1" max="12"  default="1"/>
```

The data type must be selected from the following:

- the intrinsic data types;
- the library of data types which have been derived from the intrinsic data types (see Appendix B of [8]);
- a user-defined data type;

XML 1.0 defines 10 datatypes, which may only be used to constrain attribute values, and essentially one datatype, PCDATA, that can be used to constrain element content. P547 proposes a much richer set of datatypes, applicable to both attribute and property content. The data types available are based on the SOX data typing capabilities [24].

The list of primitive or intrinsic datatypes are tabulated below. They are the ten XML 1.0 datatypes plus the SOX intrinsic data types [24].

Name	Examples	Description
id	X	XML ID
idref	X	XML IDREF
idrefs	X Y Z	XML IDREFS
entity	Foo	XML ENTITY
entities	Foo Bar	XML ENTITIES
nmtoken	Name	XML NMTOKEN
nmtokens	Name1 Name2	XML NMTOKENS
enumeration	Red Blue Green	XML ENUMERATION Legal values must be specified
notation	GIF	XML NOTATION
string	Give me liberty or give me death!	PCDATA
binary	[01]*	A sequence of bits, represented by 0 or 1.
boolean	0, 1 (1=="true")	"1" or "0"
char	char	A single character.
number	15, 3.14, -123.456E+10	A numeric value.Used when a more specific numeric representation is not required or practical. There are no contraints on the minimum or maximum values, number of digits or number of decimal places.
date	YYYY-MM-DD	A date in a subset ISO 8601 format (no time)
time	HH:MM:SS	A time in a subset ISO 8601 format, with no date and no time zone. Fractional seconds may be as precise as nanoseconds.
uri	urn:schemas-microsoft-com:Office9 http://www.ics.uci.edu/pub/ietf/uri/	Universal Resource Identifier

The enumeration data type requires the legal values to be specified. For example:

```
<propertyType id="cameraDistance" datatype="enumeration">
      <Values>close-up medium-shot long-shot</Values>
</propertyType>
```

6.6.1 User-defined datatypes

P547 proposes using the same approach as the SOX W3C submission [24]. SOX documents provide a mechanism, for defining datatypes that can be used to specify the datatype of an attribute or property content. We provide a similar mechanism through the *dataType* element. A datatype id may be referenced in the datatype attribute of the *propertyType, attributeType, enumeration, format, scalar*, and *string* elements. It is a fatal error to re-assign a datatype name or to reference a datatype that has not been defined.

User-defined datatypes may only be derived from the intrinsic datatypes. There are three types: *scalar*, *enumeration* and *format*. The valid mask values are listed in Appendix C of [8]. The MPEG-7 Schema processor must be capable of generating code to perform validation on the values of user-defined datatypes.

6.6.1.1 User-defined scalar datatypes

User-defined scalar datatypes are derived from the intrinsic number datatype. A derived datatype must specify the number of digits and decimal places, and the minimum and maximum values permitted. An optional mask describes the required format of values that conform to the datatype. The minimum and maximum permitted values may be further constrained by setting the boolean minexclusive and maxexclusive attributes to "1". A processor must be able to generate code that will validate a value against the datatype definition. Recommended scalar datatypes which should be provided include: *byte, double, float, int, long* (Appendix B of [8]).

```
<dataType id="inch">
   <scalar datatype="number" digits="4" decimals="2" min="0" max="12">
      <mask>Z#.##</mask>
   </scalar>
</dataType>
```

6.6.1.2 User-defined enumeration datatypes

User-defined enumeration datatypes may be derived from any of the intrinsic datatypes. Each of the values specified in an enumerated datatype must conform to the specified type. A processor must be able to generate code that will validate the value against the datatype definition.

```
<dataType id="transition">
   <enumeration datatype="string">
      <option>cut</option>
      <option>fade</option>
      <option>wipe</option>
      <option>dissolve</option>
   </enumeration >
</dataType>
```

6.6.1.3 User-defined format datatypes

User-defined format datatypes may be derived from any of the intrinsic datatypes, but will most commonly be used to specialize string values. A required mask describes the required format of values that conform to the datatype. A processor must be able to generate code that will validate the value against the datatype definition.

```
<dataType id="part-number">
   <format datatype="string">
      <mask>AAA-###.##-aa</mask>
   </format>
</dataType>
```

6.6.2 Examples of User-defined MPEG-7 Complex Data Types

Complex data types which the DDL should be capable of supporting include : Colour histograms, 3D vectors, graphs, RGB values etc.

RGB-Values
For example, orange-red is represented by the RGB value 255;69;0 and each value must lie between 0 and 255.

```
<dataType id="rgb">
   <scalar datatype="int" min="0" max="255">
      <mask>ZZ#</mask>
   </scalar>
</dataType>

<dataType id="RGB-value">
   <format datatype="string">
      <mask>rgb;rgb;rgb</mask>
   </format>
</dataType>
```

A Colour histogram

```
<propertyType id="colourHistogram">
      <Seq> <Seq>
            <propertyType id="RGB-data" datatype="#RGB-value"/>
            <propertyType id="frequency" datatype="float"/>
      </Seq> </Seq>
</propertyType>
```

6.7 Attribute Definitions

Attributes are handled in P547 similarly to attributes within the DCD Schema [23]. Attributes can be specified for classes using attributeType. Properties which apply to attributeType are: *id*, *datatype*, *occurs* and *default*. *occurs* indicates whether the presence of this attribute is required. It can take one of two values: *required* or *optional*.

```
<attributeType id="duration" datatype=int occurs="optional" default="1" >

<attributeType id="employment" occurs="required" datatype="enumeration">
     <Values>Temporary Permanent Retired</Values>
</attributeType>
```

An example of the use of attribute and attributeType:

```
<class id="MM_Document">
  <attributeType id="src" datatype="uri"/>
  <attribute type="begin" datatype="timestamp"/>
  <attribute type="end" datatype="timestamp"/>
  <attribute type="duration" datatype="timestamp"/>
  <attribute type="copyright" datatype="uri"/>
</class>
```

In this example, the properties of the attribute whose name is *src* are declared within the declaration of the MM_Document class. This would make sense if MM_Document is the only class for which the *src* attribute applies.

The second attribute, *begin*, has a declaration stored separately, referenced by its id. This declaration style is suitable when such an attribute is applicable to multiple classes; it allows maintaining the declaration in one location.

6.8 Synchronisation and Temporal Specifications

P547 proposes using timing controls similar to those used in SMIL [27]. SMIL provides coarse-grained and fine-grained declarative temporal structuring. Coarse-grained temporal information is provided by the following two structuring elements:

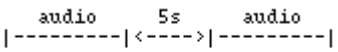
- <seq> ... </seq>: A set of objects that occur in sequence.
- <par> ... </par>: A collection of objects that occur in parallel.

Elements defined within a <seq> group have the semantics that a successor is guaranteed to start after the completion of a predecessor element. Elements within a <par> group have the semantics that, by default, they all start at the same time. Once started, all elements are active for the time determined by their encoding or for an explicitly defined duration. Elements within a <par> group can also be defined to end at the same time, either based on the length of the longest or shortest component or on the end time of an explicit master element. If objects within a <par> group are of unequal length, they will either start or end at different times, depending on the attributes of the group.

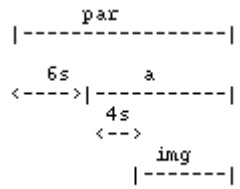
Fine-grained synchronization control is specified in each of the continuous media object references through the following values expressed as either attributes or properties and using the comprehensive set of date and time data types provided in the library:

- *duration*="length" attribute can be used to state the presentation time of the object;
- *begin*: specifies the explicit start of an object. It can be given as an absolute offset from the start time of the enclosing structural element by using *begin*=" time " attribute e.g. begin=="5s"; Alternatively it can be given as a relative offset to the start or end time of another sibling object using *begin*="object_id +time" attribute e.g. begin="id(x)(45s)" , begin="id(x)(end)"
- *end*: this attribute specifies the explicit end of the element

```
<seq>
  <audio_track src="audio1" />
  <audio_track begin="5s" src="audio2" />
</seq>
```




```
<par>
  <audio_track id="a" begin="6s" src="audio1"/>
  <img  begin="id(a)(4s)" />
</par>
```



6.9 Spatial Specifications

It is possible to break an element into spatial subparts by specifying the outline of each region. P547 proposes two methods for describing outlines: rect and outline. *rect* specifies a rectangle within a visual media object. The

syntax and semantics of this attribute are similar to the coords attribute in HTML image maps, when the link is associated with a rectangular shape. The rectangle is specified by four values. The first two values specify the coordinates of the upper left corner of the rectangle.The second two values specify the coordinates of the lower right corner of the rectangle. Coordinates are relative to the top left corner of the visual media object. If a coordinate is specified as a percentage value, it is relative to the total width or height of the media object display area. Similarly non-rectangular outlines are specified by a sequential list of (x,y) pairs measured relative to the top left corner of the container visual media object. The rect and outline region properties have the following syntax:

rectangular_region = "left-x , top-y , right-x , bottom-y"
non-rectangular_region = "x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x6,y6"

For example:
region="0%, 0%, 50%, 50%"
region="10, 10, 90, 90"
region="10,25,15,85,85,110,56,96"

7. Simple Example Based on P547

The example below illustrates the differences between classes, properties, attributes and relations as used in proposal P547 :

- *Classes* are equivalent to XML entities.
- *Properties* and *Attributes* are equivalent to XML elements and attributes respectively. They differ in where they appear in the actual description.
- *Relations* are used to express structural, temporal, spatial and conceptual relations between classes.

Figure 3 illustrates the data model for a Scene class from a fictitious video description scheme.

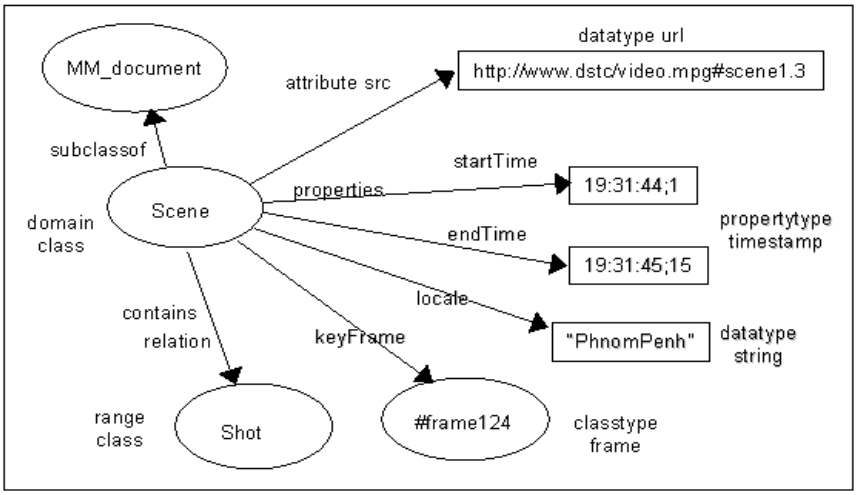


Figure 3: Example of the Data Model for A Scene Description Scheme

Below is the portion of the Schema which represents the data model illustrated in Figure 3.

```
<class id="MM_Document">
    <attributeType id="src" datatype="url"/>
    <property type="#startTime" occurs="required"/>
    <property type="#endTime" occurs="required"/>
    <relation type="#contains"/>
</class>

<class id="Scene">
    <subclassof type="#MM_Document"/>
    <relation type="#contains" range="#Shot"/>
    <propertyType id="locale" occurs="optional" datatype="string"/>
    <propertyType id="keyFrame" occurs="optional" classtype="#Frame"/>
</class>

<relationType id="contains" direction="uni" inverse="contained_by">
    <domain="#MM_Document"/>
    <range="#MM_Document"/>
    <constraint type="boolean"
        value="((range.startTime>=domain.startTime)&&(range.endTime<=domain.endTime))"/>
</relationType>

<propertyType id="startTime">
  <subpropertyof type="#timeStamp"/>
</propertyType>

<propertyType id="endTime">
  <subpropertyof type="#timeStamp"/>
</propertyType>

<propertyType id="timeStamp">
  <Alt>
    <propertyType id="SMPTE" datatype="dateTime"/>
    <propertyType id="frameNum" datatype="int" min="1" max="2000"/>
    <propertyType id="secs" datatype="float"/>
  </Alt>
</propertyType>
```

Below is an example of a description of a scene, based on the MPEG-7 schema described above.

```
<Scene id="#scene1.3" src="http://www.dstc/videos/98-02-20.mpg#scene1.3"/>
      <locale>"Phnom Penh"</Locale>
      <startTime>"19:31:44;1"</startTime>
      <endTime>"19:31:45;14"</endTime>
      <keyFrame>"#frame124"</keyFrame>
      <contains>
          <Seq>
            <Shot id="#shot1.3.1" src="http://www.dstc/videos/98-02-20.mpg#shot1.3.1"/>
            <Shot id="#shot1.3.2" src="http://www.dstc/videos/98-02-20.mpg#shot1.3.2">
           ..
            </Shot>
            <Shot id="#shot1.3.3" src="http://www.dstc/videos/98-02-20.mpg#shot1.3.3"/>
          </Seq>
      </contains>
</Scene>
```

8. Recommended Extensions to P547

Some of the recommended changes to P547 which arose out of discussions during the Lancaster evaluation meeting and from contributions by other proposers included improved support for:

- spatial and complex data types;
- spatial specifications;
- coordinate systems;
- linking mechanisms and
- parser validation.

8.1 Datatype Extensions

It was suggested [1] that the library of provided data types should be extended to include the following base set of spatial data types: 1-D, 2-D, 3-D points, discrete point sets, curves, regions and surfaces. These have been added to Appendix B of [8].

In addition, it was recommended that certain complex data types such as arrays, vectors and lists of given data types and sizes, should also be provided in the data type library.

8.2 Spatial Specifications

It was recommended that spatial specifications be extended to cover 3 dimensions and shapes other than rectangles and polygons i.e. it should be possible to define intervals, lines, planes, polygons, bounding boxes etc. in 1D, 2D and 3D space. If a coordinate is specified as a percentage value, it is relative to the total width or height or depth of the media object display area.

Interval_1d = "left-x , right-x "
Interval_1d = "0%, 50%"
BoundingBox_2d (axis aligned) = "left-x , top-y , right-x , bottom-y"
BoundingBox_2d (axis aligned) = "0%, 0%, 50%, 50%"
BoundingPolygon_2d = "x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x6,y6…"
BoundingBox_3d (axis aligned) = "left-x , top-y , back-z, right-x , bottom-y, front-z"
BoundingPolygon_3d = "x1,y1,z1,x2,y2,z2,x3,y3,z3,x4,y4,z4,x5,y5,z5,x6,y6,z6…"

8.3 Coordinate Systems

The need for coordinate systems was identified during the DDL evaluation meeting. Coordinate systems are specifications of axis, dimensions and units, used in descriptions to define n-dimensional references (e.g. temporal, spatial, etc.). As different applications and different contexts require different notions of dimension, the DDL has to provide a way to express new coordinate systems. The mechanism for defining coordinate systems will be based on the one provided by proposal P124 [10]. The proposal is that the DDL shall provide a standard set of coordinate systems, a set of core affine transformations and the mechanism for defining transformations between coordinate systems.

Coordinate Systems:
1-D, 2-D, 3-D Euclidean
2-D Polar
3-D Cylindrical and Spherical

Transforms:
General Affine transform 1-D, 2-D, 3-D
(2x2, 3x3, 4x4 Affine Matrix specification)

8.4 Links

During the evaluation process, the team identified the need for mechanisms which can support the following types of links [1]:
- Links between parts within a single description (XLinks);
- Links between separate descriptions (XLinks);
- Links between parts of separate descriptions (XLinks);
- Links to texts (SGML/XML) – (XPointers);
- Links to multimedia objects – SMIL-type temporal, spatial and spatio-temporal locators;
- Links to fragments of multimedia objects – SMIL-type temporal, spatial and spatio-temporal locators.

For example, by using SMIL temporal anchors as shown below:

```
<video src = "http://www.dstc/videos/98-02-20.mpg">
    <anchor id="seq3" begin="00:54:24;01" end ="00:56:32;25"/>
</video>
```

A reference to sequence 3 is given by a fragment id: http://www.dstc/videos/98-02-20.mpg#seq3.

Open issues which needed further clarification included:
- When/if the SMIL group intends implementing temporal, spatial and spatio-temporal locators;
- How links can be constrained within the Schema;
- How to define links to knowledge representations and ontologies;
- How to implement links from procedural code to the document and vice versa.

8.5 Implementation of the Parser

A major discussion topic during the evaluation process was how the parser would implement some of the richer constraints provided by the DDL. It was recommended that this should be clarified as part of the ongoing DDL development work.

The MPEG-7 parser or processor checks that a given MPEG-7 description conforms to the particular MPEG-7 schema being used. This includes checking the syntax and grammar of the description as well as validating the constraints. The parser must perform the following types of validation and return appropriate error messages when the MPEG-7 description is non-conformant:

- **Validation of Property Values**. Through the API, the processor will have to provide procedures which can convert strings (PCDATA) to each of the supported data types. It will also need to check that the given value lies within the Min and Max or MinExclusive and MaxExclusive bounds or that the default value is substituted if no value is supplied. If a property is defined to be a particular class, then this must also be checked.
- **Validation of Cardinality**. The processor must check that all cardinality constraints are conformed to.
- **Validation of Inheritance**. Single inheritance only will be supported. Multiple inheritance can easily lead to complex inconsistencies. Inheritance will simply be provided by repetition of the properties and attributes of the superclass within the subclass and repetition of the property components in the sub-property.
- **Validation of Relations and Structures**. The processor must check that the properties and relations used in the MPEG-7 description all conform to the range and domain constraints specified in the MPEG-7 schema.
- **Validation of Property Constraints on Related Classes**. The processor will also need to be able to parse and evaluate the algebraic expressions used to define constraints on the property values of related classes.

9. Current State of the DDL Development

At the MPEG meeting in Seoul in March [29], attempts were made to modify the terminology used in the revised DSTC input [8] to reflect the MPEG-7 terminology based on 'Description Schemes' and 'Descriptors' and to implement extended data typing and inheritance mechanisms by extending the grammar of XML 1.0. In hindsight this approach was inadvisable because it involved modifications to the BNF of XML 1.0 and also breached the copyright of XML 1.0. However at that stage, the XML Schema Language working drafts [30,31, 32] had not yet been released and those members of the MPEG-7 DDL AHG present at the Seoul meeting felt that they had no alternative.

In May this year, the XML Schema WG produced a 2-part working draft of the XML Schema language: XML Schema Part 1: Structures [31] and XML Schema Part 2 : Datatypes [32]. Discussions on the reflector and preliminary encoding of the Generic Audiovisual DS [33] using the XML Schema language, led the DDL group to the decision to use XML Schema language as the basis for the DDL. However certain reservations were raised at the Vancouver MPEG meeting in July concerning this approach. The major concerns were:

- MPEG-7's dependency on the output and time schedule of W3C XML Schema WG;
- Restricted access to internal documents associated with XML Schema development;
- The effect of W3C's copyright of XML Schema language on the ability to add MPEG-7-specific extensions.

As a result of these concerns, further discussions at the Vancouver meeting, led to the decision to develop an MPEG-7-specific language in parallel with the XML Schema development being carried out within W3C [4]. A new grammar based on DSTC's proposal but using MPEG-7 terminology (Description Schemes and Descriptors) and with modifications to ensure simple mapping to XML Schema, was recently developed. Based on this grammar, the following tasks are currently being performed:

- Specification of the BNF and an XML DTD for the new grammar;
- Specification of the validation mechanisms which must be provided by a parser;
- Development of a validating parser for this DDL.

References

[1] MPEG-7 DDL Evaluation Team Summary Report, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[2] MPEG-7 Requirements Document V.7, Doc ISO/IEC JTC1/SC29/WG11 MPEG98/N2461, MPEG Atlantic City Meeting, October 1998.

[3] Hunter J., DSTC , “A Proposal for an MPEG-7 DDL”, P547, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[4] XML Schema Working Group. http://www.w3.org/XML/Group/Schemas.html

[5] XML Linking Language (XLink) W3C Working Draft, http://www.w3.org/TR/xlink/

[6] XML Path Language (XPath) Version 1.0 W3C Working Draft http://www.w3.org/TR/xpath

[7] XML Pointer Language (XPointer) W3C Working Draft http://www.w3.org/TR/WD-xptr

[8] Hunter J., DSTC , “An 'Improved' Proposal for an MPEG-7 DDL”, M4518, 47th MPEG Meeting, Seoul, March, 1999

[9] Kunieda T., Wakita Y., Day N., Ricoh Company Ltd, "MINDS Description Definition Language", P109, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[10] Douglass R.J., DiamondBack Systems Inc. for DARPA Consortium, "Knowledge Representation Language for MPEG-7 DDL", P124, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[11] ACTS-DICEMAN, "DICEMAN Description Definition Language", P184, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[12] Preteux F., INT, "DDL Proposal - Extendible Mark-up Language (XML)", P354, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[13] AT&T, IBM, Columbia University, "MPEG Multimedia Language (MML): A Proposal for MPEG-7 DDL", P484, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[14] McDermid E., Avid Technology, "SMPTE P18.279.981017 - Format for the Interchange of Essence and Metadata in Complex Content Packages", P486, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[15] Lennon A., Wan E., CISRA, "Dynamic Description Framework", P487, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[16] Faudemay P., Joly P., Thienot C. and Seyrat C., Multimedia Indexing Group, LIP6, UPMC, "An Extensible DDL Framework based on RDF and Ontologies", P625, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[17] Krasinski R. and Safadi Y., Philips Research, "A Process for Describing Multimedia Content", P644, MPEG-7 AHG Test and Evaluation Meeting, Lancaster, Feb 1999

[18] Hu M. and Jian Y., NTU, Singapore, "Multimedia Description Definition Language (MDL): Proposal", ISO/IEC JTC1/SC29/WG11 M4386, MPEG Seoul Meeting, March 1999

[19] A Comparison of Schemas for Dublin Core-based Video Metadata Representation, N4212, 47th MPEG Meeting, 7-11 Dec 1998, Rome

[20] XML Namespaces. http://www.w3.org/TR/wd-xml-names.

[21] Resource Description Framework (RDF) Schema Specification", WD-rdf-schema-19981030, W3C Working Draft, October 1998. http://www.w3.org/TR/WD-rdf-schema

[22] Extensible Markup Language (XML) 1.0, REC-xml-19980210, W3C Recommendation 10 February 1 998. http://www.w3.org/TR/REC-xml

[23] Document Content Description for XML, Submission to W3C, 31 July 1998 http://www.w3.org/TR/NOTE-dcd

[24] Schema for Object-Oriented XML (SOX), NOTE-SOX-19980930, Submission to W3C, 15 September 1998. http://www.w3.org/TR/NOTE-SOX

[25] XML-Data, W3C Note, 5 January 1998. http://www.w3.org/TR/1998/NOTE-XML-Data

[26] XML Software, http://www.w3.org/XML/#software

[27] Synchronized Multimedia Integration Language, WD-smil-0202, W3C Working Draft, 2 February 1998. http://www.w3c.org/TR/WD-smil

[28] Brickley D., "Open Issue C23: Class-specific Constraints - Proposed Closure", http://lists.w3.org/Archives/Member/w3c-rdf-schema-wg/1998AprJun/0300.html

[29] MPEG-7 DDL Version 0.01, ISO/IEC JTC1/SC29/WG11 N2731, 47th MPEG Meeting, Seoul, March 1999.
[30] XML Schema Requirements, W3C Note, 15 Feb 1999 http://www.w3.org/TR/NOTE-xml-schema-req
[31] XML Schema Part 1: Structures, W3C Working Draft, 6 May 1999 http://www.w3.org/1999/05/06-xmlschema-1/
[32] XML Schema Part 2: Datatypes, W3C Working Draft, 6 May 1999 http://www.w3.org/1999/05/06-xmlschema-2/
[33] MPEG-7 Description Definition Language Document V 1.0, ISO/IEC JTC1/SC29/WG11 N2862, 48th MPEG Meeting, Vancouver, July 1999.

Biographies

Jane Hunter
Jane Hunter is a Senior Research Scientist within the Resource Discovery Unit at DSTC. She received a PhD in "Integrated Sound Synchronisation for Computer Animation" from the University of Cambridge, UK in 1994. Her current research focus is the development of standards (Dublin Core, MPEG-7), schemas (RDF, XML) and metadata generation tools enabling multimedia resource discovery. She is co-chair of the MPEG-7 Definition Description Language (DDL) group and also a principal investigator in the "HARMONY" International Digital Library project with Cornell and the University of Bristol.

Dr. Jane Hunter
DSTC Pty Ltd e-mail jane@dstc.edu.au
Level 7 GP South tel +617 33654310
University of Qld fax +617 33654311
St Lucia 4072 url http://archive.dstc.edu.au/RDU/staff/jane-hunter.html
Australia

Frank Nack
Frank Nack is a member of the Mobile Interactive Media (MOBILE) division at GMD-IPSI. He took his Ph.D. in 'The Application of Video Semantics and Theme Representation for Automated Film Editing', at Lancaster University, UK. The main thrust of his research is on video representation, digital video production, interactive storytelling and media-networked oriented agent technology. He is active member of the MPEG-7 standardisation group where he served as editor of the Context and Objectives Document and the Requirements Document. He is now chairing the MPEG-7 DDL development group.

Dr. Frank Nack
GMD-IPSI e-mail nack@darmstadt.gmd.de
Dolivostr.15 tel +49 6151 869833
64293 Darmstadt fax +49 6151 869818
Germany url http://www.darmstadt.gmd.de/~nack/