Int J Digit Libr (2004) 4: 139–155 / Digital Object Identifier (DOI) 10.1007/s00799-004-0080-1

An authorization model for multimedia digital libraries*

Naren Kodali[1], **Csilla Farkas**[3,4], **Duminda Wijesekera**[1,2]

[1] Department of Information and Software Engineering, George Mason University, Fairfax, VA 22030–4444, USA
[2] Center for Secure Information Systems, George Mason University, Fairfax, VA 22030–4444, USA
[3] Information Security Laboratory, University of South Carolina, Columbia, SC 29208, USA
[4] Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208, USA
e-mail: {nkodali, dwijesek}@gmu.edu, farkas@cse.sc.edu

Published online: 20 October 2004 – © Springer-Verlag 2004

Abstract. In this paper we present a *generalized authorization model* for multimedia digital libraries. Our aim is to support the enforcement of access control requirements of the original data sources without the need to create a new, unified model for the digital library. We integrate the three most widely used access control models (i.e., mandatory, discretionary, and role-based) within a single framework, allowing seamless accesses to data protected by these security models. In particular, we address the access control needs of *continuous media data* while supporting quality of service (QoS) requirements and preserving operational semantics. The technical core of the paper focuses on the development of *metadata* and the corresponding *metastructure* to represent authorization policies and QoS requirements and shows their applicabilty to continuous media. We define our security objects based on the Synchronized Multimedia Integration Language (SMIL), which controls multimedia presentations. Following the synchronization constructs ⟨par⟩ and ⟨seq⟩ of SMIL, we define a normal form for multimedia streams, called *SMIL normal form.* SMIL normal form provides a syntax-independent representation of semantically equivalent multimedia data. SMIL normal form compositions are extended (decorated) with RDF statements, representing security and QoS metadata. Interpretation of these statements and, therefore, the authorization and QoS requirements of the decorated multimedia object are defined by the metastructure, represented as a DAML+OIL ontology. We propose the concept of *generalized subject* that encompasses all access permissions of a given user regardless of the multiple permissions in different access control models. Finally, we develop methods to generate secure views for each generalized subject and retrieve them using a secure multimedia server.

Keywords: Digital library – Access control – Continuous media – Semantics – Metadata

1 Introduction

Digital libraries support a wide variety of applications, ranging from educational and research activities to government and private sector use. The main focus of digital library research is to develop methods enabling efficient data manipulation for library services. These works encompass open access [30] digital library design. Recently, the need to develop security models for digital libraries has emerged due to the increased dependence of a variety of applications on digital libraries and the potential privacy and copyright requirements. Digital libraries for multimedia data, including continuous media, is a rapidly emerging field with critical applications like surveillance and remote video-audio conferencing.

Existing works for digital library access control [1, 4, 8, 10, 27] assume the existence of a uniform authorization model for the digital library. For example, Adam et al. [1] discuss a content-based authorization model for digital libraries, where access decisions are evaluated based on user credentials and security object identities and their content. In [8] Bertino et al. describe MaX, an access control system based on user credentials and data content. They define a set of privileges, i.e., browsing and authoring, that can be assigned to users. Access control policies are defined by a tuple over credential specification, entity specification (content), privilege, and a sign representing permission or denial. While the model seems expressive to incorporate the most widely used access control models, they require the construction of a new access control for the library collection. To satisfy the original access control needs, it is crucial that each original access control

* This work was partially supported by the National Science Foundation under Grants CCS-0113515 and IIS-0237782.

model be mapped accurately to the new model. Due to the large amounts of data stored in digital libraries with heterogeneous access control requirements, this approach is error prone and cumbersome.

In this paper we present a different approach, one that allows each data object to remain under the protection of its original access control. Our main aim is to allow a user transparent access to data protected by different access control models. We develop a generalized security framework to represent discretionary (DAC), mandatory (MAC), and role-based access control (RBAC) models. We propose the abstract entity generalizing the "subject" of the three security models. Our framework provides transparent retrieval of data by transforming all capabilities of a subject into *(s,o,a)* triples, where s is the subject (acting on behalf of the user), o is the security object, and a is the access permitted on object o to subject s. In this paper we focus on multimedia retrieval only.

We also develop a security ontology (metastructure). This ontology provides interpretation of the metadata attributes of the data objects and enables the enforcement of the access control requirements. We use DAML+OIL (Darpa Agent Markup Language + Ontology Inference Layer) to represent our metastructure.

In addition to the security requirements, multimedia digital libraries also require the preservation of operational semantics and quality of service (QoS) requirements. We use SMIL [5], an XML-like language for authoring multimedia documents, to define protection objects and to represent access control and QoS requirements. SMIL composition operators ⟨seq⟩ (sequential) and ⟨par⟩ (parallel) define a rudimentary semantics of multimedia documents by controlling the presentation timing of multimedia frames. Our aim is to satisfy these timing constraints, thereby preserving the operational semantics of continuous media, while enforcing access control requirements. Due to the properties of ⟨seq⟩ and ⟨par⟩, several syntactic representations are possible for the same operational semantics (Fig. 2). Access control models based on the syntactic structure of protected objects, like the current XML access control models [10, 13–15, 36], are not sufficient to resolve this conflict of multiple syntactic representations of semantically equivalent objects.

We address this problem by developing techniques to transform any SMIL document into a specific syntactic form, called *SMIL normal form (smilNF)*, that preserves runtime semantics. SmilNF allows one to represent SMIL documents in a syntactically consistent manner, where any two semantically equivalent SMIL representations will result in, syntactically, the same smilNF. We use the *smilmetadata* attribute to represent security and QoS restrictions over smilNF. Like the interpretation of the security metadata, we also develop an ontology to describe the QoS metadata.

Finally, we address the need for developing retrieval methods for this extended multimedia data representation. We build upon existing query languages for multimedia retrieval while using the metadata and metastructure information to enforce the security and QoS requirements.

The rest of the paper is organized as follows. In Sect. 2 we give a brief overview of related work. Section 3 introduces SMIL. Section 4 discusses the generalized security framework, including the object identity in SMIL and the generalized subject concept. Section 5 defines the secure normal forms of different security paradigms. The security and the QoS metastructures are presented in Sect. 6. Section 7 gives an example of the usage of metadata in SMIL. Section 8 contains a runtime operation and the query language to manipulate multimedia data. We conclude and recommend future directions for research in Sect. 9.

2 Related work

2.1 Secure digital libraries

Research and applications for digital libraries have increased during the last years. The main aims of these works are to provide standards and technologies for digital library development and to support library services (see [4] for an overview). Recently, the need to develop security models and mechanisms that are applicable to digital libraries has emerged [1, 4, 8, 10, 27]. These studies address storage media security, development of authorization framework, and enforcement mechanisms. They require the development of a uniform access control model for the digital library. However, this requirement not only increases the risk of misclassification of data but is also costly and time consuming. Further, to the authors' best knowledge, none of these studies considers the semantics of continuous media.

2.2 Access control models

The three most widely implemented access control models are discretionary (DAC), mandatory (MAC), and role-based access control (RBAC) [33, 34]. DAC allows database owners (or security administrators) to define access permissions on a user-based manner. That is, there is a direct relation between the subjects (users) and the objects (data items) determined by the access privilege, like that given by access control lists. MAC policies provide controlled information flow between security layers based on the security labels of the protection objects and subjects. Security labels form a lattice structure with the dominance relation between the labels. To provide information confidentiality, data are permitted to flow only from a dominated security label to a dominating security label. In RBAC, the *role* that a user plays within the organization determines his/her access privileges. Privileges are assigned to roles, and roles are assigned to users. A user is allowed to activate any assigned roles and the corresponding privileges within a session.

2.3 Semantic Web technologies

Our authorization model is built on existing technologies to provide data representation and integration on the Semantic Web [7]. In particular, we use XML, RDF, SMIL, and DAML+OIL technologies. The eXtensible Markup Language (XML), a subset of Standard Generalized Markup Language (SGML), has become widely used to provide an efficient format for information exchange. Built upon the XML syntax, the Resource Description Framework (RDF) [21, 28] and RDF Schema [11] provide an enhanced semantic context by allowing one to define entities and their properties and relationships. Hayes et al. [19] and Patel-Schneider et al. [32] describe semantic aspects of RDF. Kodali et al. [24] use RDF vocabulary to specify a secure metastructure for multimedia libraries in different security paradigms.

The developments of ontologies support interoperation among different applications. Current research on DARPA Agent Markup Language (DAML) aims to link information on the Web to domain ontologies. Similarly, Ontology Inference Layer (OIL) integrates ontologies with Web standards like RDF and XML. The latest release of DAML (DAML+OIL) provides constructs to define and integrate ontologies with information resources. The specification of DAML with respect to semantics is explained by Horrocks [20] and Ankolekar et al. [2]. The security aspects of DAML have been discussed by Denker [16] with respect to DAML Web services described in DAML-S [29]. A query specification for DAML ontologies has been proposed by Anyanwu et al. [3]. The DAML Query Language [17, 18] (DQL) is a formal language that enables query-answer interaction between the server and the client agents. A DQL query is a query pattern written in DAML+OIL. The answer to the query is constructed by finding instantiations to the variables of the query.

3 SMIL: Synchronized Multimedia Integration Language

Synchronized Multimedia Integration Language (SMIL) [5] is an extension XML, developed by W3C, for authoring multimedia presentations with audio, video, text, and images. The distinguishing features of SMIL over XML are the syntactic constructs for timing and synchronizing live and stored media streams with qualitative requirements. In addition, SMIL provides syntax for the spatial layout of the media, including nontextual and nonimage media and hyperlinks.

In this paper we use SMIL constructs ⟨seq⟩ and ⟨par⟩ to synchronize media hierarchically. The ⟨seq⟩ element plays its children one after another in sequence. The ⟨par⟩ plays all children elements as a group, allowing parallel playout. For example, the SMIL specifications ⟨par⟩ ⟨video src = camera1⟩ ⟨audio src = microphone1⟩ ⟨/par⟩ specify that media sources camera1 and microphone1 are played in parallel.

In SMIL, the time period that a media clip is played out is referred to as its *active duration*. For parallel play to be meaningful, both sources must have equal active durations. When clips do not have equal active durations, SMIL provides many constructs to equate them. Some examples are *begin* (allows one to begin components after a given amount of time), *dur* (controls the duration), *end* (specifies the ending time of the component with respect to the whole construct), and *repeatCount* (allows a media clip to be repeated a maximum number of times). In addition, attributes such as *syncTolerance* and *syncMaster* control runtime synchronization, where the former specifies the tolerable missynchronization (such as tolerable lip-synchronization delays) and the latter specifies a master-slave relationship between synchronized streams. In this paper we assume that children of ⟨par⟩ have equal active durations.

Retrieving SMIL-formatted multimedia streams need to preserve the sense of continuity and synchronization. Kodali et al. [22, 25] (also Kodali et al., unpublished manuscript) propose three different models for enforcing different security paradigms. A release control for SMIL-formatted multimedia objects for pay-per-view movies on the Internet that enforces DAC is described in [25]. The cinematic structure consisting of acts, scenes, and frames of actual movies are written as a SMIL document without losing the sense of a story. Here access is restricted to the granularity of an *act* in a movie. A secure and progressively updatable SMIL document (Kodali et al., unpublished manuscript) is used to enforce RBAC and respond to traffic emergencies. Kodali et al. [22] describe an MLS application for secure surveillance of physical facilities. Multimedia streams and guards are associated with security labels. Data distribution is governed by these labels.

4 Generalized access control framework

Figure 1 shows our access control architecture and its components. We present transformations that compute all permitted accesses to a user regardless of the access control models used to define these requirements. For this, we need to specify the *objects* (media intervals) and *subjects* (users). We assume that a user may possess multiple security clearances; however, each security object belongs to a single security model. A digital library contains a collection of objects where different access control models specify accesses to different objects. The following sections define our security objects and subjects and the access control granularity.

4.1 Security object identity in SMIL

Unlike XML for textual documents, SMIL constructs have *intended meanings* (i.e., presentation restrictions)

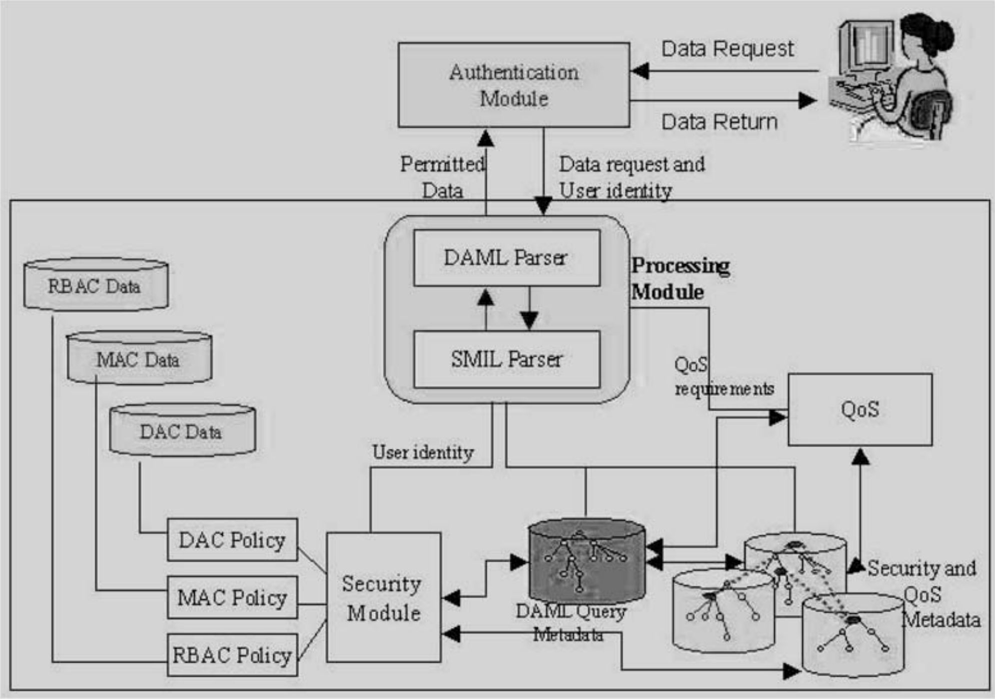


Fig. 1. Architecture of authorization model for multimedia digital libraries

that must be enforced at runtime. In SMIL, the basic objects are media intervals constructed from frames and timing constraints. A media interval begins at a specified time, plays for a specified duration, and ends at a specified time. This time-dependent presentation of the media intervals constitutes a rudimentary semantics. Consider the audio (A_1, A_2) and video (V_1, V_2) intervals shown in Fig. 2a. We assume that all intervals have the same duration. SMIL constructs $\langle$par$\rangle$ and $\langle$seq$\rangle$ can be used to ensure that A_1 and V_1, and, similarly, A_2 and V_2, are played together and presentation of A_2, V_2 follows the presentation of A_1, V_1. A possible representation of this requirement using SMIL is $\langle$par$\rangle$ $\langle$seq$\rangle$ A_1, A_2 $\langle$/seq$\rangle$ $\langle$seq$\rangle$ V_1, V_2 $\langle$/seq$\rangle$ $\langle$/par$\rangle$ (see left tree of Fig. 2c). Assume that this syntactic form is used to enforce presentation constraints on the multimedia document.

However, any of the following four, syntactically different representations of audio(A_1, A_2) and video(V_1, V_2) intervals satisfies the timing constraints (Fig. 2c,d).

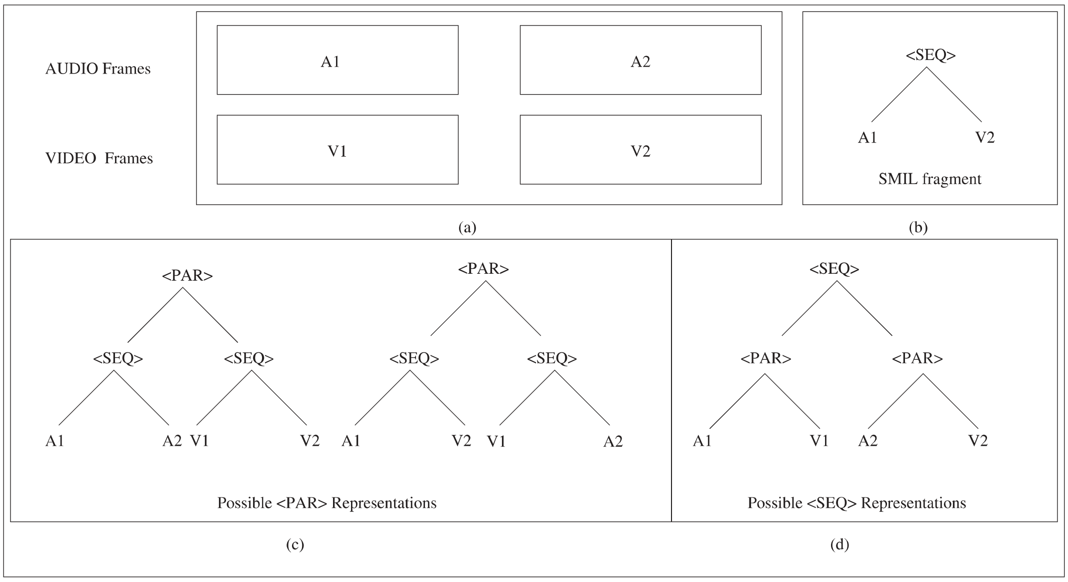


Fig. 2. Equivalence class of SMIL constructs

1. ⟨par⟩ ⟨seq⟩ A_1, A_2 ⟨/seq⟩ ⟨seq⟩ V_1, V_2 ⟨/seq⟩ ⟨/par⟩
2. ⟨par⟩ ⟨seq⟩ A_1, V_2 ⟨/seq⟩ ⟨seq⟩ V_1, A_2 ⟨/seq⟩ ⟨/par⟩
3. ⟨seq⟩ ⟨par⟩ A_1, V_1 ⟨/par⟩ ⟨par⟩ A_2, V_2 ⟨/par⟩ ⟨/seq⟩
4. Because ⟨par⟩ is *commutative* ⟨par⟩ A_1, V_1 ⟨/par⟩ is the same as ⟨par⟩ V_1, A_1 ⟨/par⟩ and ⟨par⟩ A_2, V_2 ⟨/par⟩ is the same as ⟨par⟩ V_2, A_2 ⟨/par⟩.

Now, consider that the system administrator, aware of the presentation requirement, used the right tree of Fig. 2c to define ⟨par⟩ ⟨seq⟩ A_1, V_2 ⟨/seq⟩ ⟨/par⟩ as a disallowed object. Unfortunately, this object is not contained in the tree used for presentation, and thus the security requirement cannot be enforced.

Security objects need to be defined in a clear and unambiguous manner. Syntax-dependent representations, like the models for XML-formatted textual documents [14, 15, 36], where the *protection objects* are nodes of the XML tree, do not capture the data semantics needed for multimedia. This approach may lead to incorrect security enforcement. We propose a new approach by defining a normal form for SMIL documents, representing the identity of the *protection object*. This object is not a node in the XML tree, but an equivalence class. The definition of the normal form is given in Definition 1.

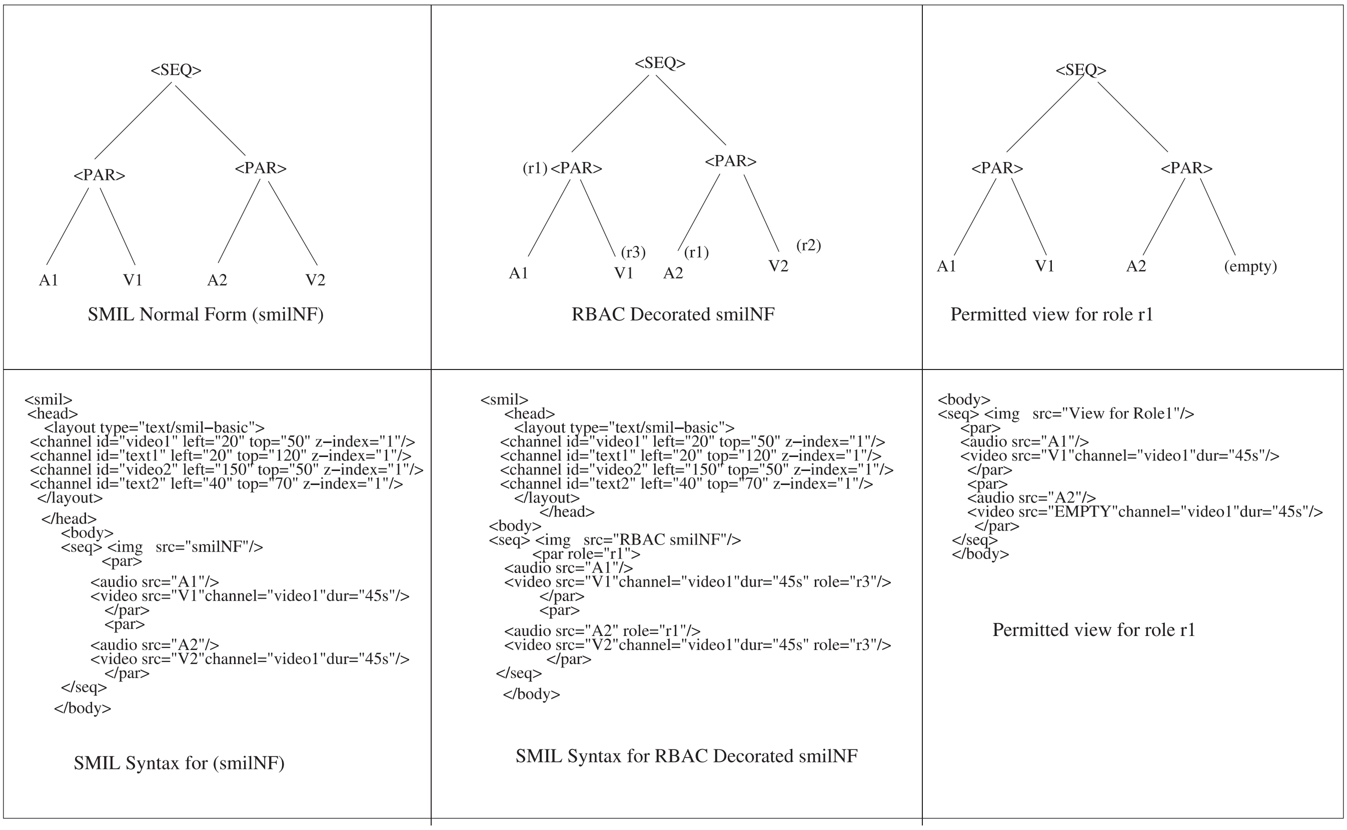
Definition 1 (SMIL Normal Form). *We say that a SMIL specification is in SMIL normal form (smilNF) if it is of the following form:* ⟨*seq*⟩ ⟨*par*⟩ $C_{1,1}(s)$ $C_{1,2}(s)$ … $C_{1,n}(s)$ ⟨/*par*⟩ … ⟨*par*⟩ $C_{m,1}(s)$ $C_{m,2}(s)$ … $C_{m,l}(s)$ ⟨/*par*⟩ ⟨/*seq*⟩, *where* $C_{i,j}$ *are audio or video intervals.*

SmilNF is the basic data object of our security model. It provides a syntax-independent and semantics-aware representation format. We also provide constructs to decorate smilNF with access control and QoS metadata.

Security classification for smilNF can be defined at ⟨seq⟩, ⟨par⟩, or at leaf levels (A_1, A_2, V_1, V_2). Figure 3 shows an example of smilNF decorated with RBAC metadata. A decoration represents the classification of the corresponding node and its subtree. We define classification propagation as follows:

Definition 2 (Classification Propagation).

1. *If the node* ⟨*seq*⟩ *is decorated with security metadata* m, *then all its descendent nodes are also decorated with* m *and all frames must be played according to the presentation requirements.*
2. *If any of the nodes* ⟨*par*⟩ *is decorated with security metadata* m, *then all of its descendent nodes are also decorated with* m *and all frames under this* ⟨*par*⟩ *must be presented together.*
3. *If any of the nodes representing an audio or video frame is decorated with security metadata* m, *then this node can be released to a user with comparable security clearance.*



SMIL Normal Form (smilNF)

RBAC Decorated smilNF

Permitted view for role r1

```
<smil>
 <head>
    <layout type="text/smil−basic">
 <channel id="video1" left="20" top="50" z−index="1"/>
 <channel id="text1" left="20" top="120" z−index="1"/>
 <channel id="video2" left="150" top="50" z−index="1"/>
 <channel id="text2" left="40" top="70" z−index="1"/>
  </layout>
   </head>
      <body>
      <seq> <img   src="smilNF"/>
            <par>
          <audio src="A1"/>
          <video src="V1"channel="video1"dur="45s"/>
            </par>
            <par>
          <audio src="A2"/>
          <video src="V2"channel="video1"dur="45s"/>
            </par>
      </seq>
     </body>
```

SMIL Syntax for (smilNF)

```
<smil>
    <head>
      <layout type="text/smil−basic">
   <channel id="video1" left="20" top="50" z−index="1"/>
   <channel id="text1" left="20" top="120" z−index="1"/>
   <channel id="video2" left="150" top="50" z−index="1"/>
   <channel id="text2" left="40" top="70" z−index="1"/>
      </layout>
           </head>
  <body>
  <seq> <img   src="RBAC smilNF"/>
          <par role="r1">
    <audio src="A1"/>
    <video src="V1"channel="video1"dur="45s" role="r3"/>
           </par>
           <par>
    <audio src="A2" role="r1"/>
    <video src="V2"channel="video1"dur="45s" role="r3"/>
            </par>
   </seq>
    </body>
```

SMIL Syntax for RBAC Decorated smilNF

```
<body>
<seq> <img   src="View for Role1"/>
    <par>
    <audio src="A1"/>
    <video src="V1"channel="video1"dur="45s"/>
     </par>
     <par>
     <audio src="A2"/>
     <video src="EMPTY"channel="video1"dur="45s"/>
      </par>
  </seq>
  </body>
```

Permitted view for role r1

Fig. 3. Decoration and reduction: RBAC

4. *If a node has two inconsistent security metadata decorations, i.e., two different security labels of MAC, then the dominating security label is considered active.*[1]

Using Definition 2, a consistent and systematic labeling of the semantic object components is possible. The granularity of our access control model is the frame level; however, the minimal presentation granularity is smilNF. Further, our model also supports association-based constraints, forming a second layer of access control.

4.2 Generalized security subject

In this section we present a method to integrate access permissions of different models to define all permitted accesses of a subject. We call this subject the *generalized subject.*

4.2.1 Discretionary access control

Discretionary access control (DAC) permits an action `a` to be performed by a subject `s` on an object `o`. This permission is expressed by constructing an access control triple `(s,o,a)`. Let T denote all access control triples of the form (s, o, a), where o is the object and a is the permitted access on o to subject s.

4.2.2 Role-based access control

Role-based access control (RBAC) models have three entities – roles, users, and privileges – and two associations – subject-to-role and role-to-privilege. A subject may activate any set of authorized roles within a session[2] to obtain privileges assigned to the activated roles. Each session is associated with a single user, but a user may have several sessions active at a time.

For each subject `s`, let the set of active roles be given by $ActR(s)$, and let $AuthR(s)$ be the set of roles permitted to be invoked by `s`. Then, the restriction that a user may activate only authorized roles can be stated as $ActR(s) \subseteq AuthR(s)$.

Access permissions of each role r_i are denoted as rToPer(r_i), where rToPer(r_i) consists of pairs (object, action) permitted to play role r_i. Then (s, o, a) can be computed as follows:

1. Let s denote the subject and $T = \emptyset$ denote all access control triples of the form (s, o, a), where o is the object and a is the permitted access on o to s.
2. Generate active roles of s as the set $ActR(s) \subseteq AuthR(s)$.
3. For each role $r \in ActR(s)$ and subrole r' of r:
   (a) For each pair (o, a) in $rToPer(r)$ or in $rToPer(r')$ let $T = T \cup (s, o, a)$.

Authorized roles of a subject and permission assignments can be generated using security ontology and following the “has-role” and “has-permission” properties.

4.2.3 Multilevel security

In multilevel security, each access permission is guided by the security clearance of the subject and the security classification of the accessed object. Security labels form a lattice structure with the dominance relation among the labels. Information flow between the security labels is controlled based on the security objectives. In this paper we allow information flow from a dominated object to a dominating object. Assuming that our access permissions are “read” permissions means that a subject is allowed to access an object only if the subject’s security clearance dominates the security classification of the object.

To model the dominance relation, first we construct the transitive closure of dominance relations. We use this closure to identify the security objects that are permitted to be retrieved by the subject.

Let $SecLabel(s)$ denote the clearance of subject s and $SecLabel(o)$ denote the classification of object o. L denotes the lattice structure and $can-read(l_1, l_2)$, $l_1, l_2 \in L$ a binary relation, i.e., information can flow from l_2 to l_1. The can-read property corresponds to the dominance relation, that is, $can-read(l_1, l_2)$ means that label l_1 dominates label l_2. The following procedure generates all (s, o, a) for a subject s with security clearance $SecLabel(s)$:

1. Let s denote the subject and $T = \emptyset$ denote all access control triples of the form (s, o, a), where o is the object and a is the permitted access on o to s.
2. Generate transitive closure of *can-read*.
3. Let $Dominated(s) = \emptyset$.
4. For all pairs $can-read(l_i, l_j)$, where $l_i = Sec.Label(s)$, $Dominated(s) = Dominated(s) \cup l_j$.
5. If $SecLabel(o) \in Dominated(s)$, then (s, o, a).

That is, a subject is granted the access a to an object o if the security clearance of the subject dominates the security classification of the object. Hence MAC could be stated as an (s, o, a) triple. In effect, the generalized access control rule in all three domains could be declared as an (s, o, a) triple.

A generalized subject s acting on behalf of a user is computed based on the unique identities of the user. Our aim is to provide a seamless representation of users without going through several rounds of authentication and identification. For a given user u and the subject s running on behalf of u, we collect the set of DAC restrictions, permitted security clearance, and assigned roles. These

[1] Note that for DAC or RBAC this is not a problem since the same document may be accessible to several different roles or users. However, an object can belong only to one of the security classes.

[2] Constraints like separation of duties may further limit the activation of roles within a session.

data, along with the security ontology, are used to transform all allowed accesses into (s, o, a) triples. All access permissions of a generalized subject are given by $T_{DAC} \cup T_{MAC} \cup T_{RBAC}$.

5 Secure normal forms

In the previous section we defined our security objects based on operational semantics of continuous multimedia. Each object is guarded by a particular access control model; thus the collection of all objects is guarded by heterogeneous access control. In this section, we present a method of preprocessing this heterogeneously classified collection to increase the performance of the retrieval. Given a smilNF multimedia document with security metadata attributes, we compute a view that is permitted for each subject, security level, or role. They are referred to as *security normal forms.* Definitions 3–5 describe the security normal forms.

5.1 Normal form for DAC

The smilNF specification is decorated with the DAC metadata and upon reduction would group all permitted segments of a particular subject under a single ⟨par⟩ construct. Each of these ⟨par⟩ constructs is the *view* of the corresponding subject.

Definition 3 (DAC Normal Form). *We say that a smilNF specification ($\tilde{s}$) is in the DAC normal form (dacNF) if it is of the form ⟨seq⟩ ⟨par⟩ $C_1(\tilde{s})$ ⟨/par⟩ ⟨par⟩ $C_2(\tilde{s})$ ⟨/par⟩ ... ⟨par⟩ $C_n(\tilde{s})$ ⟨/par⟩ ⟨/seq⟩, where $C_1, C_2, \dots C_n$ are media intervals permitted to be accessible to subjects $s_1, s_2, \dots , s_n$, respectively.*

5.2 Normal form for MAC

The smilNF specification is decorated with the MAC metadata. A normal form in mlsNF is one that is a parallel composition of single security level documents. This parallel compositions can be viewed as single-level views of the multilevel security multimedia document.

Definition 4 (MLS Normal Form). *We say that a smilNF specification ($\tilde{s}$) is in the mlsNF (MLS normal form) if it is of the form ⟨seq⟩ ⟨par⟩ $C_{l_1}(\tilde{s})$ ⟨/par⟩ ⟨par⟩ $C_{l_2}(\tilde{s})$ ⟨/par⟩ ... ⟨par⟩ $C_{l_n}(\tilde{s})$ ⟨/par⟩ ⟨/seq⟩, where $l_1, l_2, \dots , l_n$ represent all security labels, and in C_{l_i}, ($\tilde{s}$) is the media stream classified at label l_i.*

5.3 Normal form for RBAC

The smilNF specification is decorated with the RBAC metadata. A normal form in rbacNF is one that is of parallel composition at one or more role specifications, where each specification belongs to a particular role assignment and is said to be the view corresponding to the assigned role.

Definition 5 (RBAC Normal Form). *We say that a smilNF specification ($\tilde{s}$) is in the rbacNF (RBAC normal form) if it is of the form ⟨seq⟩ ⟨par⟩ $C_{r_1}(\tilde{s})$ ⟨/par⟩ ⟨par⟩ $C_{r_2}(\tilde{s})$ ⟨/par⟩ ... ⟨par⟩ $C_{r_n}(\tilde{s})$ ⟨/par⟩ ⟨/seq⟩, where $r_1, r_2, \dots , r_n$ are $role_1, role_2, \dots , role_n$, and $C_{r_1}(\tilde{s})$, $C_{r_2}(\tilde{s})$, ..., C_{r_n} are media streams permitted to $role_1$, $role_2, \dots , role_n$, respectively.*

Note that generating security normal forms may result in redundant data representation, i.e., if the same object is permitted to both role$_1$ and role$_2$, it appears in the view for both roles. The justification of this approach is to increase the efficiency of playback of the multimedia stream by reducing the complexity of data reconstruction. This approach is similar to the replicated multilevel database architecture.

The algorithms for the reduction of the smilNF to the appropriate secure normal forms, based on the security paradigm that we are using, are described in detail in [23, 24]. When we convert to a secure normal form, we encounter different time containers, some of which are nested. Figure 3 shows an example decoration and reduction in the role-based environment. The figure contains the syntax of (a) smilNF, (b) RBAC-decorated smilNF, and (c) the view derived for role r_1 and the corresponding tree structures. During the rewrite, some of the nodes become ⟨empty⟩. This representation is used to establish an audio or video *silence* in the playout. As we noted earlier, we assume that each multimedia interval has the same duration. When grouping elements that satisfy a particular access control rule, those that do not qualify should be eliminated. Normally, a silent audio segment or a blank video segment is used during playout to maintain continuity and synchronization.

6 DAML metastructure

The current specification of SMIL [5] does not have constructs for security and QoS, and therefore metadata are needed for specifying them. The SMIL metamodule [31] claims that RDF could be used to declare metadata to be used within a SMIL document but does not provide sufficient details. The RDF [21] and RDFS [11] enable defining metadata but not the interpretation or anticipated meaning applicable to multimedia. To interpret metadata for enforcing security and quality on SMIL documents, we propose a metastructure based on DAML+OIL.

Figure 4 represents the class hierarchy of the metadata we define in DAML+OIL for specifying security in a SMIL-formatted multimedia document. Figure 4 represents those components necessary to represent security parameters chosen for this study.

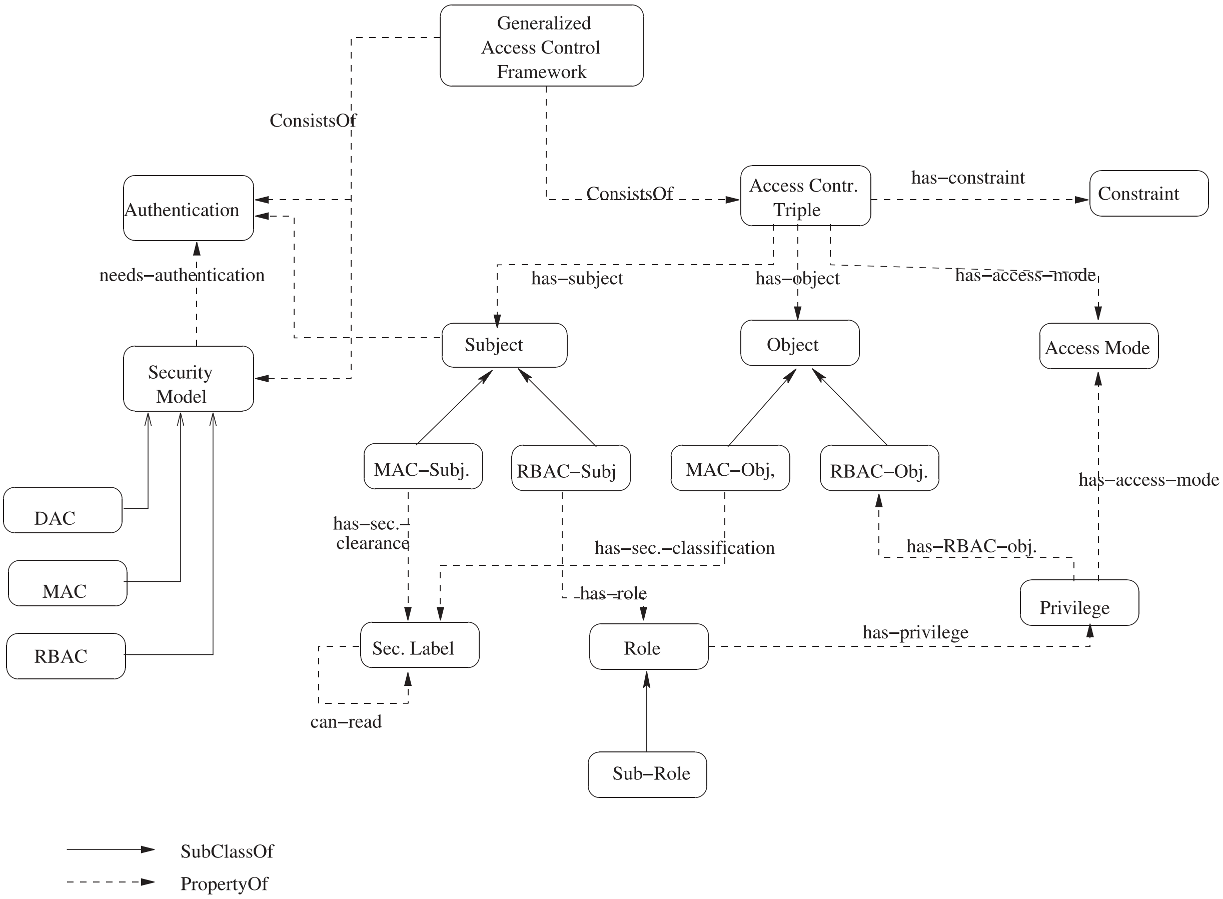


Fig. 4. Class hierarchy of security metastructure

Figure 5 represents the class hierarchy of the metadata we define in DAML+OIL for specifying QoS in a SMIL-formatted multimedia document. Figure 5 also represents those components necessary to represent QoS parameters chosen for this study.

6.1 Security metastructure

Documents of a multimedia digital library may be protected by DAC, MAC, and RBAC models. A user may be authenticated as a DAC user, MAC user, or a RBAC user. However, it is assumed that objects have only one access control specification, that is, a single object is either DAC, MAC, or a RBAC object, but only one of these. This section contains the metastructure defining the security framework for our model and corresponding concepts. We focus on access control and provide interpretation for DAC, MAC, and RBAC models. The generalized security framework consists of a description of *access control models*, *access control triples*, and *constraints* that further restrict accesses.

In the remaining sections, we require that all documents be in their appropriate security normal forms. To process a data request, the security and QoS ontologies are queried to identify relevant data.

6.2 Access control models

First, we define the three access control models used in our framework. They are subclasses of the general class *Security Model*, supporting future extension of our security framework with additional access control. Following is the DAML+OIL specification of the security models.

```
<daml:Class rdf:ID="AC Models">
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Disc Access Control">
<rdfs:comment>
Discretionary Access Control allows the owner
of a security object to define who is allowed
to access that object and what access mode,
i.e., read, write, execute.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#AC Models"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Mandatory Access Control">
```

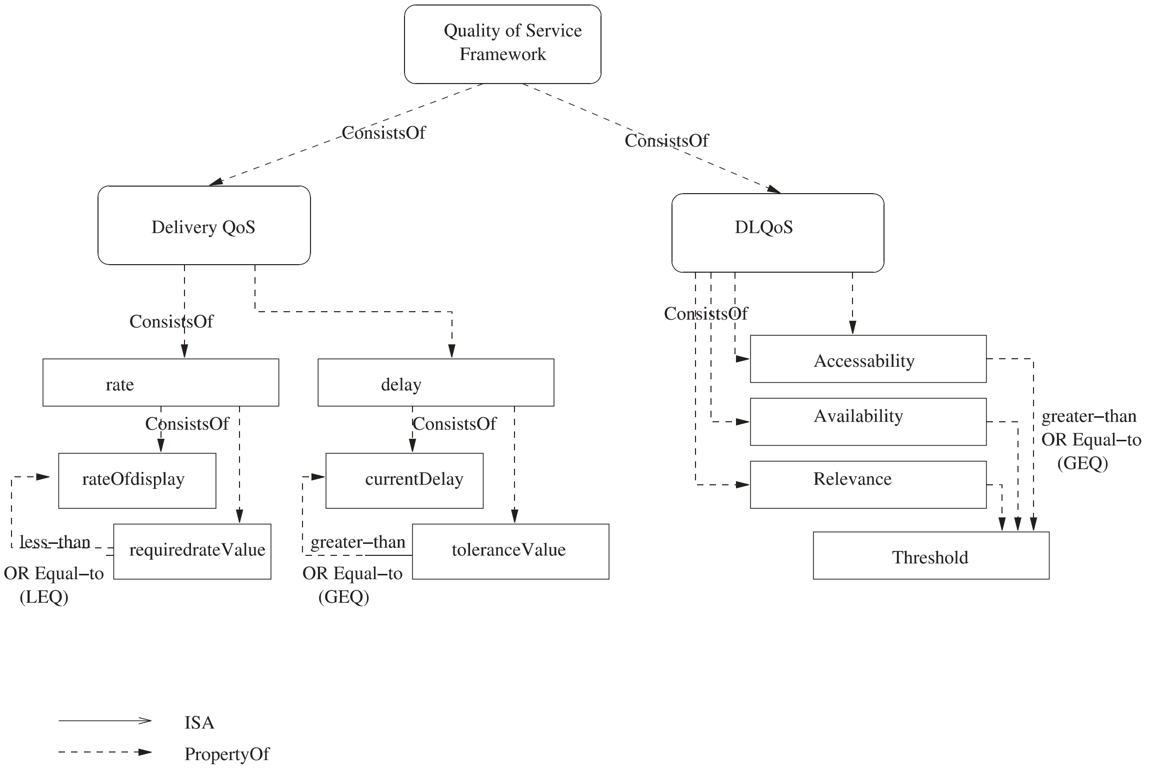


Fig. 5. Class hierarchy of QoS metastructure

```
<rdfs:comment>
   Mandatory Access Control assigns security
   labels to subject and objects. Access
   requests are evaluated based on the
   comparison of these labels.  For example,
   the BLP model allows information flow
   from low-security labels to high-security
   labels.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#AC Models"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>


<daml:Class rdf:ID="Role-Based Access Control">
<rdfs:comment>
Role-Based Access Control assigns users
to roles and roles to privilege (object,
access mode) pairs. Each access request
is evaluated based on the role a user
plays within a session and the privileges
associated with those roles.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#AC Models"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>
```

6.3 Access control triples

As mentioned earlier, we combine all access permissions for a user into a set of (s, o, a) triples. For each access control model, we define basic concepts, like security label, and their relationships. Subjects and objects are further divided into MAC and RBAC subclasses, incorporating DAC subjects into the class *Subject* itself. In the current version of our ontology, *access modes* are explicitly defined, e.g., read, but can be easily extended to incorporate abstract privileges, like activities. For our application domain, we only need the read (retrieve) permission. Appendix B contains the DAML+OIL specifications for *Access Control Triples* subtree.

```
<daml:Class rdf:ID="Access Control Triples>
 <rdfs:comment>
    Defines the (s,o,a) triples.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Subject"> <rdfs:comment>
   Subjects are the active entities in
   the system. Access permissions are
   requested by the subjects.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
```

```
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="MAC-Subject">
<rdfs:subClassOf rdf:resource="#Subject"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="RBAC-Subject">
<rdfs:subClassOf rdf:resource="#Subject"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
 </daml:Class>

<daml:Class rdf:ID="Object">
 <rdfs:comment>
 Objects are the passive entities in
 the system. Access permissions are
 requested to the objects.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="MAC-Object">
<rdfs:subClassOf rdf:resource="#Object"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="RBAC-Object">
<rdfs:subClassOf rdf:resource="#Object"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>
```

6.4 Constraints

Additional restrictions over access control triples and models are expressed as constraints. These constraints may correspond to well-understood restrictions, like time-dependent restrictions, separation of duties, and session control, but may also represent restrictions on applicable security models; for example, accesses to highly critical military objects must have MAC classification. Following are simple examples of constraint specifications in DAML+OIL.

```
<daml:Class rdf:ID="Constraint">
<rdfs:comment>
   Restricts access control rules.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID=
                 "Time-dependent Constraints">
 <rdfs:subClassOf rdf:resource="#Contraints"/>
 <rdfs:subClassOf rdf:resource="#Sec Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID=
              "Context-dependent Constraints">
 <rdfs:subClassOf rdf:resource="#Contraints"/>
 <rdfs:subClassOf rdf:resource="#Sec Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Obligatory Constraints">
 <rdfs:subClassOf rdf:resource="#Contraints"/>
 <rdfs:subClassOf rdf:resource="#Sec Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>
```

6.5 QoS metastructure

The DAML-QoS metastructure defines metadata for enabling QoS restrictions and negotiations on multimedia digital libraries. There are two sets of metadata defined for DeliveryQoS (client-server related) and DLQoS (digital library organization and retrieval related). The complete metastructure is given in Appendix B, but we describe the contents in this section.

The DeliveryQoS class and its related subclasses define metadata pertaining to the delivery of multimedia content from the library to the recipient device. The most important delivery factors are *rate* and *delay*. In the service level agreement (SLA) between the library and the clients, threshold values for the expected rate and tolerable delay are contracted. The metadata should enable conformance to such a contract by providing means of enforcement and negotiation. The threshold values are represented by the subclasses *requiredRateValue* for the rate and *toleranceValue* for delay. Constraints *greaterTHANORequal* and *lessTHANORequal* are defined to relate the current values to the threshold values and enforce conformance.

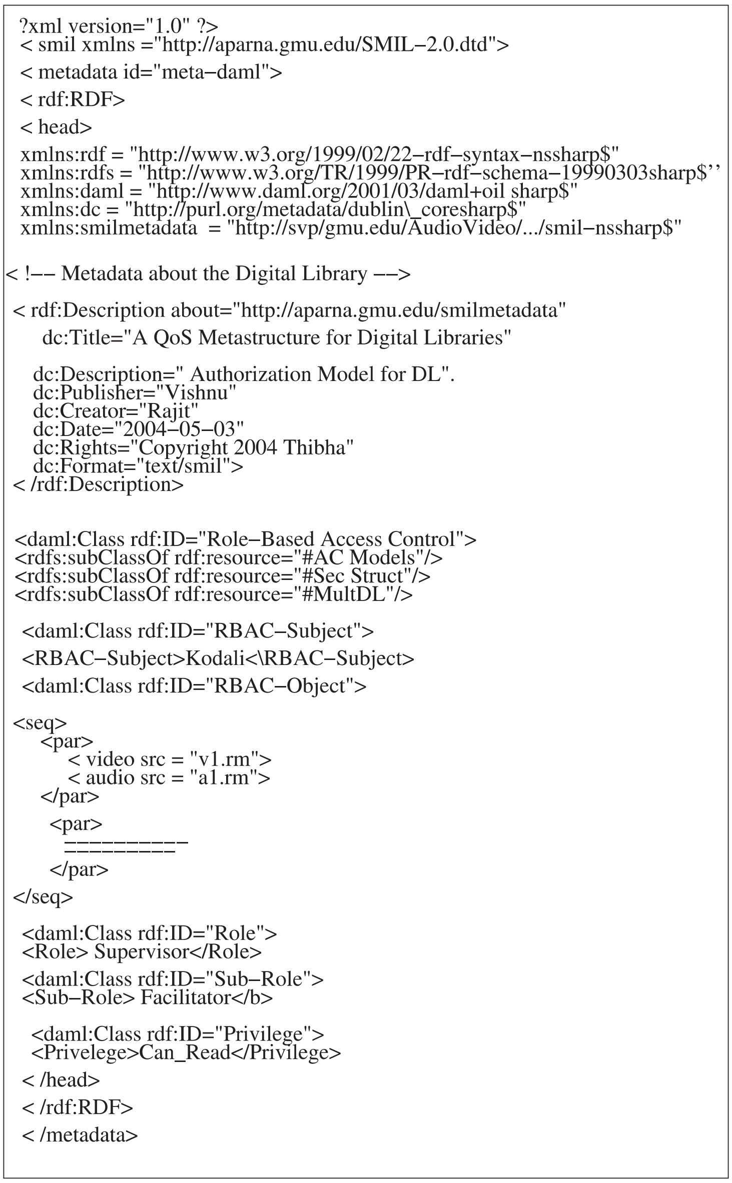
The DLQoS class and its related subclasses define metadata for the QoS issues related to the digital library itself. QoS requirements for multimedia digital libraries have been proposed by Bertino et al. [9]. We consider *accessibility*, *availability*, and *relevance* to be the driving factors for such a QoS requirement. Each of these parameters is marked on a uniform scale upon which thresholds can be dynamically defined. The allegiance to the threshold would be the deciding factor in deciding the conformance to the QoS requirements.

7 Metadata in SMIL

This section describes how the designed DAML+OIL metastructure could be used in association with a SMIL specification. The URI, DAML+OIL, and RDF metastructure enables the DAML interpreter to understand the intended meaning of the metadata. For namespace references *smilmetadata* is utilized for the DAML metastructure. Once the name space is determined, the metadata allowed by the metastructure (*smilmetadata*) are embedded within the SMIL document. The DublinCore metadata [6] are also used for describing information about the document. The SMIL document represented in Fig. 6 uses the DAML metadata that we have created to enforce security and quality restrictions. The title, description, publisher, date, rights, and format are from the Dublin Core URI, which identifies these as standard descriptors. Additional attributes defined using RDF-Schema (delay and rate) could be used to enforce QoS. These parameters are negotiated initially with the display device, even before the body of the SMIL document is interpreted during runtime. If the QoS requirements are not satisfied, the document is rejected.

8 Runtime operations

Our metastructure can be used by a multimedia client that seeks to obtain SMIL documents with proposed DAML/RDF decorations. Our client must use an RDF-based query system for this purpose to generate views for DAC, MAC, and RBAC. DAML Query [18] uses a declarative syntax for selecting DAML representations that meet specified criteria. We show how to construct



```
 ?xml version="1.0" ?>
 < smil xmlns ="http://aparna.gmu.edu/SMIL–2.0.dtd">
 < metadata id="meta–daml">
 < rdf:RDF>
 < head>
 xmlns:rdf = "http://www.w3.org/1999/02/22–rdf–syntax–nssharp$"
 xmlns:rdfs = "http://www.w3.org/TR/1999/PR–rdf–schema–19990303sharp$’’
 xmlns:daml = "http://www.daml.org/2001/03/daml+oil sharp$"
 xmlns:dc = "http://purl.org/metadata/dublin\_coresharp$"
 xmlns:smilmetadata  = "http://svp/gmu.edu/AudioVideo/.../smil–nssharp$"

< !–– Metadata about the Digital Library ––>

< rdf:Description about="http://aparna.gmu.edu/smilmetadata"
     dc:Title="A QoS Metastructure for Digital Libraries"

    dc:Description=" Authorization Model for DL".
    dc:Publisher="Vishnu"
    dc:Creator="Rajit"
    dc:Date="2004–05–03"
    dc:Rights="Copyright 2004 Thibha"
    dc:Format="text/smil">
< /rdf:Description>


<daml:Class rdf:ID="Role–Based Access Control">
<rdfs:subClassOf rdf:resource="#AC Models"/>
<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>

 <daml:Class rdf:ID="RBAC–Subject">
 <RBAC–Subject>Kodali<\RBAC–Subject>
 <daml:Class rdf:ID="RBAC–Object">

<seq>
    <par>
        < video src = "v1.rm">
        < audio src = "a1.rm">
    </par>
      <par>
        ==========
      </par>
</seq>

 <daml:Class rdf:ID="Role">
 <Role> Supervisor</Role>
 <daml:Class rdf:ID="Sub–Role">
 <Sub–Role> Facilitator</b>

   <daml:Class rdf:ID="Privilege">
   <Privelege>Can_Read</Privilege>
 < /head>
 < /rdf:RDF>
 < /metadata>
```

Fig. 6. Representing RBAC metadata within SMIL

a DAML query to retrieve the view for a given generalized subject. An example query to retrieve all objects corresponding to a particular security classification and for a particular role is given. A DAML interpreter is necessary to understand and assemble a SMIL view from an RDF-decorated SMIL document that is to be interpreted by a SMIL player at the client. Although we do not provide such an interpreter, our client needs to have two interacting interpreters, where the SMIL interpreter calls the RDF interpreter to interpret RDF decorations.

8.1 DAML query language

The DAML Query Language [17, 18] is a formal language that enables query-answer interaction between the server and the client agents. DQL is designed to support a query-answering dialog in which the answering agent may use automated reasoning methods to derive answers to queries, the knowledge to be used in answering a query may be in multiple knowledge bases on the Semantic Web, and those knowledge bases need not be specified by the querying agent. In addition, the DQL specification includes formal descriptions of the semantic relationships among a query, a query answer, and the knowledge base(s) used to produce the answer.

A DQL query contains a query pattern and an answer KB pattern that could be an answer KB, a list of answer KBs, or a variable. If the variable forms the answer KB pattern, then the replying agent sends a reference to the answer KB that was used.

All variables should either be categorized as *must-bind* or *may-bind*. The answer must contain bindings to all must-bind variables and optionally for the may-bind variables. Examples of optional elements are *query premise* and *answer bundle size*.

The response to a query (answer) has the following components: answer sentences to the query, query pattern, and agent server that replied to the query. The answer set is enveloped in an answer bundle that contains a continuation token that indicates future interaction patterns.

8.2 Operation of DAML queries

DAMLJessKB [26] is a description logic reasoner for the DARPA Agent Markup Language, implemented using JESS (Java Expert System Shell). The *defquery* component is used to obtain the answer set. The answer set is sent to the querying agent. It is convenient to use JESS defqueries for the purpose of querying the answer KB and returning the answers. We have built an ontology (MSDLQ-Ont) as suggested by Seshagiri et al. [35] to express query patterns in DAML+OIL. We represent facts in our JESS KB as follows:

```
(PropertyValue (s subject) (o object)
                                (a access))
```

We give an example for establishing facts and rules based on the DAMLQuery Ontology in the appendix. below:

```
(defrule  subclassInstances
   "An instance of a subclass is an
   instance of the parent class. This
   enforces and makes meaningful the
   daml:subClassOf relationship"
(PropertyValue daml:subClassOf ?child ?parent)
(PropertyValue rdf:type ?instance ?child) =>
(assert(PropertyValue rdf:type
                        ?instance ?parent)))
```

The *subclassInstances* rule shown above will be activated when the two facts appear in the knowledge base, i.e., when a child that is a subclass of the parent and an instance that is a subclass of child are found. The result of the rule activation is that a fact stating that the resulting instance is a type of the parent class is asserted. Ordered facts can be added to the knowledge base using the assert function. We establish rules and facts as shown above for the entire ontology. An example defquery is as follows:

```
(defquery Constraints "Find all values having
type constraint:Time-dependent"
(PropertyValue rdf:type
                 ?n constraint:Time-dependent))
(defquery RBAC-Object "Find all values having
type  subject:RBAC-Object"
(PropertyValue rdf:type
                      ?n subject:RBAC-Object))
```

By using such a query mechanism, we obtain views for the generalized subject based on the *(s,o,a) triples.* The *normal forms* ensure that all SMIL fragments associated with a generalized subject are grouped together, making the queries easier. Although we propose a syntax, queries could use arbitrary external syntax, as [17, 18] do not specify syntax.

8.3 The runtime algorithm

The runtime algorithm, described in Algorithm 1, retrieves a secure SMIL document. During the first stage, the algorithm negotiates the QoS parameters. A failure of available QoS results in termination of the media transfer. Once the query answer is obtained, the access control policy is evaluated. If access is granted, the associated action is initiated. Views are encrypted to enforce integrity. Several encryption techniques can be used, such as those suggested in [25] (also Kodali et al., unpublished manuscript).

Algorithm 1 Runtime Evaluation Algorithm

```
begin
start DAML interpreter
negotiate QoS Parameters
```

```
if rateOfDisplay = TRUE and toleranceValue = True
then
    query DAML ontology
else
    return with failure
end if
if security-domain = "DAC" then
    run DAC Query
    retrieve associated elements from dacNF
else if security-domain = "RBAC" then
    run RBAC Query
    retrieve associated elements from rbacNF
else if security-domain = "MLS" then
    run MLS Query
    retrieve associated elements from mlsNF
end if
close DAML interpreter.
GRANT or DENY access to elements
if access == GRANT then
    create SMIL-View for the set of conditions
    activate action
    start SMIL interpreter
else if access = DENY then
    return
end if
close SMIL interpreter
end
```

9 Conclusions

In this paper we presented a framework to support the representation of security and QoS requirements in multimedia digital libraries. We showed that syntactic trees used in textual XML documents to specify access control policies are insufficient to specify access control policies for SMIL-formatted multimedia documents. As a solution, we proposed to translate SMIL documents to normal forms and express restrictions on these normal forms.

We also developed a metastructure (ontology) using DAML+OIL [12] to represent access control and QoS policies for multimedia documents. We have shown via examples the applicability of the structure for DAC, MAC, and RBAC. Our security normal forms are similar to secure views computed for XML and other textual documents. We present algorithms to compute normal forms and show a runtime algorithm that uses RDF and SMIL queries to securely retrieve documents decorated with security and QoS attributes.

The presented framework and technologies will ensure efficient processing of multimedia documents, while also guaranteeing security and semantic preservation. We propose a technique to seamlessly integrate different access control paradigms.

Results presented here consider limited aspects of security models with fragments of SMIL syntax. Our future work will extend our current results by addressing additional SMIL constructs. We also plan to implement the extended model, developing joint capabilities for DAML queries and SMIL processing. Our model focuses on effective data retrieval by preprocessing multimedia data and storing then in secure normal form. While this approach slows data insertion and increases storage requirement (replicated data), it speeds up data request processing.

References

1. Adam NR, Atluri V, Bertino E, Ferrari E (2002) A content-based authorization model for digital libraries. IEEE Trans Knowl Data Eng 14(2):296–315
2. Ankolekar A, Huch F, Sycara K (2002) Concurrent execution semantics of DAML-S with subtypes. In: Horrocks I, Hendler J (eds) The Semantic Web – ISWC 2002, 1st international Semantic Web conference, Sardinia, Italy, 9–12 June 2002. Lecture notes in computer science, vol 2342. Springer, Berlin Heidelberg New York, pp 1–318
3. Anyanwu K, Sheth A (2003) P-queries: enabling querying for semantic associations on the semantic web. In: Proceedings of the 12th international conference on World Wide Web. ACM Press, New York, pp 690–699
4. Arms W (2000) Digital libraries. MIT Press, Cambridge, MA
5. Ayars J (2001) Synchronized Multimedia Integration Language. W3C Recommendation. http://www.w3.org/TR/2001/REC-smil20-20010807
6. Beckett D, Miller E, Brickley D (2002) Expressing simple Dublin Core in RDF/XML. Dublin Core Metadata Initiative, 21 July 2002
7. Berners-Lee T, Hendler J, Lassila O (2001) The semantic web. Sci Am J. http://www.scientificamerican.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
8. Bertino E, Ferrari E, Perego A (2002) Max: An access control system for digital libraries. In: Proceedings of the 26th international conference on computer software and applications, August 2002. IEEE Press, New York
9. Bertino E, Elmagarmid AK, Hacid M-S (2001) Quality of service in multimedia digital libraries. ACM SIGMOD Rec 30(1):35–40
10. Bertino E, Hammad M, Aref W, Elmagarmid A (2002) An access control model for video database systems. In: Conferece on information and knowledge management
11. Brickley D, Guha RV (2003) RDF Vocabulary Description Language 1.0:RDF Schema. W3C Working Draft, 23 January 2003. http://www.w3.org/TR/2003/WD-rdf-schema-20030123
12. Connoly D, Harmelen F, Horrocks I (2001) DAML+OIL reference description. W3C Note. http://www.w3.org/TR/daml+oil-reference
13. Damiani E, De Capitani di Vimercati S (2003) Securing xml based multimedia content. In: 18th IFIP international conference on information security
14. Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P (2000) Securing XML documents. Lecture notes in computer science, vol 1777. Springer, Berlin Heidelberg New York, pp 121–122
15. Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P (2002) A fine grained access control system for xml documents. ACM Trans Inf Syst Secur 5(2):169–202
16. Denker G (2002) Towards security in daml. Internal Report, SRI International, Menlo Park, CA
17. Fikes R, Hayes P, Horrocks I (2002) Designing a query language for the semantic web. The Knwoledge Systems Laboratory, Stanford University
18. Fikes R, Hayes P, Horrocks I (2003) DAML Query Language(DQL), April. http://www.daml.org/2003/04/dql/
19. Hayes P (2003) RDF semantics. W3C Working Draft, 23 January 2003 http://www.w3.org/TR/2003/WD-rdf-mt-20030123

20. Horrocks I (2002) Daml+oil: a reason-able web
21. Klyne G, Carroll J (2003) Resource Description Framework (RDF) concepts and abstract syntax. W3C Working Draft, 23 January 2003. http://www.w3.org/TR/2003/WD-rdf-concepts-20030123
22. Kodali N, Farkas C, Wijesekera D (2003) Enforcing integrity in multimedia surveillance. In: IFIP 11.5 working conference on integrity and internal control in information systems
23. Kodali N, Farkas C, Wijesekera D (2004) Metadata for multimedia access control. J Comput Syst Sci Eng 19(2):95–105
24. Kodali N, Farkas C, Wijesekera D (2003) Multimedia access contol using rdf metadata. In: Workshop on metadata for security (WMS'03).
25. Kodali N, Wijesekera D (2002) Regulating access to smil formatted pay-per-view movies. In: 2002 ACM workshop on XML security
26. Kopena J (2003) Daml jesskb. http://plan.mcs.drexel.edu/DAMLJessKB/
27. Lagoze C (1995) A secure repository design for digital libraries. D-Lib Mag. http://www.dlib.org/dlib/december95/12lagoze.html
28. Manola F, Miller E (2003) RDF Primer. W3C Working Draft, 23 January 2003. http://www.w3.org/TR/2003/WD-rdf-primer-20030123
29. Martin D (2003) DAML based Web-Service Ontology, May 2003
30. McCray AT, Gallagher ME (2001) Principles for digital library development. Commun ACM 44(5):48–54
31. Michel T (2001) The SMIL 2.0 MetaInformation Module. W3C Recommendation. http://www.w3.org/TR/2003/WD-rdf-mt-20030123
32. Patel-Schneider P, Siméon J (2002) The yin/yang web: Xml syntax and rdf semantics. In: Proceedings of the 11th international conference on the World Wide Web. ACM Press, New York, pp 443–453
33. Sandhu R, Ferraiolo D, Kuhn R (2000) The NISI model for role-based access control: towards a unified standard. In: ACM RBAC 2000, pp 47–64
34. Sandhu R, Samarati P (1996) Access control: principles and practices. IEEE Commun 29(2):38–47
35. Sheshagiri M, Kunjithapatham A (2003) A fipa compliant query mechanism using daml query language, June 2003., available online at http://www.cs.umbc.edu/∼finin//papers/dqlFIPA.html
36. Stoica A, Farkas C (2002) Secure xml views. In: Proc IFIP 11.3 working conference on database security

Appendices

A Access modes (roles, privileges, and classification)

```
<daml:Class rdf:ID="Access Mode">
 <rdfs:comment>
   Access mode, e.g., read, write, execute,
   defines the mode of access being
   permitted or denied.
</rdfs:comment>

<daml:oneOf rdf:parseType="daml:collection">
    <Access Mode rdf:ID="Read permitted"/>
    <Access Mode rdf:ID="Write permitted"/>
    <Access Mode rdf:ID="Execute permitted"/>
    <Access Mode rdf:ID="Read denied"/>
    <Access Mode rdf:ID="Write denied"/>
    <Access Mode rdf:ID="Execute denied"/>
  </daml:oneOf>
</daml:Class>

<daml:Class rdf:ID="Role">
 <rdfs:comment>
   Role defines the "role" a subject/user
   plays within the organization.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Subrole">
 <rdfs:subClassOf rdf:resource="#Role"/>
 <rdfs:subClassOf rdf:resource="#Sec Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Privilege">
<rdfs:comment>
  Privilege defines access rights of
  objcets. It corresponds to (object,
  access mode) pairs.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>


<daml:Class rdf:ID="Security Label">
 <rdfs:comment>
 Defines the security classification of
 data of security clearance of subject.
 Has two components: hierarchical
 component and subset components. Forms
 lattice structure and ordered by the
 "dominance" relation represented
 as "can-read" and "can-write" properties.
</rdfs:comment>

<rdfs:subClassOf rdf:resource="#Sec Struct"/>
<rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="Hierarchical Component">
<rdfs:comment>
Defines the hierarchical component of
a security label.
</rdfs:comment>

<daml:oneOf rdf:parseType="daml:collection">
 <Hierarchical Component rdf:ID="Top Secret"/>
 <Hierarchical Component rdf:ID="Secret"/>
 <Hierarchical Component rdf:ID=
                              "Unclassified"/>
</daml:oneOf>
```

```
</daml:Class>

<daml:Class rdf:ID="Subset Component">
<rdfs:comment>
Defines the subset (lattice) component of
a security label.
</rdfs:comment>

<daml:oneOf rdf:parseType="daml:collection">
    <Subset Component rdf:ID="{A,B}"/>
    <Subset Component rdf:ID="{A}"/>
    <Subset Component rdf:ID="{B}"/>
    <Subset Component rdf:ID="{ }"/>
  </daml:oneOf>
</daml:Class>
```

B QoS metastructure

```
 <daml:Class rdf:ID="Digital Library">
   <rdfs:label  >MultDL </rdfs:label >
   <rdfs:comment>
   This class of digital libraries is
   representative of metastructure.
   </rdfs:comment>
 </daml:Class>

 <daml:Class rdf:ID="QoS Framework">
  <rdfs:subClassOf rdf:resource="#MultDL"/>
  <daml:disjointWith rdf:resource=
                                "#Sec Struct"/>
 </daml:Class>

 <daml:Class rdf:ID="SystemQoS" >
  <rdfs:subClassOf rdf:resource="#QoS Struct"/>
  <rdfs:subClassOf rdf:resource="#MultDL"/>
 </daml:Class>

 <daml:Class rdf:ID="DLQoS">
  <daml:disjointWith rdf:resource=
                                 "#SystemQoS"/>
  <rdfs:subClassOf rdf:resource="#QoS Struct"/>
  <rdfs:subClassOf rdf:resource="#MultDL"/>
 </daml:Class >

 <daml:Class rdf:ID="rate" >
  <rdfs:subClassOf rdf:resource="SystemQoS"/>
  <rdfs:subClassOf rdf:resource="#QoS Struct"/>
  <rdfs:subClassOf rdf:resource="#MultDL"/>
 </daml:Class>

 <daml:Class rdf:ID="delay">
  <rdfs:subClassOf rdf:resource="SystemQoS"/ >
  <rdfs:subClassOf rdf:resource="#QoS Struct"/>
  <rdfs:subClassOf rdf:resource="#MultDL"/>
 </daml:Class>

<daml:Class rdf:ID= "rateOfDisplay">
 <rdfs:subClassOf rdf:resource="#rate"/ >
 <rdfs:domain rdf:resource="#SystemQoS"/>
</daml:Class>

<daml:Class rdf:ID= "requiredRateValue">
 <rdfs:subClassOf rdf:resource="#delay"/>
 <rdfs:domain rdf:resource="#SystemQoS"/>
</daml:Class>

<daml:Class rdf:ID= "currentDelay">
 <rdfs:subClassOf rdf:resource="#delay"/>
 <rdfs:domain rdf:resource="#SystemQoS"/>
</daml:Class>

<daml:Class rdf:ID= "toleranceValue">
 <rdfs:subClassOf rdf:resource="#delay"/>
 <rdfs:domain rdf:resource="#SystemQoS"/>
</daml:Class>

<daml:Class rdf:ID="accessibility" >
 <rdfs:subClassOf rdf:resource="#DLQoS"/>
 <rdfs:subClassOf rdf:resource="#QoS Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="availability" >
 <rdfs:subClassOf rdf:resource="#DLQoS"/>
 <rdfs:subClassOf rdf:resource="#QoS Struct"/ >
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>

<daml:Class rdf:ID="relevance">
 <rdfs:subClassOf rdf:resource="#DLQoS"/>
 <rdfs:subClassOf rdf:resource="#QoS Struct"/>
 <rdfs:subClassOf rdf:resource="#MultDL"/>
</daml:Class>
```

C DAML query ontology

```
<rdf:RDF xmlns:rdf="www.w3.org/rdf-syntax-ns#"
xmlns:rdfs="www.w3.org/2000/01/rdf-schema#"
xmlns:daml="www.daml.org/2001/03/daml+oil#"
xmlns:xsd="www.w3.org/2000/10/XMLSchema#"

<daml:Ontology rdf:about="">
   <daml:versionInfo>DLDOQLOntology.daml,v 1.0
   </daml:versionInfo>
   <rdfs:comment>
   Syntax-ontology for the DAML
   Query Language for MDL
   </rdfs:comment>
   <daml:imports rdf:resource =
   "http://www.daml.org/daml+oil" />
</daml:Ontology>
```

```
<daml:Class rdf:ID="Query">
 <rdfs:label>QueryClass</rdfs:label>
 <rdfs:comment>The class of queries.
 </rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="QueryId"> <rdfs:label>Query
Identifier</rdfs:label> <rdfs:comment>
   A unique ID assigned to queries.
</rdfs:comment> </daml:Class>

<daml:Class rdf:ID="QueryPattern">
  <rdfs:label>Query pattern as (s,o,a) Triples
</rdfs:label>
  <rdfs:comment>
  The set of triples that constitute
  the query.
  </rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="SOATriple">
   <rdfs:label>SOA Triples</rdfs:label>
   <rdfs:comment>
   Constitutes a subject and object
   and access.
   </rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="QueryPremise">
   <rdfs:label>QueryPremise</rdfs:label>
   <rdfs:comment>
   A KB  specified as a URI or included in
   the query to be included in the answerKB.
   </rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="AnswerKB">
   <rdfs:label>AnswerKB</rdfs:label>
   <rdfs:comment>
   The KB in which the query pattern needs
   to be quantified.
</rdfs:comment> </daml:Class>

<daml:Class rdf:ID="Indication">
<rdfs:label>Bind Indication</rdfs:label>
<rdfs:comment> The list of variables
in the query and the corresponding
bind-type.
</rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="Variable">
   <rdfs:label>Variable</rdfs:label>
   <rdfs:comment>
   The list of variables.
   </rdfs:comment>
</daml:Class>
```

```
<daml:Class rdf:ID="BindType">
   <daml:oneOf rdf:parseType="daml:collection">
   <daml:Literal rdf:value="may-bind" />
   <daml:Literal rdf:value="must-bind" />
</daml:oneOf>

</daml:Class>
<daml:ObjectProperty rdf:ID="QueryId">
<rdfs:domain rdf:resource="#Query" />
<rdfs:range rdf:resource=
    "http://rdf-schema#Literal"/>
<rdfs:comment />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="QueryPattern">
   <rdfs:domain rdf:resource="#Query" />
   <rdfs:range rdf:resource="#QueryPattern" />
   <rdfs:comment />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="SOATriple">
   <rdfs:domain rdf:resource="#QueryPattern" />
   <rdfs:range rdf:resource="#SOATriple" />
<rdfs:comment />

</daml:ObjectProperty>
   <daml:ObjectProperty rdf:ID="subject">
   <rdfs:domain rdf:resource="#SOATriple" />
   <rdfs:range rdf:resource=
        "http://rdf-schema#Literal"/>
<rdfs:comment />

</daml:ObjectProperty>
<daml:ObjectProperty rdf:ID="object">
   <rdfs:domain rdf:resource="#SOATriple" />
   <rdfs:range rdf:resource=
         "http://rdf-schema#Literal"/>
   <rdfs:comment />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="access">
<rdfs:domain rdf:resource="#SOATriple" />
 <rdfs:range rdf:resource=
     "http://rdf-schema#Literal"/>
<rdfs:comment /> </daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="AnswerKB">
   <rdfs:domain rdf:resource="#Query" />
   <rdfs:range rdf:resource="#SOATriple" />
   <rdfs:comment />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="QueryPremise">
   <rdfs:domain rdf:resource="#Query" />
   <rdfs:range rdf:resource="#SOATriple" />
   <rdfs:comment />
</daml:ObjectProperty>
```

```
<daml:ObjectProperty rdf:ID="Indication">
   <rdfs:domain rdf:resource="#Query" />
   <rdfs:range rdf:resource="#Indication" />
   <rdfs:comment />
</daml:ObjectProperty>


<daml:ObjectProperty rdf:ID="Variable">
   <rdfs:domain rdf:resource="#Indication" />
   <rdfs:range rdf:resource="#Literal" />
   <rdfs:comment />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="Value">
   <rdfs:domain rdf:resource="#Indication" />
   <rdfs:range rdf:resource="#BindType" />
<rdfs:comment /> </daml:ObjectProperty>

<daml:Class rdf:ID="SOATriple">
 <rdfs:subClassof>
<daml:intersectionOf
           rdf:parseType="daml:collection">
<daml:Restriction>
<daml:onProperty rdf:resource="#subject" />
<daml:cardinality>1</daml:cardinality>
</daml:Restriction>

    <daml:Restriction>
    <daml:onProperty rdf:resource="#object" />
    <daml:cardinality>1</daml:cardinality>
    </daml:Restriction>

    <daml:Restriction>
    <daml:onProperty rdf:resource="#access" />
    <daml:cardinality>1</daml:cardinality>
    </daml:Restriction>
</daml:intersectionOf>
</rdfs:subClassof>
</daml:Class>

<daml:Class rdf:ID="Variable">
<rdfs:subClassof>
    <daml:Restriction daml:cardinality="1">
    <daml:onProperty rdf:resource="#Value" />
    </daml:Restriction>
</rdfs:subClassof>
</daml:Class>
</rdf:RDF>
```