FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Ontologies for Context-Aware Applications

Ontologias em aplicações multimédia sensíveis ao contexto de utilização

Vitor Hugo dos Santos Barbosa

Dissertation developed within the framework of the Mestrado Integrado em Engenharia Electrotécnica e de Computadores under the supervision of Professor Maria Teresa Andrade, from Departamento de Engenharia Electrotécnica e de Computadores, Faculdade de Engenharia da Universidade do Porto

____
(President of the Jury)

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Electrotécnica e de Computadores
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

Porto, March 2008

Abstract

The trend towards a world where users want to receive the needed information at anytime and everywhere is now a reality. These challenges are driving the need for services and service architectures that are *aware* of the *context* of the different actors involved in a service transaction, to build content adaptation applications that maximize the user satisfaction.

This document describes a context modelling approach using ontologies, enabling the description of multimedia applications that adapts the digital content according to the characteristics and restrictions of context information, i.e., the design of context-aware content adaptation applications. To achieve this goal, it is necessary to develop ontologies capable of describing application scenarios or a domain as a formal fundament. These ontologies may be used by a reasoner which infers conclusions about the context in order to make decisions on content adaptation.

The implementation of this approach requires the identification and thorough characterisation of adequate application scenarios or real-world situations where adaptation of the content would improve the user experience. Within the framework of a research project, the VISNET-II Network of Excellence, audiovisual technologies are being developed for the implementation of advanced networked multimedia applications in different type of application domains. One of such domains is the Virtual Collaboration domain, which encompasses a number of different specific usage scenarios. In this work we have selected the virtual classroom application scenario and have accordingly performed a detailed analysis and consequently have developed a specific ontology.

This specification is based on the W3C (World Wide Web Consortium) standards, like OWL (Web Ontology Language) and MPEG-21 DIA (Digital Item Adaptation).

Keywords: Pervasive computing, Semantic Web, Context-awareness, Content adaptation, Web Ontology Language (OWL), Reasoning, MPEG-21 DIA.

Resumo

A tendência para um mundo onde os utilizadores pretendem receber toda a informação a qualquer altura e em qualquer lugar é já uma realidade. Estes desafios levam a uma busca por serviços e arquitecturas de serviços que são sensíveis ao contexto de utilização perante os diferentes actores envolvidos no acesso e uso de um serviço, por forma a desenvolver aplicações capazes de adaptar os conteúdos de forma transparente, maximizando a satisfação do utilizador.

Este documento descreve a abordagem adoptada para a modelização do contexto de utilização através da introdução de ontologias, possibilitando a descrição de aplicações multimédia que adaptam os conteúdos digitais de acordo com as características e restrições apontadas pela informação de contexto, isto é, permitindo o desenvolvimento de aplicações sensíveis ao contexto. Para a implementação desta abordagem torna-se necessário o desenvolvimento de ontologias capazes de descrever alguns cenários ou domínios de aplicação de uma maneira formal e normalizada. Estas ontologias podem ser usadas por um mecanismo de raciocínio que infere conclusões sobre o contexto em questão de forma a tomar decisões na adaptação de conteúdos.

Os objectivos deste trabalho estão enquadrados no plano de trabalhos do projecto VISNET-II, uma rede de excelência co-financiada pela Comissão Europeia no âmbito do 6º Programa Quadro de investigação. Neste projecto estão a ser desenvolvidas tecnologias audiovisuais para o desenvolvimento de aplicações avançadas de multimédia em rede.

Uma das áreas de aplicação dessas tecnologias é a da Colaboração Virtual. Foi assim decidido seleccionar um cenário dentro desta área de aplicação para o suporte de adaptação de conteúdos adoptando a estratégia proposta por esta dissertação. Nesse sentido, é feita uma análise detalhada ao cenário de "classe de aulas virtual" e através de uma ferramenta de criação de ontologias desenvolve-se a especificação de uma ontologia válida para o cenário de utilização estudado.

Esta especificação é baseada nas normas do W3C (World Wide Web Consortium), nomeadamente OWL (Web Ontology Language) e MPEG-21 DIA (Digital Item Adaptation).

Palavras-chave: Web Semântica, Aplicações sensíveis ao Contexto, Adaptação de conteúdos, Web Ontology Language (OWL), Motores de Raciocínio, MPEG-21 DIA.

To my mother and father,
who offered me unconditional love, support and always a sit on the table throughout this entire venture.

To my dear brother,
for the inexhaustible source of encouragement, inspiration and resilience.

To my girlfriend,
who has lifted me up and brought peace in the most hazardous times.

Acknowledgements

I would like to acknowledge and thank my supervisor Maria Teresa Andrade, for proof reading my masters thesis and making helpful comments and suggestions on how to improve it.

This work was partially developed within VISNET II, a European Network of Excellence (http://www.visnet-noe.org), funded under the European Commission IST FP6 programme.

I would like to leave a word of appreciation to Asdrúbal Costa, for the material and Java expertise that he has put to my disposition, and all the VISNET II partners for the ideas and knowledge exchanged.

I would also like to thank my closest friends (you know who you are) who have always stood by my side and believed in my capabilities even when I didn't recognise their existence.

I cannot end without thanking my parents, who have raised me, supported me, and loved me without knowing what they were getting themselves into.

Finally, and the most important of all, I would like to thank my brother Hélder Barbosa and my lovely girlfriend and best friend Catarina Moutinho, for being supportive at all cost, for their expertise ideas and given inspiration.

List of Acronyms

AA Adaptation Authoriser

ADE Adaptation Decision Engine

AE Adaptation Engine

AES Adaptation Engine Stack

AI Artificial Intelligence

API Application Programming Interface

AQOS Adaptation Quality of Service

CAO Context-Aware Ontology

CC/PP Composite Capability/Preference Profiles

COE CmapTools Ontology Editor

CxP Context Provider

DAML DARPA Markup Language

DAML+OIL DARPA Markup Language + Ontology Interchange Language

DAWG Data Access Working Group

DCMI Dublin Core Meta Data Initiative

DI Digital Item

DIA Digital Item Adaptation

DID Digital Item Declaration

DIDL Digital Item Declaration Language

DIG DL Implementation Group

DL Description Logics

DOM Document Object Model

DRM Digital Rights Management

DTD Document Type Definition

DS Description Schemes

EMF Eclipse Modeling Framework

EODM EMF Ontology Definition Metamodel

FOAF Friend Of A Friend

GRDDL Gleaning Resource Descriptions from Dialects of Languages

HTML HyperText Markup Language

ICOM Intelligent Conceptual Modelling

IDE Integrated Development Environment

IODT Integrated Ontology Development Toolkit

LIP Learner Information Package

MDS Media Descriptors Scheme

MPEG Motion Pictures Expert Group

NoE Network of Excellence

N3 Notation 3

ODM Ontology Definition Metamodel

OIL Ontology Interchange Language

ORIENT Ontology engineeRIng ENvironmenT

OWL Web Ontology Language

PAPI Public and Private Information

PC Personal Computer

PDA Personal Digital Assistance

QoS Quality of Service

RDF Resource Description Framework

RDFS Resource Description Framework Schema

ROI Region of Interest

SKOS Simple Knowledge Organisation Systems

SOAP Simple Object Access Protocol

SPARQL SPARQL Protocol And Query Language

SW Semantic Web

SWEDE Semantic Web Development Environment

SWT Standard Widget Toolkit

UED Usage Environment Descriptors

UCD Usage Constraints Descriptors

UML Unified Modeling Language

UNA Unique Name Assumption

URI Uniform Resource Identifier

URL Uniform Resource Locator

VCS Virtual Collaboration System

WP Work Package

W3C World Wide Web Consortium

XML eXtensible Markup Language

Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

1.1 Overview

Nowadays, users want to be able to access their computing resources and all kind of content using different types of devices, from wireless portable devices to stationary devices and computer connected to local area networks, i.e., users require ubiquitous access to information, communication, and computation.

Nevertheless, some obstacles still need to be overcome. One of such obstacles comes from the disparity of the capabilities of user devices and also from heterogeneity of content formats and required resources. Accordingly, researchers are faced with the challenge of dealing with different and varying consumption environment characteristics and conditions whilst providing seamless access to the same content.

The challenges arise because each combination of location, terminal scale, connectivity, user preferences and other local usage environment factors may require a different source and channel coding format for the content. It is clearly impossible to pre-generate and store all these formats for every item of content and so real-time adaptation of a very limited set of formats (probably only one) is required.

Ontologies are emerging as a key enabling technology for the Semantic Web, which concept was introduced by the World Wide Web creator, Tim Berners-Lee, envisioning a Web that provides a qualitatively new level of service. Thus, ontologies are seen as a key requirement to building pervasive context-aware systems, in which independently develop sensors, devices and agents are expected to share contextual information.

Within the framework of a research project, the VISNET-II Network of Excellence (NoE), audiovisual technologies are being developed for the implementation of advanced networked multimedia applications in different type of application domains. One of such domains is the Virtual Collaboration domain, which encompasses a number of different specific usage scenarios. In this work we have selected the virtual classroom application scenario and have accordingly performed a thoroughly analysis and consequently have developed a specific ontology.

The implementation strategy for the conceptualisation of the selected application domain is driven through the extraction of the fundamental contextual information associated to a user in a virtual classroom session, from the MPEG-21 DIA UED and MPEG-7 MDS standards descriptors, and the development of a data ontology providing the rules that represent as accurately as possible the real-world situations in the virtual classroom scenario application. This way, we are providing the means to achieve the interoperability required for a context-aware content adaptation system, whose main goal is to enhance the quality experienced by the user.

1.2 Objectives

The main goal of this thesis is to develop an ontology using the OWL semantic language, achieving an efficient conceptualisation of the virtual classroom application domain and providing the means for a rule-based and user-centric adaptation decision taking process. Having identified the key requirements for the implementation of a context-aware and interoperable system, the objectives that span our attention are namely:

- State-of-the-art review and the study of novel Semantic Web and context-awareness standards;
- Semantic Web tools survey to infer the most adequate open-source or freeware software to enable the ontology development and necessary reasoning services;
- Virtual classroom scenario application specification through a thoroughly analysis and Use Case(s) definition. The defined use cases should reflect the possibilities for the fundamental contextual information in a virtual classroom session;
- Main concepts definition based on the MPEG-21 DIA UED and MPEG-7 MDS standard descriptors, which includes the relative contextual information;
- Ontology overview design, identifying the main concepts in the virtual classroom scenario application, their associated datatype properties and relations between those classes;
- OWL ontology development for the conceptualisation of the virtual classroom domain;
- Using the reasoner services in order to check the consistency of the developed ontology, infer the concept satisfiability and classification;

Having concluded the aforementioned goals, we have come up with other research issues which have been developed in a second phase of this project. Therefore, we have envisioned an application developed in Java implementing the automatic extraction of data information and instantiation in the constructed Context-Aware Ontology. The goals for this second phase are namely:

- Generate contextual information associated to a selected Use Case and insert it into MPEG-21 DIA UED and MPEG-7 MDS standard descriptors;
- Develop a context-aware application in Java capable of automatically extract the contextual information from the previous generated files and insert this kind of data information into a separated data ontology;
- Using the reasoner services in order to check the consistency of the developed ontology, infer the concept satisfiability and classification;
- Envisioning possible future research directions for the developed application.

1.3 Thesis Organisation

The thesis is organised as follows. Section 2 highlights the core background technology, fundamental for the understanding of the thesis, together with an overview of the novel Web Ontology Language, OWL. Additionally, this section provides some insights into semantic Web languages, MPEG-21 and MPEG-7 standards and the ontology concept is given.

Section 3 describes the process of the ontology development focusing on the steps to create an OWL ontology.

The remaining chapters document the execution of the proposed objectives, present the associated results and draw the concluding remarks.

The analysis conducted for thoroughly characterising the virtual classroom application is presented in section 4. Representative use cases are included in this analysis, highlighting the need for different types of content adaptation operations thus providing indications of possible decision taking measures.

Section 5 provides a review of some OWL editor tools and reasoners applications, which can be seen as suitable candidates for the design ontology proposed in this work.

Section 6 details the implementation strategy for the conceptualisation of the selected application domain and presents the associated experimental results. It first presents the concepts of the ontology and it then describes the implementation of the OWL ontology. Finally, it also provides an overview of the application being developed for the integration of the designed ontology with real context description values obtained from sensors and carried as MPEG21 UED (Universal Environment Description) and MPEG7 description files.

Finally, in Section 7 we draw conclusions on the obtained results together with a brief discussion on some future research directions and trends in ontology development.

Chapter 2

2. Core Technology Overview

This chapter provides a summary of current developments and standardization efforts in context-aware content adaptation systems. It presents established concepts in context-aware computing, semantic web languages and MPEG-21 standard. Additionally, a brief review on MPEG-7 Multimedia Description Schemes is reported.

2.1 Context-aware systems

As proposed by Dey [1], context is "any information that can be used to characterize the situation of entities." In a more detailed way, *context information* is any information which can be used to characterize the state of an entity concerning a specific aspect. An *entity* is a person, a place or in general an object. An *aspect* is a classification whose subsets are a superset of all reached states, grouped in one or more dimensions called *scales*. A system is *context aware*, if it uses any kind of context information before or during service provisioning.

Context-awareness can be defined as the ability of systems, devices or software, to be aware of the characteristics and constraints of the user's preferences and environment characteristics, i.e. contextual information, and accordingly perform a number of actions/operations automatically to adapt to changes of the sensed environment without explicit user intervention and thus aim at increasing usability and effectiveness.

Contextual information can be any kind of information that characterizes or provides additional information regarding any feature or condition of the complete delivery and consumption environment. This complexity and diversity of information can be grouped into four main context classes: *Resources*, *User*, *Physical* and *Time*. Some examples of contextual information related to each class are provided below:

- **Resources Context** – description of the terminal in terms of its hardware platform, including any property such as processor, screen size or network interface; description of the terminal in terms of its software platform, such as operating system, software multimedia codecs installed or any other software application; description of the network such as maximum capacity, instantaneously available bandwidth or losses; description of multimedia servers, for example in terms of maximum number of simultaneous users or maximum throughput; description of transcoding engines in terms of their hardware and software platforms such as network interface or input/output formats allowed or bit rate range supported.

- **User Context** – description of the user general characteristics such as gender, nationality or age; description of preferences of the user related with the consumption of content such as type of media or language preferred; description of the preferences of the user in terms of his/her interests, i.e., related with the high-level semantics of the content such as local news versus international news or action movies versus comedy; description of the user's emotions such as anxious versus relaxed or happy versus sad; description of the user status such as online versus offline or stationary versus walking; description of the history/log of actions performed by the user.
- **Physical Context** – description of the natural environment surrounding the user such as lighting and sound conditions, temperature or location.
- **Time Context** – indication of the time at which variations in the context have occurred or scheduling of future events.

There are other aspects about contextual information that should be considered, like the accuracy or level of confidence of the contextual information, the period of validity and the dependencies on other types of contextual information, but they are not being characterized in this document.

2.1.1 Context-based content adaptation

Content adaptation has in fact already gained a considerable importance in today's multimedia communications, and will certainly become an essential functionality of any service, application or system in the near future. The continuing advances in technology will only emphasize the great heterogeneity that exists today in devices, systems, services and applications. Likewise, this will also bring out the desires of consumers for more choices, better quality and more personalization options. But, to empower those systems to perform meaningful content adaptation operations that meets users' expectations and satisfies their usage environment constraints, it is imperative that they use contextual information, and thus take decisions based on that information.

Context-awareness in content adaptation can be defined as the ability of a system to adapt the content to the characteristics and constraints of the consumption environment and user's preferences. It aims to increase the system usability and enhance the quality of the user experience.

For content adaptation, context-aware applications must initially acquire the contextual information and then process it and reason about it to formulate concepts and take decisions when and how to react. Different types of contextual information and their characteristics can be seen as low-level or basic contextual information in the sense that they can be directly generated by some software or hardware appliance. Based on this basic contextual information, applications may formulate higher-level concepts.

Dey has described three main steps that an application has to do in order to be context-aware. First, it must capture the context as a set of low level data from different sensors. Second, an interpreter of the captured data must build high level contextual information, more meaningful to the application. Finally, it must carry the interpreted information to the application, which uses it together with other data to offer an adapted computation or service.

The next sections present an ontology-based formal context model to address essential issues including formal context representation, knowledge sharing and logic based context reasoning.

2.2 Semantic Web

The Semantic Web (SW) concept was introduced by Tim Berners-Lee, the World Wide Web “creator”. Tim Berners-Lee [2] envisioned a Semantic Web that provides automated information access based on machine-processable semantics of data and heuristics that use these metadata. The explicit representation of the semantics of data, the metadata, accompanied with domain theories, will enable a Web that provides a qualitatively new level of service.

Within the W3C, the Semantic Web (SW) [3] is an effort to develop new tools that are able to provide richer and explicit descriptions of Web resources. The essence of the W3C SW is a set of standards for exchanging machine-understandable information.

Among these standards, Resource Description Framework (RDF) provides data model specifications and XML-based serialization syntax and Web Ontology Language (OWL) enables the definition of domain ontologies and sharing of domain vocabularies.

The key idea of Semantic Web is to develop tools, technologies and standards to enable universal access to all the information available on the Web. Nowadays, document management systems have severe weaknesses. The information associated to HTML pages is only used in some contexts or it is subjective because it includes certain terms in different meanings, depending on the creators or their applications. For that reason, though the information is searchable and likely to be found, it is only accessible within the scope of the application for which it was specifically created. However, that same information would be useful in many other contexts. Existing Web resources are usually only human understandable. This problem is known as "semantic gap" and many ways are being explored to overcome this problem ("bridging the semantic gap").

Ontologies are seen as a key enabling technology for the Semantic Web. The use of ontologies to model context is the best choice if we want to guarantee a high degree of expressiveness and semantic richness. Ontologies can also offer means to avoid conflicts that may arise when identifying context situations [4]. A more detailed view of ontologies is available on the next chapter.

2.3 Ontologies concept

A classical definition of an ontology in Artificial Intelligence (AI) is "a formal specification of a conceptualization", that is, an abstract and simplified view of the world that we wish to represent, described in a language that is equipped with formal semantics.

In knowledge representation, ontology is a data model that represents a set of concepts within a domain and the relationships between those concepts, understandable by users and/or by software agents.

An ontology defines the terms used to describe and represent an area of knowledge which enables a formal description of specific situations in that domain [5]. It is used to reason about the objects within that domain. An ontology is formal, since its understanding should be non ambiguous, both from the syntactic and the semantic point of views.

With the use of ontology schemes we can achieve *knowledge sharing*, *logic inference* and *knowledge reuse* [6]. Through logic inference, context-aware computing can exploit existing logic reasoning mechanisms to deduce high-level, conceptual context from low-level context and to check or solve inconsistent context knowledge due to imperfect sensing. This context knowledge can be shared and reused. The use of a context ontology enables computational entities to have a common set of concepts about context while interacting with each other. By reusing well-defined Web ontologies of different domains it is possible to compose large-scale context ontology without starting from scratch.

Reasoning is an important step that can be used throughout the different development phases of an ontology. During the design phase, it can be used to assess the quality and consistency of the ontology, by testing whether concepts are non-contradictory and to derive implied relations. Reasoning can also support the integration of ontologies by, for example, asserting inter-ontology relationships, testing for consistency and computing the integrated concept hierarchy. Reasoning may also be used when the ontology is deployed, allowing determining the consistency of facts and values stated in the annotation with the ontology or inferring instance relationships.

2.3.1 Elements of an Ontology

Regardless of the language in which ontologies are expressed, they share many structural similarities. Most ontologies describe individuals (instances), classes (concepts), attributes, and relations.

Individuals are the basic components of an ontology. The individuals may include concrete objects such as people, animals, etc., as well abstract individuals such as numbers and words. One of the general purposes of an ontology is to provide a means of classifying individuals, even if those individuals are not explicitly part of the ontology, so an ontology need not include any individuals.

Classes are abstract groups, sets, or collections of objects. They may contain individuals, other classes, or a combination of both. A class can be a set of elements, called the extension of the class. An important use of classes is to impose restrictions on what can be stated.

Objects in the ontology can be described by assigning **attributes** to them. Each attribute has at least a name and a value, and is used to store information that is specific to the object it is attached to.

An important use of attributes is to describe the relationships (also known as **relations**) between objects in the ontology. Typically a relation is an attribute whose value is another object in the ontology. The most important type of relation is the *subsumption* relation (*is-superclass-of*, the converse of *is-a*, *is-subtype-of* or *is-subclass-of*). It defines which objects are members of classes of objects.

2.4 Semantic Web Languages

A Semantic Web language or an ontology language provides the mean to specify at an abstract or conceptual level what is necessarily true in the domain of interest. Moreover, an ontology language should be able to express constraints, which declare what should necessarily hold in any possible concrete instantiation of the domain.

The ability to define and establish relations between any two resources, allows a better and automatic interchange of data, which is one of the driving elements towards the fulfilment of the Semantic Web. RDF (Resource Description Framework), which is one of the fundamental building blocks of the Semantic Web, gives a formal definition for that interchange. On that basis, additional building blocks are built around this central notion. Some examples are:

- Tools to query information described through such relationships (eg, SPARQL);

- Tools to have a finer and more detailed classification and characterization of those relationships. This ensures interoperability and more complex automatic behaviours. (e.g., RDF Schemas, OWL, Simple Knowledge Organisation Systems (SKOS));
- For more complex cases, tools are available to define logical relationships among resources and the relationships (e.g., OWL, Rules);
- Tools to extract from, and to bind to traditional data sources to ensure their interchange with data from other sources. (e.g., Gleaning Resource Descriptions from Dialects of Languages (GRDDL), RDFa).

The purpose of ontology languages is to allow users to write explicit, formal conceptualizations of domain models requiring a well-defined syntax, a well-defined semantics, efficient reasoning support, sufficient expressive power and convenience of expression. These requirements point to the use of a syntax based on XML. Two of the most important semantic Web languages based on XML are RDF and OWL. These languages are described below.

2.4.1 XML

XML [7], which stands for eXtensible Markup Language, is a markup language for documents containing structured information. It is classified as an extensible language because it allows it's users to define their own tags. Its primary purpose is to facilitate the sharing of structured data across different information systems, particularly via the Internet. It is used to encode documents and serialize data in a machine-readable format, being at the same time relatively human-legible.

XML does not specify the semantics nor a tag set, but only syntactic rules. In fact XML is really a meta-language for describing markup languages. In other words, XML provides the possibility to define tags and the structural relationships between them. Since there is no predefined tag set, there cannot be either any preconceived semantics. All of the semantics of an XML document will either be defined by the applications that process them or through the use of stylesheets.

There are two levels of correctness of an XML document:

- Well-formed. A well-formed document conforms to all of the XML's defined syntax rules. For example, if an element has an opening tag with no closing tag and is not self-closing, it is not well-formed. A document that is not well-formed is not considered to be XML; a conforming parser is not allowed to process it.
- Valid. A valid document additionally conforms to some semantic rules. These rules are either user-defined, or included as an XML schema or a Document Type Definition (DTD). For example, if a document contains an undefined tag, then it is

not valid within the scope it is meant to be used; a validating parser is not allowed to process it.

XML documents are composed of markup and content. There are six kinds of markup that can occur in an XML document: elements, entity references, comments, processing instructions, marked sections, and document type declarations.

A simple example of a valid XML document is presented in Table 2.1.

Table 2.1 – XML Example

```
<catalog>
     <cd>
          <title>Picture book</title>
          <artist>Simply Red</artist>
          <year>1983</year>
     </cd>
<catalog>
```

2.4.2 RDF

The Resource Description Framework (RDF) is a framework for representing information in the Web, constituting a standard for encoding knowledge.

This mechanism for describing resources is a key element of the W3C's Semantic Web activity, enabling users to deal with the information with greater efficiency and certainty.

The RDF metadata model is based upon the idea of making statements about resources in the form of subject-predicate-object expressions, called triples in RDF terminology. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object.

Most of the abstract model of RDF can be summarised into four simple rules:

1. A fact is expressed as a Subject-Predicate-Object triple, also known as a statement.
2. Subjects, predicates, and objects are given as names for entities, also called resources or nodes. Entities represent something, a person, website, or something more abstract like states and relations.
3. Names are URIs, which are global in scope, always referring to the same entity in any RDF document in which they appear.
4. Objects can also be given as text values, called literal values, which may or may not be typed using XML Schema datatypes.

Entities are named by Uniform Resource Identifiers (URIs), which provides the globally unique, distributed naming system needed for distributed knowledge. URIs can have the same syntax or format as website addresses (URLs), so it is possible to find RDF files that contain URIs, such as http://www.w3.org/1999/02/22-rdf-syntax-ns#type.

A collection of RDF statements intrinsically represents a labelled, directed pseudo-graph. As such, an RDF-based data model is more naturally suited to certain kinds of knowledge representation than the relational model or other ontological models traditionally used in computing systems today. However, for practical reasons, RDF data is often stored in relational databases. In this case, the RDF representations are called triple stores. As RDFS and OWL demonstrate, additional ontology languages can be built upon RDF.

2.4.3 RDFS

RDFS [8] (RDF Schema), also known as RDF Vocabulary Description Language, is an extensible knowledge representation language, providing basic elements for the description of ontologies, otherwise called RDF vocabularies, intended to structure RDF resources.

RDF Schema, as a RDF's vocabulary description language, is a semantic extension of RDF and it provides mechanisms for describing groups of related resources and the relationships between these resources. These resources are used to determine characteristics of other resources, such as the domains and ranges of properties.

The most elementary building block of RDFS is a class, which defines a group of individuals that belong together because they share some properties. The members of a class are known as instances of the class. Classes are themselves resources and they are often identified by RDF URI References and may be described using RDF properties.

- rdfs:Class allows to declare a resource as a class for other resources.
- rdfs:subClassOf allows to declare hierarchies of classes.

In RDFS, instances are related to other instances through properties. A RDF property is a relation between subject resources and object resources and is constrained by its range and domain.

- rdfs:domain of an rdf:property declares the class of the subject in a triple using this property as predicate.
- rdfs:range of an rdf:property declares the class or datatype of the object in a triple using this property as predicate.

Properties are also resources, named by URIs, and therefore there is also a possibility to have sub-properties.

A summary of RDFS basic features is:

- Classes and their instances;
- Binary properties between objects;
- Organization of classes and properties in hierarchies;
- Types for properties: domain and range restrictions.

One benefit of the RDF property-centric approach is that it allows anyone to extend the description of existing resources, one of the architectural principles of the Web.

A simple example of a RDF Schema document [9] is presented in Table 2.2.

Table 2.2 – RDF Schema document example

```
<?xml version="1.0"?>
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <rdfs:Class rdf:ID="Person">
     <rdfs:comment>Person Class</rdfs:comment>
     <rdfs:subClassOf
          rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Resource"/>
   </rdfs:Class>
   <rdfs:Class rdf:ID="Teacher">
     <rdfs:comment>Teacher Class</rdfs:comment>
     <rdfs:subClassOf rdf:resource="#Person"/>
   </rdfs:Class>
   <rdfs:Class rdf:ID="Course">
     <rdfs:comment>Course Class</rdfs:comment>
     <rdfs:subClassOf  rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Resource"/>
   </rdfs:Class>
   <rdf:Property rdf:ID="teacher">
      <rdfs:comment>Teacher of a course</rdfs:comment>
      <rdfs:domain rdf:resource="#Course"/>
      <rdfs:range rdf:resource="#Teacher"/>
   </rdf:Property>
</rdf:RDF>
```

2.4.4 OWL

The Web Ontology Language OWL [10] is a description logic language for defining, publishing and sharing Web ontologies. An OWL ontology may include descriptions of classes, along with their related properties and instances.

OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics. OWL is based on earlier languages OIL (Ontology Interchange Language) [11] and DAML+OIL (DARPA Markup Language + Ontology Interchange Language) [12], and is a W3C recommendation.

OWL was designed to provide a common way to process the semantic content of Web information. It was developed to augment the facilities for expressing semantics (meaning) provided by XML, RDF, and RDFS. Consequently, it may be considered an evolution of these Web languages in terms of its ability to represent machine-interpretable semantic content on the Web.

OWL can declare classes, and organize these classes in a subsumption (“subclass”) hierarchy, in a similar way to RDF Schema. OWL classes can be specified as logical combinations (intersections, unions, or complements) of other classes, or declare disjointed classes, going beyond the capabilities of RDFS. OWL can also declare properties and provide domains and ranges for these properties, again as in RDFS. The domains of OWL properties are OWL classes, and ranges can be either OWL classes or externally-defined datatypes such as string or integer. OWL can state that a property is transitive, symmetric, functional, or is the inverse of another property, here again extending RDFS. However, the major extension over RDFS is the ability in OWL to provide restrictions on how properties behave, local to a class.

OWL currently has three sublanguages: OWL Lite, OWL DL (Description Logics), and OWL Full.

One of the most important distinguishing features of logic based ontology languages like OWL from other ontology formalisms is that it has formal semantics. Staying within OWL DL, allows us to build reasoners which can make automatic inferences over our knowledge base. As referred in Section 2.3, reasoners can also be used at development time to help users to build and manage their ontologies more easily.

A concrete example of an OWL ontology is presented in Table 2.3.

Table 2.3 – OWL Ontology example

```
<rdf:RDF
  xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd ="http://www.w3.org/2000/10/XMLSchema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:exd ="http://www.w3.org/TR/@@/owl-ex-dt#"
  xmlns:dex ="http://www.w3.org/TR/@@/owl-ex#"
  xmlns     ="http://www.w3.org/TR/@@/owl-ex#"
>

   <owl:Ontology rdf:about="">
     <owl:versionInfo>$Id: owl-ex.owl,v 1.6 2002/07/25 17:33:19 mdean
Exp $</owl:versionInfo>
     <rdfs:comment>
       An example ontology, with data types taken from XML Schema
     </rdfs:comment>
     <owl:imports rdf:resource="http://www.w3.org/2002/07/owl"/>
   </owl:Ontology>
   <owl:Class rdf:ID="Animal">
     <rdfs:label>Animal</rdfs:label>
     <rdfs:comment>
       This class of animals is illustrative of a number of ontological
idioms.
     </rdfs:comment>
   </owl:Class>
   <owl:Class rdf:ID="Male">
     <rdfs:subClassOf rdf:resource="#Animal"/>
   </owl:Class>
   <owl:Class rdf:ID="Female">
     <rdfs:subClassOf rdf:resource="#Animal"/>
     <owl:disjointWith rdf:resource="#Male"/>
```

```
  </owl:Class>

  <owl:DatatypeProperty rdf:ID="age">
    <rdfs:comment>
      age is a DatatypeProperty whose range is xsd:decimal.
      age is also a FunctionalProperty (can only have one age)
    </rdfs:comment>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2000/10/XMLSchema#nonNegativeInteger"/>
  </owl:DatatypeProperty>
  …
  <owl:Class rdf:ID="HumanBeing">
    <owl:sameClassAs rdf:resource="#Person"/>
  </owl:Class>
</rdf:RDF>
```

More information about ontologies and OWL should be further analysed in Chapter 3.

2.4.5 SPARQL

The SPARQL Protocol And RDF Query Language (SPARQL) [13] is a Semantic Web protocol and query language, recently (January, 2008) recommended by W3C and standardised by the RDF Data Access Working Group (DAWG).

Most uses of the SPARQL acronym refer to the RDF query language. In this usage, SPARQL is a syntactically-SQL-like language for querying RDF graphs via pattern matching. The language's features include basic conjunctive patterns, value filters, optional patterns, and pattern disjunction. Table 2.4 illustrates a simple SPARQL query.

Table 2.4 – Simple SPARQL query

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name
WHERE  { ?x foaf:name ?name }
```

The SPARQL protocol is a method for remote invocation of SPARQL queries. It specifies a simple interface that can be supported via HTTP or SOAP that a client can use to issue SPARQL queries against some endpoint.

2.5 MPEG-7 Overview

MPEG-7 [14] is a standard developed by MPEG (Moving Picture Experts Group) for describing multimedia content data, offering a comprehensive set of audiovisual Description Tools. These descriptions are key elements of applications enabling effective access to multimedia content, by implementing efficient search, filtering and browsing operations.

MPEG-7 description data may be physically located with the associated audiovisual content, (in the same date stream or storage system), but the description could also exist somewhere else. When the content and its descriptions are not co-located, mechanisms that link the multimedia content and associated MPEG-7 descriptions are needed.

There is a large variety of MPEG-7 description possibilities, thus they may include:

- Information describing the creation and production processes of the content;
- Information related to the usage of the content;
- Structural information on spatial, temporal or spatial-temporal components of the content;
- Information about low level features in the content;
- Information about how to browse the content in an efficient way;
- Information about the interaction of the user with the content (user preferences, usage history);
- Information of the technical features of the content (encoding format, visual spatial dimensions, bit rate, number of audio channels, etc.).

The latter is the most relevant for the development of this project, as we are interested in describing the multimedia audiovisual encoding information from the point of view of required resources, to decide if and how to modify those characteristics.

Within the scope of content adaptation applications, MPEG-7 plays an important role in describing information about the content, notably related with the format of the content, i.e., with the technical parameters with which the content is encoded and the media characteristics.

The following subsection briefly brings into focus the particular part from MPEG-7 related to the Description Tools dealing with generic features and multimedia descriptions.

2.5.1 MPEG-7 Multimedia Description Schemes

MPEG-7 Multimedia Description Schemes (MDS) [15] is composed by the set of Description Tools (Descriptors and Description Schemes) dealing with audio and visual descriptions as well as multimedia entities.

MPEG-7 MDS provides support for the description of user preferences and usage history as well as for adaptation tools. Additionally, the MPEG-21 standard makes use of these kinds of MPEG-7 description tools within the same scope.

MDS Description Tools can be grouped in terms of the functionality they provide. In this manner, there are tools that characterise the Basic Elements, the Schema Tools, the Content Description Tools, the Content Organization Description Tools, the Navigation and Access Description Tools and finally the User Description Tools.

The Content Description Tools are the most relevant tools for the prosecution and development of this thesis purposes since these tools provide information about the features of the multimedia content. The Media Description Tools allow the description of the storage media, the coding format, the quality and the transcoding hints for adapting the content to different networks and terminals. The Creation Description Tools allow the description of the creation process (title, agents, materials, places, and dates), classification (genre, subject, parental rating, and languages) and related materials. The Usage Description Tools allow the description of the conditions for use (rights, availability) and the history of use.

The most important subset of the Content Description Tools is the one related with Media Description Tools as we need to describe the multimedia format, audiovisual encoding format and associated information.

2.6 MPEG-21 Overview

MPEG-21 [16] aims at defining a normative open framework for multimedia delivery and consumption for use by all the players in the delivery and consumption chain.

MPEG-21 is based on two essential concepts: the definition of a fundamental unit of distribution and transaction (the Digital Item) and the concept of Users interacting with Digital Items.

A Digital Item (DI) is a structured digital object with a standard representation, identification and metadata. This entity is also the fundamental unit of distribution and transaction within the MPEG-21 framework. More concretely, a DI can be seen as a package of multimedia resources, together with associated descriptions and including the respective Digital Item Declaration (DID).

Within any system involving DIs, there is the need for a very concrete description of what constitutes such an “item”, as there are many kinds of content and many possible ways of describing it. DID goal is to establish a uniform and flexible abstraction and interoperable schema for defining DIs. Therefore, Digital Items are declared using the Digital Item Declaration Language (DIDL) defined in *MPEG-21 Part 2 – Digital Item Declaration*, where declaring a Digital Item involves specifying its resources, metadata and their interrelationships.

The goal of MPEG-21 can thus be rephrased to: defining the technology needed to support Users to exchange, access, consume, trade and otherwise manipulate DIs in an efficient, transparent and interoperable way.

MPEG-21 identifies and defines the mechanisms and elements needed to support the multimedia delivery chain as described above as well as the relationships between and the operations supported by them. Within the parts of MPEG-21, these elements are elaborated by defining the syntax and semantics of their characteristics, such as interfaces to the elements.

Due to the diversity of topics addressed by MPEG-21, the standard is divided in eighteen parts. The part which is more relevant for the implementation of context-aware content adaptation is *MPEG-21 Part 7 – Digital Item Adaptation*.

2.6.1 MPEG-21 DIA

MPEG-21 Digital Item Adaptation (DIA) [17][18] specifies the syntax and semantics of tools that may be used to assist the adaptation of Digital Items, i.e., the Digital Item Declaration, metadata and resources referenced by the declaration.

DIA provides a set of tools, allowing to describe characteristics and capabilities of networks, terminals and environments as well as of preferences of users. In addition, the set of tools also provides the definition of the operations that can be performed upon the content and the result that can be expected.

The Digital Item Adaptation tools are grouped into eight major categories as illustrated in Figure 2.1 [17]. The categories are clustered according to their functionality and use for Digital Item Adaptation.

Among others, specific adaptation tools and descriptions of the MPEG-21 DIA standard, such as Usage Environment Descriptor (UED), Adaptation Quality of Service (AQoS) and Universal Constraints Descriptor (UCD) define a set of descriptors and methodologies to describe the context of usage, the operations that can be performed upon the content and the result that can be expected.

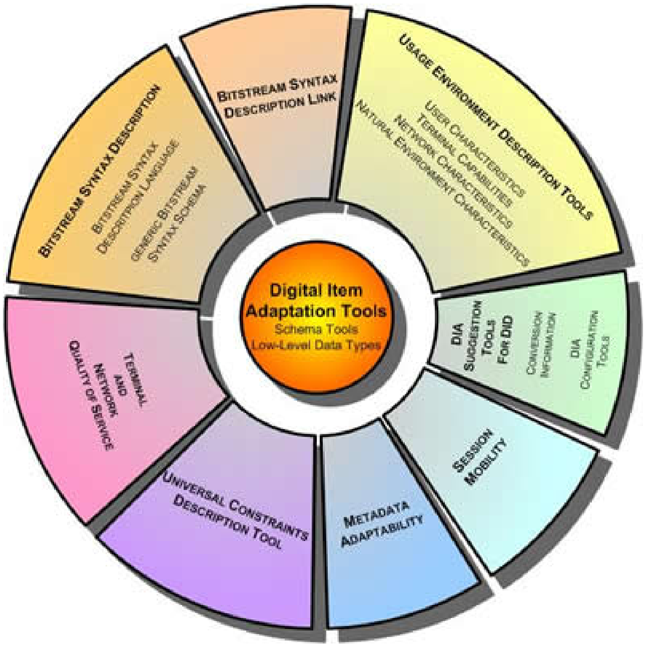


Figure 2.1 – Digital Item Adaptation tools

Accordingly, these tools can be used to implement context-aware content adaptation systems, providing the basis for thoroughly adaptation decisions taking. In this manner, an Adaptation Decision Engine (ADE) platform can use these DIA tools to analyse the current status of the consumption environment and decide upon the need to perform adaptation, including the type of adaptation to perform. If any adaptation is required this information is passed from the ADE to an Adaptation Engine (AE) to perform the request content adaptation operation.

The DIA tools mentioned above have been identified as the most relevant for the successful achievement of the objectives of this thesis. Accordingly, a detailed description of these tools is provided below.

Adaptation of DIs may involve both resource and descriptor adaptation and is represented on Figure 2.2 [17]. Various functions, such as temporal and spatial scaling, cropping, improving error resilience, prioritisation of parts of the content and format conversion, can be assigned to the AE. Implementation of an AE has not been normatively defined in the MPEG-21 standard, and therefore many technologies can be utilised.

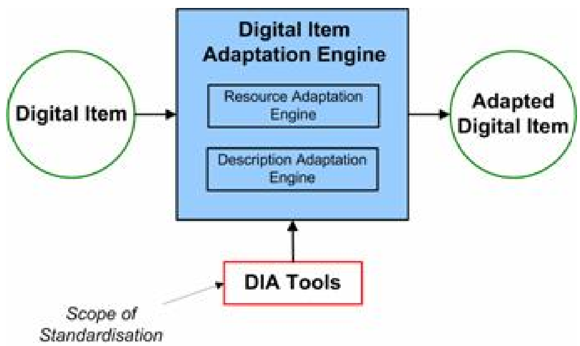


Figure 2.2 – Digital Item Adaptation Engine

The first amendment to MPEG-21 DIA deals with rights expressions to govern adaptations in an interoperable way, i.e., it specifies whether the permission to perform a particular adaptation has been granted or not. This amendment provides a framework to describe conversions, including the adaptation operation, parameters of the adaptation, and change conditions. Conversion descriptions can be used to identify suggested or permitted conversions for particular resources, or to describe terminal capabilities in terms of its supported conversions.

2.6.2 UED

The Usage Environment Description (UED) tools describe the terminal capabilities, in which the content is consumed, as well as characteristics of the network, User, and natural environment. In the context of this standard, natural environment pertains to the physical environmental conditions around a User such as lighting condition or noise level, or a circumstance such as the time and location. The aforementioned characteristics are described more precisely in the following subsections.

2.6.2.1 Terminal capabilities

In addition to enabling media format compatibility, the terminal capabilities description lets Users adapt various forms of multimedia for consumption on a particular terminal. The following classes of description tools are specified as part of DIA:

- **Codec capabilities** specify the format that a particular terminal is capable of encoding or decoding, for example, MPEG-4 Simple Profile at Level 3. The specification is symmetric with the MPEG-7 description tools for media format.
- **Input-output capabilities** include a description of display characteristics, audio output capabilities, and various properties of several input device types.
- **Device properties** characterize power-related attributes of a device, as well as storage and data I/O characteristics.

2.6.2.2 Network Characteristics

The specification considers two main categories: network capabilities and network conditions. One application for these descriptions is to enable multimedia adaptation for improved transmission efficiency. For instance, we can lower the delivery bandwidth of an audio stream if the available bandwidth is insufficient, or we can increase the rate of intra-coded blocks in a video stream if the packet loss rate is high.

- **Network capabilities** define a network's static attributes, such as the maximum capacity of a network and the minimum guaranteed bandwidth.
- **Network conditions** describe network parameters that tend to be more dynamic and time varying, such as the available bandwidth, error, and delay characteristics.

The former tools are used to assist the selection of the optimum operation point during the set up of the connection, whereas the latter ones are used to monitor the state of the service and update the initial set-up accordingly.

2.6.2.3 User Characteristics

We can classify the User characteristics specification into the following subcategories:

- **User info, usage preferences, and usage history** have imported description schemes (DSs) from MPEG-7 to describe a User's general characteristics as well as user preferences and usage history of Digital Items.
- **Presentation preferences** define preferences related to the means by which audiovisual information is presented or rendered to the User. For audio, the specification describes preferred audio power and equalizer settings. For visual information, the specification defines display preferences, such as the preferred color temperature, brightness, saturation, and contrast.
- **Accessibility characteristics** provide descriptions that enable Users to adapt content according to certain auditory or visual impairments. For audio, an audiogram specifies a person's hearing thresholds at various frequencies in the left and right ears. For visual impairments, the standard specifies the type and degree of color vision deficiencies.
- **Location characteristics** include a description of mobility and destination. Mobility describes a User's movement over time, particularly information about a User's angular changes and degree of random movement. Destination, as the name implies, indicates a User's destination. We can use these tools together to provide adaptive location-aware services.

2.6.2.4 Natural Environment Characteristics

Digital Item Adaptation (DIA) specifies the following natural environment description tools:

- **Location and time** refer to a Digital Item's location and time of usage. Both descriptions use MPEG-7 description schemes, in particular the Place DS and Time DS. Besides being standalone tools, we can also use location and time to specify User Characteristics.

- **Audiovisual environment** describes audiovisual attributes that can be measured from the natural environment and affect the way content is delivered and/or consumed by a User in this environment. For audio, the specification describes the noise levels and a noise frequency spectrum. For the visual environment, the specification defines illumination characteristics that may affect the perceived display of visual information.

2.6.3 AQoS

The AdaptationQoS (AQoS) descriptor specifies the relationship between constraints, feasible adaptation operations satisfying these constraints, and associated utilities (qualities). Therefore, the AQoS tool lets an adaptation engine know what adaptation operations are feasible for satisfying the given constraints and the quality resulting from each adaptation. In this way, terminal and network QoS (Quality of Service) management is efficiently achieved by adaptation of media resources to constraints.

In general, the AQoS description is generated in a media resource server and is delivered along with the associated media resource to an AE located at a network proxy or a terminal. The generation of the AQoS description can be done for each media resource stored in a server in advance in the case of on-demand applications. In the case of streaming of live events, the description could be generated by a prediction-based approach in real-time.

2.6.4 UCD

The Universal Constraint Descriptions (UCD) tools allow descriptions of constraints on adaptation operations, and a control over the type of operations that are executed upon the content when interacting with it.

Through the use of AQoS and UCD description tools, ADEs can obtain the best possible operation point and the correspondent transformation to perform. While the former provides the indication of different sets of encoding parameters and the resulting quality of the encoded bit stream for each of those sets, the latter enables the transformation of that information together with the information about the current conditions of the usage context conveyed as UED, into the form of restrictions that can further be used by the ADEs.

Chapter 3

3. OWL Web Ontology Language Overview

This chapter contains a deep analysis on OWL Ontologies.

Ontologies typically have two distinct components, names for important concepts in the domain and background knowledge/constraints on the domain. While the former is achieved by creating the classes that models the domain, the latter is carried out by imposing some specific properties to those classes. These concepts will be further analysed throughout this section.

3.1 OWL Lite/DL/Full Ontologies

There are three OWL species or sub-languages: OWL-Lite, OWL-DL and OWL-Full. A defining feature of each sub-language is its expressiveness. OWL-Lite is the least expressive sub-language while OWL-Full is the most expressive sub-language.

OWL Lite is the simplest sub-language. It was designed for easy implementation and to provide users with a functional subset that will get them started in the use of OWL.

OWL-DL is based on *Description Logics*, hence the suffix DL. This sub-language benefits from many years of DL research, thus it is build with well defined semantics and its formal properties are well understood in terms of complexity or decidability. DL is also known for its reasoning algorithms and highly optimised implemented systems.

It is therefore possible to automatically compute the classification hierarchy and check for inconsistencies in an ontology that conforms to OWL-DL.

OWL Full is union of OWL and RDF syntax and relaxes some of the constraints on OWL DL so as to make available features which may be of use to many database and knowledge representation systems, but which violate the constraints of Description Logic reasoners. For that reason, OWL Full is intended to be used in situations where very high expressiveness is more important than being able to guarantee the decidability or computational completeness of the language.

According to [19], choosing the sub-language to use is to simply decide – between OWL-Lite and OWL-DL – whether the simple constructs of OWL-Lite are sufficient or not, or to decide – between OWL-DL and OWL-Full – whether it is important to be able to do some reasoning on the ontology or whether it is important to be fully expressive and use powerful modelling facilities such as meta-classes.

3.2 OWL Syntax and Semantics

In order to write an ontology that can be interpreted unambiguously, a syntax and formal semantics for OWL are required.

An OWL ontology is an RDF graph, which is in turn a set of RDF triples. As with any RDF graph, an OWL ontology graph can be written in many different syntactic forms, allowing using different syntactic RDF/XML forms as long as these result in the same fundamental set of RDF triples.

An example could be the description of an OWL class where there are two alternative syntactic forms resulting in the same RDF triples. Consider the following XML/RDF syntax:

Table 3.1 – OWL Syntax Example

```
<owl:Class rdf:ID="Animal"/>
```

The following XML/RDF syntax encodes the same set of RDF triples, thus these alternative syntaxes have the same meaning.

Table 3.2 – XML/RDF Syntax Example

```
<rdf:Description rdf:about="#Animal">
 <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
</rdf:Description>
```

OWL is considered to be a vocabulary extension of RDF semantics, thus any RDF forms an OWL Full Ontology. Further, OWL assigns an additional meaning to certain RDF triples. The OWL Semantics and Abstract Syntax document [20] specifies which triples are assigned a specific meaning, and what that meaning is.

The semantics for OWL DL are formal and very similar to the semantics provided for Description Logics, in particular the ones that includes datatypes. But OWL DL as some other aspects to be considered, as annotations can be used to associate information with classes, properties and individuals, and even the ontology can be given annotation information. The import feature, another singular aspect of OWL, involves finding the referenced ontology and adding its meaning to the current ontology meaning.

3.3 Components of an OWL Ontology

Subchapter 2.3.1 as already mentioned the elements of an ontology, regardless the language of which the ontology is expressed, consisting of individuals, classes, attributes and relations. By this means, the basic elements of OWL ontologies are classes, properties, individuals also known as instances of classes and relations between these instances. These elements are explained in the following sections.

3.3.1 Classes

The most basic concept in a domain corresponds to classes. A class can be a set of elements, or in this case, a set of individuals, called the extension of the class. The individuals in the class extension are called the *instances* of the class. This construct is very import because it gives the ability to reason about individuals.

Classes may be organised into a superclass-subclass hierarchy, which is also known as taxonomy, where subclasses specialise their superclasses. Thus, every user-defined class is a subclass of `owl:Thing` that also contains every individual. This taxonomy relations expressed in OWL DL can be computed automatically by a reasoner.

A class can be described either by a class name or by describing an anonymous class with specific constraints on the class extension.

The latter class description is formulated in many ways. For example, an anonymous class can be described by an enumeration of the individuals that together forms the instances of a class, by satisfying a property restriction or satisfying a Boolean combination (union, intersection and complement) of class descriptions.

These class descriptions can be combined into class axioms, containing additional components used to provide information about classes and properties. OWL contains three language constructs for combining class descriptions into class axioms, allowing to say that the class extension of a class description is a subset (`rdfs:subClassOf`), has exactly the same class extension (`owl:equivalentClass`) or has no members in common (`owl:disjointWith`) with the class extension of another class description.

3.3.2 Individuals

Individuals represent objects in the domain, and may include concrete objects such as people, animals, etc., as well as abstracts individuals such as number and words. Individuals are also known as instances and can be referred to as being instances of classes. Many uses of an ontology will depend on the ability to reason about individuals.

Individuals are defined with individual axioms, also called facts. These facts are divided in two types, one about class membership and property values of individuals and the second about individual identity.

The latter type of facts is important to understand because OWL does not make the Unique Name Assumption (UNA), meaning that two different names, or URI references, could actually refer to the same individual. Therefore OWL provides three constructs for stating facts about the identity of individuals, such as `owl:`sameAs to state that two URI references refer to the same individual, `owl:differentFrom` to state that two URI references refer to different individuals or `owl:AllDifferent` for stating that a list of individuals are all different.

3.3.3 Properties

A property is a binary relation, i.e. a relation between two things, capable to declare general facts about the class extensions and specific facts about individuals. We can distinguish two main types of properties:

- Object properties – relations between instances of two classes;
- Datatype properties – relations between an instance of a class and a RDF literal or a XML Schema datatype value.

There are various ways to restrict the relation declared by the property, including the specification of a domain and range. The property can be defined to have a relation to other properties or to be a specialisation of an existing property. Therefore, properties, like classes, can be arranged in a hierarchy.

A property axiom defines characteristics of a property. In its simplest form, a property axiom just defines the existence of a property but it can provide the mechanisms to further specify property characteristics used for enhanced reasoning about a class.

The following sections discuss the various characteristics that properties may have.

3.3.3.1 Properties Domains and Ranges

Properties may have a *domain* and a *range* specified, meaning that properties link individuals from the *domain* to individuals from the *range*. It is important to realise that in OWL, domains and ranges should not be viewed as constraints to be checked, but they are used as ‘axioms’ in reasoning.

3.3.3.2 Property Characteristics

Different possibilities for designing properties are provided below as we introduce the property characteristics.

- **Functional properties**: A functional property is a property that can have only a value for each instance, i.e. for a given individual, there can be at most one individual that is related to the individual via the property. Both object properties and datatype properties can be declared as "functional".
- **Inverse properties**: Each object property may have a corresponding inverse property. If some property P1 links individual a to individual b then its inverse property P2 will link individual b to individual a. So, property P2 is tagged as the owl:inverseOf P1.
- **Inverse Functional properties**: If a property is declared to be inverse-functional then there can be at most one individual related to that individual via the property, i.e. the object of a property statement uniquely determines the individual.
- **Equivalent properties**: The `owl:equivalentProperty` construct can be used to state that two properties have the same property extension, i.e. that two properties are equivalent.
- **Transitive properties**: If a property is transitive, and the property relates individual a to individual b, and also individual b to individual c, then we can infer that individual a is related to individual c via property P. Syntactically, a property is defined as being transitive by making it an instance of the built-in OWL class owl:TransitiveProperty. If a property is transitive it cannot be functional.
- **Symmetric properties**: Properties may be stated to be symmetric. A symmetric property is a property for which holds that if the pair (x,y) is an instance of P, then the pair (y,x) is also an instance of P. In other words, the property is its own inverse property.

3.3.3.3 Property Restrictions

In addition to designating property characteristics it is possible to impose some constraints on the properties range in specific contexts by specifying some *property restrictions*. Restrictions in OWL are classified in three main categories:

- Quantifier Restrictions
- Cardinality Restrictions
- Has Value Restrictions

The quantifier restrictions are divided into existential quantifiers which can be read as *at least one*, or *some* (`owl:someValuesFrom`) and universal quantifiers which can be read as only (`owl:allValuesFrom`). A restriction actually describes an anonymous class, which contains all of the individuals that satisfies the restriction.

Universal restrictions describe the set of individuals that, for a given property, only have relationships to other individuals that are members of a specific class while existential restrictions describe the set of individuals that have at least one specific kind of relationship to individuals that are members of a specific class.

Cardinality restrictions are used to specify the number of relationships that an individual may participate in for a given property. This way, we can specify the exact number (`owl:cardinality`) of relationships that an individual must participate in for a given property, or some bounders as the minimum number (`owl:minCardinality`) of relationships that an individual must participate in and the maximum number (`owl:maxCardinality`) of relationships that an individual must participate in for a given property.

A has value restriction (owl:hasValue) describes an anonymous class of individuals that are related to another specific individual along a specified property.

3.3.3.4 Property Value Types

A value-type aspect describes what types of values the property can have. The value types are defined as xml schema value types where the most common are string, integer, float or Boolean. Other kind of value types are enumeration or instance type.

Enumerated properties specify a list of specific allowed values for the property.

The instance value type is restricted to be used in object properties, where the domain and range of the property is an instance.

3.4 Building an Ontology

Building an ontology is a difficult process but it can be done in many different ways. In fact, there is no correct way or standardised methodology for developing ontologies for any domain.

Therefore, I’ll try to describe a simple method for developing an ontology, in an iterative approach, starting from a rough decision and then refining the ontology in each formulated detail.

The first step is to determine the domain and scope of the ontology. Thus, we decide what domain of knowledge will be covered, the purpose for building the ontology and what type of answers should the ontology provide. This decision may change during the ontology-design process, but it will help to limit the scope of the model.

As several ontologies covering different domains or areas of knowledge are already publicly available it is almost always worth considering the reuse of an available ontology so we can refine and extend existing sources for our particular domain and task.

The reuse of ontologies is achieved by importing or simply editing the available document. Some information about importing ontologies is provided in Section 3.5 and Section 3.5.2 reviews some of publicly available ontology worth to consider the reuse in our scenario application.

The following two steps are the most important steps in the ontology-design process and are closely intertwined, because it is hard to do one of them first and then the other.

A few approaches for developing the class hierarchy have been provided by [21]. These development processes are defined by a top-down, a bottom-up and a combination of both approaches. A top-down development process starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts while the bottom-up process starts with the definition of the most specific classes, the leaves of the hierarchy, with subsequent grouping of these classes into more general concepts. The combination development process is a combination of the former ones, starting by defining the more salient concepts first and then generalises and specialise them appropriately.

Once the classes have been defined, we must describe the internal structure of concepts by defining the properties of the classes. As we have seen in Section 3.3.3 properties are divided into object properties and datatype properties and we must take this into attention that all subclasses of a class inherit the properties of that class. Properties can have different aspects describing the value type, allowed values if possible, and the number of values that the property can have as the cardinality of the property.

The following step of ontology development is the process of creating individual instances of classes in the hierarchy. Defining an instance of a class requires choosing the class where the individual fits, creating an individual instance of that class, and filling in the property values.

3.4.1 Using a Reasoner

Throughout the ontology development process it is imperative to use a reasoner or inference engine that supports reasoning with the full expressivity of OWL-DL, which is the most adequate OWL language to maintain a consistent ontology.

The reasoner provides inference services such as consistency checking, concept satisfiability and classification. Consistency checking service ensures that an ontology does not contain any contradictory facts while the concept satisfiability service determines whether is possible for a class to have any instance. Finally, the classification service computes the subclass relations between every named class to create the complete class hierarchy.

3.5 Importing ontologies

OWL ontologies may import one or more others ontologies. In fact, ontologies are built to share and used shared knowledge about a domain, making it possible to extend the descriptions of a class, properties and individuals.

The `owl:imports` statements are transitive, meaning that, if ontology A imports B, and B imports C, then A imports both B and C.

Every ontology has its own *namespace*, known as *default namespace*. A namespace is a string of characters that prefixes the classes, properties and individuals in an ontology. In order to ensure namespaces are unique they manifest themselves as Unique Resource Identifiers (URIs). Therefore it is important to maintain different namespaces for different ontologies to reference objects in another ontology in an unambiguous way and without causing name conflicts.

Namespace declarations simply set up a shorthand for referring to identifiers, as they do not implicitly include the meaning of documents located at the URI.

On the other hand, when an ontology imports another ontology, not only can classes, properties and individuals be referenced by the importing ontology, the axioms and facts that are contained in the ontology being imported are actually included in the importing ontology.

3.5.1 Dublin Core Metadata

In order to further annotate classes and other ontology entities with more expressiveness than actually OWL originally supports, the Dublin Core Meta Data ontology must be imported. The Dublin Core Meta Data Terms were standardised by The Dublin Core Meta Data Initiative (DCMI) and the contribution to RDF vocabulary can be seen in more details in [22]. These terms are a set of elements that can be used to describe resources, or in the case of ontologies, these terms can be used to describe the resources such as classes, properties and individuals.

There are many terms developed by the DCMI as described in [23], but the most important ones can be summarised as the following: *title*, *creator*, *subject*, *descriptio*n and *contributor*.

3.5.2 Ontologies for user, place and environment description

As aforementioned in Section 3.4 it is almost always worth considering the reuse of an available ontology to provide the starting point for the ontology development process, in order to refine and extend existing sources. Other possibility is the import of an available ontology in order to reuse the terms defined in that vocabulary.

In the analysis and framework specification phase some ontologies or general vocabularies commonly used by the scientifical community, for user profiles, place and environment description were investigated. I will introduce some of them.

The Public and Private Information (PAPI) [24] for Learners (PAPI Learner) is a data interchange specification that describes learner information for communication among cooperative systems, specifying the semantics and syntax of learner information. It defines elements for descriptive information about: knowledge acquisition, abilities, personal contact information, learner relationships, learner preferences, and similar types of information.

IMS Learner Information Package (LIP) [25] is a standard based on a data model that describes learner characteristics in a very similar way than PAPI specification. It includes the identification learner information, cognitive, technical and physical preferences for the learner and records of the academic performance at an institution.

The aforementioned technologies for describing a user profile, particularly a learner user, are general standards that go beyond the scope of the virtual classroom context-aware domain and therefore are not included in the ontology development process. Besides, these specification use particularly vocabularies and schemas non-compatible with OWL ontology language.

The Composite Capability/Preference Profiles (CC/PP) [26] is a W3C specification which defines an RDF-based framework for describing device capabilities and user preferences. It provides the means to specify client capabilities (i.e., the "user agent" information) and user preferences using Uniform Resource Identifiers (URIs) and RDF text sent in HTTP requests.

A CC/PP profile can be seen as a two-level tree containing components and attributes of those components. Components can be the hardware or the software platforms of the terminal or any specific application running on top of those platforms. Since the CC/PP specification uses RDF, its profiles are composed of sets of (Subject, Predicate, and Object). The components (the Subject) have named attributes (the Predicate) and values for those attributes (the Object). CC/PP uses a vocabulary to define the format and language for specifying the names and values of components as well as their attributes.

Despite the capabilities and extensible possibilities of use of the CC/PP standard, it is incompatible with OWL ontology language and cannot be integrated. The choice for using

CC/PP to describe some parts of the ontology would prevent from reasoning the whole ontology and consequently consistency checking wouldn't be as accurate as we desired.

In terms of ontology vocabularies for user profile, place and environment description some research has been made on the following list of publicly available ontologies:

- Friend of a Friend (FOAF) Vocabulary [27] – FOAF is an ontology designed to allow integration of data across a variety of applications, providing a basic vocabulary of terms to describe people and the associated things they make and do in various areas of knowledge.
- PERVASIVE-SO Person Ontology [28] – PERVASIVE-SO is a set of RDF/OWL ontology documents to define user profiles and preferences. It involves the use of three different ontology documents to fully describe the user, user preferences and user documents.
- ConOnto Physical and Personal – Set of ontologies for modelling context and meta-context information about user profiles, physical location, and devices used by the user.
- Basic Geo Vocabulary [29] – Basic Geo Vocabulary is a RDF vocabulary that enables the representation of latitude, longitude and other information about spatially-located things.

The information available from these aforementioned ontologies is quite often too much expressive and would not be efficiently used in the scope of the Context-Aware Ontology (CAO). Additionally, the language used for the previously described ontologies is OWL Full which is not appreciable for CAO developing and reasoning. Therefore, we decided to build the Context-Aware Ontology from the scratch.

Chapter 4

4. Virtual Classroom

This section contains a brief description of a virtual collaboration system and also provides the analysis of a virtual classroom application considering the involved elements.

This usage scenario analysis and use case specification, have provided the means to identify high-level context and real word situations enabling their appliance to the virtual classroom environment and the consideration of appropriate adaptation operations.

4.1 Virtual Collaboration Systems

Virtual Collaboration [30] refers to systems, which develop environments that integrate collaborative tools and functionalities through which remotely located users are able to meet in a virtual environment created by the supported audiovisual technology providing the sensation of all the remotely located users' presence in the same room regardless of their true geographical location.

In a typical virtual collaboration scenario, there are a number of fixed collaboration units (e.g., communication terminals, shared desk spaces, displays etc) as well as a number of mobile units (e.g., laptops, PDAs, mobile phones etc), which are equipped with various user interaction devices.

Such a collaborative system will be of a heterogeneous nature due to its users accessing the network with their available connectivity and their terminal devices of different levels of capabilities.

Virtual Collaboration Systems (VCS) are employed in various application scenarios as such as in virtual offices for remote meetings/collaboration, remote working for the production of consumer goods, remote media content production or in virtual classrooms for remote lectures, etc.

Users located in remote and heterogeneous environments access and exchange pervasive yet protected and trusted content. However, given the diversity of scenarios and usage environments in these types of applications, access to content is likely to pose significant challenges, which need to be addressed through the use of context-aware content adaptation.

4.2 Virtual Classroom Application Scenario

Within the context of VISNET II project, a conceptual framework for a virtual classroom application is proposed. This application is based on a VCS with a feature for context extraction from the media streams and adaptation of the delivered content.

Generally, virtual classroom refers to a learning environment where teacher and student are separated by time or space, or both, and the teacher provides course content through course management applications, multimedia resources, the internet, videoconferencing, etc.

In this case, Virtual Classroom is a learning environment envisaged to enable academic institutions to conduct a series of collaborative lectures and classes in which remotely located students can interact more efficiently and with flexibility. These students should also have the same comfort as the local audience in terms of listening to the speaker and viewing the speaker's expression, gestures, presentation materials and the whiteboard.

The lecture theatres of the participating academic institutions must be equipped with the necessary virtual collaboration infrastructure, which includes:

- A virtual whiteboard that can be used by the lecturer as well as the audience, sets projectors and screens for presentation display and video feeds provision.
- Set of cameras to capture the lecturer. One for a closed plan to capture the lecturer's expressions and another for a wider approach to inclusive capture the whiteboard.
- A microphone and a video camera associated with every each student to capture him.

The conducted lecture can also be followed remotely by enrolled students who have been unable to be in the classroom as well as the general public over a wired or wireless network using their home PC or a mobile terminal, such a PDA.

This VCS application supports multimedia content adaptation using context information, such as user preferences and characteristics, terminal capabilities, network conditions and the surrounding environment.

Before getting into the specification of virtual classroom use cases it is necessary to conduct the identification of the elements in the classroom searching for situations, contexts or concepts, events or actions and the state of each element in the system.

4.2.1 Scenario Specification

This section provides a textual description of the different contexts or the different conditions and characteristics of the usage environment that may occur during the consumption of virtual classroom application, the selected scenario based on the VCS being address in the integration Work Package (WP) of VISNET II.

Envisaging Use Cases definition, a thorough description of such an application has been made. This scenario specification has provided the means for the identification and selection of useful contextual information, of participating entities processing that contextual information and associated functionalities.

Throughout this analysis it was possible to extract a description of the participating entities such as the persons involved, objects used and physical location attended. Subsequently, the identification of events or actions and the context or state of each system element provided the means to define the low-level contextual information capable of conceptualise the virtual classroom domain.

The following subsections report the aforementioned scenario specification.

4.2.1.1 Participating Entities

Among the participating entities we can distinguish between the lecturer, local students, remotely located learners, participating guests and external users. The latter is one person that isn’t enrolled to the lecture but can attend to the class with minimal functionalities and requirements.

Every one of these entities is attached to a location. Due to the remotely located and external users a various number of locations will be incorporated in the system, in addition to the lecture theatre of each of the participating academic institution.

In order to deploy the proposed platform, these theatres are equipped with the necessary virtual collaboration infrastructure, which includes a variety of cameras to capture the lecturer and local or remote students, microphones, projectors and the whiteboard. Among these cameras, there is one for a closed plan to capture the lecturer’s expressions and another for a wider approach to inclusive capture the whiteboard.

4.2.1.2 Events and/or Actions Performed

Some of the scenario application events or action possibilities belong to the following list of options: the lecturer makes a speech without restraints, writes on the virtual electronic board, offers presentation materials using the projector and shares useful files for the class; Theorically the guest would have the same event possibilities as the lecturer but being away will have some constraints in terms of movement; Local students or even remotely located students can ask for permission to intervene in the lecture, share some opinion or make a question, write on the virtual board through an application installed in his personal computer/portable device or even receive the files provided by the lecturer.

4.2.1.3 Context or state of each participating entity

At this point, envisaging the identification of the context or state of each participating entity, it was necessary to make a distinction between the local students and the remotely located students.

For an optimal virtual classroom application functioning, there is the need for the gathering of useful students' related information about user preferences (visual and auditory preferences, Region of Interest (ROI), etc.), usage history and also impairments information (visual or auditory impairments).

Due to the constraints imposed by the terminal and network used by the remotely located user, the gathering of information about the type of terminal, terminal characteristics and capabilities, natural environment conditions (level of brightness or noise level), network conditions (available bandwidth, error rate and delay) and the quality of received information (reception of video, audio, presentation, shared files), is mandatory.

We cannot even relinquish the context of the theatre where the lecture is being addressed because it is important to perceive the quality of the arriving interventions of the guests or remotely located users and the virtual board changes.

Other type of context relevant for this scenario application is the type of user that is attending because this application is only fittable for context-aware adaptation of multimedia content that has been governed using Digital Rights Management (DRM). With this being said, some users might have or not the possibility of consuming personalised content which is context-aware enabled. For example, external students don't have the necessary rights for content adaptation and are only granted with the possibility of consuming contents with minimal requirements.

4.2.1.4 Low-Level contextual information

We are now able to perceive what low-level contextual information can and should be extracted for an effective scenario application description.

Useful user contextual information belongs to user info (name, gender, age, and role), user preferences (audiovisual, presentation, conversion), usage history and also content adaptation authorising information.

The terminal used by the learner is an essential element of the virtual classroom scenario application, thus a thorough analysis of terminal related contextual information is required.

Hence, the extraction of contextual information about terminal characteristics (CPU, device class, storage, memory and power characteristics, etc.) and terminal capabilities (codec and display capabilities) is mandatory.

Natural environment contextual information is also an important element in the adaptation decision taking process. This way, localisation, brightness and noise level, date, and also time of day information is provided.

Also network contextual information should be parametised as network capabilities and network conditions (available bandwidth, BER, and delay information).

We should also consider the state of the available sensors in the scenario, i.e., the information about microphones, cameras and the virtual board is also important. This information may be used to infer the physical or emotional state of user, or in the identification of indoor/outdoor situations.

As aforementioned, the concern with the governed content access and adaptation authorisation is vital. Therefore, we must take attention into the security and DRM contextual information associated to the consumed content throughout the lecture.

4.2.1.5 Possible adaptation operations

Envisaging the need for content adaptation, Table 4.1 previews the possible adaptation operations, in a way of maximising the quality of the user experience. This table is divided in groups of contextual information providers, events and adaptation operation. Different groups before different events or situations trigger the need for different adaptation operations.

Table 4.1 – Possible Adaptation Operations

Group	Events (Nature/origin)	Content adaptation operation
User	Auditory Impairments	Prioritising video context, audio-to-text transmoding, subtitle presentation. Prioritising a selected area of visual content in a scene (e.g. focus on lecturer lips)
	Visual Impairments	Prioritising audio context, text-to-speech transmoding
	Preferences – Region of interest and/or visual preferences	Video scaling transcoding according to user preferences
	Preferences – A user wants to watch highlights	Summarising the session
	Preferences – Delay sensitive content transmission	According to user preferences prioritising audio content by scaling video content or bit rate transcoding

	Usage history	Audiovisual content adaptation according to user usage history (presenting ROI, volume level, etc.)
	User authorisation – characteristics of the user who may be authorised or not to consume specific content	Present only the user-authorised segments of the content
Terminal	Inadequate display size	Downscaling to lower resolution or cropping to a selected region
	Non-supported audiovisual content	Audiovisual content transcoding
	Non-supported multimedia content	Transmoding documents to a video sequence
	Remaining terminal battery power is not enough for the full session	Lowering spatial/temporal resolution and/or fidelity of the video to minimise the utilisation of the processor
Network	Bandwidth scarcity constraint	Bit rate transcoding; Prioritising bit rates for important regions of the frame or separating the background from the foreground and prioritising the foreground
	Low signal reception or large bit error rate	Improving error resilience and/or use stronger error protection
Natural Environment	Lighting conditions	Increasing or decreasing the brightness of the presented material according to the illumination
	Present background noise level	Audio level/quality improvement; Audio-to-text; Prioritising a selected area of visual content in a scene

4.3 Virtual Classroom Use Case

A collaborative lecture is being conducted in an academic institution, within a lecture theatre equipped with the necessary virtual collaboration infrastructure. In the theatre of a different academic institution a female student with an auditory impairment is attending that lesson. Nevertheless, this student is very outgoing and most of the times she puts her doubts into a question.

On the opposite faculty site, a more relaxing male student is at the students bar and also trying to follow this lecture despite the loud noise in that place.

There are also some people interested in the lecture´s subject that tries to assist the same lecture but they aren't enrolled in.

The adaptation decision engine goal is to take a decision regarding the actions to perform when contextual information is available through it can maximize the quality experience by the user. By this, ADE will act differently before the aforementioned users.

4.3.1 Use Case 1

The remotely located female student has auditory impairments. This way, the decision platform takes measures to content adaptation, prioritising video content and presenting subtitles provided by audio to text transmoding. According to the user preferences it is decided to select an area of visual content in a scene providing visual content so she can read the lecturer lips. Still according to user preferences, interventions in the class will be exclusively done through the whiteboard.

4.3.2 Use Case 2

For the male student in the bar, the adaptation decision platform needs to detect and extract the context information about the terminal capabilities, network conditions and the surrounding environment.

This user is using a PDA, which has a small display. Therefore, ADE decides to downscale the spatial resolution accordingly. Since the aforementioned user is in a very noisy space, which is sensed by the context service manager, ADE decides to increase the volume level and display relevant subtitles. Meanwhile, the ADE realises that the remaining battery power level of the end terminal (i.e., the PDA) is not adequate for presenting the entire lecture session, and moreover the available network bandwidth is much less than the required data rate for delivering the audiovisual material at its best quality. Through sensing these constraints, ADE decides to further decrease the spatial and temporal resolution to minimise the processor utilisation as well as the required bit rate.

4.3.3 Use Case 3

The general public can only visualize a low resolution version of the video and do not have the privileges to view any of the adapted versions or participate in the lecture. These restrictions are imposed by ADE after contacting the Authorizer.

4.3.4 Use Case conclusion

As described in the former use cases, users located in remote and heterogeneous environments access and exchange pervasive yet protected and trusted content. These simple examples describe the diversity of scenarios and usage environments in this type of applications, thus access to content needs to be addressed through the use of context-aware content adaptation.

The big challenge in the design and development of an ADE relates to the reasoning engine, which uses the sensed context to infer high-level context once low-level context is acquired. This reasoning model is possible with the use of ontologies.

The goal of this model is to create a two-layer ontology approach using OWL. The basic layer provides descriptions of generic concepts and rules that can be used for any generic application scenario while the second layer provides specific rules for the virtual classroom application.

It is clear now the big challenge, how to obtain descriptions and set of relationships in the form of rules that represent as accurately as possible the real-world situations in virtual classroom applications.

4.4 VISNET II proposed architecture

The proposed context-aware adaptation platform for a VCS, which is conceptually illustrated in Figure 4.1, consists of the following four major modules: (1) Context Providers (CxPs), (2) Adaptation Decision Engine (ADE), (3) Adaptation Authoriser (AA), and (4) Adaptation Engine Stacks (AESs) comprising Adaptation Engines (AEs) within. These modules are developed as independent units that interact with each other through Web Services-based interfaces using Simple Object Access Protocol (SOAP), a simple and extensible web service protocol [31].

Well-defined interfaces based on open standards also guarantee interoperability and flexibility of freely adding, removing and migrating modules. The use of ontologies in the ADE, while being also a vehicle for interoperability, provides the platform with context-aware analysis capabilities closer to real-world situations. The AA ensures the governed use of protected content. Flexible AEs enable the execution of a variety of adaptations that can be dynamically configured and requested on the fly.

4.4.1 Context Providers

Entities, either software or hardware, that are able to generate and provide this explicit contextual information are designated as Context Providers (CxPs). The low-level contextual information generated by these entities, once acquired and represented according to the MPEG-21 DIA UED standard format, will be used to infer higher-level concepts, and thus assist the adaptation decision operation.

In particular, the MPEG-21 UED tool provides four main types of descriptors: *User*, *Terminal*, *Network* and *Natural Environment*. Based on this division, four context profiles have been created, as illustrated in Figure 4.2. With these profiles, each CxP needs only to know and implement its own sphere of action resulting in a level of interoperability enhancement.

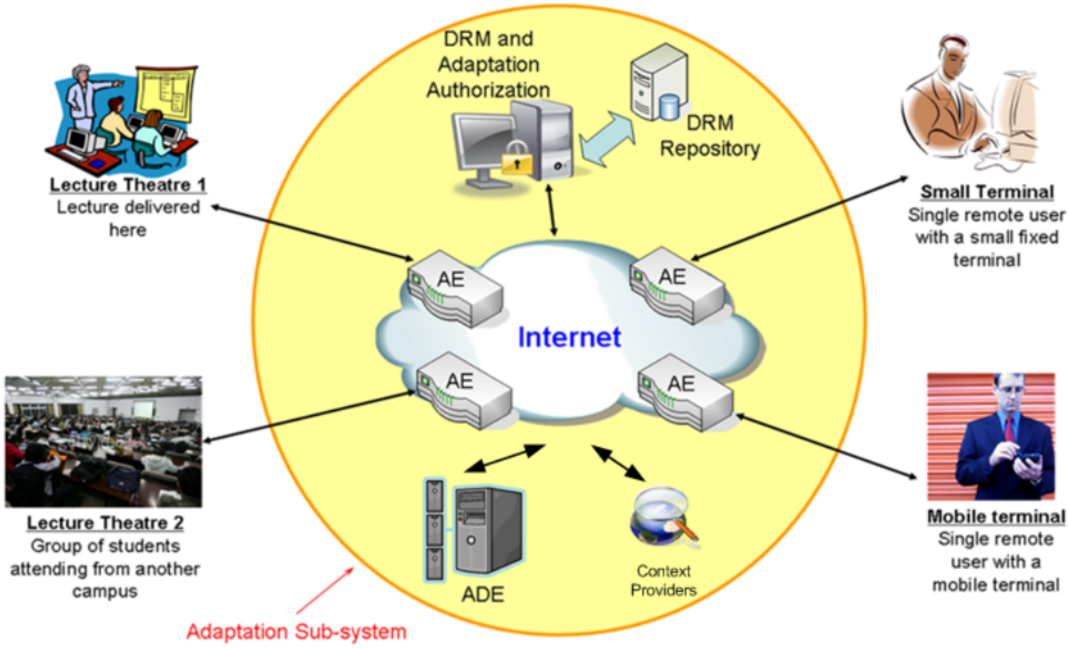


Figure 4.1 – Context-aware content adaptation platform in a virtual classroom collaboration scenario

4.4.2 Adaptation Decision Engine

The ADE is the module responsible for taking decisions regarding the actions to perform to maximise the user's quality of experience when the contextual information is available.

In the architecture described in [32], a central coordinator interacts with dedicated modules responsible for sensing low-level context generated by the CxPs, and other required content-related metadata (media characteristics) and rules for reasoning specific to the application in view.

The greatest challenge in the design and development of the ADE relates to the *Reasoner*. Whenever rules are available, the *Reasoner* is invoked by the *ContextServiceManager* and interacts with the *DecisionTaking* module to select the most appropriate adaptation and corresponding service parameters. This will be done through the use of ontologies.

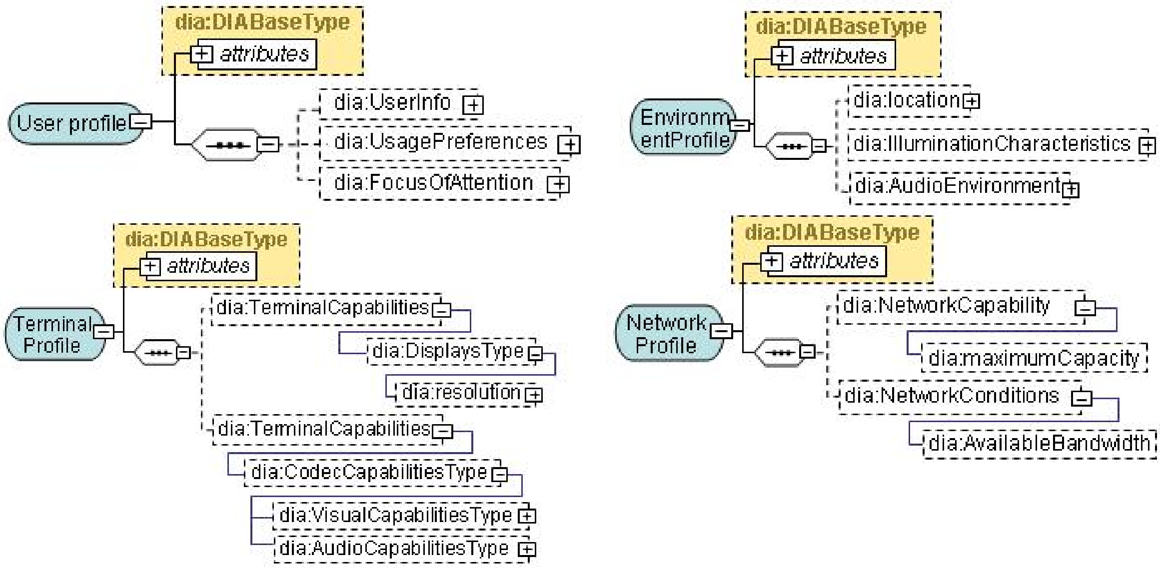


Figure 4.2 – Virtual collaboration context profiles

4.4.3 Adaptation Authoriser

Within the proposed modular platform, the Adaptation Authoriser (AA) acts as a new context provider, which converts licenses into adaptation constraints.

The main role of an AA in a governed system is to allow (or disallow) adaptation operations based on whether they violate or not any conditions expressed in the licenses. The AA looks into the DRM repository (Figure 4.1) to find all the licenses associated with a certain resource and user, and passes relevant adaptation constraints to the ADE, so that it can take an appropriate adaptation decision.

Complementing those presented in Figure 4.2, a novel context, namely *Authorisation Profile* is developed for this contextual information, which comes from the AA, as shown in Figure 4.3.

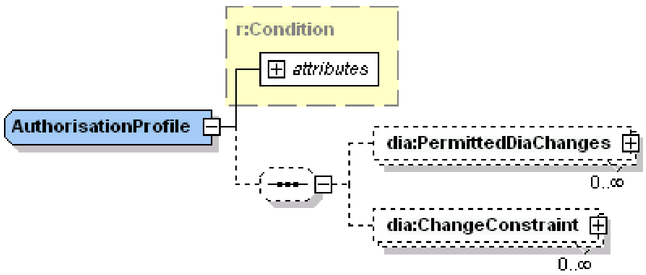


Figure 4.3 – Authorisation profile

4.4.4 Adaptation Engine Stacks

The adaptation Engine Stacks (AESs) is the module responsible for performing the adaptation operations when requested by the ADE. The progress of the adaptation operation is monitored by the Adaptation Engine monitoring service and if necessary, it informs the progress back to the ADE.

Chapter 5

5. Semantic Web Tools

This section provides an overview of tools that are being used worldwide to build Semantic Web application with a special emphasis on the development of OWL ontologies and applications that make use of ontologies.

5.1 OWL Editors

Included in a preliminary phase of semantic web study and standardization overview, a selection and installation of open source or freeware tools enabling ontology creation mechanisms, was carried out. Several tools were found and tested in the most various semantic web languages (e.g. SPARQL, DAML+OIL, RDF, and OWL).

An OWL editor is a software capable to develop, view and edit a semantic web ontology on the OWL language. UML-based (Unified Modeling Language-based) ontology editors were considered to be an intelligent approach for ontology development, but there was not free software that could prove to be an added value to OWL development.

The following subsections highlight the most important OWL editor's tools that must be taken into account on the development phase.

5.1.1 Protégé-OWL

Protégé [33] is a graphical ontology-editing and knowledge-acquisition environment. Its component-based architecture enables system builders to add new functionality to it by creating appropriate plugins.

Protégé has a very intuitive and user-friendly interface for the development of OWL ontologies and it provides the most advanced OWL edition capabilities. Plus, there is a great variety of plugins that complements the functionalities offered by the core tool. Protégé doesn't allow a UML-based nor graphic OWL editing but some plugins allow visualizing the diagrams that represents the OWL code developed by Protégé. It also allows collaborative ontology development and the exportation of ontologies into a variety of formats including RDF(S), OWL, and XML Schema as its output formats can be readily extended to support other Semantic Web languages. Protégé is written in Java and available at http://protege.stanford.edu/.

It uses a DIG (DL Implementation Group) interface to call the reasoner so it can check constraint axioms and consistency of the developed ontology, but on the latest version it is possible to use pre-installed reasoners like Pellet and FaCT++.

A large community of developers and academic users supports Protégé, providing a rich documentation available, one of the good reasons to choose Protégé as the favourite OWL editor.

Figure 5.1 represents Protégé OWL editor environment and *Figure 5.2* represents the module OWLViz enabling OWL visualization.

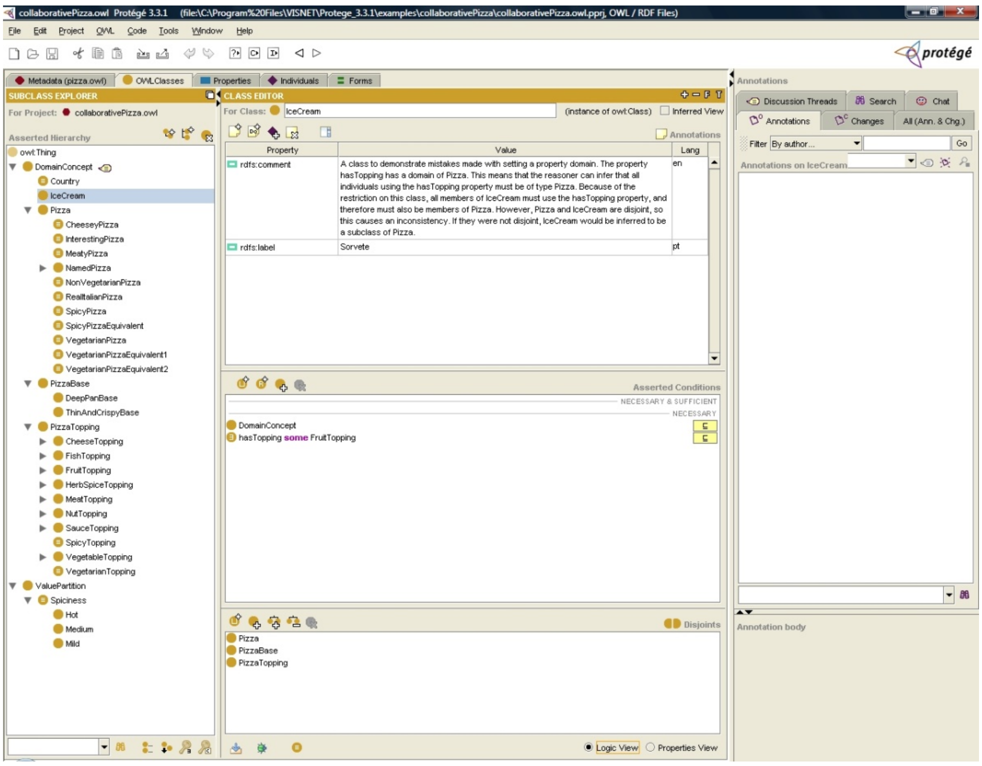


Figure 5.1 – Protégé OWL Environment

Protégé has also an OWL API that extends the functionalities to developers.

The Protégé-OWL API [34] is an open-source Java library for OWL and RDF(S). The API provides classes and methods to load and save OWL files, to query and manipulate OWL data models, and to perform reasoning based on Description Logic engines. Furthermore, the API is based on Jena Ontology API (Section 5.1.2).

The API is designed to be used for the development of components that are executed inside of the Protégé user interface and also for the development of stand-alone applications. The latter will be crucial for the goal of developing an application capable to integrate the design of an ontology with the extraction of individuals' data.

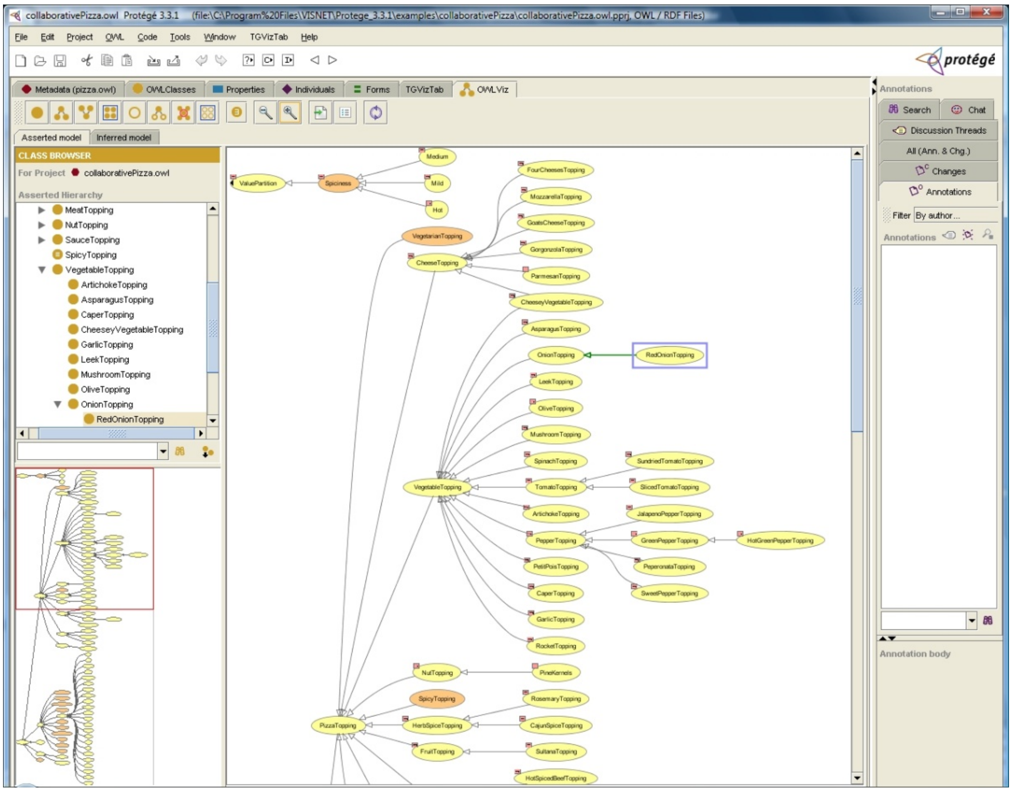


Figure 5.2 – Protégé OWL Visualization

Additionally, the Protégé main application also supports the newly standardised relational query language SPARQL, enabling users to query the data instantiated in the ontologies.

5.1.2 Jena

Jena [35] is an open source Java framework for building Semantic Web Applications. This framework provides a programmatic environment to extract data from and write to RDF graphs and an API for ontology processing considering RDFS and OWL languages. It also enables SPARQL queries and includes a rule-based inference engine.

The framework has various internal reasoners and also provides support for external reasoners through the DIG interface.

Many tested tools are built with Jena RDF/Ontology API and as mentioned before it provides the background for the Protégé-OWL API.

This framework is available at http://jena.sourceforge.net/.

5.1.3 Metatomix M3t4.Studio Semantic Toolkit

The Metatomix Semantic Toolkit [36] is a set of Eclipse plugins that allow developers to create and manage ontologies based on the OWL and RDF standards. The standards based Semantic Toolkit is available as a standalone eclipse feature. Providing a native Standard Widget Toolkit (SWT) editing tool for these semantic standards enables developers to maintain and develop ontologies in the Java Integrated Development Environment (IDE). Available at http://www.m3t4.com/semantic.jsp.

Metatomix has proven to be a good alternative to Protégé also providing a user-friendly interface and easy but advanced ontology creation. It also includes import/export functionalities enabling the edition of ontologies in other languages than OWL. It includes an ontology validator but it lacks on the ability to graphically visualise the created ontology.

Figure 5.3 represents a screenshot of Metatomix Semantic Toolkit environment.

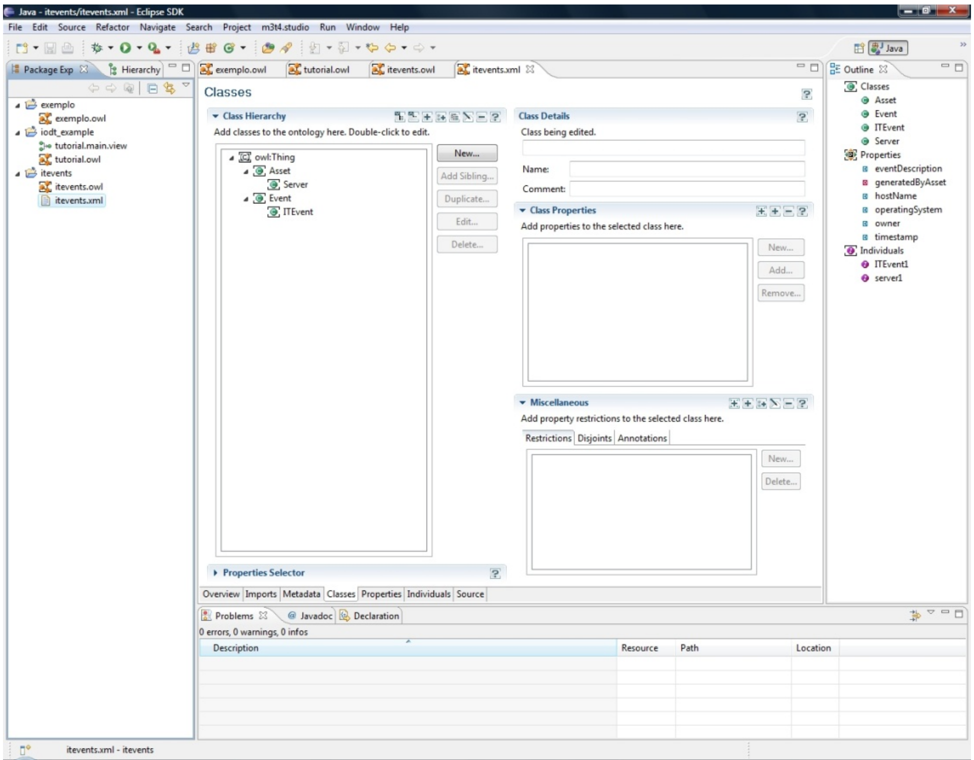


Figure 5.3 – Metatomix Semantic Toolkit environment

5.1.4 IBM Integrated Ontology Development Toolkit

IBM Integrated Ontology Development Toolkit (IODT) [37] is a toolkit for ontology-driven development. This toolkit includes the Eclipse Modeling Framework (EMF) Ontology Definition Metamodel, an OWL Ontology Repository (named Minerva) and it is available at http://www.alphaworks.ibm.com/tech/semanticstk.

EODM (EMF Ontology Definition Metamodel) is built on top of EMF and conforms to the ODM (Ontology Definition Metamodel) standard of Object Management Group (OMG). It provides a set of programming APIs so users can create, modify, and navigate RDF/OWL models using EODM. In order to facilitate software development and execution, EODM includes RDFS/OWL parsing and serialization, reasoning, and transformation between RDFS/OWL and other data-modelling languages.

This tool has two separate modes, a RDFS/OWL Basic Editor and an OWL Visual Editor. Although the name RDFS/OWL Basic Editor, this mode doesn't appear to be easily understandable. It doesn't have any hierarchy listing to support the creation and edition of classes and/or properties nor a separation between the edition of classes and properties.

The OWL Visual Editor has a good user-friendly interface but it lacks on necessary functionalities and has some editing limitations. It doesn't have instance editing capabilities and doesn't support operators such as intersectionOf, unionOf and complementOf. Plus, the import functionality doesn't work as supposed to with OWL files created on other ontology software editors.

Figure 5.4 represents the RDFS/OWL Basic Editor and *Figure 5.5* represents the OWL Visual Editor.

5.1.5 CMap COE

CMAP COE (CmapTools Ontology Editor) [38] is an integrated suite of software tools for constructing, sharing and viewing OWL encoded ontologies based on CmapTools, a concept mapping software used in educational settings, training, and knowledge capturing. Concept maps provide a human-centred interface to display the structure, content, and scope of an ontology.

COE is an RDF/OWL ontology viewing/composing/editing tool providing OWL or RDF import from RDF/XML files. It enables quickly navigation through an ontology and a graphical edition on new nodes and arcs by click and drag operations. It is possible to save or publish the ontology as an image or a XML file and to export to OWL/RDF in various formats, including RDF/XML, Turtle and N3.

Figure 5.6 presents a screenshot of the graphical editor environment of CMap COE.

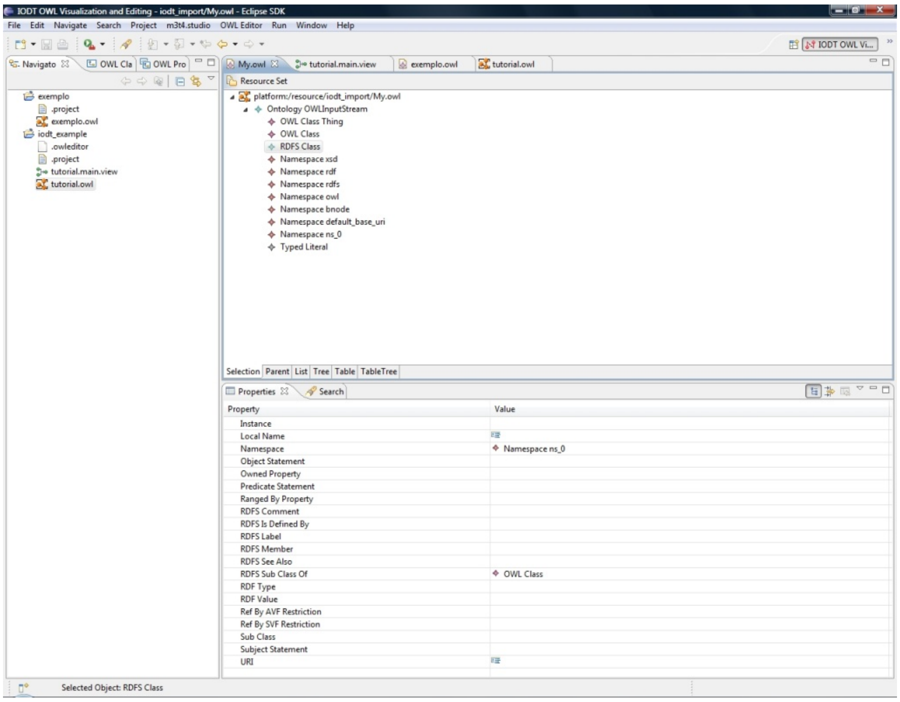


Figure 5.4 – Integrated Ontology Development Toolkit RDF/OWL Basic Editor

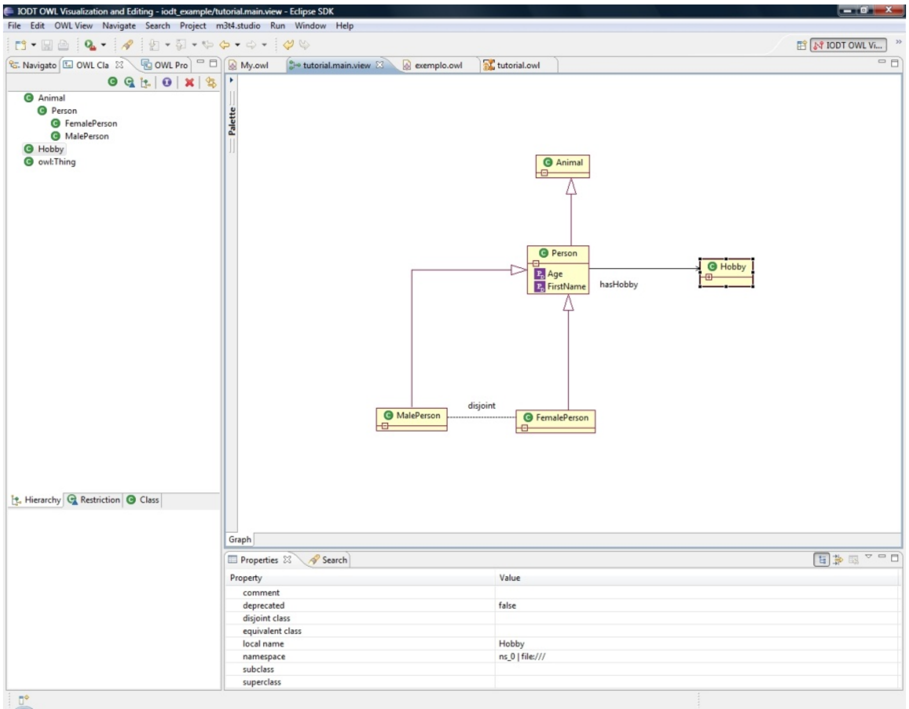


Figure 5.5 – Integrated Ontology Development Toolkit OWL Visual Editor

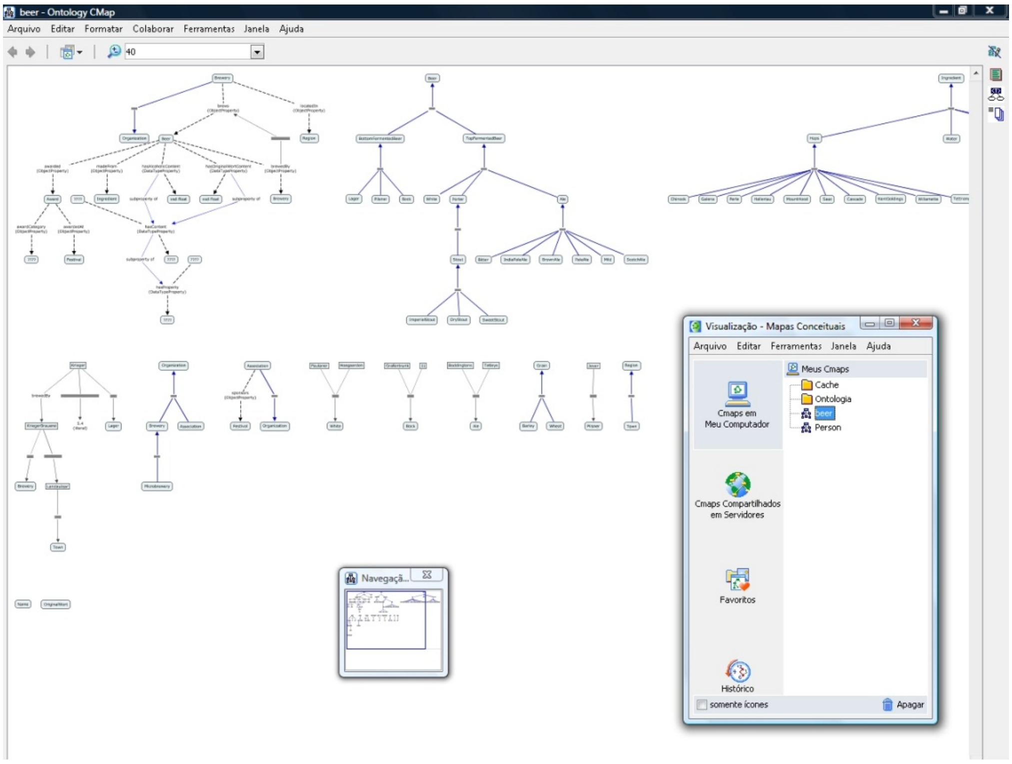


Figure 5.6 – CMap COE graphical editor

5.1.6 Others

On the listing available below, there is a brief preview of some OWL ontology editors.

- **ICOM** - ICOM (Intelligent Conceptual Modelling) [39] allows the user to design multiple extended Entity-Relationship diagrams with inter- and intra-schema constraints. Complete logical reasoning is employed by the tool to verify the specification, infer implicit facts, devise stricter constraints, and manifest any inconsistency. It is said to support UML-based editing in the future but it doesn't have a new release for a quite long time.
- **Topbraid Composer** - Top Quandrant's TopBraid Composer [40] is a complete standards-based platform for developing, testing and maintaining Semantic Web applications. The tool also implements RDFa and GRDDL. Composer provides access to various reasoning engines and serves as an agile development toolset for semantic applications. Based on Eclipse, Composer can also be extended with custom plugins. Unfortunately TopBraid Composer is a professional development environment, thus this tool is not freeware. Some other professional tools were tested but this is the only one referred.
- **OilEd** - OilEd [41] is an ontology editor allowing the user to build ontologies using DAML+OIL and in the latest version it envisioned to support OWL ontologies.

- **Swoop** - SWOOP [42] is a tool for creating, editing, and debugging OWL ontologies, a hypermedia based lightweight OWL Ontology editor. Very complex editing mechanism with no undo functionality. It doesn't allow defining domain and range properties.
- **DumpOnt** - DumpOnt [43] is a program to display the class and property hierarchies present in RDF Schema, DAML+OIL or OWL ontology. This implementation, based on Jena 2 replaces a previous implementation based on RDF API. This tool enables to visualise the ontology but it isn't capable to be an OWL development environment.
- **Orient** - ORIENT (Ontology engineeRIng ENvironmenT) [44] is a project that develops an Eclipse-based integrated ontology engineering environment that supports RDF(S) and OWL ontology standards and it also tries to be integrated with other modelling formalism and tools such as EMF and UML. This tool has an intuitive interface but it doesn't allow OWL graph visualisation neither UML-based editing.
- **SWeDE** - The Semantic Web Development Environment [45] is built on Eclipse and includes an OWL editor with helpful features like syntax highlighting, auto completion, and error-detection. It doesn't provide a graphical nor UML editor or visualisation, only an OWL code editor, so it is not recommended for the development of large ontologies.
- **POWL** - POWL [46] is a semantic web platform, also an integrated ontology editing and management solution developed in PHP. It covers database model storage, layered RDFS/OWL APIs and a web based user interface for collaborative ontology development. It depends on a local database, therefore it was not tested but still, due to the novel approach for a OWL development environment, a reference is made here.
- **Apollo** – Apollo [47] is a knowledge modelling application. Modelling is based around the basic primitives, such as classes, instances, functions, relations etc and it does a full consistency check while editing. It has a very weak user interface and it doesn't have an export functionality being only capable to save the ontology in XML.

5.2 Reasoners

This section contains previews on description logic (DL) reasoners and inference engines.

A Description Logic reasoner performs various inference services, such as computing the inferred superclasses of a class, determining whether or not a class is consistent (a class is inconsistent if it cannot possibly have any instances), deciding whether or not one class is subsumed by another, etc.

An inference engine is a computer program that tries to derive answers from a knowledge base. It is the "brain" that expert systems use to reason about the information in the knowledge base for the ultimate purpose of formulating new conclusions.

An inference engine has three main elements. They are:

- An interpreter. The interpreter executes the chosen agenda items by applying the corresponding base rules.
- A scheduler. The scheduler maintains control over the agenda by estimating the effects of applying inference rules in light of item priorities or other criteria on the agenda.
- A consistency enforcer. The consistency enforcer attempts to maintain a consistent representation of the emerging solution.

Some reasoning mechanisms have a DIG interface to Description Logics systems. The DIG interface is an emerging standard for providing access to description-logic reasoning via an HTTP-based interface to a separate reasoning process. This interface is very important because it provides a standardized XML interface to some Java reasoners API with a very elegant and efficient way of parsing, creating, and manipulating XML documents. Protégé uses this interface to check the ontology consistency and classify the ontology taxonomy.

Below, there is a list of DL Reasoners or Inference Engines:

- **FaCT++** - FaCT++ [48] is a free open-source C++-based reasoner for SHOIQ with simple datatypes (i.e., for OWL-DL with qualifying cardinality restrictions). It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and incomplete support of ABoxes (retrieval). It supports the lisp-API and the DIG-API.
- **Pellet** – Pellet [49] is a free open-source Java-based reasoner for SROIQ with simple datatypes (i.e., for OWL 1.1). It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, conjunctive query answering). It supports the OWL-API, the DIG-API, and Jena interface and it comes with numerous other features.
- **Bossam** – Bossam [50] is an inference engine for the semantic web. It is basically a RETE-based rule engine with native supports for reasoning over OWL ontologies, SWRL ontologies, and RuleML rules.
- **KAON2** - KAON2 [51] is a free Java reasoner for SHIQ extended with the DL-safe fragment of SWRL. It implements a resolution-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, conjunctive

query answering). It comes with its own, Java-based interface, and supports the DIG-API.

Some other reasoners were installed but they prove to be too complex to install and use. For example, **SweetRules** [52] has about eleven necessary 3rd party components before a successful installation and after that, it only could result in bad utilisation.

The DIG interface was an important starting point to establish what reasoning mechanism should be taken into account on the development phase. FaCT++ and Pellet proved to be useful reasoning mechanisms to be used along the DIG interface used by the Protégé OWL editor software and further tests should be done to prove which is the most efficient/fast in reasoning tasks.

Chapter 6

6. Context-Aware Ontology

As we have seen in Section 2, contextual information can be considerably large as it comprises any kind of information that characterises any aspect of the content consumption environments. The adoption of an ontology-based approach to model context, provides a powerful mean to describe those different contexts in a semantically richer form, thus allowing a finer characterisation of the situation the user is in while consuming multimedia content and thus closer to the reality. In addition, and of utmost importance, it enables formal analysis of the domain of knowledge through reasoning. However, encapsulating all the knowledge about the great diversity of existing contexts in a single context ontology, may prove to be inefficient and extremely CPU consuming. Considering that many different situations of consumption of multimedia content share common concepts and characteristics, exhibiting then a limit number of specific concepts, we have opted for the definition of a two-layer approach context ontology, with a basic, generic layer encapsulating knowledge common to any domain. Moreover, given the great interest in building an interoperable system, capable to operate in distinct environments and gather varied low-level context from different sources, it was decided to develop this generic layer based on existing open standards for context representation, namely the MPEG-21 DIA standard. Figure 6.1 illustrates this two-layer ontology model, adopted for the construction of the Context-Aware Ontology (CAO).

The second layer, i.e., the domain-specific layer, provides rules dedicated to a given application. Multiple domain-specific ontologies can thus co-exist in this layer. For example, the virtual classroom-specific layer provides the means of reasoning various adaptation options to help the user understand the classroom session better.

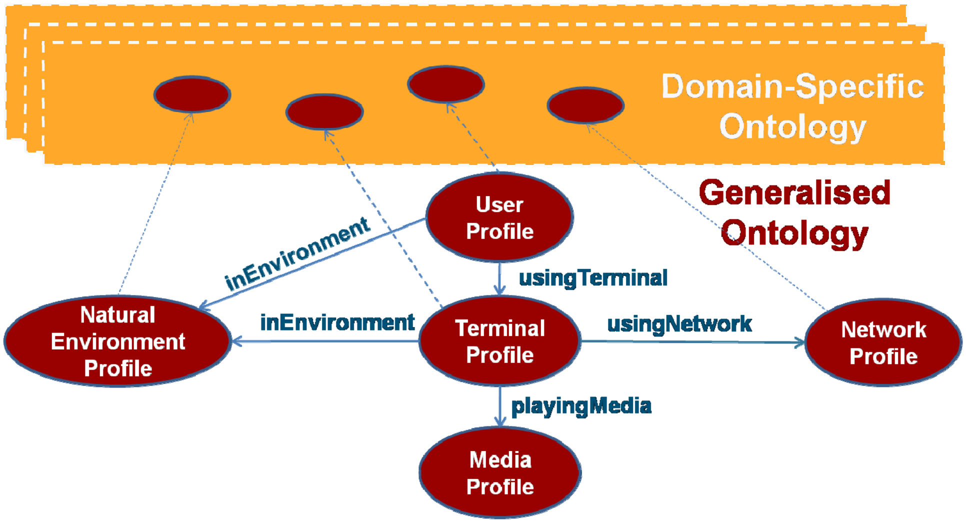


Figure 6.1 – 2-Layer Context Ontology Overview

6.1 CAO Ontology Design

The context-aware ontology is constructed with five important entities, four of which are designed based on four profiles extracted from the MPEG-21 DIA UED descriptor tool. As discussed in Section 4.4 these profiles are namely, User, Terminal, Natural Environment and Network. Another fundamental entity, the Media profile descriptor, is based on the MPEG-7 MDS description capabilities providing the means to describe the consumed multimedia content. The first group addresses the concepts concerning the context environment, whereas the latter captures knowledge concerning the context of the content, i.e., content structural metadata.

The following subsections present the concepts associated to the aforementioned main entities.

6.1.1 User concept

The user is an important asset in the Ambient Intelligence domain, where an application should adapt to every user. Contextual information is only relevant if it influences the way the user performs the task he is engaged in. Hence, there is the need to consider the user as a central and/or fundamental part on context-aware applications.

The acquisition of user related contextual information enables applications and services to improve the user quality of experience. Through the use of this kind of information it will be possible to adapt content according to user preferences or physical characteristics as auditory or visual impairments.

In the developed ontology a distinction is made between user preferences, user profile information such as name, genre, age, and also user impairments information. While the user preferences information is considered as dynamic and dependent on the current situation or event, personal information remain in a static way.

Figure 6.2 represents relevant user concept elements and associated object properties, omitting the datatype properties associated to each class for better readability.

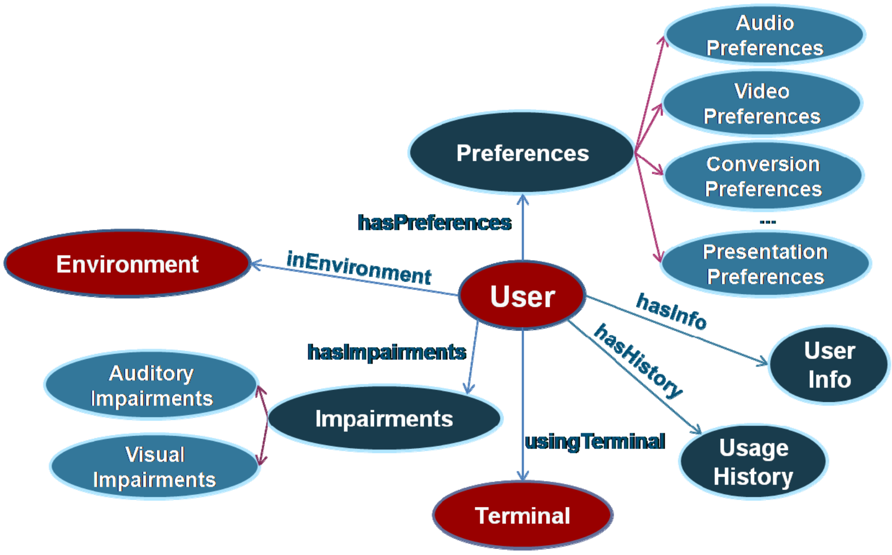


Figure 6.2 – User concept overview

6.1.2 Terminal concept

The development of the terminal concept design requires to have a good description about terminal characteristics and capabilities. This kind of information is crucial to take the best decision possible according to the terminal used.

The terminal section of the ontology provides a description about terminal characteristics and capabilities extracted from the terminal profile which is based on the MPEG-21 DIA UED, and define aspects as the type of terminal, device class, storage capacity, battery status, available input/output interfaces, and codec and display capabilities.

Characteristics as battery and storage status are important aspects in the sense that is necessary to perceive if the user, in the virtual collaboration scenario, will be capable of visualising the full lecture session. If, for any reason, the remaining battery isn't enough for the full session, a content adaptation decision must be taken (bit rate transcoding, decreasing the brightness, lowering spatio/temporal resolution) to minimise the utilisation of the processor.

It is mandatory to gather the contextual information about terminal capabilities in terms of audiovisual content format reproduction capabilities and display capabilities information in order effectively consume the audiovisual multimedia content. The size of the display is an important constraint that must be filled, where content with higher dimensions must be adapted (downscale to a lower resolution, cropping a selected region) to perform an enjoyable reproduction. Other aspects like capabilities of consuming various audiovisual content formats, colour capabilities, display resolution and refresh rate are also considered to be important in the adaptation decision taking.

Figure 6.3 presents an overview of the ontology section related to the terminal concept. Again, only object properties associated to the terminal profile elements are represented.

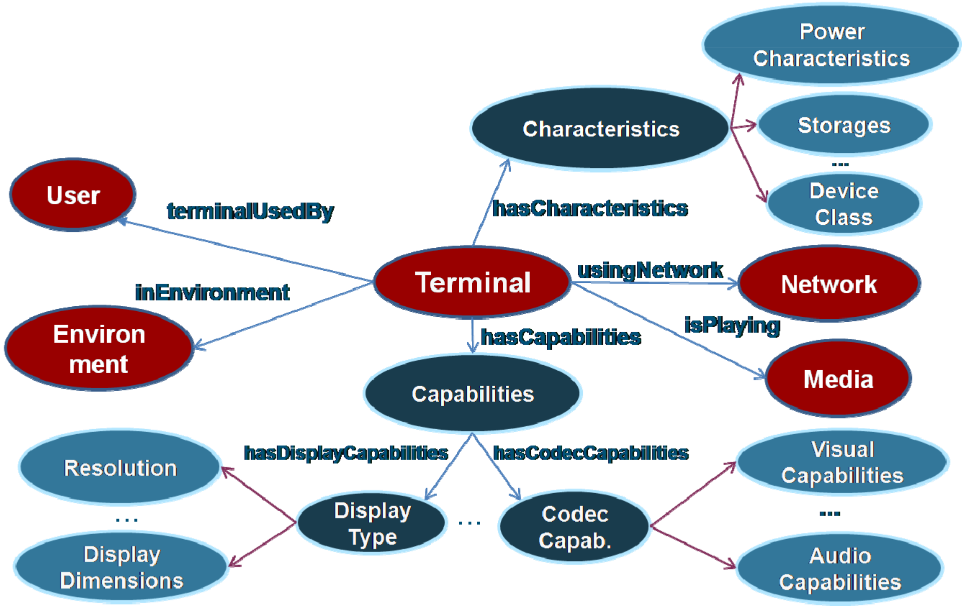


Figure 6.3 - Terminal concept overview

6.1.3 Natural Environment concept

The natural environment in which the user is interacting provides information that allows the ADE to take well-informed adaptation decisions to enhance the user quality of experience.

The fundamental concept that allows to describe the natural environment includes time, localisation, and environment conditions information such as noise level and illumination status. The latter can be acquired through terminal sensors. For example, a camera measures the brightness of the environment and a microphone is able to measure the surrounding noise level.

Although the environment isn't directly related to the user, as its' contextual information is only sensed by the terminal, any change on the environment can influence the user behaviour and it is therefore a serious aspect that the ontology regards.

This information might be sensed by varying sources with different accuracies, resulting into different measures. This issue goes beyond of this specification scope and will not be taken into account.

Figure 6.4 represents the relevant Natural Environment elements and associated object properties.

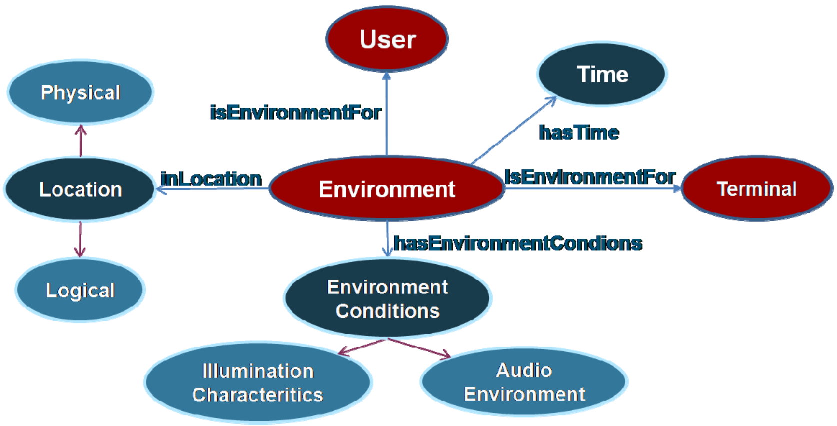


Figure 6.4 – Natural environment concept overview

6.1.4 Network concept

The network provides specific functionalities to a user through the terminal device where different types of networks are used. Therefore, the network section of the ontology provides a formal description of the used network capabilities and conditions, defining aspects as the maximum bandwidth, minimum guaranteed bandwidth, error correction type, bit error rate, delay information and of utmost importance, the current available bandwidth.

The aforementioned contextual information is important to foreseen the quality of the multimedia content reception and it may trigger an adaptation decision. The adaptation decision taking mechanism will consider the network capabilities in a way that the adapted bit rate doesn't overcome the maximum bandwidth or become lower than the minimum guaranteed.

Figure 6.5 represents the network conceptualisation and presents the most important elements and associated object properties.

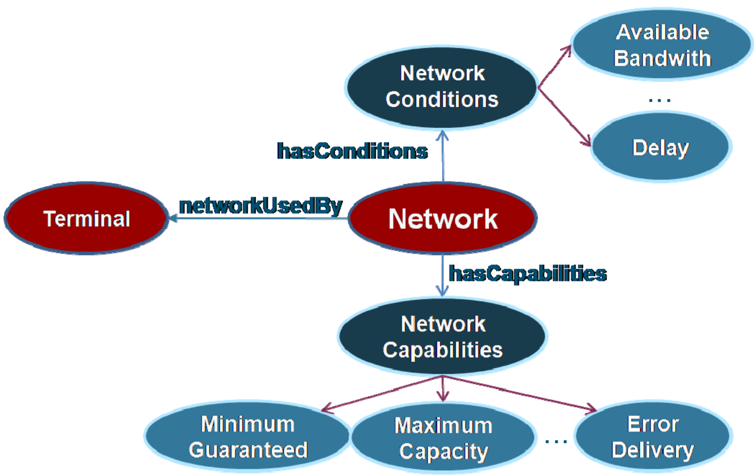


Figure 6.5 – Network concept overview

6.1.5 Multimedia concept

The consumed multimedia content takes an important role in the developed ontology. This concept is designed based on a few of MPEG-7 MDS descriptors and represents the contextual information relative to the current state of the multimedia content being consumed. This contextual information is divided in its basic parts such as visual and audio components, presentation and content format.

In the visual component we may found a description of the visual content format, frame rate, frame spatial dimensions, aspect ratio, and resolution. This information is important to perceive the impact of the current state of the visual component on the user.

The audio component is described by aspects as audio content format, number of audio channels and frequency sample rate. In the same way of the visual component, this information is important to perceive the impact of the current state of the audio component on the user.

The presentation format component includes information related to the presentation of ROI, subtitles, current brightness level and audio level.

Finally, the media format component is described by the scalability (spatial, temporal), type of content (audio, image, video, audiovisual), medium and file size if available, bit rate and the multimedia content format used.

An overview of this section of the ontology is show in Figure 6.6. It represents the relevant elements and relative properties except for the datatype properties.

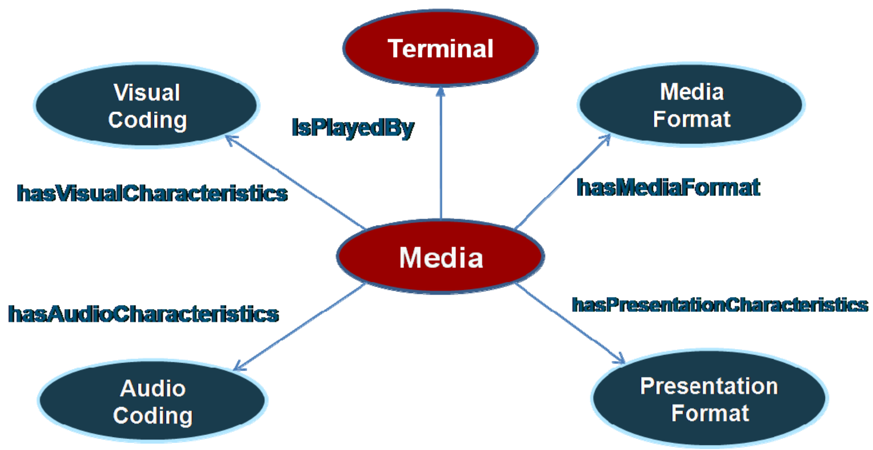


Figure 6.6 – Multimedia concept overview

6.2 CAO Ontology development process

Following from the design strategy presented in the previous section, the next step is to implement the formulated ontology using the OWL-DL sublanguage. In this section, we outline some relevant parts of the ontology development process.

Regarding our OWL editor tools and reasoners applications survey presented in Section 5, we decided to develop the CAO ontology using the Protégé-OWL editor and the reasoner Pellet. Figure 6.7 represents the platform used, merging the OWL editor, the Pellet reasoner and the SPARQL query panel.

The Context-Aware Ontology (CAO) is being designed for the Virtual Classroom domain, focusing the interoperability that can be achieved by specifying the relevant knowledge about a user, terminal, network and natural environment associated to the user and the multimedia content being processed in the virtual classroom session.

As discussed in Section 3.5.2, several types of general vocabularies and ontologies for user, place and environment description were investigated for the integration and reusing on the Virtual Classroom domain, but none of them as proved to be reasonable and worth to reuse. Therefore we will be starting to develop the ontology from scratch.

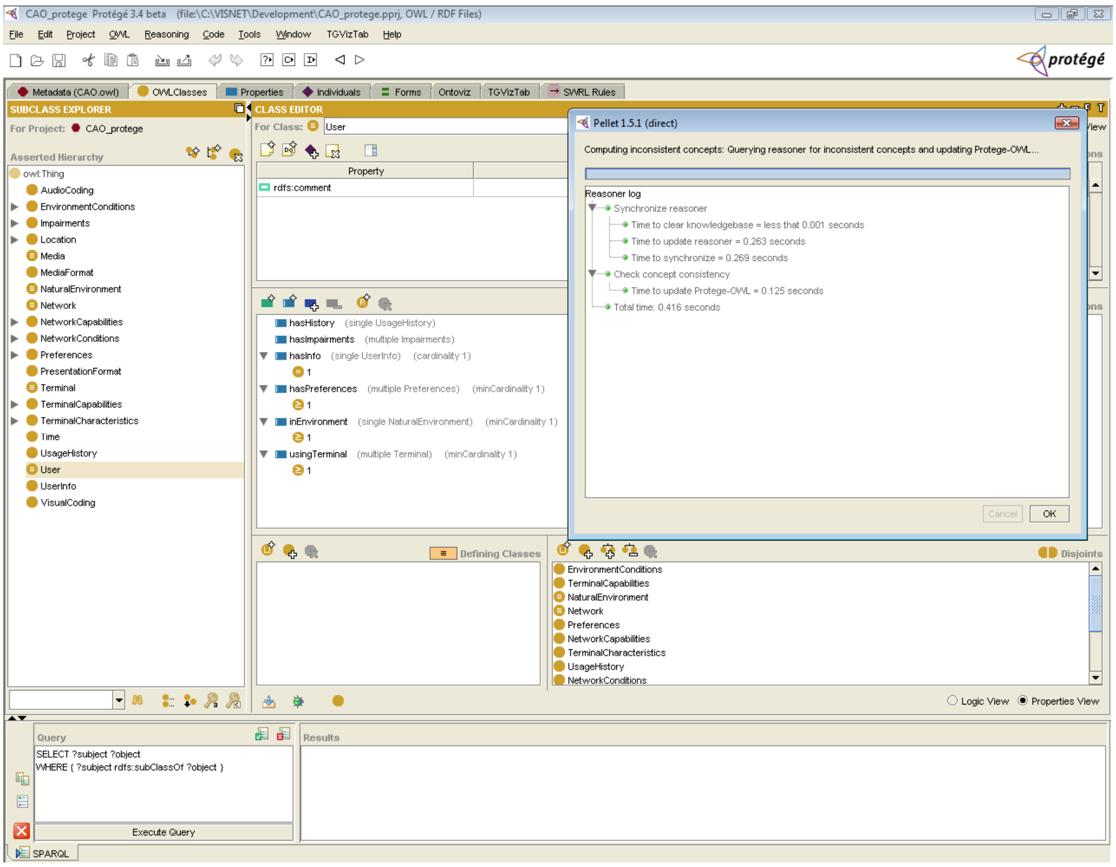


Figure 6.7 - Protégé-OWL and integrated Pellet editor platform

The more adequate process for developing the class hierarchy for the CAO ontology is similar to the top-down approach presented in Section 3.4. This way, we have started by defining the classes based on the main concepts presented in the previous section (6.1) and subsequently constructed the profiles descriptions, i.e., the concept subclasses and corresponding properties (object and datatype properties). The defined class hierarchy can be fully analysed in Table 6.1. Every class is constructed as disjoint with the same level classes so that an individual (or object) cannot be an instance of more than one class.

Table 6.1 – Class asserted hierarchy

- owl:Thing
  - AudioCoding
  - EnvironmentConditions
  - Impairments
  - Location
  - Media
  - MediaFormat
  - NaturalEnvironment
  - Network
  - NetworkCapabilities
  - NetworkConditions
  - Preferences
  - PresentationFormat
  - Terminal
  - TerminalCapabilities
  - TerminalCharacteristics
  - Time
  - UsageHistory
  - User
  - UserInfo
  - VisualCoding

- EnvironmentConditions
  - AudioEnvironment
  - IlluminationEnvironment
- Impairments
  - AuditoryImpairments
  - ColorVisionDeficiency
  - VisualImpairments
- Location
  - LogicalLocation
  - PhysicalLocation
- NetworkCapabilities
  - Error
  - MaximumCapacity
  - MinimumGuaranteed
- NetworkConditions
  - AvailableBandwidth
  - BER
  - Delay

- Preferences
  - AudioPreferences
  - ConversionPreferences
  - GraphicsPreferences
  - LanguagePreferences
  - PresentationPreferences
  - UsagePreferences
  - VideoPreferences
- TerminalCapabilities
  - CodecCapabilities
    - AudioCapabilities
    - VideoCapabilities
  - DisplayCapabilities
    - Resolution
    - Screen
- TerminalCharacteristics
  - CPU
  - DeviceClass
  - MemoryCharacteristics
  - PowerCharacteristics
  - Storages

Figure 6.8 and Table 6.2 represent the object properties and datatype properties hierarchy, respectively, while Figure 6.9 presents the object properties characteristics, domain and range attributes, that we may assign. Specifically in the presented case, the *usingTerminal* property is defined as inverse functional (Section 3.3.3.2), where its inverse property is *terminalUsedBy*. The domain and range defined for this property are respectively the classes *User* and *Terminal*.

As we may inspect in Table 6.2, properties (object or datatype) may also be constructed in a subsumption hierarchy, where sub properties inherit the aspects from the super properties. Thus, sub properties can be considered to be a specialisation of the upper class. Table 6.2 shows only a resume of the datatype properties hierarchy, while Table A.1in Appendix A represents all the available datatype properties in CAO ontology.

Note that for datatype properties, assigning the domain is done in a similar way as for object properties, while the range is chosen from a set of xml value datatypes.Furthermore, the only property characteristic available for datatype properties is the functional property.

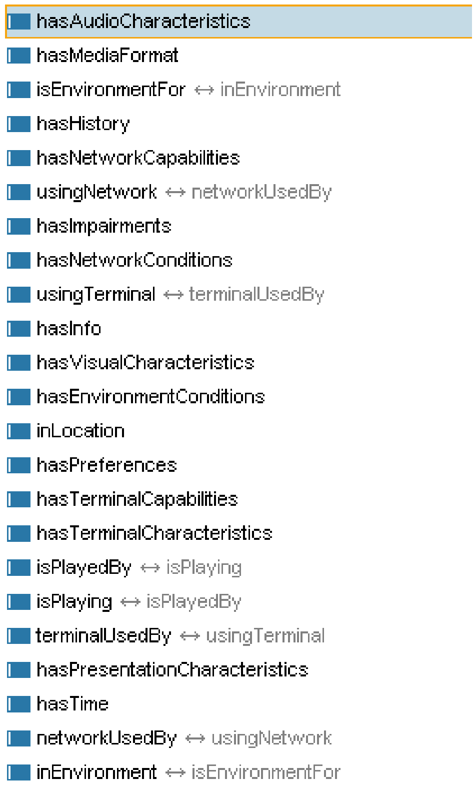


Figure 6.8 – Object properties hierarchy

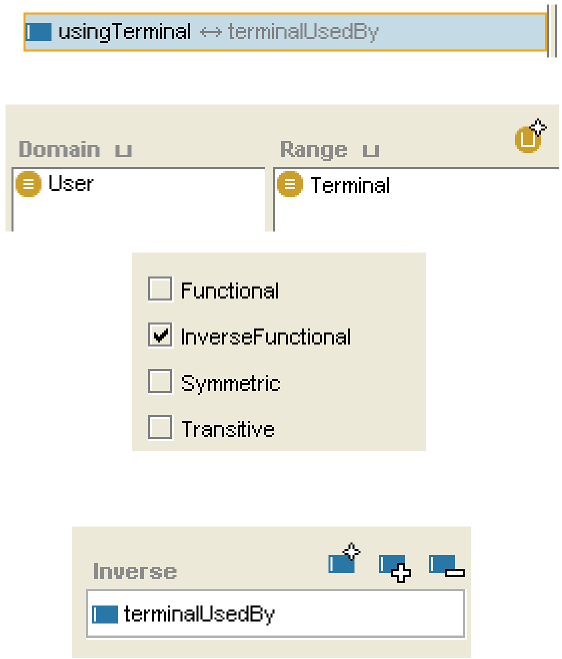
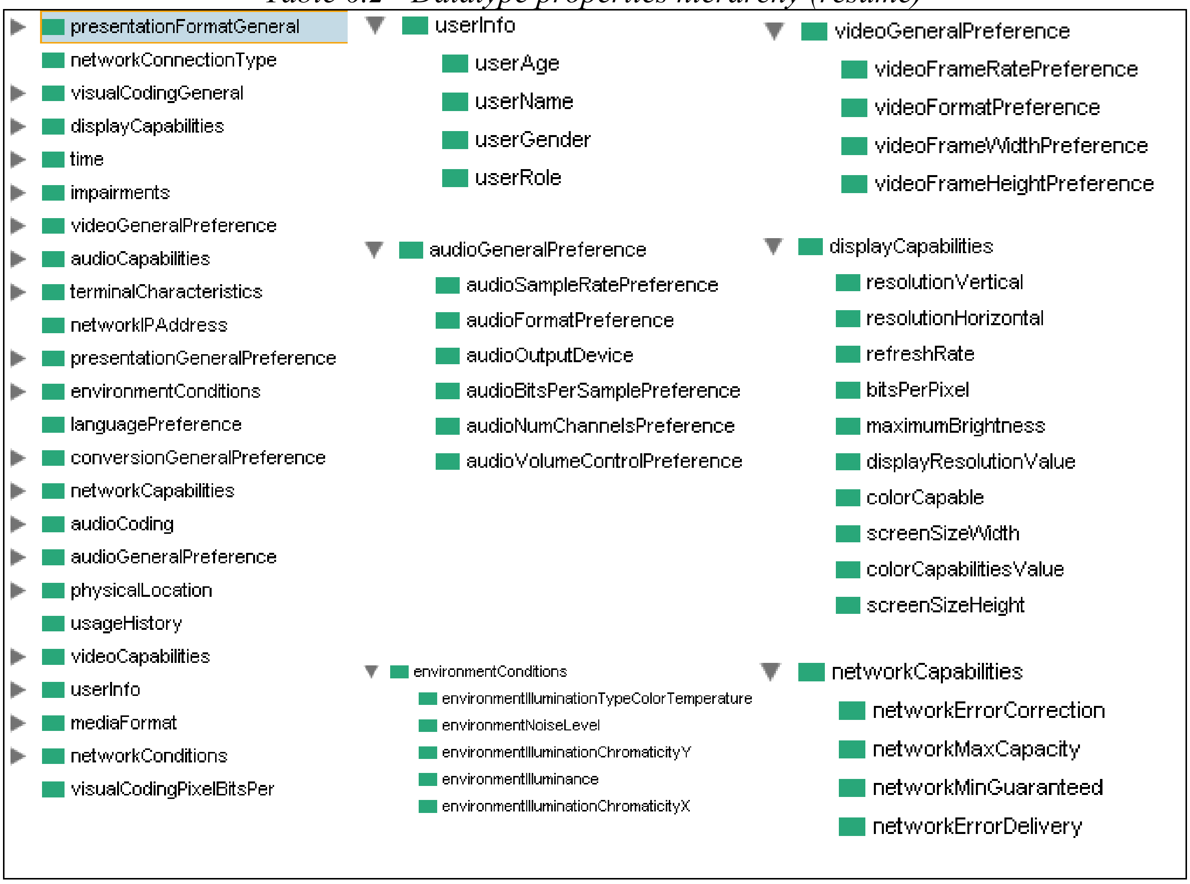


Figure 6.9 – Object property functionalities

Table 6.2 - Datatype properties hierarchy (resume)



The definition of the value datatypes for each datatype property has been done by a thoroughly study of the MPEG-21 DIA and MPEG-7 MDS schemas, which are publicly available on [53].

The result of this development process can be visualised in Figure 6.10. In this figure the classes are presented in a descriptive way while the relations between these classes are show in a blue line. Figure 6.11 presents the main information about the five main concepts described in Section 6.1, divided in datatype properties and the relations between these concepts (object properties). For better visualisation and readability please consider analysing Appendix B.

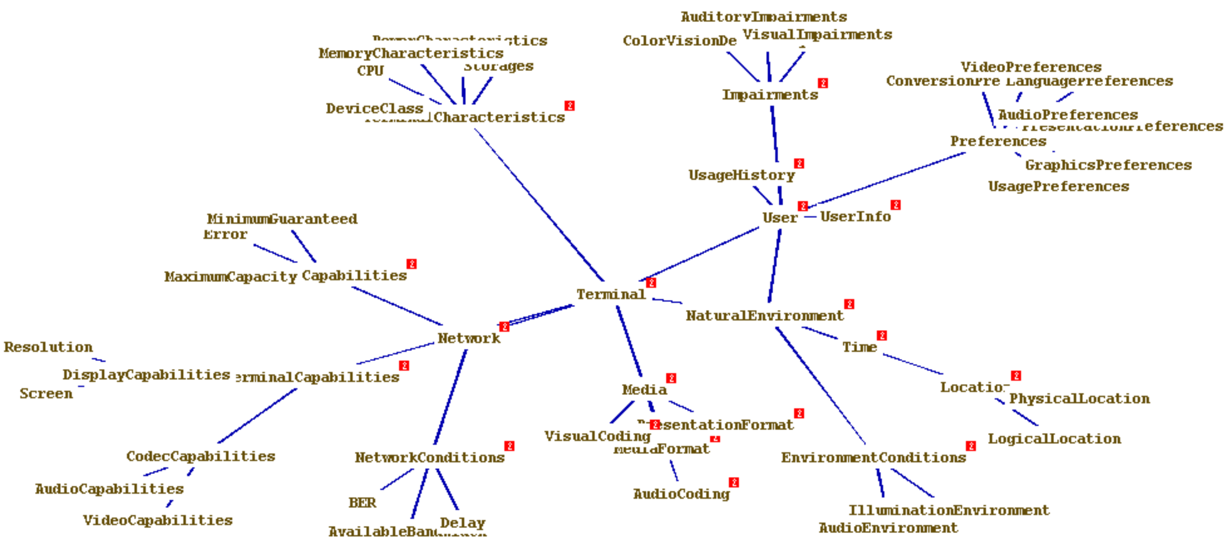


Figure 6.10 – CAO ontology overview

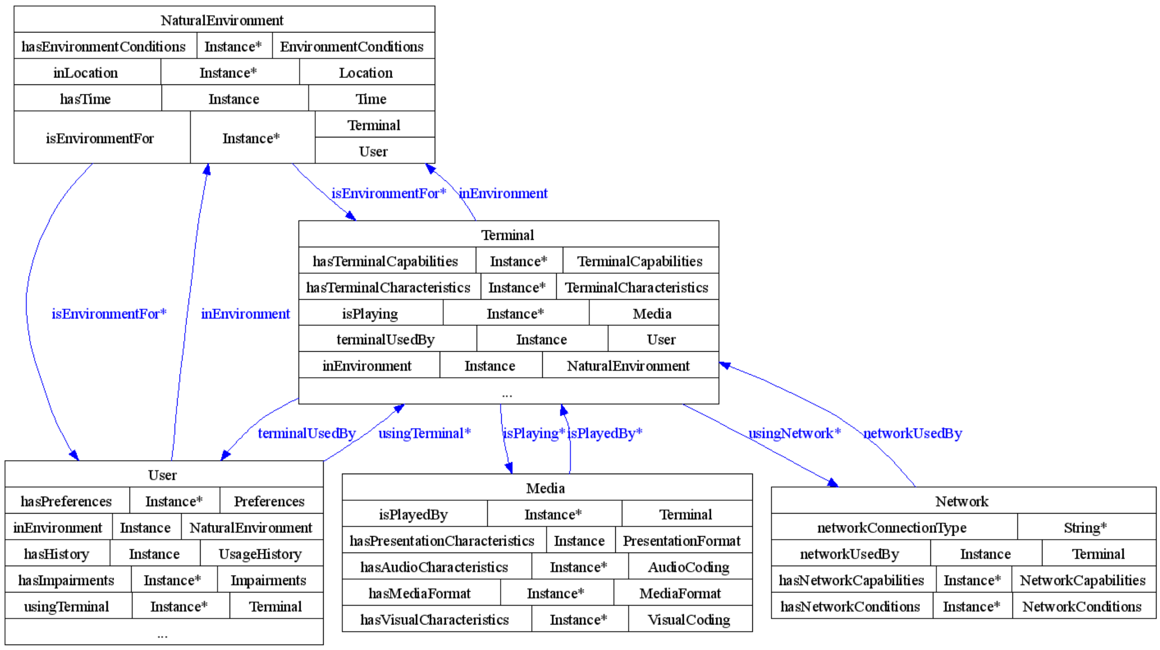


Figure 6.11 – CAO ontology main concepts

The final phase of the ontology development respects to creating individual instances of classes in the hierarchy. Adopting the virtual classroom selected scenario application and complying with the Use Case 2 defined in Section 4.3, contextual information data was inserted manually into the created instances. This step has proven to be very helpful in the CAO ontology development process because it enabled to point out some possible adjustments in the preceding steps, i.e., in the class and properties definition. Therefore, the development process became an iterative one.

The finalised CAO ontology consists of 52 classes, 23 object properties 138 datatype properties. It has also 33 subclass axioms and 5 equivalent classes axioms. The full code is available on Appendix C.1 while **Erro! Auto-referência de marcador inválida.** represents a partial code extracted from the developed CAO OWL ontology, including the definition of the class "*User*", and the datatype properties "*audioNumChannelsPreference*" and "*visualCodingFrameRate*".

Table 6.3 - Partial code from CAO ontology

```
<owl:Class rdf:ID="User">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="#hasInfo"/>
           <owl:cardinality rdf:datatype="&xsd;int">1</owl:cardinality>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#hasPreferences"/>
          <owl:minCardinality rdf:datatype="&xsd;int">1</owl:minCardinality>
          </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#inEnvironment"/>
          <owl:minCardinality rdf:datatype="&xsd;int">1</owl:minCardinality>
          </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#usingTerminal"/>
          <owl:minCardinality rdf:datatype="&xsd;int">1</owl:minCardinality>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
<owl:DatatypeProperty rdf:ID="audioNumChannelsPreference">
  <rdfs:domain rdf:resource="#AudioPreferences"/>
  <rdfs:range rdf:resource="&xsd;int"/>
  <rdfs:subPropertyOf rdf:resource="#audioGeneralPreference"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="visualCodingFrameRate">
   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
   <rdfs:domain rdf:resource="#VisualCoding"/>
   <rdfs:range rdf:resource="&xsd;float"/>
   <rdfs:subPropertyOf rdf:resource="#visualCodingGeneral"/>
   <rdfs:comment xml:lang="en"
      >Indicates the frame rate in Hz.</rdfs:comment>
</owl:DatatypeProperty>
```

6.2.1 Notational Conventions

According to the OWL Web Ontology Language XML Presentation Syntax [54] a number of namespaces prefixes are used in the ontology and are presented in Table 6.4.

As in any XML document, the choice of any namespace prefix is arbitrary, and not semantically significant. This way, we define the namespace prefix for the Context-Aware Ontology as "cao".

Table 6.4 - OWL Web Ontology Language XML Presentation Syntax

Prefix	**Namespace**	**Notes**
rdf	`"http://www.w3.org/1999/02/22-rdf-syntax-ns#"`	The namespace of the RDF in XML syntax
rdfs	`"http://www.w3.org/2000/01/rdf-schema#"`	The namespace of the RDF Schema
xsd	`"http://www.w3.org/2001/XMLSchema#"`	The namespace of the XML Schema
owl	`"http://www.w3.org/2002/07/owl#"`	The namespace of OWL in RDF/XML syntax
cao	`"http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl#"`	The namespace of the proposed CAO owl document in RDF/XML ABBREV syntax

To encourage reusability, the CAO OWL schema should be maintained on a centralised repository and accessible via Web. Using the OWL import functionality, the schema can then be referenced from the ontology application using the `owl:imports` construct. Assuming that the CAO OWL model schema is available on the address http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl, the model can then be imported using:

Table 6.5 - OWL import functionality

```
<owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl"/>
</owl:Ontology>
```

6.3 Application development

This section contains a preview of the implemented application which is responsible for the instantiation of the contextual information data extracted from the MPEG-21 DIA UED and MPEG-7 MDS descriptors, into the CAO scheme ontology. This application has been designed in Java using the J2SE 1.6 and developed in Eclipse platform.

This application was implemented using the Protégé-OWL API [34] (Section 5.1.1) which incorporates the API from Jena framework [35] (Section 5.1.2) providing functionalities in a transparent way for the developer.

MPEG-21 DIA UED and MPEG-7 MDS descriptors files were generated based on the Use Case 2 providing the required data to instantiate individuals into the CAO ontology. Table 6.6 and Table 6.7 represent the created MPEG-21 and MPEG-7 files, respectively.

Table 6.6 - Partial MPEG-21 DIA UED code

```
<Description xsi:type="UsageEnvironmentType">
      <UsageEnvironmentProperty xsi:type="NetworksType">
      <Network>
            <NetworkCharacteristic            xsi:type="NetworkCapabilityType"
maxCapacity="512000"        minGuaranteed="32000"        errorCorrection="true"
errorDelivery="false"/>
            <NetworkCharacteristic xsi:type="NetworkConditionType">
            <AvailableBandwidth maximum="256000" average="128000"/>
            <Delay packetTwoWay="330" delayVariation="66"/>
            <Error packetLossRate="0.05"/>
            </NetworkCharacteristic>
      </Network>
      </UsageEnvironmentProperty>
      <UsageEnvironmentProperty xsi:type="NaturalEnvironmentsType">
      <NaturalEnvironment>
            <NaturalEnvironmentCharacteristic xsi:type="AudioEnvironmentType">
                  <NoiseLevel>80</NoiseLevel>
            </NaturalEnvironmentCharacteristic>
            <NaturalEnvironmentCharacteristic
xsi:type="IlluminationCharacteristicsType">
                  <TypeOfIllumination>
                        <ColorTemperature>159</ColorTemperature>
                  </TypeOfIllumination>
                  <Illuminance>500</Illuminance>
            </NaturalEnvironmentCharacteristic>
      </NaturalEnvironment>
      </UsageEnvironmentProperty>
</Description>
```

Table 6.7 - Partial MPEG-7 MDS code

```
<MediaInformation id="lecture_media">
      <MediaProfile>
            <MediaFormat>
                  <Content href="MPEG7ContentCS">
                        <Name>audiovisual</Name>
                  </Content>
                  <FileFormat href="urn:mpeg:mpeg7:cs:FileFormatCS:2001:3">
                        <Name xml:lang="en">mpeg</Name>
                  </FileFormat>
                  <VisualCoding>
                        <Format
href="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:1" colorDomain="color">
                              <Name xml:lang="en">MPEG-4</Name>
                        </Format>
                        <Pixel aspectRatio="0.75" bitsPer="8"/>
                        <Frame height="144" width="174" rate="25"/>
                  </VisualCoding>
```

```
                    <AudioCoding>
                          <Format href="MPEG7FileFormatCS">
                                <Name>MPEG-1 Layer II</Name>
                          </Format>
                          <AudioChannels>1</AudioChannels>
                          <Sample rate="44100"/>
                    </AudioCoding>
              </MediaFormat>
        </MediaProfile>
</MediaInformation>
```

The following subsection provides the insights of the integration of the extracted data in the new constructed ontology for representing this instances data.

6.3.1 CAO OWL Ontology and Data

In the standard use of OWL for ontology modelling, two different ways of storing data emerge as a possibility. The ontology scheme and data may co-exist in same OWL documents or, in other cases, the ontology is defined and stored in a central OWL library and referenced in the OWL data document using the external namespace reference.

For our purposes, the latter seems more reasonable due to the dimension of the developed CAO ontology. Thus, the previously described CAO ontology is used for the conceptualisation of the scenario application domain providing the taxonomy definition for the classes, properties and restrictions, while a second ontology will be created to include the individual instances and associated contextual information data. This data ontology will reference the CAO ontology by its external reference namespace, as introduced in Section 6.2.1, enabling reasoning services such as consistency checking, concept satisfiability and classification.

6.3.2 Data Ontology Implementation

This section will present some of the important steps in the data ontology implementation phase considering the developed application.

The implemented Java application constructs the data ontology by creating an OWL model and ordering the import of the CAO ontology as represented in Table 6.8.

Table 6.8 - Importing CAO ontology

```
JenaOWLModel owlModel = ProtegeOWL.createJenaOWLModel();
owlModel = OWLUtils.importOWL(owlModel, uri_ns, uri_impns, uri_imp,
prefix);
```

The contextual information is extracted from the files inserted into the application, requiring the parsing of MPEG-7 MDS and MPEG-21 DIA xml files.

The MPEG-7 MDS file parsing, due to its relative easier schema definition and smaller size of descriptors, was implemented manually in the Java application through the use of XML Document Object Model (DOM) parsing facilities.

The parsing of MPEG-21 was done with the reuse of the reference software defined in *MPEG-21 Part 8 – Reference Software*, which is publicly available on [53]. This reference software framework was edited in order to suit the data extraction and application needs.

The contextual information associated to the Media concept is extracted from MPEG-7 MDS file and the contextual information associated to the User, Terminal, Network and Natural Environment concepts is extracted from MPEG-21 DIA UED. This step is reported in Table 6.9 where the mentioned MPEG files can be previewed in Table 6.6 and Table 6.7.

Table 6.9 - MPEG-21 and MPEG-7 parsing

```
Document xmldoc = XMLUtils.createDOMDocURI(file_xml_mpeg7, false);
MPEG7 mpeg7info = ParseMPEG7.processMediaProfile(xmldoc);
String[] argstring = {"-in "+file_xml_mpeg21};
MPEG21UED mpeg21info = ParseMPEG21UED.processUEDInfo(argstring);
```

The CAO application has taken some benefits of an extremely important feature from the Protégé-OWL main application, as it can generate Java code which models the ontology scheme created. This code, when used with the Protégé-OWL API, provides the necessary methods for the creation of instances and the setup of instances data. This way, the code for the CAO ontology has been previously generated and imported into the implemented application, enabling an easier instantiation of the data ontology.

Table 6.10 presents the methods used for the individuals' instantiation and the setup of the data associated to each concept profile.

The final result is then saved in an OWL file as shown in Table 6.11. The full code can be analysed in Appendix C.2.

Following the instantiation process, the CAO application automatically invokes the Protégé-OWL main application. This feature enables the visualisation of the constructed CAO Data ontology and some services become available. The user has the possibility to use the integrated reasoner to validate or check inconsistencies on the created data ontology but also to make queries on the ontology with the SPARQL query panel. Table 6.11 shows the code for the application call and the possibilities for the analysis on the created data ontology are illustrated in Figure 6.12.

Table 6.10 – Instantiation and data mining

```
CAO cao = new CAO(owlModel);
cao = OWLUtils.joinAllInfo(cao,mpeg7info,mpeg21info,userInfo,extra);

(...)

public static CAO joinAllInfo(CAO cao, MPEG7 mpeg7info, MPEG21UED
mpeg21info, UserAppInfo userappinfo, ExtraInfo extra) {

      String suffix = "UseCase2";

      Network network = cao.createNetwork("Network_"+suffix);
      Terminal terminal = cao.createTerminal("Terminal_"+suffix);
      NaturalEnvironment natEnv =
cao.createNaturalEnvironment("NaturalEnvironment_"+suffix);
      User user = cao.createUser("User_"+suffix);
      UserInfo userinfo = cao.createUserInfo("UserInfo_"+suffix);
      Media media = cao.createMedia("Media_"+suffix);

      cao = joinNetorkInfo(cao, mpeg21info, network, extra, suffix);
      cao = joinNaturalEnvironmentInfo(cao, mpeg21info, natEnv, extra,
suffix);
      cao = joinTerminalInfo(cao, mpeg21info, terminal, user, natEnv,
network, media, suffix);
      cao = joinUserCharacteristicsInfo(cao, mpeg21info, user,
userinfo, userappinfo, natEnv, suffix);
      cao = joinMediaInfo(cao, mpeg7info, media, terminal, extra,
suffix);

      return cao;
}
```

Table 6.11 – CAO data ontology save feature and Protégé application call

```
static String fileName = "docs/project/CAO-saved.owl";
owlModel.save(new   File(fileName).toURI(),   FileUtils.langXMLAbbrev,
errors);

(...)

project = new Project(fileNamePprjAux,errors);
project.setProjectFilePath(fileNamePprj);
String[] argsprotege = {""+fileNamePprj};
Application.main(argsprotege);
```

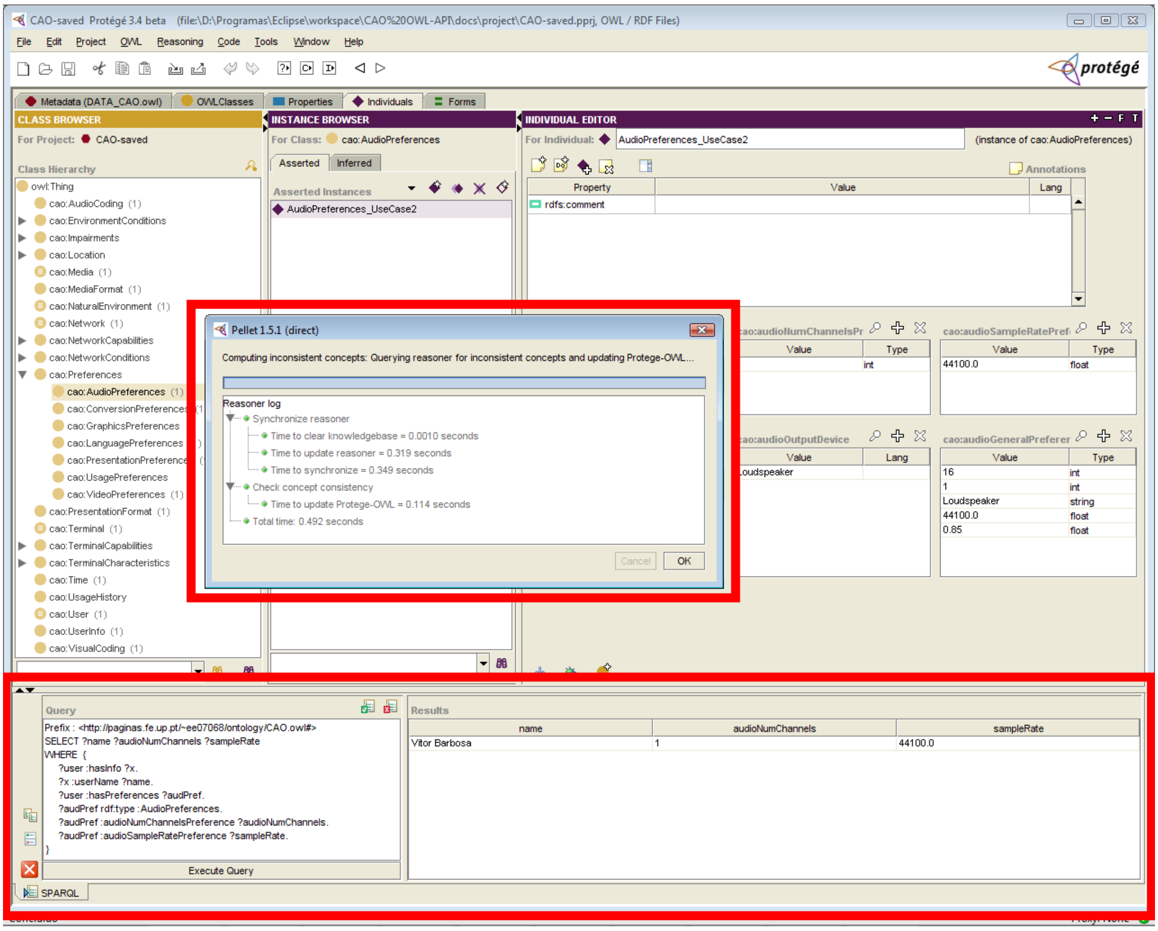


Figure 6.12 – CAO Data ontology post-processing

6.4 CAO Ontology Results

The developed CAO ontology has enabled the conceptualisation of the context-aware virtual classroom domain in a formal and efficient way, enabling to share, reuse and instantiate contextual information data related to the user (learner) in a virtual classroom session.

The implemented CAO Data application has provided the means to construct a data ontology and instantiate individuals, resulting in a valid and informative ontology.

The Adaptation Decision Engine (ADE) can thus use the resulting data ontology to query and infer the user satisfaction and preferences, media information, terminal characteristics and capabilities, network conditions and surrounding natural environment, in order to perform a well-informed decision to select the most appropriate adaptation and corresponding service parameters, maximising the user quality of experience.

Table 6.12 presents a query on the resulting data ontology using the SPARQL query language and the results are presented in Figure 6.13.

Table 6.12 – SPARQL query

```
Prefix : <http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl#>
SELECT ?name ?audioNumChannels ?sampleRate
WHERE  {
     ?user :hasInfo ?x.
     ?x :userName ?name.
     ?user :hasPreferences ?audPref.
     ?audPref rdf:type :AudioPreferences.
     ?audPref :audioNumChannelsPreference ?audioNumChannels.
     ?audPref :audioSampleRatePreference ?sampleRate.
}
```

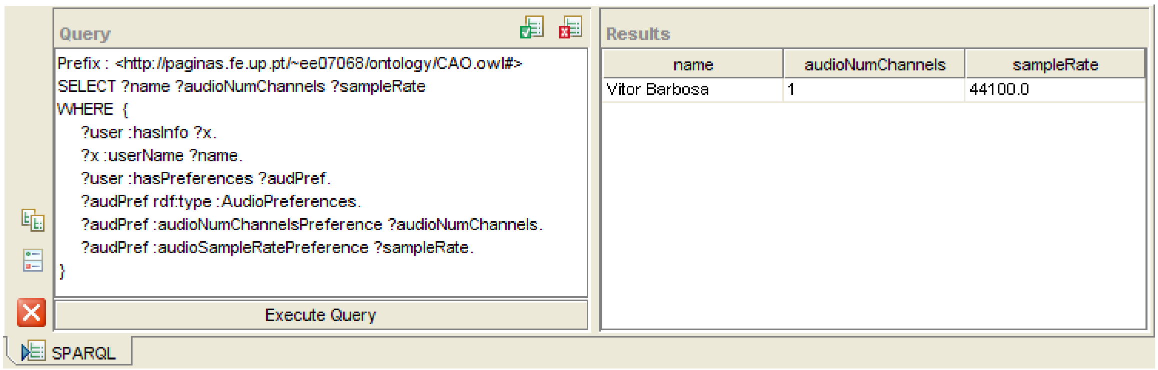


Figure 6.13 – SPARQL query results

Chapter 7

7. Conclusions

Ontologies are key requirements for building pervasive context-aware systems, in which independently developed sensors, devices and agents are expected to share contextual information.

In summary, I have presented a formal context-aware ontology model develop using the OWL language, which is the W3C recommended technology to achieve the Semantic Web goal. The developed Context-Aware Ontology (CAO) as provided the means to represent, access and query contextual information associated to the use cases defined for the VISNET-II selected application scenario, the virtual classroom.

The Context-Aware Ontology (CAO) consists of five fundamental context entities: (1) user, the central concept in context-aware computing, (2) terminal, the characteristics and capabilities of the device through which a user interacts with the application, (3) network, the characteristics and current conditions of the network through which a user connects to the desired functionalities, (4) natural environment, the description of relevant aspects of the user`s surroundings and (5) media, the description of technical features of the consumed multimedia content. The design of these concepts has been based on the MPEG-21 DIA UED and MPEG-7 MDS description standards, in a way to achieve the necessary interoperability among different systems.

The prototype application developed for the automatic extraction of contextual information and contextual data instantiation into a separated OWL data ontology demonstrate an expert mechanism to be used within the Adaptation Decision Engine module, enabling the use of rules to infer high-level concepts and thus assist the adaptation decision operation. Therefore, the implemented application has proved to meet the requirements of a context-aware system and is capable to assist the selection of the best possible operation point.

With the growing adoption of OWL to construct ontologies, the gradual realisation of the Semantic Web vision and the use of MPEG-21 standard to describe the contextual information, we foresee that this work is instrumental in bridging the gap between being able to understand and use context and evaluate the quality of the result achieved by the adaptation mechanisms.

This project allowed me to become familiar with the current state of context-aware applications and ontologies development, an area experiencing increased research activity in a bid to achieve the Semantic Web activity goals and endeavour the conceptual design of an interoperable system for context-aware content adaptation. The proposed goals were entirely fulfilled although the strict time constraints bounded to the project.

The completion of this project as opened up other research issues which I hope to be exploring in the future.

Within VISNET-II virtual classroom application scenario, intellectual property and digital rights are managed in the service initialisation but also during the lecture session. Therefore, the Adaptation Decision Engine should also be aware of the constraints referred in the license during the adaptation decision taking process. This will force the creation of a novel but fundamental concept in the ontology model, in order to instantiate the newly contextual information associated to the constraints, conditions, and possible adaptation operations.

Furthermore, a number of relevant emerging standards have been recently recommended by The World Wide Web Consortium (W3C), including Semantic Web Rule Language (SWRL) and SPARQL Query Language (SPARQL). The integration of these newly standards in the developed Context-Aware Ontology (CAO) will enable to develop a more comprehensive rule-based ontology model to reason about the low-level contextual information and represent as accurately as possible the real-world situations in virtual-classroom applications.

References

[1] Dey, Anind K. (2001). *Understanding and Using Context,* Personal Ubiquitous Computing 5 (1): 4-7.
[2] T. Berners-Lee, J. Hendler, and O. Lassila, The Semantic Web, Scientific American, May 2001.
[3] W3C, *W3C Semantic Web Activity*. [Online] Available on http://www.w3.org/2001/sw/ .
[4] H. Chen, F. Perich, T.W. Finin and A. Joshi. "SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications", *MobiQuitous 2004, 1st Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services*, pp. 258–267, Cambridge, MA, 2004.
[5] W. Pratt, A. Faquhar, R. Fikes, and J. Rice, "Collaborative Ontology construction for information integration". Research Report KSL-95-63, Stanford University, August 1995.
[6] Wang, X.H.; Zhang, D.Q.; Gu, T.; Pung, H.K.; Ontology Based Context Modeling and Reasoning using OWL, in Pervasive Computing and Communications Workshops, March 2004.
[7] Extensible Markup Language (XML) 1.0 (Fourth Edition), W3C, September 2006. [Online] Available on http://www.w3.org/TR/xml/ .
[8] RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, February 2004. [Online] Available on http://www.w3.org/TR/rdf-schema/ .
[9] RDF Schema Example, XMLJ – Programming XML in Java, Spring 2002. [Online] Available on http://www.cs.rpi.edu/~puninj/XMLJ/classes/class8/slide35-0.html .
[10] OWL Web Ontology Language: Guide, W3C Recommendation, February 2004. [Online] Available on http://www.w3.org/TR/owl-guide/ .
[11] D. Fensel, F. van Harmelen, I. Horrocks, D. L. McGuinness. P. F. Patel-Schneider, "*OIL: An Ontology Infrastructure for the Semantic Web*", IEEE Intelligent Systems, vo. 16, nr. 2, 2001. [Online] Available on http://www.cs.man.ac.uk/~horrocks/Publications/download/2001/IEEE-IS01.pdf
[12] T. Berners-Lee, *et a.l*, DAML+OIL, March 2001. [Online] Available on http://www.daml.org/2001/03/daml+oil-index .
[13] SPARQL Query Language for RDF, W3C Recommendation, January 2008. [Online] Available on http://www.w3.org/TR/rdf-sparql-query/
[14] Moving Picture Experts Group, *MPEG-7 Overview*, October 2004. [Online] Available on http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm .
[15] *Multimedia Content Description Interface - Part 5: Multimedia Description Scheme*, ISO/IEC Standard, MPEG-7 Part-5: ISO/IEC 15938-5, 2003.
[16] *Information Technology – Multimedia Framework (MPEG-21) – Part 1: Vision, Technologies and Strategy*, ISO/IEC Standard, ISO/IEC 21000-1:2001, December 2001.
[17] *Information Technology – Multimedia Framework (MPEG-21) – Part 7: Digital Item Adaptation*, ISO/IEC Standard, ISO/IEC 21000-7:2004, October 2004.
[18] *Introducing MPEG-21 DIA – an Overview, Multimedia Description Scheme Group*, April 2005, [Online]. Available on http://www.chiariglione.org/mpeg/tutorials/technologies/mp21-dia/index.htm .
[19] Horridge, Matthew; Knublauch, Holger; Rector, Alan; Stevens, Robert and Wroe, Chris; *A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools*, Edition 1.0. The University Of Manchester, August 27, 2004. [Online] Available on http://www.co-ode.org/resources/tutorials/protege-owl-tutorial.php .
[20] W3C, OWL Web Ontology Language Semantics and Abstract Syntax, February 2004. [Online] Available on http://www.w3.org/TR/owl-semantics/ .
[21] Uschold, M. and Gruninger, M., *Ontologies: Principles, Methods and Applications.* Knowledge Engineering Review 11, 1996
[22] Nilsson, Mikael; Powell, Andy; Johnston, Pete; Naeve, Ambjorn; *Expressing Dublin Core metadata using the Resource Description Framework (RDF),* Dublin Core Meta Data Initiative, January 14, 2008, [Online]. Available on http://dublincore.org/documents/dc-rdf/ .
[23] DCMI Usage Board, *DCMI Metadata Terms*, Dublic Core Meta Data Initiative, January 14, 2008, [Online]. Available on http://dublincore.org/documents/dcmi-terms/ .
[24] IEEE P1484.2.2/D8, *Draft Guide for Learning Technology — Public and Private Information (PAPI) for Learners (PAPI Learner)*, October 25, 2001.
[25] IMS, *IMS Learner Information Package*, Version 1.0.1 Final Specification, January 04, 2005.
[26] W3C Recommendation, Composite Capability/ Preference Profiles (CC/PP): Structure and Vocabularies 1.0, Standard, 15 January 2004. [Online] Available on http://www.w3.org/TR/CCPP-struct-vocab/ .

[27]Brickley, D., Miller, L., FOAF Vocabulary Specification 0.91 - OpenID Edition, November 2007. [Online] Available at http://xmlns.com/foaf/spec/ .

[28]Harry Chen, *Pervasive Computing Standard Ontology (PERVASIVE-SO) Guide - Describing User Profile and Preferences*, University of Maryland Baltimore County, January 2004. [Online] Available on http://pervasive.semanticweb.org/doc/2004-01-ont-guide/part1/ .

[29]W3C Semantic Web Interest Group, *Basic Geo (WGS84 lat/long) Vocabulary*, January 2003. [Online] Available on http://www.w3.org/2003/01/geo/

[30]C.T.E.R. Hewage, H. Kodikara Arachchi, T. Masterton, A.C. Yu, H. Uzuner, S. Dogan, and A.M. Kondoz, "Content adaptation for virtual office environment using scalable video coding", *Proceedings of the 16th IST Mobile and Wireless Communications Summit (IST Mobile Summit'2007)*, Budapest, Hungary, 1-5 July 2007.

[31]SOAP Version 1.2 Part 1: Messaging Framework, W3C Recommendation, 2007. [Online] Available: http://www.w3.org/TR/soap12-part1/

[32]M.T. Andrade, H. Kodikara Arachchi, S. Nasir, S. Dogan, H. Uzuner, A.M. Kondoz, J. Delgado, E. Rodriguez, A. Carreras, T. Masterton, and R. Craddock, "Using context to assist the adaptation of protected multimedia content in virtual collaboration applications", *Proceedings of the 3rd IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom'2007)*, New York, NY, USA, 12-15 November 2007.

[33]*The Protégé Ontology Editor and Knowledge Acquisition System*, Stanford Center for Biomedical Informatics Research. [Online] Available on http://protege.stanford.edu/

[34]H. Knublauch*,et al., Protégé-OWL API Programmer's Guide,* Stanford Center for Biomedical Informatics Research, September 2006. [Online] Available on http://protege.stanford.edu/plugins/owl/api/guide.html

[35]*Jena – A Semantic Web Framework for Java*, HP Labs Semantic Web Research. [Online] Available on http://jena.sourceforge.net/

[36]*Metatomix m3t4.studio Semantic Toolkit*, Metatomix, Inc. [Online] Available on http://www.m3t4.com/semantic.jsp

[37]*IBM Integrated Ontology Development Toolkit*, IBM AlphaWorks Services. [Online] Available on http://www.alphaworks.ibm.com/tech/semanticstk

[38]P. Hayes, et al., *CmapTools Ontology Editor*. [Online] Available on http://cmap.ihmc.us/coe/

[39]E. Franconi, *ICOM – A tool for Intelligent Conceptual Modelling*. [Online] http://www.inf.unibz.it/~franconi/icom/

[40]TopQuadrant, *TopBraid Composer*. [Online] Available on http://www.topbraidcomposer.com/

[41]*OilEd*. [Online] Available on http://oiled.man.ac.uk/index.shtml

[42]*Swoop – Semantic Web Ontology Editor*. [Online] Available on http://code.google.com/p/swoop/

[43]*Dumpont*. [Online] Available on http://projects.semwebcentral.org/projects/dumpont/

[44]*ORIENT – Ontology engineeRIng ENvironmenT*. [Online] Available on http://apex.sjtu.edu.cn/projects/orient/News.htm

[45]*Swede – Semantic Web Development Environment*. [Online] Available on http://owl-eclipse.projects.semwebcentral.org/

[46]*Powl – Semantic Web Development Plattform*, Agile Knowledge Engineering and Semantic Web, University of Leipzig. [Online] Available on http://ontowiki.net/Projects/Powl?v=287

[47]M. Falc, et al., *Apollo*, Knowledge Media Institute. Available on http://apollo.open.ac.uk/

[48]D. Tsarkov, et al., FaCT++, Department of Computer Science, University of Manchester. [Online] Available on http://owl.man.ac.uk/factplusplus/

[49]*Pellet: The Open Source OWL DL Reasoner*, Clark & Parsia. [Online] Available on http://pellet.owldl.com/

[50]M. Jang, *Bossam Rule/OWL Reasoner*. [Online] Available on http://bossam.wordpress.com/

[51]*KAON2 – Ontology Management for the Semantic Web*. [Online] Available on http://kaon2.semanticweb.org/

[52]B. Grosof, et. al, *SweetRules: Tools for Semantic Web Rules and Ontologies*. [Online] Available on http://sweetrules.projects.semwebcentral.org/

[53]Publicly Available Standards, ISO/IEC Standards. [Online] Available on http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

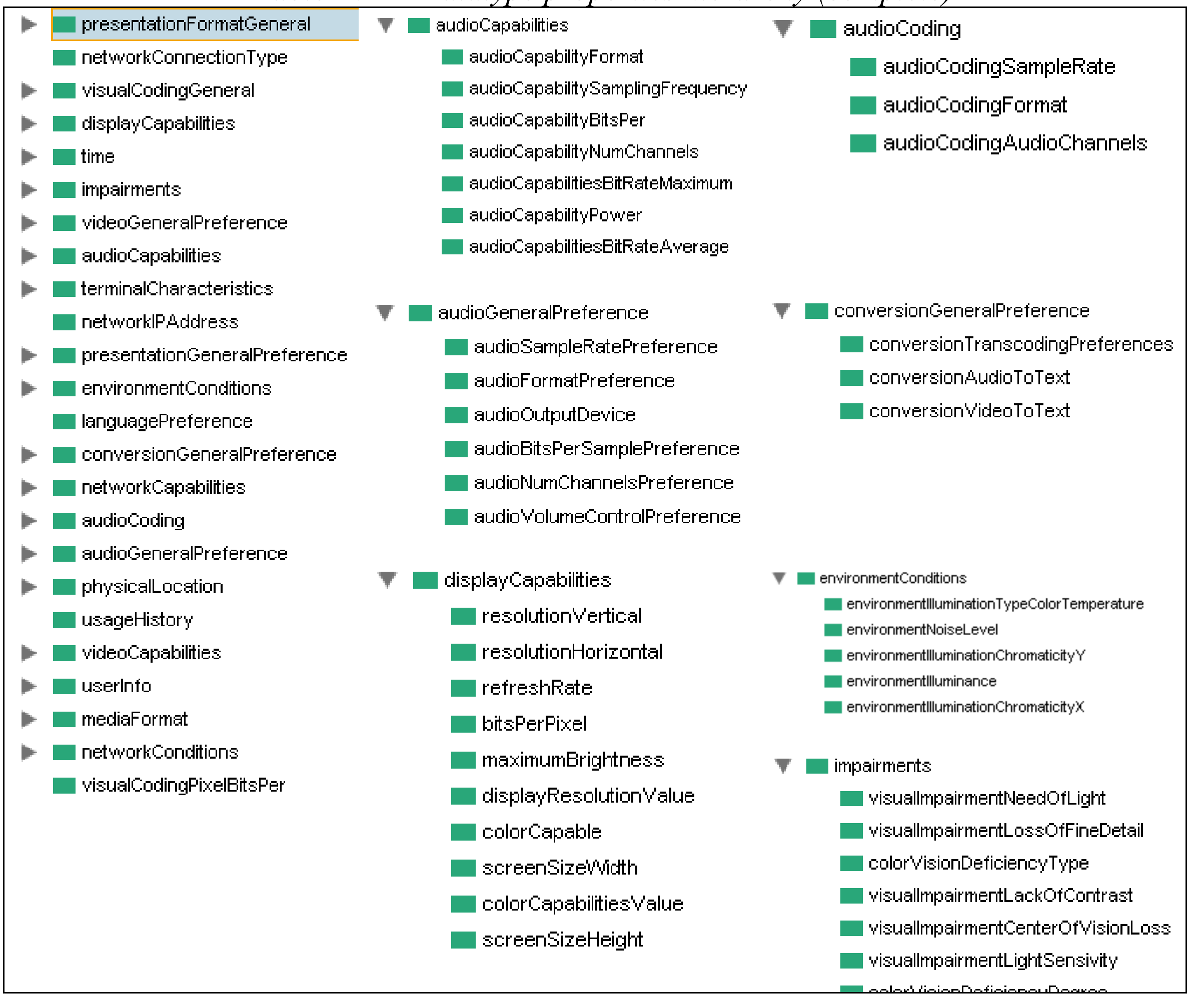
[54]M. Hori, J. Euzenat, P. F. Patel-Schneider, "OWL Web Ontology Language XML Presentation Syntax", W3C Note 11 June 2003, 2003. [Online] Available on http://www.w3.org/TR/owl-xmlsyntax/

Appendix A

A Datatype Properties Hierarchy

This appendix contains further information about the datatype properties hierarchy defined for the developed Context-Aware Ontology (CAO).

Table A.1 – Datatype properties hierarchy (complete)




mediaFormat
mediaFormatFileFormat
mediaFormatMedium
mediaFormatFileSize
mediaFormatScalableCoding
mediaFormatBitRate
mediaFormatSystem
mediaFormatContent
networkCapabilities
networkErrorCorrection
networkMaxCapacity
networkMinGuaranteed
networkErrorDelivery
networkConditions
availableBandwidthMax
networkDelayPacketTwoWay
availableBandwidthAvg
availableBandwidthMin
networkBERValue
networkDelayVariation
networkPacketLossRatio
physicalLocation
physicalLocationLongitude
physicalLocationRoom
physicalLocationLatitude
physicalLocationCountry
physicalLocationRegion
physicalLocationAddress
physicalLocationAltitude
physicalLocationBuilding
physicalLocationCity
presentationFormatGeneral
presentationFormatBrightnessLevel
presentationFormatROI
presentationFormatSubtitles
presentationFormatAudioLevel
terminalCharacteristics
powerCharacteristicsBatteryCapacityRemaining
powerCharecteristicsBatteryTimeRemaining
terminalStorageWritable
terminalStorageInputTransferRate
terminalStorageOutputTransferRate
deviceClass
cpuFrequency
cpuCache
totalMemory
terminalStorageSize
availableMemory
powerCharacteristicsRunningOnBatteries
visualCodingGeneral
visualCodingFrameWidth
visualCodingFrameHeight
visualCodingFrameAspectRatio
visualCodingFrameRate
visualCodingPixelAspectRatio
visualCodingFormat
visualCodingPixelResolution
videoCapabilities
videoCapabilitiesFormat
videoCapabilitiesBitRateAvera
videoCapabilitiesBitRateMaxim
userInfo
userAge
userName
userGender
userRole
videoGeneralPreference
videoFrameRatePreference
videoFormatPreference
videoFrameWidthPreference
videoFrameHeightPreference
time
dateStart
timeStart

Appendix B

B Context-Aware Ontology (CAO) visualisation

This appendix contains further information about Context-Aware Ontology (CAO) providing an overview of the developed classes and associated properties. The following sections represent the User, Terminal, Natural Environment, Network and Media concepts, respectively.

B.1 CAO Overview – User Concept

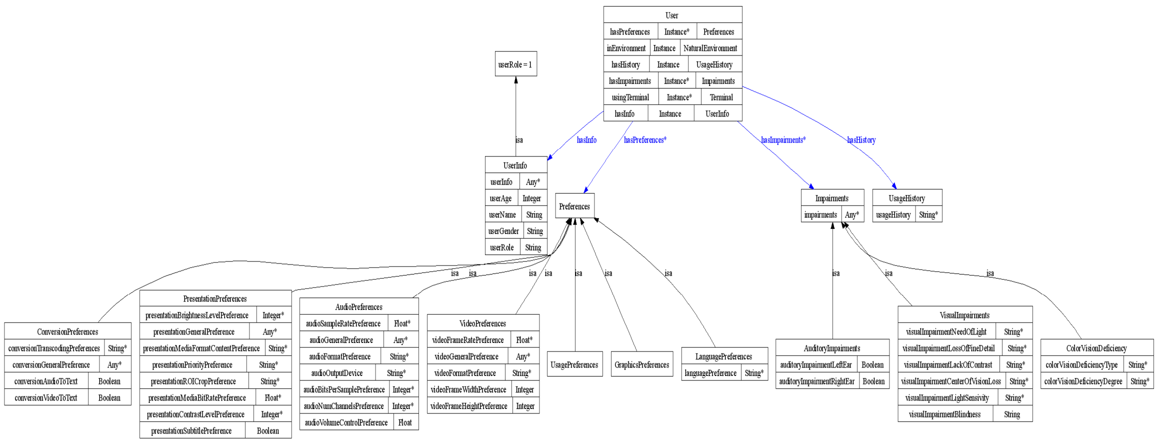


Figure B.1 – CAO ontology – User concept overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_User.gif

B.2 CAO Overview – Terminal Concept

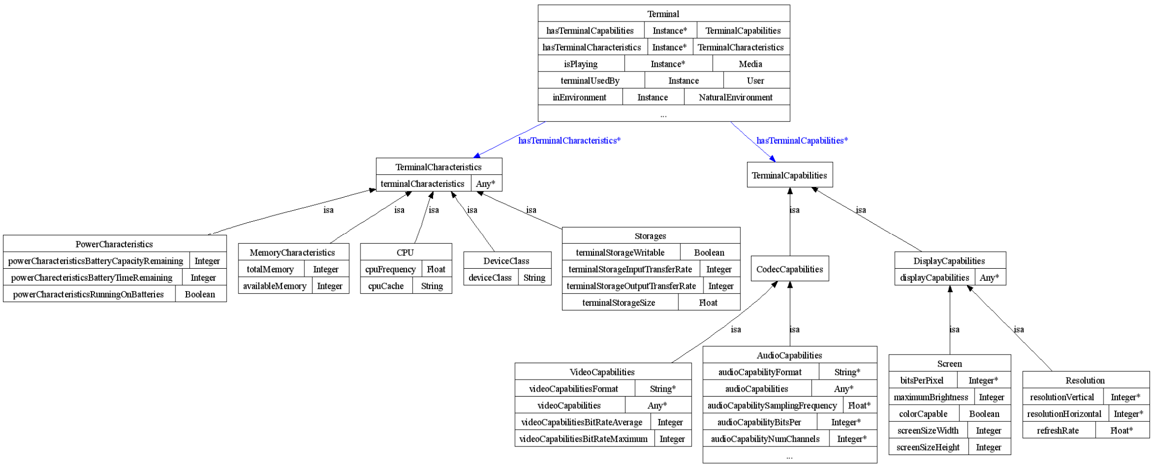


Figure B.2 – CAO Ontology – Terminal concept overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_Terminal.gif

B.3 CAO Overview – Natural Environment Concept

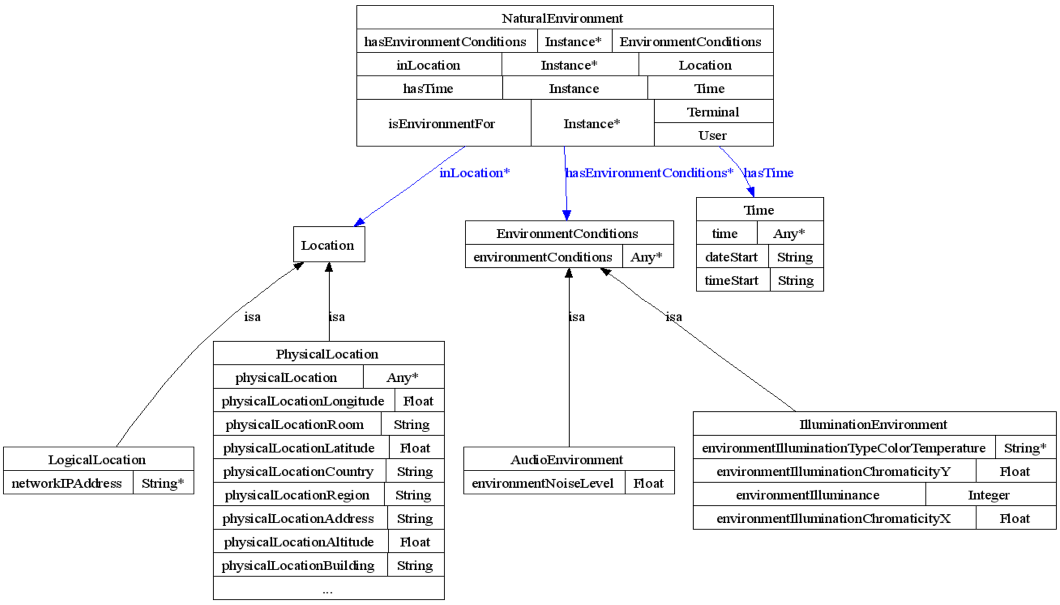


Figure B.3 – CAO Ontology – Natural Environment concept overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_NaturalEnvironment.gif

B.4 CAO Overview – Network Concept

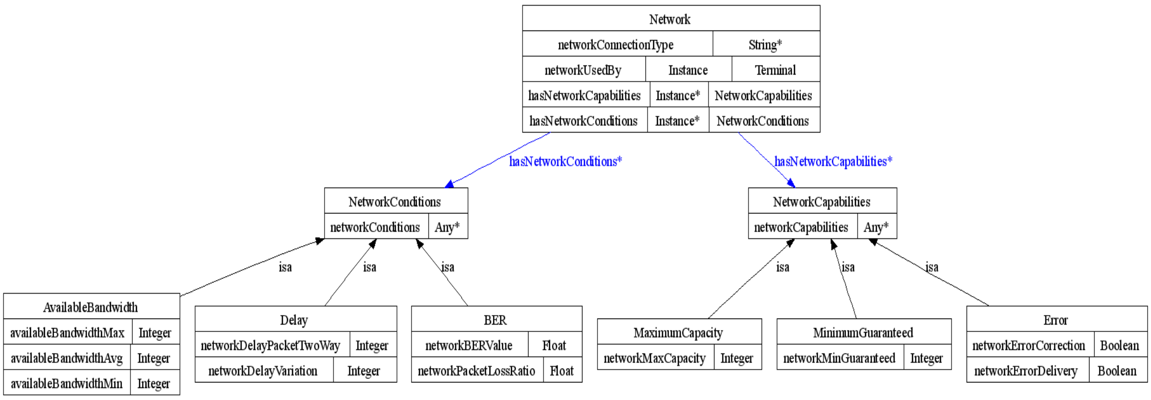


Figure B.4 – CAO Ontology – Network concept overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_Network.gif

B.5 CAO Overview – Media Concept

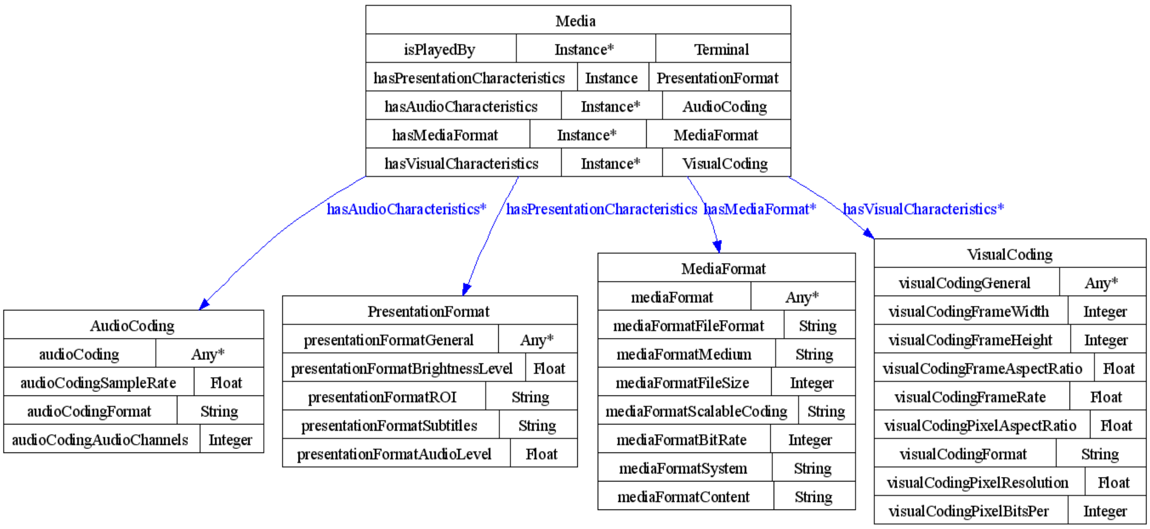


Figure B.5 – CAO Ontology – Media concept overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_Media.gif

B.6 CAO General Overview

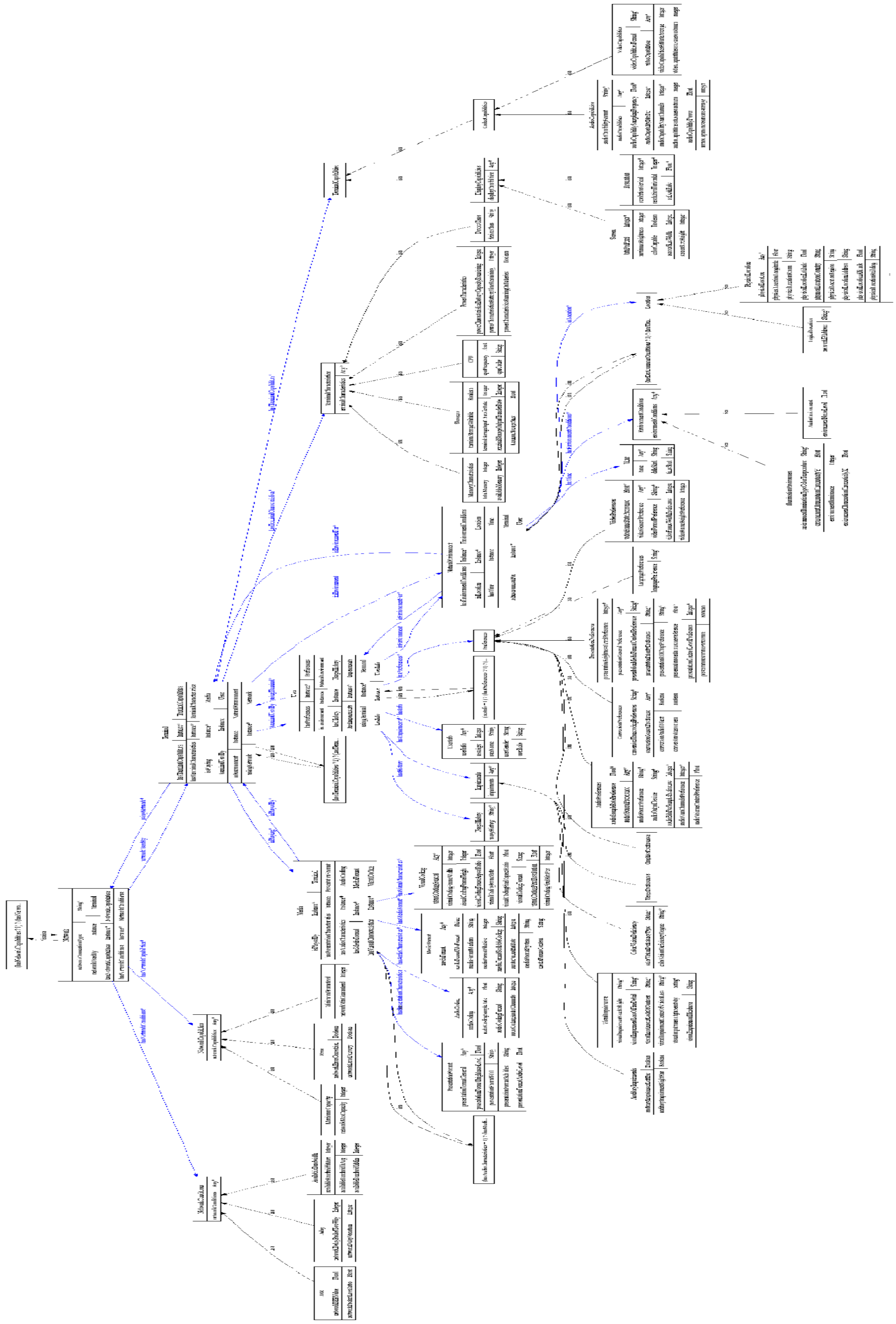


Figure B.6 – CAO Ontology – General overview

This figure is also available in the following address for better visualisation:

- http://paginas.fe.up.pt/~ee07068/ontology/view/CAO_Overview.gif

Appendix C

C CAO OWL code

C.1 CAO – OWL Code

Due to the large size of the generated OWL code for the constructed Context-Aware Ontology, it is only available on:

- http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl

C.2 CAO (Instances) Data – OWL Code

presents the code generated from the CAO Data application using the contextual information based on the Use Case 2, extracted from the MPEG-21 DIA UED and MPEG-7 MDS files which are represented in and , respectively.

Table C.1 – CAO Data OWL code

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://paginas.fe.up.pt/~ee07068/instances/DATA_CAO.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:cao="http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://paginas.fe.up.pt/~ee07068/instances/DATA_CAO.owl">
  <owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://paginas.fe.up.pt/~ee07068/ontology/CAO.owl"/>
  </owl:Ontology>
  <cao:MinimumGuaranteed rdf:ID="MinimumGuaranteed_Instance">
    <cao:networkMinGuaranteed rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >32000</cao:networkMinGuaranteed>
  </cao:MinimumGuaranteed>
  <cao:AudioEnvironment rdf:ID="AudioEnvironment_Instance">
    <cao:environmentNoiseLevel
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
```

```
    >80.0</cao:environmentNoiseLevel>
  </cao:AudioEnvironment>
  <cao:IlluminationEnvironment rdf:ID="Illumination_Instance">
    <cao:environmentIlluminance
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >500</cao:environmentIlluminance>
    <cao:environmentIlluminationTypeColorTemperature rdf:datatype=

"http://www.w3.org/2001/XMLSchema#string">159</cao:environmentIlluminationTypeCol
orTemperature>
  </cao:IlluminationEnvironment>
  <cao:PresentationPreferences rdf:ID="PresentationPreferences_Instance">
    <cao:presentationPriorityPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Video:1.5</cao:presentationPriorityPreference>
    <cao:presentationContrastLevelPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >80</cao:presentationContrastLevelPreference>
    <cao:presentationROICropPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >stillroi.xml#region1</cao:presentationROICropPreference>
    <cao:presentationSubtitlePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
    >true</cao:presentationSubtitlePreference>
  </cao:PresentationPreferences>
  <cao:AudioPreferences rdf:ID="AudioPreferences_Instance">
    <cao:audioVolumeControlPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >0.85</cao:audioVolumeControlPreference>
    <cao:audioSampleRatePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >44100.0</cao:audioSampleRatePreference>
    <cao:audioNumChannelsPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</cao:audioNumChannelsPreference>
    <cao:audioOutputDevice rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Loudspeaker</cao:audioOutputDevice>
    <cao:audioBitsPerSamplePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >16</cao:audioBitsPerSamplePreference>
  </cao:AudioPreferences>
  <cao:BER rdf:ID="BER_Instance">
    <cao:networkPacketLossRatio
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >0.05</cao:networkPacketLossRatio>
  </cao:BER>
  <cao:VideoCapabilities rdf:ID="VideoCapabilities_Instance">
    <cao:videoCapabilitiesBitRateAverage
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >50000</cao:videoCapabilitiesBitRateAverage>
    <cao:videoCapabilitiesFormat
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >MPEG-4 Visual Simple Profile @ Level 1</cao:videoCapabilitiesFormat>
    <cao:videoCapabilitiesBitRateMaximum
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >200000</cao:videoCapabilitiesBitRateMaximum>
  </cao:VideoCapabilities>
  <cao:PresentationFormat rdf:ID="PresentationFormat_Instance">
    <cao:presentationFormatBrightnessLevel
```

```
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >100.0</cao:presentationFormatBrightnessLevel>
    <cao:presentationFormatAudioLevel
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >0.6</cao:presentationFormatAudioLevel>
    <cao:presentationFormatROI
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ROI_1</cao:presentationFormatROI>
  </cao:PresentationFormat>
  <cao:Terminal rdf:ID="Terminal_Instance">
    <cao:terminalUsedBy>
      <cao:User rdf:ID="User_Instance">
        <cao:hasInfo>
          <cao:UserInfo rdf:ID="UserInfo_Instance">
            <cao:userRole rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >student</cao:userRole>
            <cao:userName rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >Vitor Barbosa</cao:userName>
            <cao:userGender
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >M</cao:userGender>
            <cao:userAge rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >23</cao:userAge>
          </cao:UserInfo>
        </cao:hasInfo>
        <cao:usingTerminal rdf:resource="#Terminal_Instance"/>
        <cao:inEnvironment>
          <cao:NaturalEnvironment rdf:ID="NaturalEnvironment_Instance">
            <cao:hasTime>
              <cao:Time rdf:ID="Time_Instance">
                <cao:timeStart
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >15:22</cao:timeStart>
                <cao:dateStart
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >2008-03-06</cao:dateStart>
              </cao:Time>
            </cao:hasTime>
            <cao:isEnvironmentFor rdf:resource="#Terminal_Instance"/>
            <cao:isEnvironmentFor rdf:resource="#User_Instance"/>
            <cao:hasEnvironmentConditions rdf:resource="#Illumination_Instance"/>
            <cao:hasEnvironmentConditions
rdf:resource="#AudioEnvironment_Instance"/>
            <cao:inLocation>
              <cao:LogicalLocation rdf:ID="LogicalLocation_Instance">
                <cao:networkIPAddress
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >172.116.10.206</cao:networkIPAddress>
                <cao:networkIPAddress
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >192.168.10.210</cao:networkIPAddress>
              </cao:LogicalLocation>
            </cao:inLocation>
            <cao:inLocation>
              <cao:PhysicalLocation rdf:ID="PhysicalLocation_Instance">
                <cao:physicalLocationAltitude
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
                >10.0</cao:physicalLocationAltitude>
                <cao:physicalLocationLatitude
```

```
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
                >41.15</cao:physicalLocationLatitude>
                <cao:physicalLocationLongitude
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
                >-8.61</cao:physicalLocationLongitude>
                <cao:physicalLocationRegion
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >pt</cao:physicalLocationRegion>
              </cao:PhysicalLocation>
            </cao:inLocation>
          </cao:NaturalEnvironment>
        </cao:inEnvironment>
        <cao:hasPreferences rdf:resource="#AudioPreferences_Instance"/>
        <cao:hasPreferences>
          <cao:ConversionPreferences rdf:ID="ConversionPreferences_Instance">
            <cao:conversionTranscodingPreferences rdf:datatype=
            "http://www.w3.org/2001/XMLSchema#string">From:    Audio    -    To:
Text</cao:conversionTranscodingPreferences>
            <cao:conversionTranscodingPreferences rdf:datatype=
            "http://www.w3.org/2001/XMLSchema#string">From:    Video    -    To:
Text</cao:conversionTranscodingPreferences>
            <cao:conversionVideoToText
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
            >true</cao:conversionVideoToText>
            <cao:conversionAudioToText
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
            >true</cao:conversionAudioToText>
          </cao:ConversionPreferences>
        </cao:hasPreferences>
        <cao:hasPreferences>
          <cao:LanguagePreferences rdf:ID="LanguagePreference_Instance">
            <cao:languagePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >en</cao:languagePreference>
            <cao:languagePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >pt</cao:languagePreference>
          </cao:LanguagePreferences>
        </cao:hasPreferences>
        <cao:hasPreferences rdf:resource="#PresentationPreferences_Instance"/>
        <cao:hasPreferences>
          <cao:VideoPreferences rdf:ID="VideoPreferences_Instance">
            <cao:videoFrameHeightPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >1024</cao:videoFrameHeightPreference>
            <cao:videoFormatPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >MPEG-4</cao:videoFormatPreference>
            <cao:videoFrameWidthPreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >768</cao:videoFrameWidthPreference>
            <cao:videoFrameRatePreference
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
            >25.0</cao:videoFrameRatePreference>
          </cao:VideoPreferences>
        </cao:hasPreferences>
      </cao:User>
    </cao:terminalUsedBy>
    <cao:isPlaying>
```

```
      <cao:Media rdf:ID="Media_Instance">
        <cao:hasMediaFormat>
          <cao:MediaFormat rdf:ID="MediaFormat_Instance">
            <cao:mediaFormatBitRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >0</cao:mediaFormatBitRate>
            <cao:mediaFormatMedium
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >CD</cao:mediaFormatMedium>
            <cao:mediaFormatFileFormat
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >mpeg</cao:mediaFormatFileFormat>
            <cao:mediaFormatContent
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >audiovisual</cao:mediaFormatContent>
            <cao:mediaFormatFileSize
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >666478608</cao:mediaFormatFileSize>
          </cao:MediaFormat>
        </cao:hasMediaFormat>
        <cao:hasPresentationCharacteristics
rdf:resource="#PresentationFormat_Instance"/>
        <cao:hasAudioCharacteristics>
          <cao:AudioCoding rdf:ID="AudioCoding_Instance"/>
        </cao:hasAudioCharacteristics>
        <cao:isPlayedBy rdf:resource="#Terminal_Instance"/>
        <cao:hasVisualCharacteristics>
          <cao:VisualCoding rdf:ID="VisualCoding_Instance">
            <cao:visualCodingFrameWidth
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >174</cao:visualCodingFrameWidth>
            <cao:visualCodingPixelAspectRatio
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
            >0.75</cao:visualCodingPixelAspectRatio>
            <cao:visualCodingPixelResolution
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
            >0.0</cao:visualCodingPixelResolution>
            <cao:visualCodingFormat
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >MPEG-4</cao:visualCodingFormat>
            <cao:visualCodingFrameHeight
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >144</cao:visualCodingFrameHeight>
            <cao:visualCodingFrameRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
            >25.0</cao:visualCodingFrameRate>
          </cao:VisualCoding>
        </cao:hasVisualCharacteristics>
      </cao:Media>
    </cao:isPlaying>
    <cao:hasTerminalCapabilities>
      <cao:Resolution rdf:ID="Resolution_Instance">
        <cao:resolutionVertical
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >144</cao:resolutionVertical>
        <cao:resolutionHorizontal
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >176</cao:resolutionHorizontal>
        <cao:refreshRate rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
```

```
        >70.0</cao:refreshRate>
      </cao:Resolution>
    </cao:hasTerminalCapabilities>
    <cao:hasTerminalCapabilities>
      <cao:AudioCapabilities rdf:ID="AudioCapabilities_Instance">
        <cao:audioCapabilityPower
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
        >60.0</cao:audioCapabilityPower>
        <cao:audioCapabilitySamplingFrequency
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
        >44100.0</cao:audioCapabilitySamplingFrequency>
        <cao:audioCapabilityNumChannels
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >2</cao:audioCapabilityNumChannels>
        <cao:audioCapabilityBitsPer
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >16</cao:audioCapabilityBitsPer>
        <cao:audioCapabilityFormat
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >MP3</cao:audioCapabilityFormat>
        <cao:audioCapabilityFormat
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >AMR</cao:audioCapabilityFormat>
        <cao:audioCapabilitiesBitRateMaximum
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >500</cao:audioCapabilitiesBitRateMaximum>
        <cao:audioCapabilitiesBitRateAverage
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >200</cao:audioCapabilitiesBitRateAverage>
      </cao:AudioCapabilities>
    </cao:hasTerminalCapabilities>
    <cao:inEnvironment rdf:resource="#NaturalEnvironment_Instance"/>
    <cao:hasTerminalCharacteristics>
      <cao:DeviceClass rdf:ID="DeviceClass_Instance">
        <cao:deviceClass rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >PC</cao:deviceClass>
      </cao:DeviceClass>
    </cao:hasTerminalCharacteristics>
    <cao:hasTerminalCapabilities rdf:resource="#VideoCapabilities_Instance"/>
    <cao:hasTerminalCapabilities>
      <cao:Screen rdf:ID="Screen_Instance">
        <cao:maximumBrightness
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1500</cao:maximumBrightness>
        <cao:screenSizeWidth rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >176</cao:screenSizeWidth>
        <cao:screenSizeHeight rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >144</cao:screenSizeHeight>
        <cao:bitsPerPixel rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >8</cao:bitsPerPixel>
        <cao:colorCapable rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
        >true</cao:colorCapable>
      </cao:Screen>
    </cao:hasTerminalCapabilities>
    <cao:hasTerminalCharacteristics>
      <cao:PowerCharacteristics rdf:ID="PowerCharacteristics_Instance">
        <cao:powerCharecteristicsBatteryTimeRemaining rdf:datatype=

"http://www.w3.org/2001/XMLSchema#int">15</cao:powerCharecteristicsBatteryTimeRem
```

```
aining>
        <cao:powerCharacteristicsBatteryCapacityRemaining rdf:datatype=

"http://www.w3.org/2001/XMLSchema#int">20</cao:powerCharacteristicsBatteryCapacit
yRemaining>
        <cao:powerCharacteristicsRunningOnBatteries rdf:datatype=

"http://www.w3.org/2001/XMLSchema#boolean">true</cao:powerCharacteristicsRunningO
nBatteries>
      </cao:PowerCharacteristics>
    </cao:hasTerminalCharacteristics>
    <cao:hasTerminalCharacteristics>
      <cao:Storages rdf:ID="Storage_Instance">
        <cao:terminalStorageWritable
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
        >true</cao:terminalStorageWritable>
        <cao:terminalStorageSize
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
        >60000.0</cao:terminalStorageSize>
        <cao:terminalStorageInputTransferRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >100</cao:terminalStorageInputTransferRate>
        <cao:terminalStorageOutputTransferRate
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >200</cao:terminalStorageOutputTransferRate>
      </cao:Storages>
    </cao:hasTerminalCharacteristics>
    <cao:usingNetwork>
      <cao:Network rdf:ID="Network_Instance">
        <cao:hasNetworkConditions rdf:resource="#BER_Instance"/>
        <cao:hasNetworkCapabilities>
          <cao:Error rdf:ID="Error_Instance">
            <cao:networkErrorDelivery
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
            >false</cao:networkErrorDelivery>
            <cao:networkErrorCorrection
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
            >true</cao:networkErrorCorrection>
          </cao:Error>
        </cao:hasNetworkCapabilities>
        <cao:networkUsedBy rdf:resource="#Terminal_Instance"/>
        <cao:networkConnectionType
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >wlan</cao:networkConnectionType>
        <cao:hasNetworkConditions>
          <cao:Delay rdf:ID="Delay_Instance">
            <cao:networkDelayPacketTwoWay
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >330</cao:networkDelayPacketTwoWay>
            <cao:networkDelayVariation
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >66</cao:networkDelayVariation>
          </cao:Delay>
        </cao:hasNetworkConditions>
        <cao:hasNetworkCapabilities rdf:resource="#MinimumGuaranteed_Instance"/>
        <cao:hasNetworkCapabilities>
          <cao:MaximumCapacity rdf:ID="MaximumCapacity_Instance">
            <cao:networkMaxCapacity
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
```

```
            >512000</cao:networkMaxCapacity>
          </cao:MaximumCapacity>
        </cao:hasNetworkCapabilities>
        <cao:networkConnectionType
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >ethernet</cao:networkConnectionType>
        <cao:hasNetworkConditions>
          <cao:AvailableBandwidth rdf:ID="AvailableBandwith_Instance">
            <cao:availableBandwidthAvg
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >128000</cao:availableBandwidthAvg>
            <cao:availableBandwidthMax
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >256000</cao:availableBandwidthMax>
          </cao:AvailableBandwidth>
        </cao:hasNetworkConditions>
      </cao:Network>
    </cao:usingNetwork>
  </cao:Terminal>
</rdf:RDF>
```

Table C.2 – MPEG-21 DIA UED code for the Use Case 2

```
<?xml version="1.0" encoding="UTF-8"?>
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001" >
<Description xsi:type="UsageEnvironmentType">
	<UsageEnvironmentProperty xsi:type="NetworksType">
		<Network>
			<NetworkCharacteristic xsi:type="NetworkCapabilityType"
maxCapacity="512000" minGuaranteed="32000" errorCorrection="true"
errorDelivery="false"/>
			<NetworkCharacteristic xsi:type="NetworkConditionType">
				<AvailableBandwidth maximum="256000"
average="128000"/>
				<Delay packetTwoWay="330" delayVariation="66"/>
				<Error packetLossRate="0.05"/>
			</NetworkCharacteristic>
		</Network>
	</UsageEnvironmentProperty>
	<UsageEnvironmentProperty xsi:type="UsersType">
		<User>
			<UserCharacteristic
xsi:type="AudioPresentationPreferencesType">
				<VolumeControl>0.85</VolumeControl>
				<AudibleFrequencyRange>
					<StartFrequency>20</StartFrequency>
					<EndFrequency>20000</EndFrequency>
				</AudibleFrequencyRange>
				<AudioOutputDevice>Loudspeaker</AudioOutputDevice>
				<Soundfield>
					<ImpulseResponse
href="http://www.sac.or.kr/concertHall/hallImp.wav">
					<SamplingFrequency>44100</SamplingFrequency>
					<BitsPerSample>16</BitsPerSample>
					<NumOfChannels>1</NumOfChannels>
					</ImpulseResponse>
				</Soundfield>
```

```
                          <SoniferousSpeed>0.5</SoniferousSpeed>
                    </UserCharacteristic>
                    <UserCharacteristic xsi:type="UserInfoType">
                          <UserInfo xsi:type="mpeg7:PersonType">
                          <mpeg7:Name>
                                <mpeg7:GivenName>Barbosa</mpeg7:GivenName>
                                <mpeg7:FamilyName>Vitor</mpeg7:FamilyName>
                          </mpeg7:Name>
                          </UserInfo>
                    </UserCharacteristic>
                    <UserCharacteristic
xsi:type="DisplayPresentationPreferencesType">
                          <BrightnessPreference>
                                <BinNumber>255</BinNumber>
                                <Value>
                                      <PreferredValue>110</PreferredValue>
                                      <ReferenceValue>127</ReferenceValue>
                                </Value>
                                <Value>
                                      <PreferredValue>156</PreferredValue>
                                      <ReferenceValue>151</ReferenceValue>
                                </Value>
                          </BrightnessPreference>
                          <ContrastPreference>
                                <BinNumber>255</BinNumber>
                                <Value>
                                      <PreferredValue>80</PreferredValue>
                                      <ReferenceValue>70</ReferenceValue>
                                </Value>
                          </ContrastPreference>
                    </UserCharacteristic>
                    <UserCharacteristic xsi:type="ConversionPreferenceType">
                          <GeneralResourceConversions>
                                <Conversion order="1" weight="1.0">
                                      <From
href="urn:mpeg:mpeg7:cs:ContentCS:2001:1">
                                      <mpeg7:Name>Audio</mpeg7:Name>
                                      </From>
                                      <To
href="urn:mpeg:mpeg7:cs:ContentCS:2001:5">
                                      <mpeg7:Name>Text</mpeg7:Name>
                                      </To>
                                </Conversion>
                                <Conversion order="2" weight="1.0">
                                      <From
href="urn:mpeg:mpeg7:cs:ContentCS:2001:4.2">
                                      <mpeg7:Name>Video</mpeg7:Name>
                                      </From>
                                      <To
href="urn:mpeg:mpeg7:cs:ContentCS:2001:5">
                                      <mpeg7:Name>Text</mpeg7:Name>
                                      </To>
                                </Conversion>
                          </GeneralResourceConversions>
                    </UserCharacteristic>
                    <UserCharacteristic
xsi:type="PresentationPriorityPreferenceType">
                          <GeneralResourcePriorities>
                                <ModalityPriorities>
```

```
                                        <Modality priorityLevel="1.5"
href="urn:mpeg:mpeg7:cs:ContentCS:2001:4.2">
                                        <mpeg7:Name>Video</mpeg7:Name>
                                        </Modality>
                                  </ModalityPriorities>
                            </GeneralResourcePriorities>
                      </UserCharacteristic>
                      <UserCharacteristic xsi:type="FocusOfAttentionType">
                            <ROI uri="stillroi.xml#region1"
updateInterval="0.5"/>
                      </UserCharacteristic>
                </User>
          </UsageEnvironmentProperty>
          <UsageEnvironmentProperty xsi:type="NaturalEnvironmentsType">
                <NaturalEnvironment>
                      <NaturalEnvironmentCharacteristic xsi:type="LocationType">
                            <Location>
                                  <mpeg7:GeographicPosition>
                                  <mpeg7:Point longitude="-8.61" altitude="10.0"
latitude="41.150"/>
                                  </mpeg7:GeographicPosition>
                                  <mpeg7:Region>pt</mpeg7:Region>
                            </Location>
                      </NaturalEnvironmentCharacteristic>
                      <NaturalEnvironmentCharacteristic xsi:type="TimeType">
                            <Time>
                            <mpeg7:TimePoint>2008-03-
06T15:22+01:00</mpeg7:TimePoint>
                            </Time>
                      </NaturalEnvironmentCharacteristic>
                      <NaturalEnvironmentCharacteristic
xsi:type="AudioEnvironmentType">
                            <NoiseLevel>80</NoiseLevel>
                      </NaturalEnvironmentCharacteristic>
                      <NaturalEnvironmentCharacteristic
xsi:type="IlluminationCharacteristicsType">
                            <TypeOfIllumination>
                                  <ColorTemperature>159</ColorTemperature>
                            </TypeOfIllumination>
                            <Illuminance>500</Illuminance>
                      </NaturalEnvironmentCharacteristic>
                </NaturalEnvironment>
          </UsageEnvironmentProperty>
          <UsageEnvironmentProperty xsi:type="TerminalsType">
                <Terminal>
                      <TerminalCapability xsi:type="DisplaysType">
                            <Display id="d0">
                                  <DisplayCapability
xsi:type="DisplayCapabilityType" bitsPerPixel="8" colorCapable="true"
maximumBrightness="1500">
                                  <Mode refreshRate="70">
                                  <Resolution horizontal="176" vertical="144"/>
                                  </Mode>
                                  <ScreenSize horizontal="176" vertical="144"/>
                                  </DisplayCapability>
                            </Display>
                      </TerminalCapability>
                      <TerminalCapability xsi:type="CodecCapabilitiesType">
                            <Decoding xsi:type="AudioCapabilitiesType">
```

```
                          <Format
href="urn:mpeg:mpeg7:cs:AudioCodingFormatCS:2001:4.4">
                                <mpeg7:Name xml:lang="en">MP3</mpeg7:Name>
                          </Format>
                          <Format
href="urn:mpeg:mpeg7:cs:AudioCodingFormatCS:2001:6">
                                <mpeg7:Name xml:lang="en">AMR</mpeg7:Name>
                          </Format>
                          <CodecParameter xsi:type="CodecParameterBitRateType">
                                <BitRate average="200" maximum="500"/>
                          </CodecParameter>
                          <CodecParameter
xsi:type="CodecParameterFillRateType">
                                <FillRate>20000</FillRate>
                          </CodecParameter>
                          </Decoding>
                          <Decoding xsi:type="VideoCapabilitiesType">
                                <Format
href="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:3.1.2">
                                <mpeg7:Name xml:lang="en">MPEG-4 Visual Simple
Profile @ Level 1</mpeg7:Name>
                                </Format>
                                <CodecParameter
xsi:type="CodecParameterBitRateType">
                                <BitRate average="50000" maximum="200000"/>
                                </CodecParameter>
                          </Decoding>
                    </TerminalCapability>
                    <TerminalCapability xsi:type="DeviceClassType">
                          <DeviceClass href="urn:mpeg:mpeg21:2003:01-DIA-
DeviceClassCS-NS:1">
                                <mpeg7:Name xml:lang="en">PC</mpeg7:Name>
                          </DeviceClass>
                    </TerminalCapability>
                    <TerminalCapability xsi:type="PowerCharacteristicsType"
batteryCapacityRemaining="20" batteryTimeRemaining="15"
runningOnBatteries="true"/>
                    <TerminalCapability xsi:type="StoragesType">
                          <Storage>
                                <StorageCharacteristic
xsi:type="StorageCharacteristicsType" inputTransferRate="100"
outputTransferRate="200" size="60000" writable="true"/>
                          </Storage>
                    </TerminalCapability>
                    <TerminalCapability xsi:type="AudioOutputsType">
                          <AudioOutput xsi:type="AudioOutputType">
                                <AudioOutputCapability
xsi:type="AudioOutputCapabilitiesType" lowFrequency="20" highFrequency="22000"
numChannels="2" power="60">
                          <Mode bitsPerSample="16" samplingFrequency="44100"/>
                                </AudioOutputCapability>
                          </AudioOutput>
                    </TerminalCapability>
              </Terminal>
              </UsageEnvironmentProperty>
       </Description>
</DIA>
```

Table C.3 – MPEG-7 MDS code for the Use Case 2

```
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001">
	<Description xsi:type="MediaDescriptionType">
		<MediaInformation id="news1_media">
			<MediaIdentification>
				<EntityIdentifier organization="MPEG"
type="MPEG7ContentSetId">mpeg7_content:news1
				</EntityIdentifier>
				<VideoDomain
href="urn:mpeg:mpeg7:cs:VideoDomainCS:2001:1.2">
					<Name xml:lang="en">Natural</Name>
				</VideoDomain>
			</MediaIdentification>
			<MediaProfile>
				<MediaFormat>
					<Content href="MPEG7ContentCS">
						<Name>audiovisual</Name>
					</Content>
					<Medium
href="urn:mpeg:mpeg7:cs:MediumCS:2001:1.1">
						<Name xml:lang="en">CD</Name>
					</Medium>
					<FileFormat
href="urn:mpeg:mpeg7:cs:FileFormatCS:2001:3">
						<Name xml:lang="en">mpeg</Name>
					</FileFormat>
					<FileSize>666478608</FileSize>
					<VisualCoding>
						<Format
href="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:1" colorDomain="color">
							<Name xml:lang="en">MPEG-4</Name>
						</Format>
						<Pixel aspectRatio="0.75" bitsPer="8"/>
						<Frame height="144" width="174"
rate="25"/>
					</VisualCoding>
				</MediaFormat>
			</MediaProfile>
		</MediaInformation>
	</Description>
</Mpeg7>
```