

# Delivery Context Descriptions – A Comparison and Mapping Model

Christian Timmerer, Johannes Jabornig, and Hermann Hellwagner

Department of Information Technology (ITEC)  
Klagenfurt University  
Universitätsstrasse 65-67  
A-9020 Klagenfurt  
{christian.timmerer, hermann.hellwagner}@itec.uni-klu.ac.at  
j.jabornig@jabjo.net

**Abstract:** Nowadays, mobile devices have implemented several transmission technologies which enable access to the Internet and increase the bit rate for data exchange. Despite modern mobile processors and high-resolution displays, mobile devices will never reach the stage of a powerful notebook or desktop system (for example, due to the fact of battery powered CPUs or just concerning the small-sized displays). Due to these limitations, the deliverable content for these devices should be adapted based on their capabilities including a variety of aspects (e.g., from terminal to network characteristics). These capabilities should be described in an interoperable way. In practice, however, there are many standards available and a common mapping model between these standards is not in place. Therefore, in this paper we describe such a mapping model and its implementation aspects. In particular, we focus on the whole delivery context (i.e., terminal capabilities, network characteristics, user preferences, etc.) and investigated the two most prominent state-of-the-art description schemes, namely User Agent Profile (UAProf) and Usage Environment Description (UED).

## 1 Introduction and Motivation

Today, the access to the Internet via mobile phones and other devices which are limited in power capacity and/or rendering functionality increases. Additionally, manufacturers equip their devices with technologies to access various networks, and mobile providers offer services for connecting these devices to the Internet. Thus, using small, mobile devices enables the access to the Internet but often the available content of Web pages is not suitable according to their capabilities. In order to solve this issue, some projects and standards have been released which deal with the description of capabilities and characteristics of all kind of devices. Terminal capabilities and network conditions as well as user characteristics may allow the adaptation of the content for certain purposes.

Amongst others, there are two standards which were designed to meet the requirements of device and user descriptions and are often compared to each other. The first one was released by the WAP Forum (now the Open Mobile Alliance) and is named User Agent Profile (UAProf) [OM06] and the second one is the Usage Environment Description (UED) standard which was standardized within MPEG-21 Digital Item Adaptation (DIA) [VT05]. The aim of this paper is to build a mapping model for these description formats enabling context-aware content delivery independent of the actual description format used. Recently, W3C has started a new work item with the aim to define a delivery context ontology [LF08] but, still, the mapping issue remains.

While the User Agent Profile standard is very popular and has been implemented in a wide range of mobile devices the Usage Environment Descriptions are only limited available and tools which would ease the creation of such descriptions rarely exist. However, a high availability of Usage Environment Descriptions is desired by research projects [Da06][Ax08][En08] and, thus, obtaining information about terminals, networks and users from projects with similar aims to that of Usage Environment Descriptions should be enabled. For example, an implementation compliant to a standardized delivery context description format A requires a mapping module if it receives descriptions compliant to another standard B and vice versa. In order to keep the mapping effort minimal and scalable the proposed method in this paper enable the implementation of a service that performs such kind of mapping.

This paper is organized as follows. Section 2 highlights the main requirements on standards for delivery context descriptions by classifying terminals and their properties. An analysis and comparison of delivery context description formats is presented in Section 3 while Section 4 describes the actual mapping model. Finally, the implementation details are described in Section 5 and the paper is concluded with Section 6 which contains also future work items.

## **2 Requirements on Standards for Context Descriptions**

In this section we have identified different classes of terminals including their hardware and software capabilities. This kind of information should provide us a rough estimation on the requirements for delivery context description standards from the terminal's point of view which also includes the access networks.

When the Internet became popular, the only way to access the Web was through a personal computer (PC) or a workstation. In general, these computers had large color displays with full graphic capabilities, sufficient computational power without battery issues, and a decent network connection [GLS06]. Nowadays, people tend to access the Web using smaller and mobile devices with various constraints on display capabilities, user input/output facilities, computational power, electric power, and access networks ranging from high-speed Wireless Local Area Network (WLAN) to low-speed General Packet Radio Service (GPRS). In the following we will provide a classification of the various terminals based on their hardware (HW) and software (SW) characteristics.

Table 2 in the Appendix provides an overview of HW/SW characteristics of different end user devices (i.e., desktop PC/workstation, notebook/tablet PC, sub-notebook/netbook, handheld, smart phone, and mobile phone) with respect to performance (i.e., CPU), display, permanent storage, memory, network connectivity, electric power, user input/output, extensibility, operating system support, and software in general.

A summary and comparison of terminal's display and memory properties is depicted in Figure 1. As one can see there is still a huge gap between classical mobile devices (i.e., phones) and devices that may have full power supply. Thus, a comprehensive delivery context standard needs to accommodate all these HW/SW properties in an easy-to-understand/use, extensible, and manageable way.

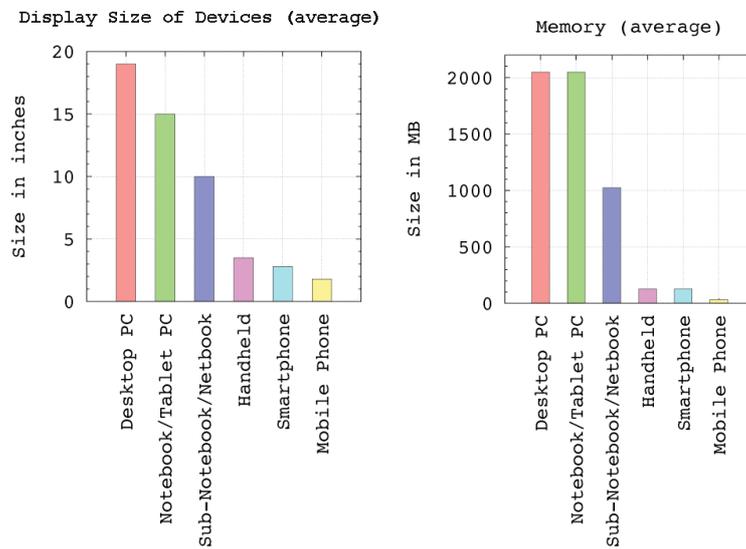


Figure 1. Summary and comparison of terminal's display and memory properties.

However, not only HW properties like screen size, color capabilities, or user input/output facilities are important, also SW properties like supported operating systems, audio/video/image codecs, etc. become more and more important as diversity of devices increases. In particular, the number of different coding formats a terminal is capable to support – both for encoding and decoding – is of interest for delivery context description formats. As there are so many coding formats available, some may have certain profile/level definitions, and even a class of terminals may define its own constraints, there exists a strong requirement to describe these properties effectively. A key functionality is the possibility to add new coding formats – e.g., through a registration authority – in a convenient and relatively unbureaucratic way as they arise rather rapidly on the market.

## 3 Analysis and Comparison of Context Description Formats

### 3.1 Composite Capabilities/Preference Profiles (CC/PP)

The Composite Capabilities/Preference Profiles (CC/PP) [KI04] comprises descriptions based on the Resource Description Framework (RDF) which cover device capabilities and user preferences by introducing a two-level hierarchy consisting of *components* and *attributes*. Components are groups of attributes with related meaning such as the software or the hardware properties of a terminal. A CC/PP document shall contain at least one component each identified by an `rdf:type` attribute which indicates the type of the component. Attribute values may be *simple*, i.e., string and integer or rational numbers, or *complex*, i.e., set (`rdf:bag`) or sequence (`rdf:seq`) of simple values. However, CC/PP does not define a vocabulary of terms but provides a common structure for holding any arbitrary vocabulary. Thus, another description format is required which specifies the actual vocabulary, e.g., the User Agent Profile as described next.

### 3.2 User Agent Profile (UAProf)

The Open Mobile Alliance (OMA) defines the User Agent Profile (UAProf) [OM06] which is based on CC/PP and defines a vocabulary for describing characteristics and capabilities of mainly WAP-enabled mobile devices. The components can be clustered into

- `HardwarePlatform` defines manufacture, CPU type, display/audio output capabilities, device interaction possibilities, and keyboard layout;
- `SoftwarePlatform` comprises supported media types, preferred language, operating system, audio/video codecs, etc.;
- `BrowserUA` includes whether the browser supports certain (X)HTML features, frames, tables, and its JavaScript capabilities;
- `NetworkCharacteristics` holds information about supported and current bearers, security options, and some details about Bluetooth support; and
- various `WapCharacteristics` and `PushCharacteristics`.

### 3.3 Usage Environment Description (UED)

The Usage Environment Description (UED) is defined in Part 7 of MPEG-21, i.e., Digital Item Adaptation (DIA) [VT05]. The UED is a very comprehensive vocabulary organized in so-called properties. It is based on XML Schema and its properties can be divided into four categories:

- *User characteristics* provide information pertaining to the user plus his/her usage preferences/history, presentation preferences, accessibility characteristics, and location information including the user's movement.
- *Terminal capabilities* comprise codec capabilities, input/output characteristics including display/audio output capabilities, and device properties such as device class, power/storage characteristics, data input/output facilities, and CPU capabilities.

- *Network characteristics* include static and dynamic properties pertaining to the capabilities (e.g., max. bandwidth) and conditions (e.g., available bandwidth) of a network.
- *Natural environment characteristics* provide means for describing lightning conditions, noise level, time, and location.

### 3.4 Delivery Context Ontology (DCO)

The Delivery Context Ontology (DCO) [LF08] is based the Web Ontology Language (OWL) [Mv04] and is divided into *four entities*, namely:

- *Environment* including information about the location and network.
- *Hardware* provides information about various hardware capabilities including display, input, memory, camera, Bluetooth, CPU, etc.
- *Measure* defines terms related to units with respect to physical electric charges and length as well as unit conversions.
- *Software* describes whether the delivery context supports certain APIs, formats (i.e., audio, video, image, text, binary), operating systems, application-layer protocols, Java/Web browser specifics, etc.

### 3.5 Analysis/Comparison

The previous sections provided an overview of existing standards for delivery context description formats. In this section we will analyse differences and commonalities of these formats. First of all, and most importantly, all standards make use of XML that provides extensibility but UED is based on XML Schema whereas CC/PP and, consequently, UAProf are based on RDF. The most recent standard in this series is DCO which adopted already OWL which is based on RDF. Hence, we observe an incompatibility at the level of technology used for these description formats, mainly between XML Schema and RDF. Although it is possible to provide a high-level mapping between these two technologies, the mapping of concrete schemas/instances itself is a difficult and cumbersome task [HL01].

The second observation we made was that there are only a few characteristics or capabilities that are common across all delivery context description formats in question, e.g., display capabilities and file/coding formats. However, there is sometimes a huge difference in the actual syntax used. For example, display resolution described as `horizontal=1024` and `vertical=768` versus `1024x768` or using MIME types for file/coding formats versus classification schemes (i.e., URNs).

Finally, CC/PP defines only a basic structure (i.e., components and attributes) without specifying a particular vocabulary of terms. UAProf adopts CC/PP and provides a concrete vocabulary mainly targeting WAP-enabled mobile devices. A repository of some specific device profiles is available [W308]. Other industry adoptions of CC/PP are not known but some are envisaged and documented in Annex E of [Ki07]. The UED defines both the structure (i.e., properties) and a comprehensive vocabulary while DCO defines an ontology including not only a vocabulary of delivery context terms but also basic measure units.

In conclusion, there is a need to describe the relationship between commonalities of the different delivery context description formats, i.e., a mapping model which will be described in the next section. To the best of our knowledge, such a mapping model has not been published yet.

## 4 Mapping Model

UAProf and UED are based on different data models as the former is based on RDF whereas the latter is based on XML Schema with having their pros and cons [HL01]. That is, RDF provides support for rich semantic descriptions but provides limited support for the specification of local usage constraints, e.g., cardinality and datatype constraints. On the other hand, XML Schema provides support for explicit structural, cardinality and datatype constraints but provides little support for the semantic knowledge necessary to enable a flexible mapping between metadata domains.

The main issue is to find a suitable technology for the mapping process which includes the advantages of both standards. Basically, the mapping can be performed by two approaches as discussed in the following and depicted in Figure 2:

- **Direct mapping model:** creating mapping functions that perform direct mapping from one standard to another.
- **Integration model:** integrating both models into a new one with functions to convert between this new model and the initial model.

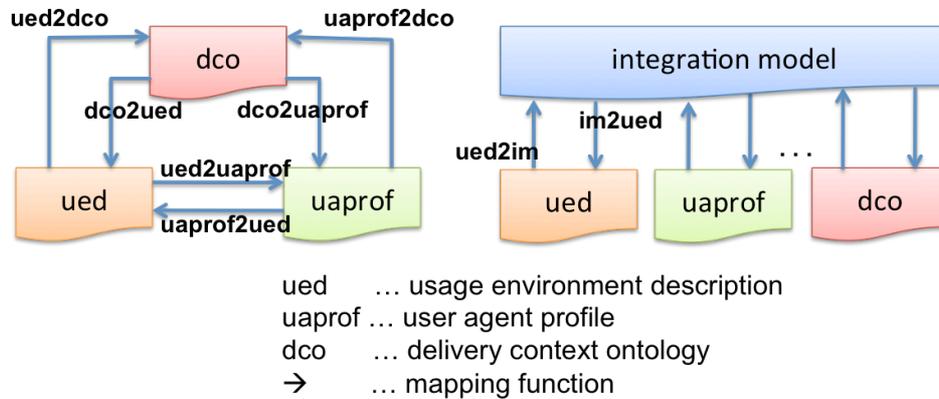


Figure 2. Direct Mapping Model vs. Integration Model.

As the *direct mapping model* provides an explicit mapping from one format to another format it lacks of flexibility with respect to the integration of other formats. Thus, it can be only applied for specific solutions whereby the number of explicit mappings increases exponentially with the number of formats between which mappings should be provided.

The *integration model* defines a common interface that allows the provisioning of the individual description formats. For new formats to be added, only the methods for converting to/from this model needs to be implemented without taking into account the existence of the other formats. Thus, the number of mappings increases linearly with the number of formats. However, the integration model needs to be implemented with a certain technology and those in question are XML Schema, RDF, or even OWL:

- One could define an **XML Schema** that covers all components of each standard and well-established XPath/XML processors could be used to extract the required data for a certain standard. Unfortunately, datatype or value range incompatibilities (e.g., UED: `colorCapable={true,false}` vs. UAProf: `ColorCapable={Yes,No}`) cannot be represented with XML Schema which requires external knowledge to be provided. Thus, it would be better to use tools which are able to express the relations between, e.g., datatypes or attribute values.
- **OWL** is based on RDF and provides means to describe the relationship between classes and properties, e.g., classes can be declared distinct or equal, restrictions on properties can be defined as transitive or functional, or the use of cardinality restricts the number of values which can be associated to properties.

#### 4.1 Mapping Levels

The relationships between different delivery context description formats can be found at different levels within the entities of their respective schemas (i.e., XML Schema or OWL). In this paper we introduce four different mapping levels, namely *component*, *datatype*, *element*, and *value*. An example thereof is shown in Table 1.

Table 1. Example of Mapping Levels for Network Characteristics.

Level	UAProf Example	UED Example
<i>Component</i>	prf:NetworkCharacteristics	dia:NetworkType
<i>Element</i>	prf:InputCharSet	dia:CharacterSetCode
<i>Datatype</i>	prf-dt:Boolean	xsd:Boolean
<i>Value</i>	Yes	true

The *component* mapping level tries to map a predefined group of elements/attributes (e.g. prf:NetworkCharacteristics) to similar group of the other description format (e.g. dia:NetworkType). Difficulties may arise in case the structure is different, e.g., one has only attributes defined whereas the other includes also elements within a nested structure.

Thus, one needs to dig a level deeper and the *element* mapping level tries to map attributes/elements with equal semantics but possibly different syntax, i.e., different tag names. Note that a mapping at the component level is not always sufficient as indicated above which requires describing relationships between individual elements/attributes or even beyond.

The *datatype* mapping level tries to map datatypes with equal domains but different syntax whereas the *value* mapping level tries to map datatypes with different domains but equal semantics.

An example of mapping attributes with the equal semantics and Boolean values is described in Listing 1 which maps the prf:ColorCapable to the map:colorCapable (assuming map:colorCapable is the RDF/XML representation of the colorCapable attribute of the dia:DisplayCapabilityType).

Listing 1. Mapping prf:ColorCapable to map:colorCapable.

```

1 ...
2 <owl:FunctionalProperty rdf:about = "&prf;ColorCapable">
3   <owl:equivalentProperty rdf:resource = "&map;colorCapable" />
4 </owl:FunctionalProperty>
5 ...

```

A problem arises as both datatypes are Boolean types but with different syntax: while xsd:boolean (as used within UED) accepts true and false, prf-dt:Boolean (as used within UAProf) accepts only Yes and No which requires an appropriate mapping. Listing 2 shows one possibility for such a mapping of these Boolean datatypes. Lines 2 to 6 and lines 8 to 12 describe properties to create a relation between a new defined resource for a Boolean value (prefixed by btd) and the Boolean values used by UAProf and UED. The rest provides the mapping between the values and the bdt Boolean datatypes.

Listing 2. Mapping the values of `prf-dt:Boolean` and `xsd:Boolean`.

```
1 ...
2 <owl:DatatypeProperty rdf:about="&bd;hasXsdBoolean">
3   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
4   <rdfs:domain rdf:resource="&prf-dt:Boolean"/>
5   <rdfs:range rdf:resource="&xsd:boolean"/>
6 </owl:DatatypeProperty>
7
8 <owl:DatatypeProperty rdf:about="&bd;hasOmaBoolean">
9   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
10  <rdfs:domain rdf:resource="&xsd:boolean"/>
11  <rdfs:range rdf:resource="&prf-dt:Boolean"/>
12 </owl:DatatypeProperty>
13
14 <bd:Boolean rdf:about="&bd;true">
15   <bd:hasXsdBoolean>true</bd:hasXsdBoolean>
16   <bd:hasOmaBoolean >Yes</bd:hasOmaBoolean>
17 </bd:Boolean>
18
19 <bd:Boolean rdf:about="&bd;false">
20   <bd:hasXsdBoolean>false</bd:hasXsdBoolean>
21   <bd:hasOmaBoolean>No</bd:hasOmaBoolean>
22 </bd:Boolean>
23 ...
```

Another example is shown in Listing 3 which maps a Uniform Resource Name (URN) identifying a certain key input type to the equivalent string representation of UAProf. The usage of URNs to uniquely identify predefined terms is heavily used within UED.

Listing 3. Mapping of key input types: `DIA-KeyInputCS-NS:1` and `Querty`.

```
1 ...
2 <owl:DatatypeProperty rdf:about="&key;hasMpegUrnRepresentation">
3   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
4   <rdfs:domain rdf:resource="&key;Keyboard"/>
5   <rdfs:range rdf:resource="&xsd:string"/>
6 </owl:DatatypeProperty>
7
8 <owl:DatatypeProperty rdf:about="&key;hasOmaKeyboard">
9   <rdf:type rdf:resource="&owl;FunctionalProperty"/>
10  <rdfs:domain rdf:resource="&key;Keyboard"/>
11  <rdfs:range rdf:resource="&xsd:string"/>
12 </owl:DatatypeProperty>
13
14 <key:Keyboard rdf:about="urn:mpeg:mpeg21:2003:01-DIA-KeyInputCS-NS:1">
15   <key:hasMpegUrnRepresentation>
16     urn:mpeg:mpeg21:2003:01-DIA-KeyInputCS-NS:1
17   </key:hasMpegUrnRepresentation>
18   <key:hasOmaKeyboard>Querty</key:hasOmaKeyboard>
19 </key:Keyboard>
20 ...
```

Datatypes such as `prf-dt:Number` and `xsd:nonNegativeInteger` can be mapped directly to each other because both cover the same range of values. However, integer values also raise problems when a mapping from `dia:bitsPerPixel` to `prf:BitsPerPixel` is provided because the UED standard defines an `xsd:integer` datatype and the UAProf standard uses the `prf-dt:Number` datatype which is equal to `xsd:nonNegativeInteger`. Of course, it is unlikely to describe the number of bits per pixel or the horizontal and vertical resolution with a negative number but there is the possibility to do that. OWL lacks to offer appropriate tools to describe what to do if such problems arise.

Another issue is raised with the datatype `prf-dt:Dimension` which is used to describe the resolution of a terminal's display within one string of the form "HorizontalResolution" x "VerticalResolution" (e.g., 1024x768). This means that two values of the UED (i.e., horizontal and vertical attributes of the `Resolution` element) are combined to represent one value within UAProf. Thus, external tools are needed to enable such mappings.

## 4.2 Mapping Classes

In practice, the tag names and datatypes of different description formats can be clustered into four classes which are described in the following.

**Direct.** Elements falling into this class have equal semantics and compatible datatypes with equal domains but may differ in their syntax (i.e., tag name).

For example, `dia:bitsPerPixel` of type `xsd:integer` and `prf:BitsPerPixel` of type `prf-dt:Number` where these datatypes are compatible. Another example is the `dia:CharacterSetCode` of type `mpeg7:characterSetCode` (equal to `xsd:string`) and `prf:CcppAccept-Charset` of type `prf-dt:Literal` (equal to `xsd:anySimpleType`) which are used to store string representations of the supported character sets.

**Advance.** The class advance comprises elements describing the same concept (i.e., equal semantics) but with different, non-compatible datatypes and/or domains. Thus, the actual format is that much different and requires major changes if mapped from one format to the other. For example, the `dia:Resolution` includes two attributes (horizontal and vertical) for describing the resolution of a screen whereas `prf:ScreenSize` uses only one value (e.g., 480x320). Another example is the usage of classification schemes versus MIME types as detailed in Section 4.3. Thus, an advanced mapping mechanism is required.

**Derive.** This class includes mappings where element values can be derived from one or more elements of the respective other description format. The difference to the advance class is that for the derive class the semantic equality is not necessarily a requirement. For example, `prf:SoundOutputCapable` indicates whether a terminal is able to output sound which could be derived from only the presence of a `dia:AudioOutputCapability` element.

**Extend.** Elements that cannot be mapped directly, in an advanced way through additional mapping rules, or derived from other elements require proprietary extensions of the respective other description format. For example, properties defined within UAProf but not defined in UED require an extension of the UED schema by adding additional elements and datatypes representing these UAProf properties.

In our example, the UAProf standard defines six components and 77 elements which have been mapped – with respect to UED – to the classes described above (quantities in brackets): direct (4), advance (7), derive (4), and extend (62). The specific support for mobile phones within UAProf causes the high number within the class extend whereas UED does not provide means for describing WAP or push characteristics. One could now argue that such a mapping is not required. Please note that for most of the application scenarios – in particular, multimedia content adaptation – the required elements/attributes/tags fall into direct, advance, and derive classes, e.g., adaptation to screen size, codec, bitrate which are covered in all delivery context description formats. Therefore, the class ‘extend’ can be ignored for this kind of applications.

### 4.3 Additional Mapping Rules for Coding Formats

This section specifically discusses means for describing supported coding formats as this seem to be an inherent part of each delivery context description standard. Unfortunately, the standards in question adopt different technologies for describing this property. In particular, the CC/PP and, thus, UAProf adopts an approach which is based on MIME media types [FB96] whereas MPEG-21 UED relies on classification schemes introduced within MPEG-7 [MSS02].

MIME media types are well known within the Internet – thanks to its adoption for HTTP, etc. – and comprises five discrete top-level media types, i.e., text, image, audio, video, and application, as well as two composite top-level media types, i.e., multipart and message. These top-level media types are referred to as content types and the actual coding format is identified through the content sub-type (e.g, `video/mp4`). It is also possible to associate an arbitrary number of parameters in form of key-value pairs to media types which could be used to describe specifics usually defined within profiles/levels. However, most of the audio/video/image MIME type definition does not make use of this possibility. Thus, it is up to the application to identify the exact data format by other means. For example, `video/mp4` may contain a bitstream compliant to MPEG-4 Part 2 (Visual) or MPEG-4 Part 10 (Advanced Video Coding), not mentioning all the available profile/level combinations.

An MPEG classification scheme is an XML document that may contain terms – identifiable by URN – and corresponding definitions of arbitrary semantics in a hierarchically fashion. Thus, it is also possible to include profile/levels of a certain coding format as shown in Listing 4. Although classification schemes are extensible as they are based on XML there is a lack of an approved registration authority to accommodate future coding formats. However, the European Broadcasting Union (EBU) maintains a set of classification schemes used within their specifications (including TV-Anytime) [EBU09].

Listing 4. Excerpt of Visual Coding Format Classification Scheme MPEG-4 Visual Simple Profile.

```
<?xml version="1.0" encoding="UTF-8"?>
<ClassificationScheme
  uri="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001">
  <!-- further terms -->
  <Term termID="3">
    <Name xml:lang="en">MPEG-4 Visual</Name>
    <Definition xml:lang="en">MPEG-4 Visual Coding Format</Definition>
    <Term termID="3.1">
      <Name xml:lang="en">MPEG-4 Visual Simple Profile</Name>
      <Term termID="3.1.1">
        <Name xml:lang="en">MPEG-4 Visual Simple Profile @ Level 0</Name>
      </Term>
      <Term termID="3.1.2">
        <Name xml:lang="en">MPEG-4 Visual Simple Profile @ Level 1</Name>
      </Term>
      <Term termID="3.1.3">
        <Name xml:lang="en">MPEG-4 Visual Simple Profile @ Level 2</Name>
      </Term>
      <Term termID="3.1.4">
        <Name xml:lang="en">MPEG-4 Visual Simple Profile @ Level 3</Name>
      </Term>
      <!-- further terms -->
    </Term>
  <!-- further terms -->
</ClassificationScheme>
```

Listing 5 describes the mapping from the MIME type image/jpeg to an equivalent URN representation which is used in UED. Line 1 of Listing 5 defines a resource for the MIME type image/jpeg. The other prefixes csm, mit and owl are shortcuts for the resources where the used vocabularies are defined (e.g., owl as shortcut for <http://www.w3.org/2002/07/owl#>). Line 2 defines the mapping from the resource `&uic;jpeg` to the resource `urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:4` which represents JPEG as a reference to a classification scheme term. Line 3 defines which string representation for JPEG should be used in UAProf descriptions. Lines 4 and 5 define all representations for the image/jpeg MIME media type. Line 6 uses standard OWL syntax to define that the resource `&uic;jpeg` is different from the resource `&uic;bitmap`.

Listing 5. Mapping MIME media type image/jpeg to  
urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:4.

```

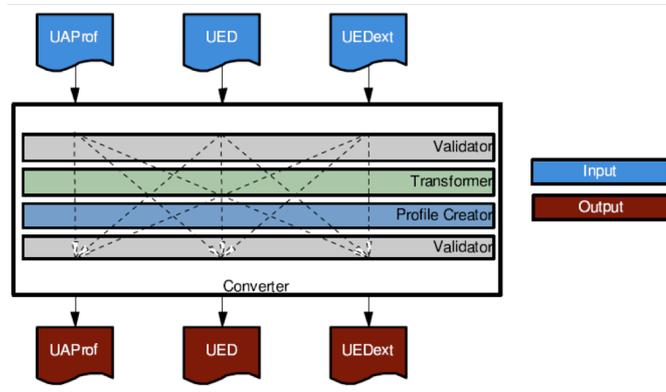
1 <uic:KluItecImage rdf:about="&uic;jpeg">
2   <csm:mapsToMpegResource
3     rdf:resource="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:4"/>
4   <mit:hasMimeType
5     rdf:datatype="&xsd:string">image/jpeg</mit:hasMimeType >
6   <mit:hasMimeRepresentation
7     rdf:datatype="&xsd:string">image/jpeg</mit:hasMimeRepresentation >
8   <mit:hasMimeRepresentation
9     rdf:datatype="&xsd:string ">image/jpg</mit:hasMimeRepresentation >
10  <owl:differentFrom rdf:resource="&uic;bitmap "/>
11  <!-- ... -->
12 </ uic:KluItecImage >

```

## 5 Implementation Details

The aim of this section is to provide an overview of our implementation that currently performs a mapping between UAProf descriptions and UEDs (and vice-versa). The high-level architecture is depicted in Figure 3 and comprises three components:

- Validator.
- Transformer.
- Profile Creator.

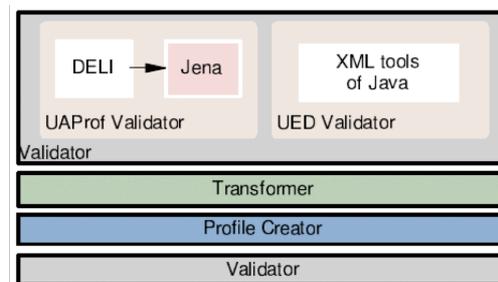


UAProf ... User Agent Profile  
UED ... Usage Environment Description  
UEDext ... Usage Environment Description extensions

Figure 3. High-Level Architecture of the UAProf/UED Mapping Implementation.

The *Validator* is responsible for validating incoming and outgoing UAProfs and UEDs. If the received profile is a UED profile, a transformation to an RDF/XML document is needed for further processing which the *Transformer* accomplishes. UAProfs need not be translated because they are already written in RDF/XML syntax which is a requirement of the profile creator. The *Profile Creator* queries data from the profile data which is available in a consistent syntax and creates the desired profile as output which is again checked by the validator before it is delivered. These three components are further detailed in the following.

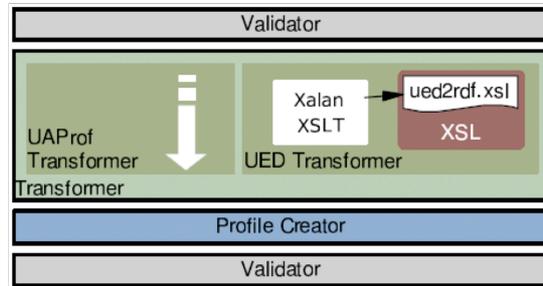
**Validator** (cf. Figure 4). The purpose of this component is to validate instances against its specification. This is performed both for inputs and outputs of our implementation. For UAProf we have integrated the DELivery context LIBrary (DELI) [Bu08] which is one of some rarely available tools that are able to validate UAProfs and to extract data from these documents. As the UED schema is an XML schema we have used standard XML schema validation tools such as the Java built-in XML validation Application Programming Interface (API).



DELI ... A Delivery Context Library For CC/PP and UAProf  
 Jena ... A Semantic Web Framework for Java  
 UAProf ... User Agent Profile  
 UED ... Usage Environment Description

Figure 4. Architecture of the Validator.

**Transformer** (cf. Figure 5). The transformer is responsible for translating the input instances into an integration model based on RDF as already introduced in Section 4. Therefore, we have implemented style sheets based on the Extensible Stylesheet Language Transformation (XSLT) [C199], i.e., only one style sheet is required for each delivery context description language keeping the overall approach scalable.



UAProf ... User Agent Profile  
 XSLT ... Extensible Stylesheet Language  
 for Transformations  
 XSL ... Extensible Stylesheet Language  
 UED ... Usage Environment Description

Figure 5. Architecture of the Transformer.

**Profile Creator** (cf. Figure 6). Finally, this component generates the designated target delivery context description based on the integration model. In order to query the RDF-based integration model we have used SPARQL Protocol And RDF Query Language (SPARQL) [PS08] and A SPARQL Processor for Jena (ARQ) [HP08] as the actual query engine. The implementation adopts predefined templates and queries to generate desired output format based on the integration model.

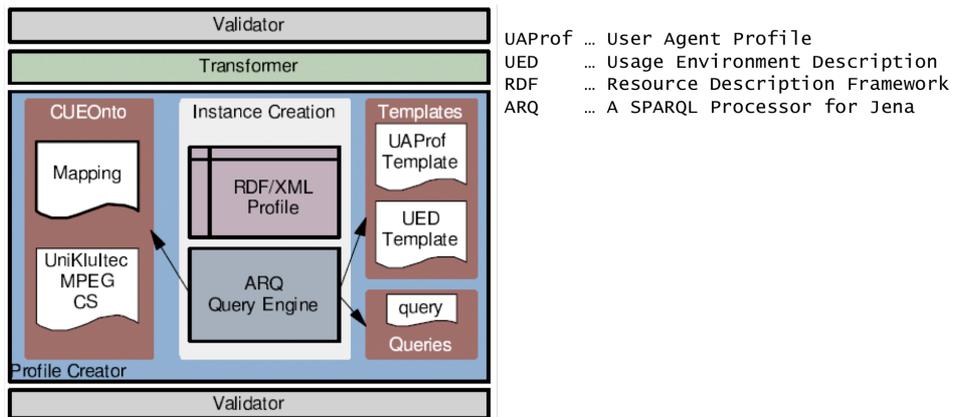


Figure 6. Architecture of the Profile Creator.

## 6 Conclusions and Future Work

In this paper we have presented a model that allows one to map context delivery descriptions between different formats (e.g., OMA UAProf and MPEG-21 UED) that are based on different technologies (i.e., XML Schema and RDF/OWL). For this model we have investigated state-of-the-art terminals in terms hardware and software capabilities as well as analyzed and compared existing delivery context description formats. Based on this analysis and comparison we concluded that there is a need for describing the commonalities and relationships between these description formats using a common model, i.e., following the integration model approach introduced earlier. The mapping model clusters the properties of the individual description formats based on their levels into four classes, namely *direct*, *advance*, *derive*, and *extend*. Based on these classes we have defined the integration model and formulated templates (i.e., using SPARQL and OWL) to query information from the integration model in order to generate the target context delivery format. The feasibility of the approach has been validated through a prototype and implementation details have been described in this paper.

The major findings can be summarized as follows:

- The overlap between different context delivery description formats is not that huge as expected but is clustered around those properties which are considered by the majority of applications areas (e.g., screen size, coding formats, etc.).
- Hence, the classes *direct*, *advance*, and *derive* are sufficient for most of the application areas.
- The relationship between different delivery context description formats needs to be described manually with respect to an integration model (i.e., the mapping function) and requires a thorough analysis of these formats which is sometimes cumbersome (cf. also [HL01]). For each format the mapping functions need to be defined only once with respect to an integration model.
- However, in this paper we have demonstrated that it is feasible – in principle – but requires the integration of many XML-based technologies ranging from XML Schema and RDF to SPARQL and OWL.

The following items are to be considered for future work. The integration of an OWL reasoner may be used to automatically recognize related data and extract specific information by using inference (e.g., mapping between different versions or slight syntax variations). Another future work item is a more detailed investigation of W3C's Delivery Context Ontology (DCO) and whether it can be used as the basis for the integration model for both UED and UAProf. Finally, as the newest description format under development (i.e., DCO) is based on OWL it confirms our decision to use OWL as underlying technology for the integration model.

## 7 References

- [Ax08] IST-FP6: AXMEDIS (Automating Production of Cross Media Content for Multi-channel Distribution), <http://www.axmedis.org/> (last accessed: December 2008).

- [Bu08] Butler, M.: DELI: A Delivery Context Library For CC/PP and UAProf, October 2006. <http://delicon.sourceforge.net/> (last accessed: December 2008).
- [CI99] Clark, J.; (ed.): XSL Transformations (XSLT) Version 1.0, W3C Recommendation, November 1999. <http://www.w3.org/TR/1999/REC-xslt-19991116> (last accessed: December 2008).
- [Da06] IST-FP6: DANAE (Dynamic and Distributed Adaptation of scalable multimedia coNtent in a context-Aware Environment), <http://danae.rd.francetelecom.com/> (last accessed: December 2008).
- [EBU09] EBU Metadata Classification Scheme Maintenance Form, [http://www.ebu.ch/metadata/maintenanceforms/index\\_cs.html](http://www.ebu.ch/metadata/maintenanceforms/index_cs.html) (last accessed 2009).
- [En08] IST-FP6: ENTHRONE (End-to-End QoS through Integrated Management of Content, Networks and Terminals), <http://www.ist-enthrone.org/> (last accessed 2008).
- [FB96] Freed, N.; Borenstein, N; (eds.): Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, RFC 2064, November 1996.
- [GLS06] Gimson, R.; Lewis, R.; Sathish, S.; (eds.): Delivery Context Overview for Device Independence, W3C Working Group Note, March 2006. <http://www.w3.org/TR/di-dco/> (Last accessed: December 2008).
- [H01] Hunter, J.: Adding Multimedia to the Semantic Web - Building an MPEG-7 Ontology, Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford, August 2001; pp. 261-281.
- [HL01] Hunter, J.; Lagoze, C.: Combining RDF and XML Schemas to Enhance Interoperability Between Metadata Application Profiles, Proceedings of the 10th World Wide Web Conference, Hong Kong, May 2001.
- [HP08] HP Laboratories: ARQ - A SPARQL Processor for Jena, <http://jena.sourceforge.net/ARQ/> (last accessed: December 2008).
- [Ki07] Kiss, C.; (ed.): Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 2.0, W3C Working Draft, April 2007. <http://www.w3.org/TR/CCPP-struct-vocab2/> (last accessed: December 2008).
- [KI04] Klyne G.; et.al. (eds.): Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0, W3C Recommendation, January 2004. <http://www.w3.org/TR/CCPP-struct-vocab/> (last accessed: December 2008).
- [LF08] Lewis, R.; Fonseca, J.M.C.; (eds.): Delivery Context Ontology, W3C Working Draft, April 2008. <http://www.w3.org/TR/dcontology/> (last accessed: December 2008).
- [MSS02] Manjunath, B.S.; Salembier, P.; Sikora, T.; (eds.): Introduction to MPEG-7: Multimedia Content Description Interface, Wiley & Sons, April 2002.
- [Mv04] McGuinness, D.L.; van Harmelen, F.; (eds.): OWL Web Ontology Language Overview, W3C Recommendation, February 2004. <http://www.w3.org/TR/owl-features/> (last accessed: December 2008).
- [OM06] Open Mobile Alliance Ltd: User Agent Profile, Approved Version 2.0. Technical report, OMA, February 2006.
- [PS08] Prud'hommeaux, E.; Seaborne, A.: SPARQL Query Language for RDF, W3C Recommendation, January 2008. <http://www.w3.org/TR/rdf-sparql-query/> (last accessed: December 2008).
- [VT05] Vetro, A.; Timmerer, C.: Digital Item Adaptation: Overview of Standardization and Research Activities, IEEE Transactions on Multimedia, vol. 7, no. 3, June 2005; pp. 418-426.
- [W308] w3development.de: UAProf profile repository. <http://w3development.de/rdf/uaprof-repository/> (last accessed: December 2008).

## 8 Appendix

Table 2. HW/SW Characteristics of Desktop PC/Workstation, Notebook/Tablet PC, Sub-Notebook/Netbook, Handheld, Smartphone, and Mobile Phone

	<b>Desktop PC/Workstation</b>	<b>Notebook/Tablet PC</b>	<b>Sub-Notebook/Netbook</b>	<b>Handheld</b>	<b>Smart Phone</b>	<b>Mobile Phone</b>
<i>Performance</i>	High performance with dual, triple or quad core processors	Power saving dual core processor	Medium performance special power saving processors	<i>Medium and low performance with various processors</i>		<i>Low performance with various processors</i>
<i>Display</i>	17" - 30", multiple displays	12" - 20"	7" - 12"	4" (or smaller), color and monochrome	1,8"-3,5", color (and monochrome)	1,8"-3,5", color and monochrome
<i>Storage</i>	Up to 2 TB	Up to 1 TB	2GB to 160GB	8MB-256MB (also 16GB)	Up to 16GB (expandable, e.g., microSD-Cards)	Up to 64MB
<i>Memory</i>	Up to 8 GB	Up to 4 GB	Up to 1 GB	Similar to permanent storage	Often 128MB	Similar to permanent storage
<i>Network</i>	Gigabit Ethernet, WLAN, cable/xDSL, etc.	Gigabit Ethernet, WLAN, Modem, Bluetooth, (UMTS, HSDPA)		IrDA, Bluetooth, WLAN, GSM, GPRS, UMTS, HSDPA, GPS	IrDA, Bluetooth, USB, WLAN, GSM, GPRS, EDGE, UMTS, HSDPA, HSUPA, GPS, DVB-T, etc.	IrDA, Bluetooth, GSM, GPRS, UMTS
<i>Power</i>	AC	AC, DC, battery 2-6h (typically 2,5h)		DC, battery (3-15h)	Battery 100h with normal use (up to 400h Standby)	
<i>User I/O</i>	Keyboard, Mouse, Monitor, Loudspeaker, Microphone, Webcam, etc.	Keyboard, Touchpad, LCD, Touchscreen and Stylus, Loudspeaker, Microphone, Webcam		Touchscreen and Stylus, Loudspeaker, Microphone	Touchscreen, QUERTY-Keyboards (often very small), limited Keyboards, Loudspeaker, Microphone, Cameras	Phone Keyboard, (joystick or navigation buttons)
<i>Extensibility</i>	USB, Firewire, PCIe, PCI, etc.	USB, Firewire, eSATA, ExpressCard, PCMCIA	USB	CF and SD card slot for devices and memory, USB	USB, TV-out	Usually not supported
<i>OS</i>	Windows Vista, Windows XP, Mac OS, Linux, etc.		Windows XP, special Linux versions, Windows Mobile	Windows Mobile, Palm OS	Symbian OS, Windows Mobile, RIM Blackberry, Apple OS X (special edition for iPhone), Google Android (Linux and Java based)	Symbian OS, other proprietary systems
<i>Software</i>	Almost unlimited		Limited to Operating System and hardware capabilities	Limited to Operating System and hardware capabilities		Preinstalled, (sometimes expandable through Java-Midlets)