From UAProf towards a Universal Device Description Repository

José Quiroga, Ignacio Marín, Javier Rodríguez, Diego Berrueta, Nicanor Gutiérrez, and Antonio Campos

R&D Department, Fundación CTIC,
C/ Ada Byron, 39 – 33203 Gijón, Spain
{jose.quiroga,ignacio.marin,javier.rodriguez, diego.berrueta,nicanor.gutierrez, antonio.campos}@fundacionctic.org

Abstract. Techniques to create software and content that adapt to different apparatus require gathering information about device features. Traditionally, Device Description Repositories (DDRs) have provided limited descriptions, in terms of description granularity and of the amount of devices included. A Universal DDR (UDDR) would allow any software developer or content creator to have complete, up-to-date and trustworthy device descriptions for any application domain. Collaboration of all stakeholders in the adaptation business would be necessary to populate the UDDR, but without compromising the quality of the information. Device manufacturers usually publish first-hand device descriptions using UAProf. Unfortunately, UAProf documents are known to contain mistakes or inaccurate/incomplete information. This work suggests a multi-step process to manipulate UAProfs in order to correct their most common mistakes, to extend their expressiveness and to allow amendments from different contributors. More specifically, amendments are annotated with provenance information, enabling device description consumers to decide whether to trust them.

Keywords: device description repository, UAProf, CC/PP, software adaptation, content adaptation, RDF, data provenance, profile resolution.

1 Introduction

The development of software solutions for the vast heterogeneity of connected computing devices in the market is a grand challenge. Multi-device development is a great opportunity for the software industry, considering the extraordinary amount of connected devices far beyond traditional PCs. Mobile and portable devices, set-top boxes, home media players, in-vehicle devices and other embedded devices are some examples of such device diversity. For instance, mobile devices, which are an obvious case of device fragmentation, have widely spread over the last years. The International Telecommunication Union claims in the 2010 edition of their annual "Measuring the Information Society" [1] report that 67% of the population of the world was subscribed to a mobile cellular connection by the end of 2009. When compared to global penetration of fixed broadband subscriptions (7%), which have traditionally been one of the most relevant driving forces of the software industry, the business opportunity seems to be significant.

However, writing software which adapts to multiple operating systems and device characteristics and capabilities is a very complex issue. One of the key aspects of this type of adaptive software is the availability of a database containing device descriptions. Organizations devoted to content and software creation and provisioning for multiple devices and/or software platforms keep their own databases for their internal developments –Google or Amazon, for example. Some others, as DeviceAtlas [2], publish commercial device databases for creators of adaptive software and content creators to use it. Additionally, there are open-source and/or free efforts (e,g., WURFL [3] and maDDR [4]) which intend to populate and maintain device databases in a collaborative manner. Even commercial proprietary device databases must be considered collaborative databases, as their maintainers watch the novelties regularly added to open and free device databases. The two greatest problems to populate a device database are that (P1) there is usually a delay between the release of a new device in the market and the insertion of its description in the database and that (P2) there is no annotation about the provenance of the information contained in the database. As an example of (P1), the WURFL database version dated on 12/31/2010 adds 11 devices to the previous version (12/6/2010). In between early owners of those devices might experience a lack of support for their corresponding new devices. This is an important business problem, as these users are candidates to be intensive consumers of data services. In what regards to (P2), even when a device description is added to the WURFL database, it may happen that it is an incomplete description to be refined in the future releases. Therefore, it is important to know who refines that information and how such information is obtained for the sake of trustworthiness. The balance between getting device descriptions of new device models as soon as possible and waiting for trustworthy device descriptions to be available is the principal warhorse in device databases guiding software and content adaptation. The existence of trustworthy collaborative device databases would invite an increasing amount of software developers and content creators to face the challenge of multi-device development.

Device description databases have been named as Device Description Repositories (DDRs) by the World Wide Web Consortium (W3C [5]) through their already extinct Device Independence Working Group [6]. It is important to note that DDRs include device descriptions which include information known a priori. In this way, a client device can perform a request to a server system and the server can subsequently obtain evidences about the identity of the device. These evidences can be used to query a DDR in order to find out the actual identity of the device and its software and hardware features. This process enables the adaptation of the response from the server to the client (for example, provisioning an application in the binary format accepted by the client device or some type of content, i.e. a raster image, in the format expected at the client side).

The authors of this work are developing a universal DDR to be populated and maintained in a collaborative way. Because of this, the authors have decided to name the proposed DDR as UDDR (Universal Device Description Repository). The term universal means that it intends to capture all the device features required by any type of application and content transformation. Therefore, one of the most important features for the UDDR is that its various contributors have mechanisms to include new device properties which make sense for novel application domains.

In what regards to the collaborative nature of the UDDR, access to the information that it will keep will be provided by means of an Application Programming Interface (API) which will seamlessly enforce submission of new device descriptions or properties, although this feature is out of the scope of this article. Although collaborative DDRs as WURFL have been successful, organizations tend to avoid contributing their amendments back to the DDR.

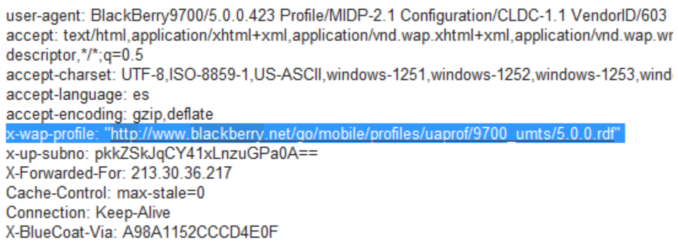
The level of granularity of the information about each device is an important issue to obtain a universal DDR. So far, DDRs have mainly focused on the description of the web browser of the device and other aspects related to the browser: supported images, sound and video formats or encryption algorithms (related to SSL and HTTPS), for instance. There is a lack of fine-grained information about the operating system, non-HTTP protocol support (RTSP, SIP and others), the presence of optional APIs (supported JSRs, for example, in the case of Java), or about technologies for connectivity (such as supported Bluetooth profiles).

In order to have a sufficient expressivity, UAProf [7] has been considered by the authors of this work as the starting point to develop a device description framework for the UDDR. The reason for this decision is that its ancestor technology (Composite Capability/Preference Profiles or CC/PP [8], which is based on RDF [9]) provides an extensible mechanism for device description which will permit the conversion of UAProf documents to UDDR-Profiles, as suggested later in this article. UDDR-profiles are individual device descriptions in the UDDR, analogous to the concept of UAProf profiles. One of the advantages of this approach is to start populating the UDDR with existing UAProf instances published by device makers. Unfortunately, UAProf was not intended to be a language to express device descriptions in a DDR but for device manufacturers to announce device characteristics to software developers and content creators.

This article is organized in sections. Section 2 provides a technological background in order to explain what UAProf is and how and why it was invented. Section 3 describes the problem suggested in the article: why device descriptions in UAProf documents are not sufficient as information items in a Universal Device Description Repository and an introduction to how they may be manipulated to obtain device descriptions useful for a UDDR. Section 4 lists and comments previous research works devoted to the improvement of the existing UAProf specification. In Section 5, the requirements expected for the UDDR-profile are actually explained. Section 6 comments the steps proposed by the authors to populate the UDDR after original UAProf documents published by device manufacturers. Section 7 describes profile resolution, the suggested mechanism to avoid ambiguity and contradiction when querying the UDDR. Section 8 details the step-by-step process to transform the UAProf description of an actual device into its UDDR-Profile description. Sections 9 and 10 present the conclusions obtained and the future work to be accomplished in order to develop the UDDR after the UDDR-Profile definition.

2 Technological Background

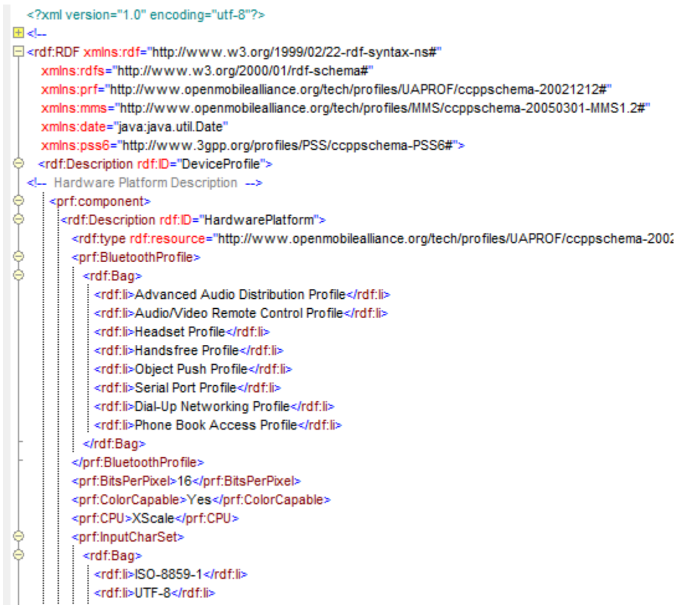
UAProf is a vocabulary proposed by the Open Mobile Alliance (OMA [10]) from the CC/PP specification defined by the W3C through the extinct Device Independence working group, which is expressed in RDF. UAProf profiles are created as documents expressed in the homonymous vocabulary. They are referenced by means of a URI provided by some web browsers (generally, a significant amount of mobile web browsers) in their HTTP requests. As an example, Figure 1. shows the *x-wap-profile* header contained in an HTTP Request as submitted by the web browser of a BlackBerry 9700 device.



```
user-agent: BlackBerry9700/5.0.0.423 Profile/MIDP-2.1 Configuration/CLDC-1.1 VendorID/603
accept: text/html,application/xhtml+xml,application/vnd.wap.xhtml+xml,application/vnd.wap.wr
descriptor,*/*;q=0.5
accept-charset: UTF-8,ISO-8859-1,US-ASCII,windows-1251,windows-1252,windows-1253,wind
accept-language: es
accept-encoding: gzip,deflate
x-wap-profile: "http://www.blackberry.net/go/mobile/profiles/uaprof/9700_umts/5.0.0.rdf"
x-up-subno: pkkZSkJqCY41xLnzuGPa0A==
X-Forwarded-For: 213.30.36.217
Cache-Control: max-stale=0
Connection: Keep-Alive
X-BlueCoat-Via: A98A1152CCCD4E0F
```

Fig. 1. URI associated to the UAProf which describes a BlackBerry 9700

The URI for a UAProf profile may also be found in the value of other headers in HTTP Requests, such as *Profile*, *wap-profile* or *xx-profile* (with xx being a number, usually indicated in an additional *Opt* header), depending on the choice of different device manufacturers and HTTP Proxy implementations. An example of the content of a UAProf document can be found in Figure 2.



```
<?xml version="1.0" encoding="utf-8"?>
<!--
<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:prf="http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20021212#"
  xmlns:mms="http://www.openmobilealliance.org/tech/profiles/MMS/ccppschema-20050301-MMS1.2#"
  xmlns:date="java:java.util.Date"
  xmlns:pss6="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#">
  <rdf:Description rdf:ID="DeviceProfile">
<!-- Hardware Platform Description -->
    <prf:component>
     <rdf:Description rdf:ID="HardwarePlatform">
       <rdf:type rdf:resource="http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-2002
       <prf:BluetoothProfile>
         <rdf:Bag>
           <rdf:li>Advanced Audio Distribution Profile</rdf:li>
           <rdf:li>Audio/Video Remote Control Profile</rdf:li>
           <rdf:li>Headset Profile</rdf:li>
           <rdf:li>Handsfree Profile</rdf:li>
           <rdf:li>Object Push Profile</rdf:li>
           <rdf:li>Serial Port Profile</rdf:li>
           <rdf:li>Dial-Up Networking Profile</rdf:li>
           <rdf:li>Phone Book Access Profile</rdf:li>
         </rdf:Bag>
       </prf:BluetoothProfile>
       <prf:BitsPerPixel>16</prf:BitsPerPixel>
       <prf:ColorCapable>Yes</prf:ColorCapable>
       <prf:CPU>XScale</prf:CPU>
       <prf:InputCharSet>
         <rdf:Bag>
           <rdf:li>ISO-8859-1</rdf:li>
           <rdf:li>UTF-8</rdf:li>
```

Fig. 2. Part of the UAProf document which describes a BlackBerry 9700 device

It is important to note that UAProf documents contain static device descriptions. This means that they contain information that is known *a priori*, such as screen resolution, Bluetooth profiles supported or the web browser(s) installed from factory. Some examples of dynamic device information are battery charge level, or screen orientation (landscape, portrait).In the early times of mobile web development, HTTP headers were parsed in search of a URI referencing the UAProf document describing the device for a later adaptation of web content. An obvious improvement to that technique was caching UAProf documents at the web server side, thus giving birth to the first UAProf-based Device Description Repositories. The main goals of this approach were (1) reduction of device identification time by avoiding repetitive HTTP interaction to access a UAProf document already accessed before, and (2) availability of device description even when the web server containing the original UAProf document is temporarily or permanently unavailable.

3 Description of the Problem

Unfortunately, UAProf documents contain mistakes despite the fact that they are generated by device manufacturers themselves. Some mistakes are caused by mere copying-and-pasting errors when device makers release a new model which is an evolution of a previous device –proofs of the copy-and-paste technique are shown in Section 7. Some others are caused by engineers not being aware of the restrictions imposed by the CC/PP and UAProf specifications. Therefore, UAProf documents are made available without being validated against the UAProf Schema in the Appendix A of the specification.

Once that an original UAProf is cached, the obvious need to correct mistakes appears. In addition to this need, DDR maintainers might want to add new properties to the device description in UAProf and to enhance the expressivity power of device descriptions in order to overcome some of the limitations of the vocabulary and of the overall CC/PP framework, as suggested in the next sections of this document.

Both UAProf and CC/PP offer an interesting framework for device description. They have been providing device descriptions used by the software industry over the last decade. Hundreds of device models expose their software and hardware characteristics by means of a UAProf document which may be cached and then enhanced. Information enhancements or amendments are required to obtain a UDDR, as suggested by some of the requirements listed in Section 5. These requirements do not only include changes in mistaken information or adding new properties to the existing categories grouping device features. They may also include new sections in UAProf documents reorganizing the categorization of device features.

Moreover, there is an aspect in device description manipulation which is the inclusion of information about the authorship of amendments to previous content. One of the foundations of the future UDDR is its collaborative nature. Collaborative approaches in the development of a DDR are not innovative, with WURFL as the flagship. Still, one of the uncovered problems of DDRs being maintained by different contributors is to keep track of contributions provenance [11] as an estimation of the fidelity of the information contained in each device description. One of the requirements in the UDDR is to ensure that information can be updated by any contributor as soon as a new device (or device feature) is detected. Thus, device descriptions will be patched by different contributors and it is likely that not all the consumers of device descriptions will trust all the amendments. They might subscribe only to changes from trusted contributors, therefore establishing their own balance between quick availability of new descriptions and relying only on trustworthy information in the DDR.

4 Related Work

Previous research work has considered UAProf and CC/PP limitations. The first efforts in the analysis of these specifications were done by Mark Butler, at the HP Labs research group, through several white papers. [12] and [13] reflect the absence of a formal specification for profile resolution, the lack of a mechanism to allow combining profiles expressed in different vocabularies and the need for a formal definition of vocabularies –unfortunately, often indicated as comments in UAProf profiles. These problems were notified later to the W3C Device Description Working Group by means of a Position Paper [14]. Moreover a CC/PP-UAProf validation tool (DELI) [15] was created by the same author.

In [16], a CC/PP-based vocabulary is proposed in order to represent more detailed context information for content and software adaptation. One of the most relevant problems found in CC/PP is the organization of device description in two layers. This forces the use of undesired syntactic sugar to express some definitions and relationships between device properties. An example of relationship is the required existence of an attribute X in the device description when another attribute Y exists. Another relevant issue mentioned in the article is the need to provide CC/PP with a mechanism for profile resolution (which is commented later in subsection 5.7), as previously suggested by Butler.

Related to the aforementioned work, an interesting study of the limitations in CC/PP and UAProf is carried out in [17]. Its conclusions state that basing CC/PP on RDF does not seem very appropriate as it basically models a hash table with name-value pairs. The study also considers that CC/PP and UAProf describe the data structures in which device profiles are represented but they do not provide an API to access the properties contained. Finally, one of the most evident problems is the diversity in vocabularies actually used by different device makers. The authors consider that a mapping mechanism between vocabularies should be provided, as it was also previously noted by the rest of related work.

5 Requirements for UDDR-Profile

In order to use UAProf documents as the central element for a Device Description Repository, we propose a number of extensions to UAProf driven by the following requirements.

5.1 Provenance Support

All the contributions must be associated to the author of such changes. The original UAProf lacks mechanisms for tracking this information, which is essential to enable a trustworthy access to a collaborative DDR. For instance, it may be relevant to know if a contribution has been originated from the decisions of a human or from a software probe which has gathered device capabilities in an automated way.

Information providing more details about amendments to the original UAProf may also be included [16] in order to allow developers to estimate the trustworthiness. For instance: temporal information (freshness and history), accuracy, confidence in correctness and digital signature to ensure the authorship.

Using RDF named graphs [18] is the proposed mechanism to fulfill this requirement. Named graphs enable the annotation of information subsets with provenance metadata about authorship and change tracking. This approach has already been applied to other domains, such as biological data resources [19]. Although the current specification of RDF does not contemplate named graphs, the SPARQL [20] query language does, and there is ongoing work at W3C to revise RDF in this direction. This fact leads the authors to avoid introducing all the expressivity requirements in the proposed UDDR-Profile and therefore use resolution rules which will be triggered after the SPARQL queries to the UDDR.

In order to increase the value of provenance annotations, it is convenient to re-use existing vocabularies, such as the ones listed in the W3C Provenance Incubator Group final report [21], as well as existing identifiers for entities, e.g., URIs of FOAF profiles to characterize the agents involved in the data evolution. All these mechanisms will be used in the UDDR-Profile format.

5.2 Correct Separation of UDDR-Profile Description Aspects

Sometimes, information about a generic software component (for instance, web browser) is provided in a UAProf description. Actually, a device may include more than one implementation of that software component (for instance, two or more web browsers) without its UAProf clarifying whether both of them are compliant to the description.

A specific example is the inclusion of the MIME types supported by “the web browser” when the device has two browsers installed from factory. This can be found in the UAProf description of the HTC Touch mobile phone[1], in which the coexistence of two web browsers is declared as:

<rdf:Description rdf:ID="BrowserUA_Opera">
<rdf:Description rdf:ID="BrowserUA_pIE">

A simple test accessing different media types from both browsers with such device indicates that some MIME types supported by the device are unevenly supported, contradicting what is stated in the *<prf:CcppAccept>* property. This is the case of the *image/svg+xml*, not included in that property, although it is supported by the Opera Mobile browser –but not by the Pocket Internet Explorer. The *<prf:CcppAccept>* property is attached to the resource *<rdf:Description rdf:ID="SoftwarePlatform">*, not to the specific instance of the browser. Consequently, UAProf lacks the ability to express these implementation details.

[1] `http://www.htcmms.com.tw/gen/HTC_Touch_HD_T8282-1.0.xml`

These limitations invite to a re-engineering of the structure of UAProf. It is likely that some attributes may need to be assigned to more specific resources. Although these changes may initially introduce an apparent redundancy due to the need to state large chunks of shared features, this drawback can be mitigated by a mechanism that permits to refine resource descriptions, as explained next.

In order to obtain a correct separation, it is necessary to eliminate expressivity restrictions in CC/PP which do not exist in RDF. CC/PP defines a hierarchical structure based on two main levels (components and attributes). It means a significant restriction over RDF, as its expressiveness is considerably reduced [16]. In practical terms, CC/PP could be seen as a kind of big table in the form key-value, where content providers are very restricted in what regards to the semantic relationships they can define.

In the UDDR-Profile format, a more comprehensive usage of the RDF model will be encouraged in order to express more complex hierarchies.

5.3 Grouping of Common Attributes

Hardware and software components from the same vendor or the same family usually share common features. It is just natural to organize the information to minimize redundancy by creating descriptions that refine or extend other descriptions. The representation of these relations and their semantics are not straightforward in RDF. We propose the use of property paths featured in the upcoming revision of SPARQL [22], to introduce basic reasoning capabilities on top of RDF datasets. For instance, variable-length property paths make it possible to formulate queries that find the value of a property that is associated to the resource directly or indirectly.

As shown in Figure 3. , there is a top level profile that defines the features of *Symbian 9.4* operative system. Then, there is another profile with the features of *Series60 5th Edition* that will extend, and override if necessary, the top level profile. At last, in the lowest level, there is a profile of the device *N97*, which extends, and override if needed, both higher level profiles.

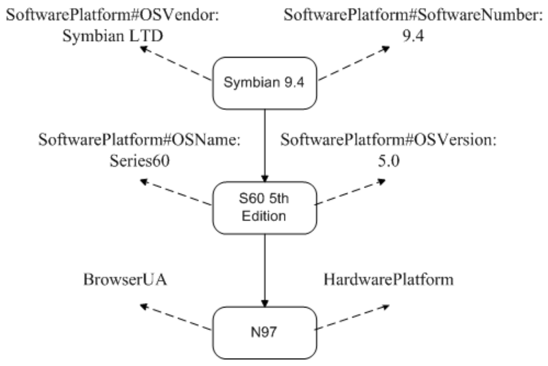

SoftwarePlatform#OSVendor: Symbian LTD
SoftwarePlatform#SoftwareNumber: 9.4
Symbian 9.4
SoftwarePlatform#OSName: Series60
SoftwarePlatform#OSVersion: 5.0
S60 5th Edition
BrowserUA
HardwarePlatform
N97

Fig. 3. Example of description hierarchy for some Symbian devices

5.4 Value Normalization

In some cases, the different values that an attribute in UAProf may take are strictly defined. This leads to incoherent device descriptions in the sense that, for instance, an attribute is valued with a string for which there is no formally specified format. As an example, the authors have reviewed the values for the attribute *BluetoothProfile* in a set of actual UAProf description files downloaded from different manufacturers' repositories. A quick read after the values accepted show that the support for the AVRCP Bluetooth profile is noted with different strings (in alphabetical order): *"audio vedio remote control"*, *"Audio Video Remote Contorl Profile"*, *"Audio Video Remote Control"*, *"Audio Video Remote Control – Target"*, *"Audio Video Remote Control Profile"*, *"audiovideoremotecontrol"*, *"AVRCP"*, etc.

Some other typical inconsistencies include the expression of the values of a same attribute by means of different types. Following the same procedure for the *BluetoothProfile* attribute, a study of the *NumberOfSoftKeys* attribute has been carried out. In addition to the expected *xsd:integer* values (0, 1, 2, 3, 4, 5, ..), a "*None*" value has been found for many Motorola devices, such as the A1600. This is due to different versions of the CC/PP schema, UAProf vocabulary and third-party schemas (such as those from the 3GPP) published over time.

For the sake of interoperability and to simplify queries to the UDDR, it becomes necessary to enforce a tighter control over the values of the attributes. Some constraints, such as type restrictions, can be defined at a syntactic level, whereas others may require defining a well-known and extensible set of entity URIs (e.g., for Bluetooth profiles). The UDDR will enforce the usage of the latest versions of schemas and vocabularies used in UAProf, defining conversion rules from previous versions to the most recent ones. Additionally, strict values for the various properties in device descriptions will be enforced, with mapping mechanisms defined after expertise –to avoid, for instance, the "string hell" in Bluetooth profile definitions.

5.5 Uniform Device Description Granularity

Distinct device descriptions of similar devices provide different granularity. For example, some Java devices include a list of the JSRs available whereas some do not. Such divergence exists even in device descriptions of device models of the same device family, as in the case of the GT B7330 Omnia PRO[2], with supported JSRs listed, and GT B7300 Lite[3], without such description detail. Even in a same UAProf document, two analogous components may provide different information granularity. For example, both web browsers in the UAProf describing the HTC Mini[4] provide information about the XHTML version supported. However, supported XHTML modules are listed for the Opera Mini but not for the Pocket Internet Explorer.

In UDDR, guidelines will guide the process to publish descriptions, so a uniform granularity baseline can be set. Moreover, the collaborative aspect of the repository evolution may help to reconcile divergences in the granularity level.

[2] `http://wap.samsungmobile.com/uaprof/GT-B7330V_2G.xml`

[3] `http://wap.samsungmobile.com/uaprof/GT-B7300.xml`

[4] `http://www.htcmms.com.tw/gen/HTC_HD_mini_T5555-1.0.xml`

5.6 Appropriate Semantic Expresiveness

Vocabulary extensibility is a key issue on using the CC/PP-UAProf stack [13], which is defined by XML/RDF namespaces. However, using multiples vocabularies causes interoperability problems. For instance, the *BrowserUA#HtmlVersion* property defined in both 2002[5] and 2007[6] versions of the CC/PP Schema will not be interpreted in the same way by a CC/PP processor.

An additional aspect to be considered is the definition of relational constraints such as cardinality (e.g. the number of values in a bag) or existence/absence of attributes within the profile [16]. This kind of restrictions makes necessary to use validation tools. An example of validation tool is the DELI validator [23] commented in section 4, but this framework presents some weakness (again [17]). Therefore, a new semantic validation is needed based on rules or query engines.

Schema and ontology languages on top of RDF, such as RDF Schema and OWL, bring in the ability to declare certain constraints. However, some of the aforementioned integrity constraints are beyond the expressiveness of these languages. Even those that are possible usually lead to consequences that are not intuitive for people trained in databases and XML. Thus, it may make sense to introduce an *ad hoc* validation tool that implements the logic behind semantic restrictions. This kind of tools is not uncommon and is often implemented by means of rule or query engines. Interestingly, the upcoming version of the SPARQL query language for RDF may be a good candidate for the task, due to the aggregated functionality and ASK queries.

Finally, UAProf documents offer many chances for improvements in the light of recent developments in Linked Data [24]. More specifically, UAProf documents, profiles and the resources they contain should be assigned HTTP-resolvable URIs. By doing so, they become extensible and linkable, and new opportunities appear for re-using shared resources.

5.7 Profile Resolution

One of the main difficulties when using CC/PP and UAProf is to carry out the resolution of profiles properly. The complexity of this issue stems from the fact that the same attributes may be defined in different source profiles. These profiles might have been created by different providers, so possible contradictions might be found. The notion of *profile resolution* in this document refers to the process of determining the correct values for each attribute in order to make up a definitive profile. Because different contributors may provide additional or overriding information, the resolution process must analyze all the available data to determine the final attribute values.

As stated in Section 4, one of the main claims of the scientific community in what regards to profile resolution is the absence of advanced formal mechanisms to automate this process. On the one hand, profiles resolution is out of the scope of the CC/PP recommendation. On the other hand, UAProf provides a mechanism to treat profile resolution. Unfortunately it presents some important drawbacks. UAProf incorporates a resolution rule associated to each attribute in the RDF schema as a

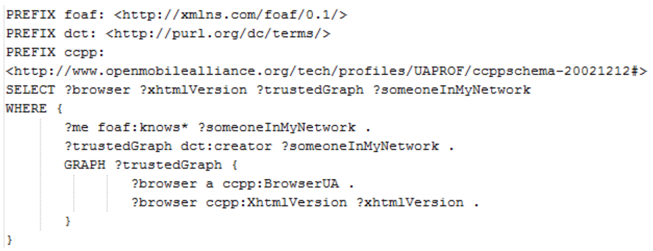
[5] `http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20021212`

[6] `http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20070511`

comment. Each attribute must take one of the following values: “locked”, “override” and “append”. For instance, the OSVersion attribute has been defined as “locked” in the RDF schema. It means that a profile provider would not be allowed to take advantage of an existing profile (e.g. a profile from a previous version of the device) and extend it with a new value for the OSVersion attribute.

The first limitation of this approach is that the resolution rules are expressed in the RDF schema itself. It means that, a given attribute will be associated to a concrete resolution rule regardless of the intention of the profile provider. Other important disadvantage is that data related to resolution rules is stored in the form of comments, which makes it hard to parse and process.

Our approach towards the resolution of profiles is based on two key pillars: the addition of meta-information within the profiles and the usage of advanced query mechanisms to take advantage of from the available data. The inclusion of semantic annotations, as outlined in section 5.1, allows us to resolve profiles taking into account important aspects such as the trustworthiness of amendment providers or the freshness of the provided data. In order to extract this kind of information from the profiles description, we use the SPARQL query language. By way of example, a SPARQL fragment is shown in Figure 4. to define the following query expressed in natural language: “Ask for all the XHTML versions supported by mobile web browsers, but just considering sources coming from my contacts network”. Note that for this example to make sense, all “my” social information is supposed to have been externally described by using FOAF vocabulary.



```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX ccpp:
<http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20021212#>
SELECT ?browser ?xhtmlVersion ?trustedGraph ?someoneInMyNetwork
WHERE {
        ?me foaf:knows* ?someoneInMyNetwork .
        ?trustedGraph dct:creator ?someoneInMyNetwork .
        GRAPH ?trustedGraph {
                ?browser a ccpp:BrowserUA .
                ?browser ccpp:XhtmlVersion ?xhtmlVersion .
        }
}
```

Fig. 4. Provenance-based SPARQL query

5.8 Corollary

After the previous requirements, a division of the improvements suggested for the UAProf format to obtain appropriate device descriptions in the UDDR is proposed. Firstly, some improvements affect the way in which information is handled in the various versions of the CC/PP and UAProf specifications but not breaking the UAProf format. In second place, some improvements imply exploiting the RDF nature of CC/PP-UAProf, thus breaking the original format. Finally, conflict resolution needs to be performed at querying time after SPARQL sentences.

The previous sentence suggests a new division, considering the moment when information manipulation occurs. The first and second types of manipulation take place when populating the UDDR, in that specific order –improvements keeping the UAProf format first and improvements breaking that format afterwards. The third type takes place at querying time, as previously stated.

6 Populating the UDDR

This section is intended to provide a step-by-step process to obtain the UDDR-Profile device descriptions that will be managed by the UDDR after original UAProf descriptions. There is an intermediate state for document conversion called UAProf+. A UAProf+ device description is the result of using properties and values in the original UAProf profile as expected in the latest versions of the different schemas used in the UAProf specification. The authors of this work consider that this will permit a later contribution to the Open Mobile Alliance, as indicated in Section 10. To illustrate it, each step of the profile definition process will be explained, as shown in Figure 5.

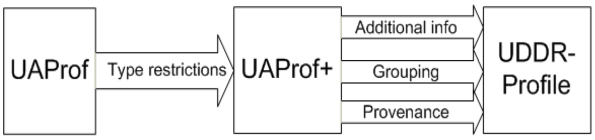


Fig. 5. Step-by-step process from UAProf to UDDR-Profile

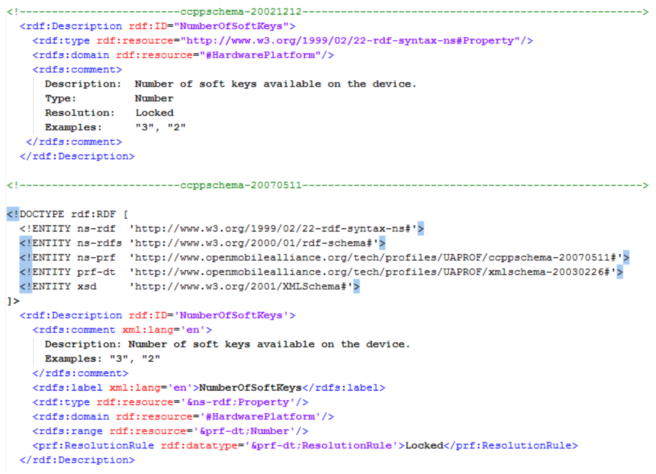
The starting point for the UDDR profile definition is an existing UAProf document as it was referenced in the *x-wap-profile* or equivalent HTTP header sent from the mobile browser to the server. Once the original UAProf is retrieved, a new version of the document, UAProf+, will be generated by the UDDR. This new UAProf+ document will impose further type restrictions in order to guarantee an appropriate document processing avoiding inconsistencies. The next step will be the addition of custom extensions to provide more information about the device and to better group the already available data. In this way, a UDDR-Profile device description is obtained after the UAProf+ device description. Extensions will be generally obtained by carrying out a set of semiautomatic tests, although annotations by a human expert will also be allowed. Finally, device descriptions are decorated by adding provenance information, which will be used in the profile resolution process.

6.1 Obtaining UAProf Documents

The initial step to start generating profile definitions for the UDDR would be providing the system with a URI referring a UAProf profile. These URIs may be obtained from product pages of device manufacturers (as it happens in Nokia's product page), by following specialized web sites and manually providing found UAProf URIs, or by analyzing the HTTP request headers coming from the device mobile browser of tested devices and check whether any of the headers points to a UAProf document If a UAProf document already exists for the device, it will be cached and used as a basic data template to be completed in step B.

6.2 Generating UAProf+ Documents

As stated in section 5.5, one of the main difficulties when working with UAProf-CC/PP is to process type restrictions. The XSD types defined in the UAProf schemas are often not sufficiently restrictive. Moreover, distinct UAProf schemas sometimes place different restrictions on the same properties. This is the case of the *NumberOfSoftKeys* property. A *"None"* value for this property would be valid according to the 2002 version of the CC/PP schema, but invalid according to the 2007 version as shown in Figure 6. In the latter case, the expected value would be *"0"* rather than *"None"*.



```
<!-------------------------ccppschema-20021212---------------------------------------------------->
  <rdf:Description rdf:ID="NumberOfSoftKeys">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#HardwarePlatform"/>
    <rdfs:comment>
      Description:  Number of soft keys available on the device.
      Type:         Number
      Resolution:   Locked
      Examples:     "3", "2"
   </rdfs:comment>
  </rdf:Description>

<!-------------------------ccppschema-20070511---------------------------------------------------->

<!DOCTYPE rdf:RDF [
  <!ENTITY ns-rdf  'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!ENTITY ns-rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
  <!ENTITY ns-prf  'http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20070511#'>
  <!ENTITY prf-dt  'http://www.openmobilealliance.org/tech/profiles/UAPROF/xmlschema-20030226#'>
  <!ENTITY xsd     'http://www.w3.org/2001/XMLSchema#'>
]>
  <rdf:Description rdf:ID='NumberOfSoftKeys'>
    <rdfs:comment xml:lang='en'>
      Description: Number of soft keys available on the device.
      Examples: "3", "2"
    </rdfs:comment>
    <rdfs:label xml:lang='en'>NumberOfSoftKeys</rdfs:label>
    <rdf:type rdf:resource='&ns-rdf;Property'/>
    <rdfs:domain rdf:resource='#HardwarePlatform'/>
    <rdfs:range rdf:resource='&prf-dt;Number'/>
    <prf:ResolutionRule rdf:datatype='&prf-dt;ResolutionRule'>Locked</prf:ResolutionRule>
  </rdf:Description>
```

Fig. 6. Difference between vocabulary definition restrictions in CC/PP Schema version 2002 and version 2007

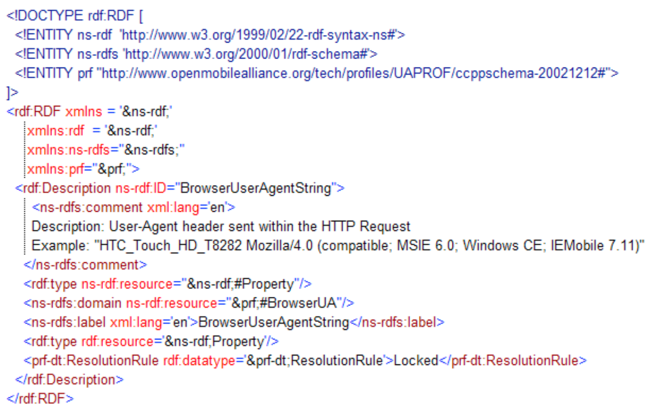
UDDR will provide a schema as restrictive as possible for the UAProf property values (as stated in section 5.3). Furthermore, a mapping mechanism to convert from those values which have been defined against a less restrictive schema into the appropriate value in the UDDR schema will be provided. For instance, each appearance of "*None*" value under the *NumberOfSoftKeys* property should be transformed into a "0" value.

The resulting document format after applying these changes to the original UAProf document has been named as UAProf+.

6.3 Generating UDDR-Profile Documents

Once the UAProf+ document has been generated, the UDDR might need to modify its structure (as exposed in 5.2 and 5.3) and to add new information to the already available data. These new extensions are required due to the universal nature of the proposed DDR and to the unevenness in the granularity of data in the existing UAProf documents (as aforementioned in 5.6).

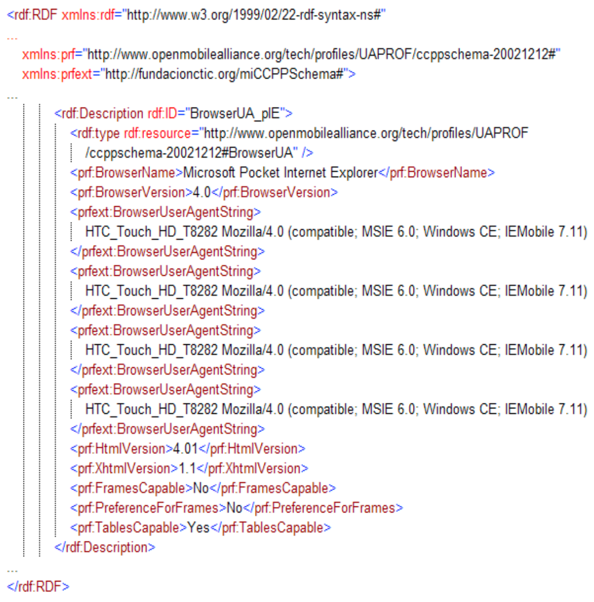
One of the extensions introduced by the UDDR is the addition of new properties over an already defined component. For instance, we might need to add a new *BrowserUserAgentString* to define all the possible User Agent strings that a given instance of the *BrowserUA* component (e.g. *BrowserUA_pIE*) might have. In order to accomplish this goal, the most recent version of the CC/PP schema (2007) has been extended. The proposed schema extension defining a new *BrowserUserAgentString* inside the *BrowserUA* component is illustrated in Figure 7.



```
<!DOCTYPE rdf:RDF [
  <!ENTITY ns-rdf  'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!ENTITY ns-rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
  <!ENTITY prf "http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20021212#">
]>
<rdf:RDF xmlns = '&ns-rdf;'
    xmlns:rdf  = '&ns-rdf;'
    xmlns:ns-rdfs="&ns-rdfs;"
    xmlns:prf="&prf;">
  <rdf:Description ns-rdf:ID="BrowserUserAgentString">
    <ns-rdfs:comment xml:lang='en'>
    Description: User-Agent header sent within the HTTP Request
    Example: "HTC_Touch_HD_T8282 Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 7.11)"
   </ns-rdfs:comment>
   <rdf:type ns-rdf:resource="&ns-rdf;#Property"/>
   <ns-rdfs:domain ns-rdf:resource="&prf;#BrowserUA"/>
   <ns-rdfs:label xml:lang='en'>BrowserUserAgentString</ns-rdfs:label>
   <rdf:type rdf:resource='&ns-rdf;Property'/>
   <prf-dt:ResolutionRule rdf:datatype='&prf-dt;ResolutionRule'>Locked</prf-dt:ResolutionRule>
  </rdf:Description>
</rdf:RDF>
```

Fig. 7. Example of extension of an existing vocabulary

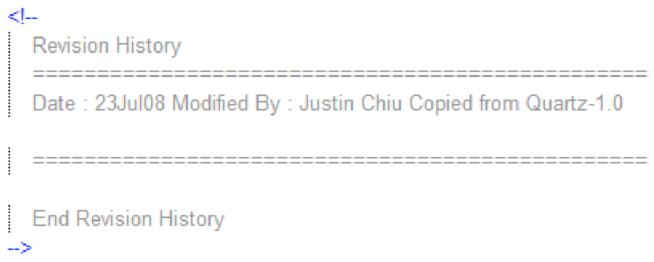
Once the extended schema has been created, it is possible to define an instance of the BrowserUA component for the Pocket Internet Explorer of the HTC Touch HD as depicted in Figure 8. :



```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
...
   xmlns:prf="http://www.openmobilealliance.org/tech/profiles/UAPROF/ccppschema-20021212#"
   xmlns:prfext="http://fundacionctic.org/miCCPPSchema#">
...
      <rdf:Description rdf:ID="BrowserUA_pIE">
        <rdf:type rdf:resource="http://www.openmobilealliance.org/tech/profiles/UAPROF
          /ccppschema-20021212#BrowserUA" />
        <prf:BrowserName>Microsoft Pocket Internet Explorer</prf:BrowserName>
        <prf:BrowserVersion>4.0</prf:BrowserVersion>
        <prfext:BrowserUserAgentString>
          HTC_Touch_HD_T8282 Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 7.11)
        </prfext:BrowserUserAgentString>
        <prfext:BrowserUserAgentString>
          HTC_Touch_HD_T8282 Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 7.11)
        </prfext:BrowserUserAgentString>
        <prfext:BrowserUserAgentString>
          HTC_Touch_HD_T8282 Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 7.11)
        </prfext:BrowserUserAgentString>
        <prfext:BrowserUserAgentString>
          HTC_Touch_HD_T8282 Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 7.11)
        </prfext:BrowserUserAgentString>
        <prf:HtmlVersion>4.01</prf:HtmlVersion>
        <prf:XhtmlVersion>1.1</prf:XhtmlVersion>
        <prf:FramesCapable>No</prf:FramesCapable>
        <prf:PreferenceForFrames>No</prf:PreferenceForFrames>
        <prf:TablesCapable>Yes</prf:TablesCapable>
      </rdf:Description>
...
</rdf:RDF>
```

Fig. 8. Example of extended profile definition

Another improvement carried out in the UDDR is the re-use of data previously defined in other profiles. Analyzing the UAProf documents created by HTC, we have noticed that they include a comment after each profile document as shown in Figure 9.:



```
<!--
Revision History
===============================================
Date : 23Jul08 Modified By : Justin Chiu Copied from Quartz-1.0

===============================================

End Revision History
-->
```

Fig. 9. Revision history as commented in some HTC device profiles

The comment is intended to note that the HTC Diamond UAProf document has been created based on the Quartz-1.0 by modifying some properties. After comparing these documents, we came to the conclusion that 463 lines out of 466 are completely the same in both profiles. This means that the 99.35% of the data is duplicated. The proposed mechanism to solve this issue is commented in Section 8.

Additionally, to create a Universal DDR, it is necessary to provide further information about the devices than those coming from the original UAProf document. To carry out this task, some tests must be performed. Some of the tests can be completely automatic, since a lot of information can be extracted by running predefined software programs on the devices. What is more, sometimes part of the information automatically extracted after the execution of specific software is already described in the original UAProf. However, if such a piece of information might have been defined in a generic way and it should have been associated to a specific component (as has been explained in section 5.2), so a refactoring process is needed. Unfortunately, not all the tests can be performed by a software component and they require the human expert intervention, for instance to guarantee that the quality of some audiovisual content is acceptable.

UDDR-Profiles need not only to add more information about the device capabilities, but also to include the provenance of the amendments. As stated in section 5.1, the provenance information must be represented using named graphs. This requires the usage of UAProf documents combined with graph definition files in charge of linking the provenance information with the associated profile definitions. The provenance information might annotate information about the provider (company name), the editor, date and time for the update, etc.

7 Profile Resolution

In the context of this work, the process of gathering available information pieces and assembling a coherent profile description is named "profile resolution". This process becomes necessary because the UDDR relies on many information sources, including UAProf+ profiles, but at the same time it is expected to hide the diversity and provide precise answers to the queries. The profile resolution also deals with amended information, descriptions with different degrees of granularity, and also with sources of different trustworthiness. Finally, profiles can be built by refining other profiles. The resolution process is in charge of assembling general and detailed profiles into a single view of the device description.

Some of the profile descriptions may be contradictory or ambiguous. Our approach to tackle this issue is twofold. In the first place, it is necessary to assess whether a conflict actually exists. Multiple concurring but different values of a property do not necessarily lead to a conflict, because the property may be multivalued. It is also possible that values can be merged or unified (e.g., the meaning of *VendorName = "HTC"* and *VendorName = "High Tech Corporation"* it is the actually same). However, in other situations, the conflict cannot be avoided, and consequently a mitigation method comes into effect. A number of strategies can be conceived, the simplest one being just to discard those information pieces that are in conflict. This conservative strategy, which can be applied in the absence of meta-data, sacrifices completeness for the sake of soundness. One of the goals of introducing provenance meta-data in the UDDR is to enable cleverer strategies. For instance, a more interesting approach would be to keep the most recently updated value, because it is assumed to be a fix for an erroneous value. Another strategy would rank values according to the popularity or trustworthiness of their source.

Figure 10. suggests the fact that profile resolution takes place after the population of the DDR commented in the previous section.

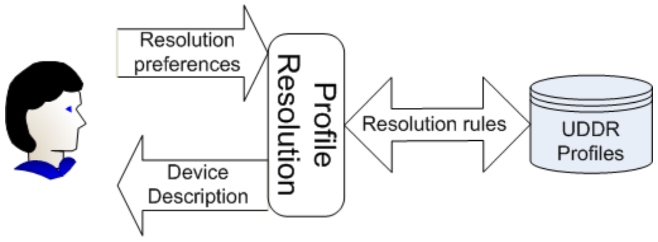


Fig. 10. Profile resolution mechanism

8 Example

In order to illustrate the populating and querying processes exposed in previous sections, a brief and concise example will be explained step-by-step. As a proof of concept, a specific device model, the HTC Touch HD (also known as Blackstone), has been chosen, and it is exemplified how to convert from its original UAProf document to the final UDDR-Profile. For the sake of brevity, the intention of the example is not to provide a full profile describing all the possible device capabilities, but to illustrate the technique to be followed in order to create a UDDR-Profile by using a reduced and comprehensive set of properties. Throughout this section some references are made to various documents, as the UAProf, UAProf+ or UDDR-Profile documents and its corresponding schemas. All the mentioned files are available online[7].

The first step to be applied is to create the UAProf+ document from the original UAProf file. To accomplish this goal, first of all those changes necessary to guarantee that the UAProf+ use the last available schemas for the distinct defined namespaces

[7] `http://idi.fundacionctic.org/mc2011/index.html`

must be performed. For instance, the *pss5:audioChannels* and *pss5:rendering ScreenSize* properties which had originally been included within the Streaming component under the pss5 namespace have been moved to the *PssCommon* component, under the last PSS namespace (pss6). Moreover, the values defined in the *pss5:pssAccept* property have been moved to the *pss6:StreamingAccept*.

The next step is to normalize the possible values for each property. For instance, the "HTC Corporation" value for the Vendor property has been converted to "HTC" as it is the notation established by convention in UDDR. Similarly, the Bluetooth profile values defined inside the *BluetoothProfile* property have been defined by using the appropriate acronym. Upon the completion of this task, the conversion from UAProf to UAProf+ can be considered as finished.

The last step to complete the populating process is the generation of the final UDDR Profile. To do that, three sub-steps are necessary:

1) Add information about the capabilities of the device. In this case, it has been decided to extend the multimedia information available for the device. A custom software tool has been developed in Windows Mobile. This software analyzes the Windows register in search of the MIME types and protocols supported by each media player pre-installed in the device. Moreover the tool is able to find out which is the default media player for each protocol. To guarantee the correctness of the information coming from the Windows register, some manual tests have been performed by a human expert.

2) Group the available information. In order to avoid data redundancy, UDDR tries to group data properly and reference other profiles when possible. For instance, the UDDR-Profile will be based on the UAProf+ descriptions, so it is not necessary to duplicate in the former all the data defined the latter. To carry out this task, a set of custom properties with a specific semantic meaning have been included. Such properties are intended to state that the subject description contains all the properties and attributes defined in the object.

3) Add provenance information. In step 1, we have added new information about the HTC Touch HD. However, in step 2, we have stated that we had also reused part of the information which was previously available in the original UAProf+ document. For the management of the UDDR it is crucial to annotate all the possible information about the provenance of the information. In this case we have created two distinct named graphs: one for the original information created by HTC and other for the information provided by the authors of this work.

Having completed the population phase, it is now possible to proceed to the querying process. Since the UDDR-Profile of the HTC Touch HD has already been defined, it is possible to query the UDDR about its properties. For instance, someone might like to ask the UDDR for the complete profile for such device by providing its URI as input evidence. Some other might want to retrieve all the possible User Agent strings for the default mobile browser of the device. Even more, and just supposing for the sake of example that we would had performed the population process over a thousand of devices, it might be interesting to ask for all the devices containing the substring "Opera" as part of the User Agent string of its mobile browser.

9 Conclusions

The main contribution of this paper is a process to obtain new device descriptions for a Universal Device Description Repository which builds on top of existing technologies, namely CC/PP and UAProf. We characterize the UDDR by means of its requirements, and we attempt to overcome the limitations found in the current alternatives. A repository without comprehensive contents would be useless, therefore we suggest a method to adapt existing (and possibly broken) UAProf profiles. UAProf+ arises as an intermediate step in such transition, and introduces a number of improvements while maintaining backward-compatibility with UAProf.

More precisely, UAProf+ aims to be fully aligned with existing schemes, and to be accepted by current conformance-checking tools such as the DELI validator. We expect UAProf+ to become the baseline of real interoperability between profile descriptions based on current practices, vocabularies and toolsets.

However, we believe that backward-compatibility must be eventually sacrificed in order to fulfill the long-term vision, particularly with respect to the requirements of provenance, adequate profile resolution and semantic precision. We argue that significant gains in these fronts can be obtained by fully exploiting the benefits of the RDF model. In other words, we propose to lift some of the XML-oriented syntactical restrictions inherited from CC/PP, as well as to introduce some refactoring in the schemes. Therefore, our final step is the UDDR, which re-uses as much as possible of its predecessor schemes, and fully leverages the RDF model in order to obtain expressivity, precision and concision. Our proposed UDDR profiles are easier to manage due to their reduced verbosity and redundancy, and can be intelligently combined taking into account provenance information. The usage of RDF becomes more idiomatic, and is aligned with the "linked data" principles. For instance, vendors, protocols or file formats are promoted from literal values to resources, a movement that enables unambiguous references and exploitation of information from external sources.

We shall not build the UDDR data sets from scratch. In addition to schema re-use, we also propose to integrate UAProf+ data in the UDDR. To this end, we introduced and exemplified a profile-extension mechanism that permits to derive UDDR profiles from UAProf+ ones (e.g., by means of the extendsProfile property). This data-extension mechanism is a custom add-on to the RDF model, and its semantics must be enforced by the UDDR, through the query evaluation engine. It is also tightly coupled to provenance tracking and profile resolution. We believe that these semantics can be implemented using SPARQL 1.1 queries.

Finally, we envisage another application of SPARQL to the implementation of advanced validation tools for UDDR profiles. The main issue here is to find the delicate balance between the permissiveness of the open-world assumption and partial descriptions on the one hand, and enforcing some essential semantic restrictions on the other one. As with other semantic web schemes, such as SKOS [25], Description Logic reasoners fall short to check all the restrictions and entailments. We propose to capture these additional restrictions as invariants expressed in SPARQL. Therefore, new UDDR profile validation tools will have to be created and made available to the industry.

10 Future Work

The authors of this article intend to further develop a formal definition for the UAProf+ format beyond the proof-of-concept suggested in this article. After that definition, a UAProf+ validator would be created in order to help device manufacturers to create interoperable device descriptions. In addition, a new version of the conversion tool from existing UAProf documents to the new UAProf+ would be developed and released. In this way, software tools might use a single language processor based on the new format to extract information from device descriptions. The success of both format and software tools (validator and legacy-UAProf conversor) will depend on their acceptance by the Open Mobile Alliance, after the corresponding submissions from the authors.

In what regards to the improvements in the current prototype of the UDDR and its population and query, a more formal model will be established and published. More specifically, the authors intend to develop the grouping mechanism to create hierarchies of device descriptions and avoid storing redundant information. The initial candidate approach to create hierarchies for different device families is the obtention of the common factor (common values for a subset of device properties) among device descriptions with the same operating system, operating system version, operating system flavor (i.e., S60 in the case of Symbian), operating system flavor version (S60 5th edition, for the previous example), etc.

References

1. Ramioul, M., Huws, U., Bollen, A.: Measuring the information society. HIVA Publication
2. DeviceAtlas: Mobile Device Detection, `http://deviceatlas.com/`
3. WURFL. The Wireless universal Resource FiLe, `http://wurfl.sourceforge.net/`
4. The maDDR Project, `http://www.maddr.org/`
5. The World Wide Web Consortium, `http://www.w3c.org/`
6. Device Independence working group. The World Wide Web Consortium, `http://www.w3.org/2001/di/`
7. Wireless Application Group–User Agent Profile Specification (1999)
8. Klyne, G., Reynolds, F., Woodrow, C., Ohto, H., Hjelm, J., Butler, M., Tran, L.: Composite Capability/Preference Profiles (CC/PP): Structure and vocabularies 1.0. W3C Recommendation 15 (2004)
9. Beckett, D., McBride, B.: RDF/XML Syntax Specification (Revised). W3C Recommendation. The World Wide Web Consortium, February 10 (2004)
10. Open Mobile Alliance, `http://www.openmobilealliance.org/`
11. Buneman, P., Khanna, S., Tan, W.-C.: Why and Where: A Characterization of Data Provenance. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 316–330. Springer, Heidelberg (2000)
12. Butler, M.: Some questions and answers on CC/PP and UAProf. Tech. rep., Hewlett Packard Laboratories (2002)
13. Butler, M.: CC/PP and UAProf: Issues, improvements and future directions. In: Proceedings of W3C Delivery Context Workshop, DIWS 2002 (2002)

14. Butler, M.: Input to Device Description Working Group (2005), http://lists.w3.org/Archives/Public/public-ddwg/2005Aug/att-0005/ddwgPositionPaper.htm
15. DELI: A Delivery Context Library For CC/PP and UAProf, http://delicon.sourceforge.net/
16. Indulska, J., Robinson, R., Rakotonirainy, A., Henricksen, K.: Experiences in Using CC/PP in Context-Aware Systems. In: Chen, M.-S., Chrysanthis, P.K., Sloman, M., Zaslavsky, A. (eds.) MDM 2003. LNCS, vol. 2574, pp. 247–261. Springer, Heidelberg (2003)
17. Gergic, J., et al.: Addressing On-Demand Assembly and Adaptation Using a Run-time Intentional Versioning Engine. Ph.D. thesis, Charles University in Prague, Faculty of Mathematics and Physics (2008)
18. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: Proceedings of the 14th International Conference on World Wide Web, WWW 2005, pp. 613–622. ACM, New York (2005), http://doi.acm.org/10.1145/1060745.1060835
19. Zhao, J., Miles, A., Klyne, G., Shotton, D.: Linked data and provenance in biological data webs. Briefings in Bioinformatics 10(2), 139–152 (2009), http://dx.doi.org/10.1093/bib/bbn044
20. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF (working draft). Tech. rep., W3C (March 2007), http://www.w3.org/TR/2007/WD-rdf-sparql-query-20070326/
21. Gil, Y., Cheney, J., Groth, P., Hartig, O., Miles, S., Moreau, L., da Silva, P.P.: Provenance XG Final Report (December 2010), http://www.w3.org/2005/Incubator/prov/XGR-prov/
22. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language. W3C working draft, W3C (October 2010)
23. OMA DELI-2 UAProf Validator, http://validator.openmobilealliance.org/cgi/
24. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space, 1st edn. Morgan & Claypool (2011), http://linkeddatabook.com/
25. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System Reference. The World Wide Web Consortium (August 2009), http://www.w3.org/TR/skos-reference/