Some Questions and Answers on CC/PP and UAProf

Mark H. Butler
Information Infrastructure Laboratory
HP Laboratories Bristol
HPL-2002-73
April 5th , 2002*

E-mail: mark-h_butler@hp.com

device independence, composite capabilities/ preferences profile (CC/PP), resource description framework (RDF), wireless access protocol (WAP), user agent profile (UAProf)

DELI is an open-source library developed at HP Labs that allows Java servlets to resolve HTTP requests containing device capability information in either CC/PP, the proposed recommendation by the W3C CC/PP, or UAProf, the standard proposed by the WAP Forum. This document consists of a set of questions and answers that explore issues that have arisen during the implementation of DELI. It is presented specifically to initiate further discussion to identify whether further work is necessary on the CC/PP recommendation track documents.

* Internal Accession Date Only
Approved for External Publication

Some Questions and Answers On CC/PP and UAProf

Mark H. Butler
mark-h_butler@hp.com
20 / 03 / 2002

Abstract

DELI is an open-source library developed at HP Labs that allows Java servlets to resolve HTTP requests containing device capability information in either CC/PP, the proposed recommendation by the W3C CC/PP, or UAProf, the standard proposed by the WAP Forum. This document consists of a set of questions and answers that explore issues that have arisen during the implementation of DELI. It is presented specifically to initiate further discussion to identify whether further work is necessary on the CC/PP recommendation track documents.

Keywords

Device Independence, Composite Capabilities / Preferences Profile (CC/PP), Resource Description Framework (RDF), Wireless Access Protocol (WAP), User Agent Profile (UAProf)

CC/PP and UAProf: A Dialogue

Q: So what do you think about the current versions of CC/PP and UAProf?

A: Well in summary my opinion is that the CC/PP documents do not give enough detail to effectively implement a CC/PP processor. The UAProf documents do give enough detail to implement a UAProf processor, but neither UAProf nor CC/PP adequately addresses profile validation so it is not possible to guarantee vendor interoperability for UAProf profiles.

Q: But surely CC/PP and UAProf work? Haven't a number of people have already implemented them?

A: Yes people have implemented CC/PP and UAProf. I've produced an implementation myself called DELI[1,2]. However DELI (and I suspect other implementations) uses a number of *bodges* i.e. temporary fixes that are likely to break in a large-scale deployment. Although DELI can process all profiles I have currently encountered, my experience is that generally DELI will fail when it encounters a new profile. There are several reasons for this but it is chiefly because there is currently no notion of profile validation in CC/PP or UAProf so there is no guarantee of vendor interoperability.

Q: Yes but I'm working on CC/PP - why should I be interested in UAProf as that's the WAP Forum's problem?

A: UAProf is the first large-scale deployment of CC/PP. Therefore the problems encountered with UAProf illustrate some important issues for CC/PP. A lot of the problems stem from the fact that the CC/PP recommendation track documents don't specify very much. As a result of this UAProf had to define functionality that CC/PP left unspecified. Really CC/PP should specify these bits of functionality in order to

guarantee vocabulary interoperability, as after all UAProf is really just a specific CC/PP vocabulary and protocol rather than an alternative standard.

Q: What do you mean CC/PP doesn't specify very much?

A: In my opinion, the requirements in the CC/PP structure and vocabularies document can be re-interpreted as saying "any RDF parser that assumes any property ending in "Defaults" is equivalent to any property ending in "defaults" constitutes an implementation of CC/PP".

Q: What do you mean - it's a long document - surely it specifies more than that?

A: When you read drafts one key thing to consider is what is required for conformance. Typically this is indicated by "REQUIRED", "MUST" and "SHOULD". If you search the CC/PP working draft[3], you'll find the following conformance issues. I've numbered them so I can make further comments:

1. *CC/PP profile components: support for these is REQUIRED.*

2. *CC/PP profile defaults: support for these is REQUIRED.*

3. *Support for the structured CC/PP attribute formats described, where relevant, is REQUIRED.*

4. *CC/PP applications are not required to support features described in the appendices, but any new attribute vocabularies defined MUST conform to the RDF schema in appendices B and C.*

5. *If a CC/PP profile uses any attribute that can appear on different component types, then the type of any component on which such an attribute appears MUST be indicated by an rdf:type property, or equivalent RDF. A CC/PP processor MUST be able to use this type information to disambiguate application of any attribute used.*

6. *A ccpp:Component resource MAY have an rdf:type property (or equivalent RDF structure) indicating what kind of client component it describes. The example in figures 3-4 is of a profile with an explicit indication of component subtype. However, CC/PP processors MUST be able to handle profiles that do not contain component type indicators. As long as the CC/PP attributes used are all specific to a given component type, a processor will have sufficient information to interpret them properly.*

7. *Default values are referenced by the property ccpp:defaults. This name conforms to the name format recommendations of the RDF model and syntax specification [3], appendix C.1. However, for compatibility with earlier versions of CC/PP used with UAPROF, CC/PP processors SHOULD recognize the property name ccpp:Defaults (i.e. with capital "D") as equivalent.*

8. *The component resources in a profile are instances of components identified in the corresponding schema, which in turn MUST be subclasses of ccpp:Component. They*

MUST be identified as such, by means of the rdf:type property whose value matches the name of the component type in the schema.

9: *NOTE: A default document uses a <Description> element as its root node. The <Description> is named using an about= attribute whose value is a URI. This URI MUST correspond to the value in the rdf:resource= attribute in the <Defaults> element in the referencing document. In the examples of default documents below, the URLs of the external default values documents are used. However the default resource URI does not have to be the document URL, as long as the URI is uniquely identified, the same URI is used in both the source document and the external default values document, and there is some way for the processing software to locate and retrieve the document containing the default resource.*

It's strange that a lot of these constraints refer to the processor when this document is meant to describe the structure and vocabularies used by the profile. There is a logical problem here because even if the processor MUST process something, it doesn't necessary follow that a profile MUST contain it. Furthermore none of the CC/PP documents discuss a processing model, so putting constraints on the processor doesn't achieve much. For example points 1 - 3 use the word "support" to describe what the processor does. However "support" is not defined anywhere so could simply mean parse. In Point 5 "use" is not defined nor "handle" in points 6 and 7. Points 8 and 9 seem okay though: 8 is a requirement on any CC/PP schema so we can say a CC/PP schema must be a valid RDF schema and condition 8 must hold whereas 9 is a requirement on "default documents" so we can say a default CC/PP document must be a valid CC/PP profile and condition 9 must hold. I think a distinction needs to be drawn between "support", "use" and "handle" and parsing otherwise any RDF parser can be regarded as a CC/PP implementation as long as it obeys point 7 e.g. it regards the property defaults equivalent to Defaults.

Q: So what's RDF? I thought CC/PP and UAProf used XML?

A: RDF, the Resource Description Framework[4], is the W3C's recommendation for metadata. It is a way of representing information as statements, each consisting of a subject, predicate and object. However it is a new technology so people are not as familiar with it as they are with XML. RDF can be serialised in many ways, but CC/PP uses the XML serialisation of RDF. Hence CC/PP is built on RDF represented using XML, so CC/PP profiles are, in effect, written in XML. However when we parse CC/PP profiles we need to remember there is another level of abstraction above the XML i.e. RDF.

Several implementations of CC/PP or UAProf, for example the IBM UAProf implementation[5] or the XCries CC/PP implementation in XSmiles[6] process UAProf and CC/PP respectively using an XML parser. Parsing the XML serialisation of RDF using an XML parser is difficult[7] as the XML serialisation allows the same RDF model to be represented in many different ways. Therefore these processors are not able to parse all possible XML serialisations of RDF and hence all possible CC/PP profiles.

Even if the CC/PP processor does use an RDF parser (for example DELI uses Jena[8]) the XML serialisation can still cause problems. For example one use case for CC/PP

is to use the profile information to adapt XML using XSLT. To investigate this, I have integrated DELI into the Apache Cocoon[9] XML / XSLT publishing framework so that XSLT stylesheets can query CC/PP profiles. XSLT uses XPath but it is difficult to query CC/PP profiles using XPath as the structure of the underlying RDF model is different to the XPath representation of the CC/PP profile. In order to resolve this problem DELI "flattens" the profile to produce an XML profile which is amenable to query via XPath. Clearly this is undesirable as it means that DELI is using two different profile representations.

Q: This all sounds rather pedantic - what does this have to do with vendor interoperability?

A: Well the problem is if a specification is not adequately constrained, there is too much room for vendors to produce different incompatible variants. This seems to be what's happening with CC/PP. When implementing DELI, the first thing I had to do was answer the question of what a CC/PP or UAProf processor does and I concluded that one important task was profile resolution. UAProf, and the proposed protocol for CC/PP, split the profile into a number of fragments so that these fragments can be sent efficiently to the server. *Profile resolution* is when the server recombines these fragments to form the original profile. Unfortunately as profile resolution is defined in the CC/PP protocol document and the protocol was outside the original charter, profile resolution is not defined in any recommendation track W3C documents. So one way of resolving the problem with requirements 1-7 would be to put a description of profile resolution in a recommendation track document. I think it is possible for resolution to be specified in a protocol independent way.

Q: So how does DELI perform profile resolution?

A: Firstly when DELI processes a HTTP request containing CC/PP information, it searches the request header for the headers "x-wap-profile" and "x-wap-profile-diff", the W-HTTP UAProf request headers. Then it has to search for the headers "profile" and "profile-diff" because currently the Nokia phone emulator uses these non-standard request header names. So Nokia and Ericsson currently implement the protocol differently! Then it has to search for HTTP-ex style request headers in case it is dealing with a CC/PP device conforming to the proposed (but not recommended) extension protocol[10] or a UAProf 1.2 device.

Secondly once DELI has identified the profiles and the profile-diffs, it builds an RDF model for each profile fragment. Some profiles, such as the Ericsson T68[11] and T39[12] do not use rdf:type to identify components. Although this behaviour is permitted under the CC/PP and UAProf recommendations, really this is incorrect use of RDF just as using a variable in a Java program before declaration would be considered incorrect[13]. For example in the following profile fragment

```
<prf:component>
  <rdf:Description rdf:ID="HardwarePlatform">
    <rdf:type
       rdf:resource =
"http://www.wapforum.org/profiles/UAPROF/ccppschema20010430#HardwarePlatform
"/>
    <prf:BitsPerPixel>2</prf:BitsPerPixel>
   </rdf:Description>
</prf:component>
```

we declare that the component is called HardwarePlatform twice. It is important to note the first use of HardwarePlatform defines the instance name whereas the second use is to define the type. This is just as if we defined an object in Java e.g.

```
HardwarePlatform hardwarePlatform;
```

In Java we would not expect the complier to determine the object type from the instance name.

Q: Is it really true that they would be required to do this in order to be valid RDF? Surely typing is not mandatory as in Java?

A: I mention Java as an analogy. By incorrect RDF I don't mean that it breaks formal things such as the EBNF rules governing the RDF syntax but it is using RDF in a way that is not intended. Here's a fragment from the T68 profile:

```
<prf:component>
  <rdf:Description ID="SoftwarePlatform">
       <prf:AcceptDownloadableSoftware>No</prf:AcceptDownloadableSoftware>
   </rdf:Description>
</prf:component>
```

As you can see in this profile a component is created with a *local* ID of "SoftwarePlatform". By local I mean it is local to this particular model. RDF processors don't place a "global" interpretation on the ID. Here's what the RDF Model and Syntax Specification document says about Description elements:

A single RDF statement seldom appears in isolation; most commonly several properties of a resource will be given together. The RDF XML syntax has been designed to accomodate this easily by grouping multiple statements for the same resource into a Description element. The Description element names, in an about attribute, the resource to which each of the statements apply. If the resource does not yet exist (i.e., does not yet have a resource identifier) then a Description element can supply the identifier for the resource using an ID attribute.

Furthermore it says this about Description elements using the type property:

The third basic abbreviation applies to the common case of a Description element containing a type property (see Section 4.1 for the meaning of type). In this case, the resource type defined in the schema corresponding to the value of the type property can be used directly as an element name. For example, using the previous RDF fragment if we wanted to add the fact that the resource http://www.w3.org/staffId/85740 represents an instance of a Person, we would write this in full serialization syntax as:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:s="http://description.org/schema/">
  <rdf:Description rdf:about="http://www.w3.org/Home/Lassila">
    <s:Creator>
      <rdf:Description rdf:about="http://www.w3.org/staffId/85740">
       <rdf:type rdf:resource="http://description.org/schema/Person"/>
       <v:Name>Ora Lassila</v:Name>
       <v:Email>lassila@w3.org</v:Email>
      </rdf:Description>
```

```
    </s:Creator>
  </rdf:Description>
</rdf:RDF>
```

So we have to use rdf:type to indicate that when we create a SoftwarePlatform resource we are referring to something we have defined in the associated schema. Presumably some UAProf processors search the profile for resources with specific local IDs e.g. "SoftwarePlatform", "HardwarePlatform" etc. This isn't the recommended approach in RDF: if we need to search for resources, then we should do it based on resource type.

Q: But doesn't the CC/PP structure and vocabularies document explicitly allow you to do this in point 6?

A: Yes that's what the document says at the moment. I suspect this requirement is primarily to guarantee compatibility with UAProf rather an explicit design decision. Specifically although it is possible to make a requirement like this of a UAProf processor as it only processes the UAProf vocabulary, I think the current CC/PP documents make it impossible to create a CC/PP processor which could process many vocabularies.

Currently DELI tries to cope with such situations by inferring the component from the attribute name using a vocabulary description. However a processor cannot guarantee that the vocabulary description will be available. All currently available UAProf profiles reference a namespace using a URL that does not contain a schema. If the processor has not been instructed a priori to associate this namespace with a local copy of the schema, then the processor cannot infer the parent components of any attribute, as they do not possess the schema. This stems from the fact that CC/PP and UAProf do not state that profiles MUST reference namespaces that make RDF schemas available.

The Mitsubishi Trium profile[14] does not use components at all. This causes a problem as RDF models, unlike XML, do not have a concept of a root node. Therefore by default DELI tries to use the components as a starting point in the RDF graph for retrieving information. With the Trium, DELI has to use a fallback mode where it tries to manually locate the root of the model (as CC/PP profiles, unlike other RDF models, always have a root node) in the hope it can find attributes there.

Q: That sounds like a good solution - what's the problem?

A: Well either I'd like to see the profile format much more tightly defined or if a fallback approach is necessary have that made explicit in a formal document.

I think the confusion about components stems from the fact it is not obvious what purpose components serve. The CC/PP structure and vocabularies document recommends that attributes should really only be associated with one component: hence components don't provide any additional information beyond attribute name. Furthermore I observe that there has been disagreement in UAProf about which component is most appropriate for various attributes e.g. should CcppAccept be in SoftwarePlatform or BrowserUA? If we don't have a concrete use case for why we

need components, perhaps we need to exercise Occam's razor and just dispose of them altogether.

Q: You also mentioned something called a vocabulary description - what's that?

A: When you perform profile resolution in UAProf, you need to know some information about the vocabulary in use. I've called this information the vocabulary definition. It is derived either from the RDF schema for the vocabulary or a proprietary DELI format file.

Q: Why did you create a proprietary format when CC/PP and UAProf use RDF schema?

A: In order to perform profile resolution in UAProf the processor needs to know the type and the resolution rule used by the attribute. In the RDF schema for UAProf, this information is stored as comments rather than as XML formatted data. This means it is hard to retrieve using an XML parser, as you have to do lots of text matching which should not be necessary. So initially I invented my own data format that stored this information as XML. As time went on I discovered there are many versions of the UAProf vocabulary in use and phones often use a vocabulary incorrectly e.g. attributes are spelt incorrectly. For example the Ericsson phones use an attribute called "PixelsAspectRatio" which I think should be "PixelAspectRatio". In the interests of fairness I decided it was important to discover whether I was making mistakes or whether it was the phone. The best way to do this seemed to be to use the official UAProf schema. This involved writing a parser that could parse the comments fields. These UAProf schemas are available at
http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#
http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405#

When I tried parsing these schemas I found that both schemas contained a large number of RDF errors. For example they do not use the correct RDF namespace. They should qualify ID with the rdf namespace i.e. use rdf:ID not ID. Similarly they should use rdf:resource not resource. They contain a few invalid URI with extraneous pre-pended and trailing white spaces. They also use several different namespaces to refer to RDFS elements when they should all use a single RDFS namespace. Finally the schema incorrectly says that Bag and Seq elements are in the RDFS namespace when they should be in the RDF namespace. If you want to see the errors for yourself, try running the schema through the W3C RDF Validation service at http://www.w3.org/RDF/Validator/. I have corrected the schemas and sent the corrected versions to the WAP Forum, but they have not yet updated them. The corrected schemas are available with DELI though.

Q: But surely companies check their profiles before they release them?

A: Well to be fair to the vendors one of the biggest problems with CC/PP is there is no automatic scheme for validating profiles. To date the biggest users of DELI have been vendors who want to test their profiles rather than content authors using CC/PP or UAProf information. To be honest, DELI is not ideal as a profile validator because CC/PP has a very weak concept of what constitutes a valid profile. For example at the moment there is no requirement that profiles reference real schemas as processors do

not have to support schemas. As I see it, the working draft should contain much stronger constraints on profiles somewhere along these lines:

A CC/PP profile MUST meet the following criteria:

- *It must be valid RDF.*
- *It must refer to a minimum of three namespaces, the RDF namespace, the CC/PP namespace and one or more vocabulary namespaces. The RDF and CC/PP namespaces MUST be the standard W3C namespaces.*
- *All vocabulary namespaces MUST be URLs that actually contain valid RDF schemas. A CC/PP profile can only use attributes that are defined in one of the RDF schemas it references.*
- *As well as defining the attribute name and its parent component, the RDF schema should also define the attribute type and whether it is simple or complex. When attributes in the vocabulary can use several different resolution policies, as in UAProf, it should also define the resolution rule. When suitable mechanisms for defining these properties are not available in the RDF schema namespace, a new standardised CC/PP schema namespace should be used rather than placing the information in comments fields.*

If profiles were formally required to meet these criteria, DELI could perform validation and check profiles before they become public. Such a validating CC/PP processor could be made available at the W3C website, in a similar way to the RDF validation service.

Q: Why can't you use this on profiles for currently available phones like the Ericsson T68?

A: Currently phones often refer to schemas that do not exist so such a validation process is not possible. For example the Ericsson T68 profile references the namespace
http://www.wapforum.org/UAPROF/ccppschema-20000405#
Whereas the correct URL is
http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405#
In DELI, I "bodge" this problem using a configuration file that maps the incorrect namespaces used by phones onto local corrected versions of the UAProf schemas. However when a manufacturer releases a new phone, there is a danger that they will use a different namespace. As namespaces have to be explicitly configured in DELI, if it encounters a new namespace it will fail. This is a major barrier to true vendor interoperability. Of course I could "bodge" this so if a profile uses a namespace that DELI has never heard of it adopts a fallback one, but this is patching the problem rather than solving it I would like to get rid of all these "bodges" - that's why I've been so vocal about the issues with CC/PP and UAProf.

Q: What about other phones?

A: Well the other phone I have encountered, the Mitsubishi Trium, references the namespace
http://www.wapforum.org/UAPROF/ccppschema-19991014#
and again no schema is available from this URL. The Trium profile contains two unrecognised attributes - SoftkeysCapable and WmlscriptCapable. I think the problem

here is caused by incorrect use of capitals e.g. WmlscriptCapable should be WmlScriptCapable.

Q: Isn't SoftkeysCapable in one of the SINs? (For people not familiar with UAProf, a SIN is a specification information note i.e. an approved specification change document)

A: Perhaps but in my opinion that isn't good enough. UAProf has been created so content providers can deliver optimized content and optimized services to different devices in a seamless fashion. In order to guarantee this, we need processors to load the correct information on demand, not hope that the processor contains hard-coded information that meets the latest revision of the specification. We also need to guard against errors using standard quality control methods like data validation. I think it's fair to say that at the moment RDF has less support for data validation than XML. This is primarily because RDF is designed to be used in an extensible manner. Extensibility comes at a price though: for example how does a processor determine whether "WmlscriptCapable" is a typing mistake or an attempt to extend a vocabulary?

Q: So are there any other problems with profile resolution?

A: Yes. To recap, we have parsed our profile fragments using Jena and we need to perform profile resolution. Unfortunately this is difficult to implement in RDF. Both CC/PP and UAProf clients split up profile information in order to send it to the server in an efficient way. They do this by using a standard profile, known as a reference profile, and a list of overrides specific to the requesting device known as a profile diff. Other devices in the communication path, such as proxies, may also add profile diffs. The process of reassembling the final profile from the reference profile(s) and profile diff(s) is known as profile resolution. CC/PP does not specify the exact mechanism for profile resolution, apart from requiring that default attribute values are always overridden by non-default attribute values. UAProf on the other hand specifies a set of resolution rules that apply to non-default values. Each attribute in the UAProf vocabulary is associated with a specific resolution rule that is applied when multiple attribute values are encountered. In UAProf these resolution rules are order dependent; for example, locked means take the first value encountered whereas override means take the last value encountered. Unfortunately these rules are difficult to implement in RDF, as RDF models do not have any implicit concept of ordering statements. Ordering must be done explicitly, e.g. using an RDF Sequence. This is in contrast to XML, which does implicitly order elements in documents. As statements in an RDF model are unordered it is impossible to apply the UAProf resolution rules to a single RDF model. UAProf keeps the reference profile and diffs as separate RDF models in the W-HTTP protocol, and in theory each model should only define an attribute once. If they did define an attribute twice, then currently the resolved value would be determined in an undefined way. DELI makes the assumption that each separate RDF model will only define an attribute once and "bodges" the resolution problem by converting the profiles to an intermediate data structure that stores attribute order before performing profile resolution. It then merges this data structure rather than the RDF models.

Q: So how do you use the CC/PP information once you've resolved it?

A: Well there are two problems with using this information in UAProf. Firstly UAProf does not define any semantic meaning for attribute values. This means that two phones from two different vendors might use different attribute values to describe the same functionality i.e. one might call a Keyboard "keypad" whereas another might call it "numeric". Alternatively they might use the same attribute value to describe different functionality. Even if there is a standard attribute with a common meaning, CC/PP and UAProf does not define any rules regarding capitalization or use of white space in attribute values. This, along with mistyping and the lack of profile validation techniques means we may not be able to use information from attribute values. Ideally there should be some means of validating attribute values as well as attribute names, although this would not resolve the problem of semantic meaning. XML Schema has mechanisms for specifying constraints on attribute values.

Assuming we have solved the problem of attribute values, the next problem is what operators you can use in association with those values. UAProf defines four data types: Boolean, Numeric, Dimension and Literal. Boolean can use operators such as True or False, whereas Numeric can use equals, not equal, less than, more than, less than equals and more than equals. However operators for Dimension and Literal are more problematic. The Dimension datatype contains two numbers separated by an X e.g. 101x48 or 160x160. Hence standard less than and more than does not work as a value could be greater than a comparison value in one dimension but less than the other comparison value in the other dimension. Clearly there is a need for some standard operators to be defined such as canContain and canBeContainedBy to determine which of two dimension values is larger or smaller. The Literal datatype also causes problems: on first glance equals and notEquals seem the most appropriate operators here but UAProf uses Literals to represent version numbers as Numeric can only store integer values. We might want to test if the device is compatible with a certain version or higher. This seems to indicate UAProf need a version number datatype and a backwardCompatible operator.

Another problem concerns merging requests containing different vocabularies. As already noted, there are already several different versions of the UAProf vocabulary currently in use. Therefore it is possible that a phone and a proxy used in combination could use different vocabulary namespaces. Currently under CC/PP and UAProf, even if the phone and the proxy made statements about the same attributes then those attributes would not be merged as they are in different namespaces. However intuitively it seems that attributes should be merged if they are identical in two different UAProf vocabularies but there are no rules defined for doing this. This task is made harder as between different versions of the UAProf specification; some attributes have retained the same name but changed type, parent component or resolution rule. How should the merge be done in such situations?

Q: So how should this be solved?

A: Well the CC/PP recommendation says don't create attributes if suitable attributes already exist. However the UAProf interpretation was to re-use attributes but place them in new namespaces. Perhaps the advice should be "if a suitable attribute already exists, use it in the namespace already defined". This would automatically overcome these namespace problems and merging would be carried out automatically.

Q: So you've talked a lot about UAProf, what does this tell us about CC/PP?

A: Here's a quick summary of the issues:

The notion of profile resolution should be part of the core CC/PP recommendation rather than being specified in the protocol. This would bring it inside the current CC/PP working group charter. CC/PP would not have to specify resolution rules like UAProf but it should define a standard processing model. Specific resolution rules for a given vocabulary would need to be compatible with this processing model to ensure that resolution can be performed on RDF models.

The CC/PP recommendation also needs to introduce the notion of profile validity. This would be done by defining a fixed set of rules that are used to determine if a profile is valid or not. These rules would specify how a validating processor would work. Such rules should not require the creation of new technology; rather they should leverage RDF schema and XML schema where necessary.

The importance of defining semantic meaning for attribute values within vocabularies needs to be emphasised in the CC/PP recommendation. People should not be encouraged to use vocabularies where attribute value meaning has not been defined. Validation should be extended to cover attribute values as well as attribute names.

CC/PP needs to propose a proper method of describing data-types, and operators that can then be used on those datatypes. This could reuse work currently underway on datatypes in RDF.

The advice the CC/PP recommendation gives on creating new attributes should be updated to encourage people to re-use existing namespaces, not just to re-use existing attribute names.

A more major change would be to reconsider the role of RDF in CC/PP. For example if CC/PP adopted an alternative serialisation then this could be designed so it is easier to parse CC/PP and perform XPath queries. However such a serialisation could still be converted to RDF so you would still be able to use RDF tools if necessary.

I'm working on a more formal issue list, along with a number of possible proposals to submit to the W3C.

Q: Are these the reasons why UAProf has not yet been widely deployed?

A: I'm not sure. I would be interested to hear the phone vendors' position on this. Apparently concern about UAProf is phone vendors have been trying to work out what kind of server capability they need for their profile repository. Apparently some people base their assumptions on the idea that the origin server queries the repository every time the phone makes a request. This leads to a very high estimated value for repository server usage, which in turn would require a considerable investment in infrastructure. This could put people off deploying UAProf! Several implementations, including DELI, cache profiles at the origin server, which drastically reduces such

estimates. Perhaps caching needs to be discussed in the CC/PP and UAProf documents.

On a different note, I'd recommend you read this paper[15] as it provides an amusing account of why getting people to provide metadata can be difficult.

---

[1] DELI, http://www-uk.hpl.hp.com/people/marbut/deli/
[2] DELI: A Delivery Context Library for CC/PP and UAProf, http://www-uk.hpl.hp.com/people/marbut/DeliUserGuideWEB.htm
[3] CC/PP: Structure and Vocabularies, http://www.w3.org/TR/2001/WD-CCPP-struct-vocab-20010315/
[4] RDF Model and Syntax Specification, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/
[5] Composite profile information, *Carl Binding, Reto Hermann, Andreas Schade,* http://www.w3.org/2002/02/DIWS/submission/aschadeCompositeProfileInformation.html
[6] XSmiles, http://www.xsmiles.org/
[7] Co-Parsing of RDF and XML, Jeremy Carroll, http://www-uk.hpl.hp.com/people/jjc/docs/r292.pdf
[8] Jena, http://www.hpl.hp.com/semweb/jena-top.html
[9] Apache Cocoon, http://xml.apache.org/cocoon/
[10] CC/PP exchange protocol using HTTP Extension Framework, http://www.w3.org/TR/NOTE-CCPPexchange
[11] Ericsson T68 profile, http://mobileinternet.ericsson.com/UAprof/T68R1.xml
[12] Ericsson T39 profile, http://mobileinternet.ericsson.com/UAprof/T39.xml
[13] CC/PP and UAProf: Issues, improvements and future directions, http://www-uk.hpl.hp.com/people/marbut/deliverycontextFinal.html
[14] Mitsubishi Trium profile http://www.mitsubishi-telecom.com/profiles/eclipse.ua
[15] The Seven Straw Men Of The Meta-Utopia, http://www.well.com/~doctorow/metacrap.htm