WAG UAPROF

Version 10-Nov-1999

Wireless Application Group
User Agent Profile Specification

© Wireless Application Protocol Forum, Ltd. 1999. All rights reserved.

1 Contents

2 References

[CC/PP] Franklin Reynolds, Johan Hjelm, Spencer Dawkins, and Sandeep Singhal, "Composite Capability/ Preferences Profiles: A user side framework for content negotiation," W3C Note, 27 July 1999, http://www.w3.org/TR/1998/NOTE-CCPP-19990727. (Note that this specification refers to the 7/99 version of this specification, irrespective of the availability of future versions of that specification).

[CC/PPex] Hidetaka Ohto and Johan Hjelm, "Composite Capability/Preference Profiles (CC/PP) Exchange protocol based on HTTP Extension Framework," W3C NOTE 17 June 1999, http://www.w3.org/Mobile/Group/IG/1999/06/NOTE-CCPPexchange-19990617. (Note that this specification refers to the 6/99 version of this specification, irrespective of the availability of future versions of that specification).

[CHTR] "Wireless Application Group: UAPROF Drafting Committee Charter," WAP Forum, 12/98, http://www.wapforum.org/member/wg/wag/Activities/ActivityUAPROF/UAPCharter.htm.

[CONNEG] Graham Klyne, "Requirements for Protocol-Independent Content Negotiation," IETF draft RFC, 3/98.

[HTTP] "Hypertext Transfer Protocol -- HTTP/1.1," RFC2616.

[HTTPext] H. Frystyk Nielsen, P. Leach, and Scott Lawrence, "HTTP Extension Framework," Internet Draft, Jan 1999, http://www.w3.org/Protocols/HTTP/ietf-http-ext/draft-frystyk-http-extensions-03.

[MExE] ETSI Special Mobile Group (SMG) 4, Mobile Station Application Execution Environment (MExE), http://www.etsi.org.

[P3P] "Platform for Privacy Preferences Project," W3C, 11/98, http://www.w3.org/TR/WD-P3P/.

[RDF] Lassila, "Resource Description Framework (RDF) Model and Syntax Specification," W3C Recommendation, 2/99, http://www.w3.org/TR/1999/PR-rdf-syntax-19990105.

[RDF-Schema] Brickley, et al., "RDF Schema Specification," W3C Proposed Recommendation, 3/99, http://www.w3.org/TR/WD-rdf-schema.

[RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396.

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119.

[Salutation] Salutation Consortium Specification, http://www.salutation.org.

[SiRPAC] SiRPAC Compiler and Parser, http://www.w3.org/RDF/Implementation/SiRPAC.

[UA-Attrs] Tomihisa Kamada, et. al., "Client-specific Web Services by using User Agent Attributes," 12/97, http://www.w3.org/TR/NOTE-agent-attributes.

[WAP-PAP] "WAP Push Access Protocol," 16-August-1999, WAP Forum Ltd.

[WAP-PushArch] "WAP Push Architecture Specification," 16-August 1999, WAP Forum Ltd.

[WBXML] "Binary XML Specification, version 1.2," WAP Forum Ltd.

[WINA] "WAP Interim Naming Authority," WAP Forum Ltd.

[WAE] "WAP Application Environment Specification, version 1.2," WAP Forum Ltd.

[WSP] "Wireless Session Protocol, version 1.2," WAP Forum Ltd.

[WTAI] "Wireless Telephony Application Interface Specification," 10 February 1999, WAP Forum Ltd.

[XHTML] "XHTML TM 1.0: The Extensible HypterText Markup Language," W3C Proposed Recommendation, 8/99, http://www.w3.org/TR/PR-xhtml1-19990824.html.

[XML] "eXtensible Markup Langauge," W3C.

[XML-NS] Bray, et al., "Namespaces in XML," W3C Recommendation, 1/99, http://www.w3.org/TR/1999/REC-xml-names-19990114..

3 Definitions, Abbreviations, and Conventions

The key words "*MUST*," "*MUST NOT*," "*SHOULD*," "*SHOULD NOT*," "*MAY*," and "*MAY NOT*" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3.1 Definitions

Attribute: An RDF attribute refers to the data elements describing the CPI and is denoted as an RDF property. Each attribute is associated with a value or a list of values or resource.

CC/PP Repository: A server that stores CPI persistently in a form that may be referenced by and incorporated into a profile. A CC/PP repository is typically a Web server that provides CC/PP profiles in response to HTTP requests.

Component: Elements of a high level classification of the information in the CPI. For UAProf, these include *HardwarePlatform*, *SoftwarePlatform*, *NetworkCharacteristics*, *WAPCharacteristics*, and *BrowserUA*.

CPI: Capabilities and Preference Information pertaining to the capabilities of the device, the operating and network environment, and user's personal preferences for receiving content and/or resource. CPI are represented by means of a Profile.

Gateway: Software that is capable of bridging disparate network protocols. For the purposes of this specification, "gateway" refers to protocol bridging functionality, which may exist in a stand-alone gateway or may be co-located with a proxy or origin server.

Home Location Register: A permanent database used in cellular networks. The HLR is located on the SCP (Signal Control Point) of the cellular provider of record, and is used to identify/verify a subscriber; it also contains subscriber data related to features and services.

Origin Server: Software that can respond to requests from a WAP terminal by delivering appropriate content or error messages. The origin server may receive requests via either WSP or HTTP. Application programs executing on the origin server can use UAProf to deliver content that is tailored in accordance with the CPI that can be found within the provided Profile. For the purpose of this specification, "origin server" refers to content generation capabilities, which may physically exist in a stand-alone Web server or may be co-located with a proxy or gateway.

Profile: An instance of the schema that describe capabilities for a specific device and network. A profile need not have all the attributes identified in the vocabulary/schema.

Property: An RDF property is a specific aspect, characteristic, capability or relation (metadata) used to describe a resource.

Proxy: Software that receives HTTP requests and forwards that request toward the origin server (possibly by way of an upstream proxy) using HTTP. The proxy receives the response from the origin server and forwards it to the requesting client. In providing its forwarding functions, the proxy may modify either the request or response or provide other value-added functions. For the purposes of this specification, "proxy" refers to request/response forwarding functionality, which may exist in a stand-alone HTTP proxy or may be co-located with a gateway or origin server.

Resource: An object or element being described by RDF expressions is a resource. An RDF resource is typically identified by a URI.

Schema: An RDF schema denotes resources which constitute the particular unchanging versions of an RDF vocabulary at any point in time. It is used to provide semantic information (such as organization and relationship) about the interpretation of the statements in an RDF data model. It does not include the values associated with the attributes.

User: An individual or group of individuals acting as a single entity. The user is further qualified as an entity who uses a device to request content and/or resource from a server.

User agent: A program, such as a browser, running on the device that acts on a user's behalf. Users may use different user agents at different times.

Vocabulary: A collection of attributes that adequately describe the CPI. A vocabulary is associated with a schema. The vocabulary for UAProf includes attributes pertaining to the device capabilities and network characteristics.

3.2 Abbreviations

For the purposes of this specification the following abbreviations apply.

CC/PP	Composite Capability/ Preferences Profiles
CC/PP-HTTP	CC/PP Exchange Protocol over HTTP
CC/PP-WSP	CC/PP Exchange Protocol over WSP
CPI	Capability and Preference Information
HTML	Hyper-Text Markup Language
HTTP	Hyper-Text Transfer Protocol
IANA	Internet Assigned Numbers Authority
P3P	Platform for Privacy Preferences Project
RDF	Resource Description Framework
SiRPAC	Simple RDF Parser and Compiler
UAProf	User Agent Profile
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WAP	Wireless Application Protocol
WBXML	WAP Binary XML
WML	Wireless Markup Language
WSP	Wireless Session Protocol
WTA	Wireless Telephony Application
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language

4 Introduction

This section is informative.

Existing markup languages and content written in those markup languages presume that devices have similar display sizes, memory capacities, and software capabilities. Content is also largely oblivious to the available network bandwidth and perceived network latency. As WAP-enabled devices come of age, this homogeneity assumption is no longer universally valid. In particular, mobile devices can be expected to have an an ever-divergent range of input and output capabilities, network connectivity, and levels of scripting language support. Moreover, users may have content presentation preferences that also cannot be transferred to the server for consideration. As a result of this device heterogeneity and the limited ability of users to convey their content presentation preferences to the server, clients may receive content that they cannot store, that they cannot display, that violates the desires of the user, or that takes too long to convey over the network to the client device. Recently, work has begun within the World-Wide Web Consortium (W3C) to define mechanisms for describing and transmitting information about the capabilities of Web clients and the display preferences of Web users. The Composite Capabilities/Preferences Profile (CC/PP) note [CC/PP] defines a high-level structured framework for describing this information using the Resource Description Framework (RDF) [RDF]. CC/PP profiles are structured into named "components," each containing a collection of attribute-value pairs, or properties. Each component may optionally provide a default description block containing either a set of default values for attributes of that component or a URI that refers to a document containing those default alues. Any attributes explicitly provided in the component description therefore override the default values provided in the default description block or through that URI. The CC/PP specification does not mandate a particular set of components or attributes, choosing instead to defer that definition to other standards bodies. The CC/PP Exchange Protocol over HTTP [CC/PPex] enables CC/PP profiles to be transported over the HTTP 1.1 protocol [HTTP] with the HTTP Extension Framework [HTTPex]. This protocol enables effective profile caching at Web servers and proxies.

The User Agent Profile (UAProf) specification extends WAP 1.1 to enable the end-to-end flow of a User Agent Profile (UAProf), also referred to as ***Capability and Preference Information*** (CPI), between the WAP client, the intermediate network points, and the origin server. It seeks to interoperate seamlessly with the emerging standards for Composite Capability/Preference Profile (CC/PP) [CCPP,CCPPex] distribution over the Internet. It uses the CC/PP model to define a robust, extensible framework for describing and transmitting CPI about the client, user, and network that will be processing the content contained in a WSP response. The specification defines a set of components and attributes that WAP-enabled devices may convey within the CPI.[1] This CPI may include, but is not limited to, hardware characteristics (screen size, color capabilities, image capabilities, manufacturer, etc.), software characteristics (operating system vendor and version, support for MExE, list of audio and video encoders, etc.), application/user preferences (browser manufacturer and version, markup languages and versions supported, scripting languages supported, etc.), WAP characteristics (WML script libraries, WAP version, WML deck size, etc.), and network characteristics (bearer characteristics such as latency and reliability, etc.). This specification seeks to minimize wireless bandwidth consumption by using a binary encoding for the CPI and by supporting efficient transmission and caching over WSP in a manner that allows easy interoperability with the CC/PP Exchange Protocol over HTTP.

As a request travels over the network from the client device to the origin server, each network element may optionally add additional profile information to the transmitted CPI. These additions may provide information available solely to that particular network element. Alternatively, this information may override the capabilities exposed by the client, particularly in cases where that network element is capable of performing in-band content transformations to meet the capability requirements of the requesting client device.

Origin servers, gateways, and proxies can use the CPI to ensure that the user receives content that is particularly tailored for the environment in which it will be presented. Moreover, this specification permits the origin server to

[1] Though a set of well-known components and attributes are defined within this specification, individual implementors are free to provide additional components and attributes with their CPI. However, most origin servers or proxies are unlikely to properly interpret those extensions unless they are standardized by another standards body.

select and deliver services that are appropriate to the capabilities of the requesting client. Finally, it is expected that this specification will be used to enhance content personalization based on user preferences, and other factors, as enabled by the Platform for Privacy Preferences Project (P3P) [P3P].

4.1 Scope

The *user agent profile* is concerned with capturing classes of device preference information. These classes include (but are not restricted to) the hardware and software characteristics of the device as well as information about the network to which the device is connected. The user agent profile contains information used for *content* formatting *purposes.* A user agent profile is distinct from a *user preference profile* that would contain application-specific information about the user for content *selection* purposes. For example, a user preference profile might designate whether the user is interested in receiving sports scores and, if so, the particular teams. The specification of user preference profiles is beyond the scope of this document.

In accordance with the CC/PP note [CCPP], the user agent profile schema is defined using an RDF schema and vocabulary [RDF]. This document defines the structure of this schema in terms of the class definitions and semantics of attributes for devices adhering to the WAP standard. Extensibilty guidelines are also offered to assist in extending the schema with new components.

Regarding the user agent profile, this document assumes that:

- The information contained within the profile is provided on behalf of the user who will be receiving the content contained in the associated WSP response.
- The CC/PP repository storing the profile is secure, meaning that it does not permit unauthorized modification to stored profile information. The functionality associated with this repository may be implemented in a WAP gateway or intermediate proxy or as a separate standalone element in the network.
- WSP/HTTP headers are generally not encrypted. Because this specification does not define any security mechanisms, care will be taken when including user-confidential information[2] in the profile unless the profile is transmitted over an end-to-end secure channel (e.g. WTLS to the origin server).
- An implicit chain of trust exists between the client and origin server. The integrity of the profile is maintained (in other words, not compromised) as it is transmitted through or cached within the network. It is assumed that the network elements that contribute property descriptions to the profile are trusted. Network elements will not assemble a "history" about users by tracking the deltas in their profile over time.
- To ensure the integritiy of the profile, lower-level mechanisms, such as WTLS, must be used.

Use of the profile by the origin server is intended to enhance the user's experience. Therefore, should the profile be discarded, be corrupted, or be otherwise inaccessible, then the origin server will still provide and deliver content to the client in a best-efforts fashion.

4.2 Areas for Further Work

Areas for further work include—but are not limited to—the following:

- Billing
- Security of CPI transmission, persistence, and update
- Client-side capture and storage of application-preferences related to a user
- Transmission of CPI profile subsets by a client or proxy

[2] The definition of "*confidential*" varies among different users, cultures, and service providers. It is therefore impossible to define a profile schema that meets the security needs of all users while still conveying any useful information.

- Third-party WSP session invocation and termination, and third-party update to cached CPI information associated with a WSP session
- WSP session redirection and gateway-to-gateway transfer of the cached CPI associated with the transferred session
- Support for third-party requests to a gateway or proxy to retrieve, update, and manage profile information about a particular client or group of clients.
    - A naming scheme allowing a third-party requester to designate a particular WSP session or particular profile attributes or components of interest. For example, a PUSH operation of an e-mail message to an e-mail client may require different information about the client than a push operation involving a graphic image that may be handled by an image viewer.
    - A persistence model to allow profile queries to be serviced by WAP gateways without contacting the client. This allows queries to be handled while no active WSP session is in place, while the client cannot be contacted, or when wireless network bandwidth is limited. Moreover, a caching, or "freshness," policy controls when the WAP gateway must fetch the profile directly from the client(s).
    - A protocol and format for retrieving the profile in the form of unresolved Profile and Profile-Diff headers.

4.3 Relationship to Other Standards

This specification builds upon and coexists with numerous WAP and Internet standards. These relationships are summarized below:

- Composite Capability/Preferences Profiles (CC/PP) [CC/PP]: This specification defines the CPI structure according to the structure mandated by the CC/PP specification note.
- Resource Description Framework (RDF) [RDF]: The CC/PP specification uses the RDF syntax to represent the CPI. In designing or extending the schema, a schema designer must be familiar with the RDF concepts.
- Wireless Session Protocol (WSP) [WSP]: CPI is transmitted over wireless networks within WSP headers.
- WAP Binary XML (WBXML) [WBXML]: When the CPI is transmitted over the WSP protocol, this specification mandates that it be encoded according to the WBXML specification.
- CC/PP Exchange Protocol over HTTP [CC/PPex]: When transmitted over the Internet, this specification requires that the CPI be transmitted using the CC/PP Exchange Protocol over HTTP which, in turn, defines headers in HTTP 1.1 [HTTP] with the HTTP Extension Framework [HTTPex].
- WAP Push Access Protocol (PAP) [WAP-PAP]: This protocol is used by push origin servers to retrieve the CPI from the WAP gateway or Push Protocol Proxy (PPG). The request is issued over HTTP, and the response contains the profile, with MIME type **text/xml**.

5 End-to-End Architecture

This section describes the end-to-end architecture within which this specification operates. It is informative.

This specification provides for the end-to-end specification, delivery, and processing of composite capability information from the device. The information is collected on the client device, encoded into an efficient binary form, transmitted and cached within a WSP session, optionally enhanced with information provided with a particular request, optionally combined with other information available over the network, and made available to the origin server. Over the Internet, this specification assumes the use of the CC/PP [CC/PP], CC/PP Exchange Protocol over HTTP [CC/PPex], and HTTP 1.1 [HTTP] with the HTTP Extension Framework [HTTPex].

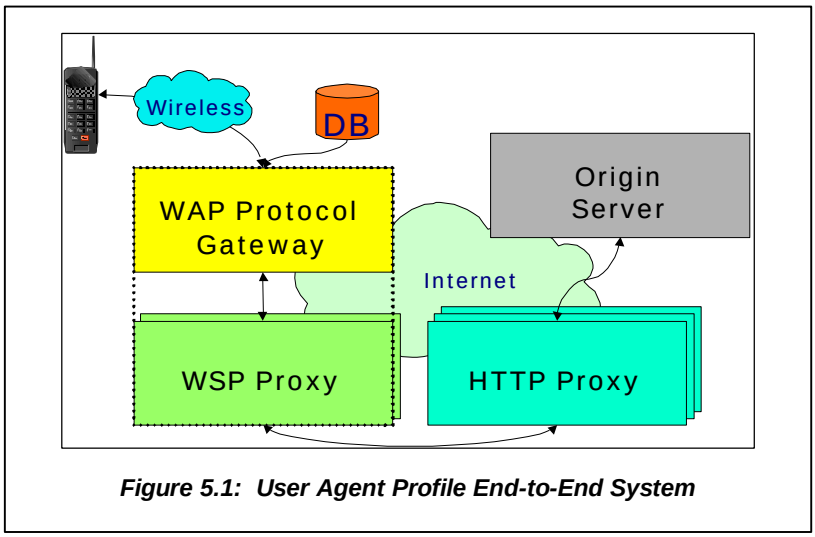


Figure 5.1: User Agent Profile End-to-End System

The end-to-end system consists of five logical components:

- A client device capable of requesting and rendering WAP content.
- A wireless network employing WAP 1.1 or later protocols.
- A WAP-capable gateway capable of translating WAP protocol requests into corresponding requests over the Internet and translating responses from the Internet into corresponding responses over the WAP protocols.
- The Internet or an intranet using TCP/IP-based protocols and possibly having one or more protocol gateways and Web/HTTP proxies.
- An origin (Web) server that can generate requested content.

Though this specification refers to five end-to-end system components, actual configurations may physically deploy those components in many forms. For example, the latter three components (WAP gateway, Internet/intranet, and origin server) might easily be merged into a single server-side system connected to the WAP network. Moreover, the WAP gateway may itself be distributed, with different hosts serving as endpoints for different layers of the WAP protocol stack.

Each of the following sections highlights the architectural assumptions behind each of the logical components.

5.1 Client Device

The CPI consists of information gathered from the device hardware, active user agent software, and user preferences. In many cases, much of this information must be pre-installed directly on the device, possibly in the firmware. For instance, the device may publish a single URI that points to default device capability information made available by the device manufacturer. Similarly, the user agent may publish a single URI that points to default software information made available by the software developer.

The client device is assumed to employ the WAP communications protocols, particularly WSP [WSP], to request content from an origin server. The CPI is transmitted and maintained using designated WSP headers in accordance with this specification (see Section 9). This information is initially conveyed when a WSP session is established with

a compliant WAP protocol gateway. The client thereafter assumes that the WAP gateway caches the CPI and will apply it on all requests initiated during the lifetime of the WSP session.

The CPI is scoped to the particular WSP session within which it is transmitted; each of a client's active WSP sessions therefore may be associated with a different set of CPI. The client may update the cached CPI for a WSP session at any time, with such updates taking effect either only for a particular request or for the remaining lifespan of the session. However, network efficiency concerns dictate that the CPI should not change frequently during a WSP session. Therefore, though multiple client user agents may share a single WSP session, optimal network behavior is typically achieved by restricting use of the session to a single user agent or application.

5.2 Wireless Network

WSP sessions are carried over wireless networks that are capable of implementing the WAP protocols.

5.3 WAP Gateway

The WAP gateway represents the server-side endpoint for the client's WSP session. To support these sessions, the gateway must suport the WDP and WTP protocol layers and, optionally, WTLS. As part of its WSP session implementation, the WAP gateway must implement WSP header caching, thereby allowing it to hold the CPI conveyed by the client device during session establishment.

The WAP gateway is responsible for translating WSP requests into appropriate HTTP 1.1 requests for delivery over an intranet or the Internet to the designated origin server. In forwarding these requests, the gateway must also forward the current CPI associated with the session and/or request. This specification requires that the gateway use the HTTP Extension Framework to convey the CPI information within HTTP headers, as discussed in Section 9.3.3. When generating the HTTP request, the gateway may optionally augment the received CPI with additional data obtained from local databases, such as a network Home Location Register (HLR.).

The WAP gateway is also responsible for translating HTTP responses into appropriate WSP responses for delivery over the wireless network to the requesting client device. In forwarding these responses, the gateway must also forward any CPI usage headers provided by the origin server and/or any intermediate HTTP proxies.

5.4 Internet or Intranet

The HTTP requests generated by the WAP gateway are conveyed over an intranet or the Internet, capable of carrying TCP/IP-based requests and responses. In passing through these networks, the request may pass through one or more proxies, each responsible for forwarding the request toward the particular origin server designated in the request. These proxies may conform to either the HTTP 1.0 or HTTP 1.1 protocol standards. It is important to note that conformant HTTP 1.0 proxies will discard all CPI information contained in the HTTP request. Conformant HTTP 1.1 proxies may or may not forward the CPI information in tact, depending on whether the CPI information is conveyed in Mandatory or Optional headers. Conformant HTTP 1.1 proxies that are aware of the HTTP Extension Framework and CC/PP Exchange Protocol over HTTP optionally may add information to the CPI conveyed in the outbound HTTP request. Conformant proxies optionally may also manipulate the content contained in the associated HTTP response to make it better conform to the active CPI offered by the device.

Internet network elements, both proxies and origin servers, may provide content caching capabilities. Caching is complicated by the presence of CPI information because the content associated with a particular URI may differ according to the CPI information presented to the origin server. As a rule, therefore, a conformant HTTP proxy or origin server will only deliver content from its cache if both of the following conditions hold:

- The content has not expired from the cache, in accordance with standard HTTP caching semantics
- The CPI associated with the cached request exactly matches the CPI associated with the new request

To minimize the possibility that an intermediate proxy that is unaware of CC/PP accidentally sources content from its cache without first checking for a matching CC/PP profile, an origin server may set the Cache-Control headers in the HTTP response to prevent the proxy from doing any caching.

5.5 Origin Server

The origin server is the ultimate recipient of the request initiated by the client device (and forwarded as an HTTP request from the WAP gateway). The origin server is responsible for receiving the request and generating appropriate content that is subsequently transported as an HTTP response to the WAP gateway. In generating this response, the origin server extracts the CPI conveyed with the HTTP request, resolves all indirect references to information stored at other repositories in the network, if necessary, and uses that information to select or otherwise customize the content being delivered to the client. In generating the HTTP response using the CC/PP Exchange Protocol over HTTP, a conformant server must indicate the extent to which the CPI was honored in producing the content contained within the HTTP response.

Conformant origin servers are expected to honor the HTTP 1.1 protocol with the HTTP Extension Framework [HTTPex]. However, non-conformant HTTP 1.0 and HTTP 1.1 origin servers may also be capable of processing HTTP requests containing CPI by specifically parsing headers designated by the CC/PP Exchange Protocol over HTTP.

6 Usage Scenarios

This Section describes several scenarios in which CPI is conveyed and used to support information delivery to a WAP-enabled client. This Section is informative.

6.1 Opening a WSP Session and Establishing an Initial UAProf

Upon opening a WSP session, the UAProf-aware client conveys its profile information using Profile and Profile-Diff headers within the WSP **Connect** request. The values of these headers are constructed by encoding the CPI using a WBXML encoding. Upon receiving the profile, a WAP gateway that is aware of the UAProf capability responds with a Profile-Warning header value of 100 ("OK"). This header signals to the client that the CPI is being cached by the WAP gateway and will be effective for the lifetime of the session. The client device may update the CPI at any time during the session lifetime.

If the client does not receive the "OK" Profile-Warning header in the WSP **Connect** response, it assumes that the gateway does not support this UAProf specification and therefore that the CPI is not being cached by the WAP gateway. The client device may neither convey nor update the CPI during the session lifetime.

6.2 Updating the UAProf During an Active WSP Session

While a UAProf-aware session is established, the client may update the active UAProf at any time. To do this, the client transmits a WSP Session **Resume** message to the WAP gateway, said message containing Profile and Profile-Diff headers with the new CPI. Upon updating the cached headers, the gateway responds with a Profile-Warning header value of 100 ("OK"). All future requests issued on the WSP session will be associated with the newly cached Profile and Profile-Diff headers.

6.3 Resuming a Suspended WSP Session

To resume a suspended WSP session, the client initiates a standard WSP **Resume** request. The session, once resumed, retains all of the cached header state at the gateway, including the Profile and Profile-Diff headers containing the CPI.

The lifespan of a WSP session is not tied to power cycles of the device. In many situations, therefore, the client's capabilities may change significantly while a WSP session is suspended. This is the case, for example, if hardware is added or removed from the device while it is turned off. It may also occur if the WSP session is held by a smart card that may be moved to a new device while the session is suspended. In these situations, the client device must update the cached WSP Profile and Profile-Diff headers upon resuming the session. These updated headers are conveyed in the WSP **Resume** request and cached at the WAP gateway, in the same manner described in Section 6.2.

6.4 Suspending an Active WSP Session

To suspend a WSP session, the client initiates a standard WSP **Suspend** request. As long as the session is established, the WAP gateway must cache all negotiated headers, including the Profile and Profile-Diff headers associated with the WSP session. However, should the gateway choose to discard the session, it may also discard these cached headers. To resume the session, the client device follows the steps outlined in Section 6.3.

6.5 Issuing a Request for Content

To request content during a UAProf-aware WSP session, the client issues a standard WSP request to the WAP gateway. The WAP gateway is responsible for forwarding this WSP request to the designated origin server (typically via HTTP using the CC/PP Exchange Protocol over HTTP). In forwarding the request to the origin server, the WAP gateway must include the CPI associated with the WSP session over which the request was conveyed. The origin server receives the HTTP request (which may have been modified by one or more intermediate HTTP proxies), resolves the CPI, and generates a response along with a Profile-Warning header indicating whether the CPI was honored as the content was generated.

The HTTP response may itself be modified by intermediate HTTP proxies to better meet the needs of the requesting client. The WAP gateway forwards the content to the client device over WSP, encoding the Profile-Warning header for efficient transmission over the wireless network.

6.5.1 CPI Provided by the WAP Gateway

As it forwards the request, the WAP gateway may optionally add profile information, reflecting information only available to the gateway. For instance, if a network operator controls the gateway, then the gateway may provide additional network information (such as from a Home Location Register (HLR)) that is not otherwise available to the requesting client. Similarly, the WAP gateway may add information to the profile to override information provided by the requesting device/user. These overrides may, for example, reflect policies in place by the network or gateway operator.

6.5.2 Overriding the CPI Within a Single Request

When issuing a WSP request, the client may provide additional information to override or augment the basic CPI already cached at the WAP gateway. This additional information is only applied during the scope of the associated request, and it is not used by the gateway during subsequent requests.

To augment the profile during a request, the client includes Profile and/or Profile-Diff headers with the WSP request. As it generates an HTTP request, the WAP gateway overrides the cached WSP Profile and Profile-Diff headers with any headers provided in the WSP request. As described above in Section 6.5.1, the gateway may also add additional information to the forwarded profile.

6.6 Provisioned WSP Sessions and CPI

In many situations, a device will be provisioned with one or more static WSP sessions; when the device is issued, the WAP gateway will similarly be provided with information about these sessions. This situation arises, for instance, with one-way devices that are otherwise incapable of transmitting requests to dynamically establish WSP sessions. As the WSP session is provisioned, the session may be associated with one or more Profile and Profile-Diff headers which are consequently cached by the WAP gateway as a standard part of the WSP session state. For the purposes of this specification, these provisioned headers are treated in the same manner as cached headers that are provided by the client device during WSP session establishment. In particular, these headers are delivered to origin servers during gateway-initiated requests, as illustrated in Section 6.5 and specified in Section 10, and they are delivered to third-party hosts upon request, as illustrated to Section 6.8.

6.7 Resolving Attribute Values in the CPI

An origin server or proxy that needs to determine the correct values for CPI attributes must resolve the profile. This resolution process applies a collection of default attribute values and then applies appropriate overrides to those

defaults. Because different network elements may provide additional (or overriding) profile information, the resolution process must apply this additional information to determine the final attribute values.

The User Agent Profile is constructed in three stages:

- Resolve all indirect references by retrieving URI references contained within the profile
- Resolve each Profile and Profile-Diff document by first applying attribute values contained in the default URI references and by second applying overriding attribute values contained within the category blocks of that Profile or Profile-Diff.
- Determine the final value of the attributes by applying the resolved attribute values from each Profile and Profile-Diff in order, with the attribute values determined by the resolution rules provided in the schema. Where no resolution rules are provided for a particular attribute in the schema, values provided in profile diffs are assumed to override values provided in previous profiles or profile diffs.

6.8 Third-Party Requests for Cached Profile, Including PUSH

While a WSP session is established (whether active or suspended), the WAP gateway caches all Profile and Profile-Diff headers associated with that session. A third party host may issue a request for this CPI to, for instance, generate content that wil subsequently be pushed to the client device.

The request is initiated from the third-party host and delivered to a Push Protocol Gateway (PPG). This specification defines neither a protocol for issuing this request nor a means for addressing the requested information. However, it is expected that such requests will typically be made to the WAP gateway using HTTP, as suggested by [WAP-PAP].

Upon receiving the profile request, the gateway accesses the cached Profile and Profile-Diff headers and resolves them to form a complete CPI profile. It responds to the CPI request with the resolved profile (a CC/PP document) using a MIME type of **text/xml**.

It is important to note that the WAP gateway has incomplete information about the current CPI. In particular, the gateway is not aware of any request-specific profile information that the client would have provided to the requesting third-party server. Moreover, the WAP gateway cannot incorporate attribute information from Profile-Diff headers that would have been added by intermediate proxies through which the request would have passed had it originated at the client device. Finally, if the WSP session is currently suspended, then the gateway may be caching out-of-date profile information.

7 Composite Profile Segments and Attributes

With the exception of this introduction, this section is normative. Throughout this section, sentences in Italics are also informative.

The CC/PP framework [CC/PP] enables information about the capabilities and characteristics of a device and network to be communicated to web servers and gateways/proxies so that suitable content is rendered to the device. These capabilities and characteristics are referred to as attributes, and together form a vocabulary. The semantics associated with these attributes are identified in a schema for that vocabulary. A profile is an instance of the schema and contains one or more attributes from the vocabulary. The attributes in the schema are classified into one of several components, each of which represents a distinguished set of characteristics.

CC/PP uses Resource Description Framework (RDF) [RDF], the technology to describe the meta data for communicating the profile. An RDF description consists of a structured collection of RDF properties, each of which is associated with a property type and value and expressed as a directed labeled graph with nodes, leaves, and arcs or an object oriented data model with tuples (object/ attribute/value).

7.1 Schema Layout

Definition of a CC/PP schema is governed by the following rules:

- The schema MUST be associated with a well-defined vocabulary. A unique URI (identified as **prf**) in the XML namespace MUST serve as an unambiguous identifier for the particular vocabulary.
- The schema for a profile MUST consist of one or more components, each describing a set of attributes within one or more description blocks.
- All components in a CC/PP profile MUST have the same schema structure (layout).
- Each component MUST be an object of type *Class* and MAY contain a subordinate description block for default attributes.
- Attributes considered as default capabilities or characteristics MUST be described within the Default description block. *Typically, these include attributes that are common to a group or class of devices or user agents across multiple profiles. For example, profiles for users of a Palm III device would all have the same Default hardware characteristics as specified by the vendor.*
- Descriptions to override the default values MUST also be included in the component description but outside the Default description block. The final value of an attribute described in the profile is resolved based on the semantics associated with the attribute.

7.2 User Agent Profile Components

The schema for WAP User Agent Profiles consists of description blocks for the following key components:

HardwarePlatform: A collection of properties that adequately describe the hardware characteristics of the terminal device. This includes, the type of device, model number, display size, input and output methods, etc.

SoftwarePlatform: A collection of attributes associated with the operating environment of the device. Attributes provide information on the operating system software, video and audio encoders supported by the device, and user's preference on language .

BrowserUA: A set of attributes to describe the HTML browser application

NetworkCharacteristics: Information about the network-related infrastructure and environment such as bearer information. *These attributes can influence the resulting content, due to the variation in capabilities and characteristics of various network infrastructures in terms of bandwidth and device accessibility.*

WapCharacteristics: A set of attributes pertaining to WAP capabilities supported on the device. This includes details on the capabilities and characteristics related to the WML Browser, WTA [WTA], etc.

Additional components can be added to the schema to describe capabilities pertaining to other user agents such as an Email application or hardware extensions. See Section 7.5 for details.

7.3 Attributes

A profile attribute MUST belong to one and only one component of the schema. Since CC/PP creates an unambiguous distinction between the component classes, a domain constraint (**rdfs:domain**) MUST be used to describe the attribute-component relationship in the schema. If an origin server application is interested in a set of attributes that spans multiple components, it MUST parse the complete profile to obtain the necessary information.

Profile attributes MUST be defined using a traditional name and value pair syntax. The first half of the name-value pair describes the attribute, and the other half provides the value itself. The RDF property descriptions MUST be unique and unambiguous in both semantics and value.

Within an RDF component description block, the attribute description MUST be either embedded lexically inline to denote the value or expressed as an RDF resource (using indirect or remote reference such as URIs) that must be resolved to obtain the full description.

Attributes with composite or multiple values MUST be described as RDF resources. For instance, the *Default* attribute points to a collection of attributes and should therefore be described as a URI resource. Similarly, an RDF container (Bag or Sequence) MUST be used to describe a list of values (ordered or unordered) associated with a given attribute. *For example, the InputCharSet attribute would be expressed as follows*

```
<prf:InputCharSet>
      <rdf:Bag>
            <rdf:li>US-ASCII</rdf:li>
            <rdf:li>ShiftJIS</rdf:li>
      </rdf:Bag>
</prf:InputCharSet>
```

The server parsing and interpreting the profile MAY carry out validation of the attribute descriptions in terms of units and value.

The value of an attribute with multiple descriptions MUST be resolved as follows:

- The description of the attribute within the Default tag MUST be resolved first
- Any other description of the attribute identified in a subsequent instantiation of the attribute MUST override the default description
- Where multiple descriptions of the attribute exist outside the default description block, the ultimate value of the attribute MUST be determined by the resolution rules for that attribute. An attribute MUST be associated with one of the following three resolution rules:

    1. **Locked:** The final value is determined by the first description outside the default description block
    2. **Override:** The final value equals the last description of the attribute
    3. **Append:** The final value is a list of all the descriptions of the attribute

    For example, the *ModelNumber* attribute has a resolution semantic identified as *Locked*. This implies that the first non-default description of the attribute is the final value or that subsequent instantiations of the attribute description cannot change the value of that attribute. On the other hand, an attribute such as *ImageCapable* has a resolution semantic of *Override* which means that of the multiple descriptions of the attribute, only the last one is ultimately used to determine the final value. Finally, an attribute with a resolution semantic *Append* has a final value that includes all the descriptions for that attribute in the profile.

An attribute (RDF property) MUST closely represent the semantics of the capability or characteristics being represented and MUST follow the naming conventions identified in [RDF] and [RDF-Schema]. For example, the *SoftKeysCapable* attribute indicates whether or not the device supports programmable soft keys.

7.4 Profile

A profile is an instance of the schema. The following rules govern the formation of a profile:

- The root node of the profile MUST be identified with an invariant node name such as "MyProfile". This name is fixed for all user agents implementing the profile.
- The profile MUST contain one or more attributes identified in the base vocabulary. It is not required for a profile to contain all the attributes in the base vocabulary. In other words, the profile instantiates a subset of the attributes in the vocabulary.

- The components in the profile are instances of the components identified in the schema. They MUST be identified as such, by means of the **rdf:type** attribute whose value matches the name of the component type in the schema. For example, **SBCNetworkChar** is an instantiated component of a profile, which is of type **NetworkCharacteristics**. The profile must therefore include an RDF statement such as

```
<rdf:Description ID="SBCNetworkChar">
<rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#NetworkCharacteristics"/>
.....
```

Furthermore, all profile parsers MUST parse the profile based on the type of the component and not based on the name (since the name can vary, but the type is defined in the schema). This applies to the merging of the various delta profiles (diffs) that are generated or appended by different network elements.

7.5 User Agent Profile Schema and Base Vocabulary

This section specifies the schema and base vocabulary for WAP User Agent Profiles. The schema is expressed in RDF and encoded in XML, and it includes semantic descriptions of all attributes identified in the vocabulary. However, a profile that conforms to the schema is not required to define all attributes contained in the schema. Because of limitations in the existing XML and RDF standards, certain information in the schema such as attribute type is currently described in the form of RDF or XML comments. For the purposes of determining whether a profile conforms to this specification, the entire schema--including comments--is considered normative. *In future versions of this specification, information currently contained within comments is likely to be described more formally using standard XML and/or RDF methods.*

The base vocabulary is tabulated in Appendix A.1 for easy reference.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf  = "http://www.w3.org/TR/1999/02/22-rdf-syntax-ns##"
         xmlns:rdfs = "http://www.w3.org/1999/PR-rdf-schema-19990303#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">


<rdf:Description ID="Component">
   <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
   <rdfs:subClassOf
     rdf:resource="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Resource"/>
   <rdfs:label>Component</rdfs:label>
   <rdfs:comment>
      A Component within the CC/PP Schema is a class of related properties
      that describe the capabilities and preferences information.
   </rdfs:comment>
</rdf:Description>



<!-- ************************************************************** -->
<!-- ***** Properties shared among the components***** -->

<rdf:Description ID="component">
   <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:label>component</rdfs:label>
   <rdfs:comment>
      The component attribute links the various components to the root node
      (profile).
    </rdfs:comment>
 </rdf:Description>

<rdf:Description ID="Defaults">
   <rdfs:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
```

```
      <rdfs:domain rdf:resource="#WapCharacteristics"/>
      <rdfs:domain rdf:resource="#BrowserUA"/>
      <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
      <rdfs:comment>
         An attribute used to identify the default capabilities.
      </rdfs:comment>
   </rdf:Description>



   <!-- ************************************************************** -->
   <!-- ***** Component Definitions ***** -->

   <rdf:Description ID="HardwarePlatform">
      <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Component"/>
      <rdfs:label>Component: HardwarePlatform</rdfs:label>
      <rdfs:comment>
         The HardwarePlatform component contains properties of the device's
         Hardware, such as display size, supported character sets, etc.
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="SoftwarePlatform">
      <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Component"/>
      <rdfs:label>Component: SoftwarePlatform</rdfs:label>
      <rdfs:comment>
         The SoftwarePlatform component contains properties of the device's
         application environment, operating system, and installed software.
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="BrowserUA">
      <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Component"/>
      <rdfs:label>Component: BrowserUA</rdfs:label>
      <rdfs:comment>
         The BrowserUA component contains attributes related to the browser
         user agent running on the device.
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="NetworkCharacteristics">
      <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Component"/>
      <rdfs:label>Component: NetworkCharacteristics</rdfs:label>
      <rdfs:comment>
         The NetworkCharacteristics component contains properties describing the
         network environment including the supported bearers.
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="WapCharacteristics">
      <rdf:type resource="http://www.w3.org/TR/PR-rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Component"/>
      <rdfs:label>Component: WapCharacteristics</rdfs:label>
      <rdfs:comment>
         The WapCharacteristics component contains properties of the WAP
         environment supported by the device.
      </rdfs:comment>
   </rdf:Description>
```

```
<!-- **
     ** In the following property definitions, the defined types
     ** are as follows:
     **
     **    Number:     A positive integer
     **                  [0-9]+
     **    Boolean:    A yes or no value
     **                  Yes|No
     **    Literal:    An alphanumeric string
     **                  [A-Za-z0-9/.\-_]+
     **    Dimension:  A pair of numbers
     **                  [0-9]+x[0-9]+
     **
-->


<!-- ********************************************************************* -->
<!-- ***** Component: HardwarePlatform ***** -->

<rdf:Description ID="BitsPerPixel">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  The number of bits of color or grayscale information per
                    pixel, related to the number of colors or shades of gray
                    the device can display.
      Type:         Number
      Resolution:   Override
      Examples:     "2", "8"
</rdfs:comment>
</rdf:Description>

<rdf:Description ID="ColorCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates whether the device's display supports color.
                    "Yes" means color is supported. "No" means the display
                    supports only grayscale or black and white.
      Type:         Boolean
      Resolution:   Override
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="CPU">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Name and model number of the device CPU.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Pentium III", "PowerPC 750"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="ImageCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates whether the device supports the display of
                    images. If the value is "Yes", the property CcppAccept
                    may list the types of images supported.
      Type:         Boolean
      Resolution:   Locked
```

```
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="InputCharSet">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  List of character sets supported by the device for text
                    entry. Property's value is a list of character sets,
                    where each item in the list is a character set name, as
                    registered with IANA.
      Type:         Literal
      Resolution:   Locked
      Examples:     "US-ASCII", "ISO-8859-1", "Shift_JIS"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Keyboard">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Type of keyboard supported by the device, as an indicator
                    of ease of text entry.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Disambiguating", "Qwerty", "PhoneKeypad"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="MaxScreenChar">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Size of the virtual page onto which a document is
                    rendered, in units of characters. Property value is
                    composed of the screen width and screen height. The
                    device's standard font should be used to determine this
                    property's value. This property may not apply to all
                    devices.
      Type:         Dimension
      Resolution:   Locked
      Examples:     "16x80", "48x32"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Model">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Model number assigned to the terminal device by the
                    vendor or manufacturer.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Mustang GT", "Q30"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="OutputCharSet">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  List of character sets supported by the device for
```

```
                    output to the display. Property value is a list of
                    character sets, where each item in the list is a
                    character set name, as registered with IANA. List
                    items are separated by white space.
      Type:         Literal
      Resolution:   Append
      Examples:     "US-ASCII", "ISO-8859-1", "Shift_JIS"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="PointingResolution">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Type of resolution of the pointing accessory supported
                    by the device.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Character", "Line", "Pixel"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="ScreenSize">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  The size of the device's screen in units of pixels,
                    composed of the screen width and the screen height.
      Type:         Dimension
      Resolution:   Locked
      Examples:     "160x160", "640x480"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="ScreenSizeChar">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Size of the device's screen in units of characters,
                    composed of the screen width and screen height. The
                    device's standard font should be used to determine
                    this property's value.
      Type:         Dimension
      Resolution:   Locked
      Examples:     "12x4", "16x8"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="SoftKeysCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates whether the device supports programmable soft
                    keys. A soft key is a physical key whose label and
                    function can change programmatically.
      Type:         Boolean
      Resolution:   Locked
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="SoundOutputCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform "/>
   <rdfs:comment>
```

```
      Description:  Indicates whether the device supports sound output
                    through an external speaker, headphone jack, or other
                    sound output mechanism.
      Type:         Boolean
      Resolution:   Locked
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="TextInputCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates whether the device supports alpha-numeric text
                    entry. "Yes" means the device supports entry of both
                    letters and digits. "No" means the device supports only
                    entry of digits.
      Type:         Boolean
      Resolution:   Locked
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Vendor">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Name of the vendor manufacturing the terminal device.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Ford", "Lexus"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="VoiceInputCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#HardwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates whether the device supports any form of voice
                    input, including speech recognition. This includes voice-
                    enabled browsers.
      Type:         Boolean
      Resolution:   Locked
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>



<!-- ************************************************************** -->
<!-- ***** Component: SoftwarePlatform ***** -->

<rdf:Description ID="AcceptDownloadableSoftware">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  Indicates the user's preference on whether to accept
                    downloadable software.
      Type:         Boolean
      Resolution:   Locked
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="AudioInputEncoder">
```

```
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
      <rdfs:domain rdf:resource="#SoftwarePlatform"/>
      <rdfs:comment>
         Description:  List of audio input encoders supported by the device.
         Type:         Literal
         Resolution:   Append
         Example:      "G.711", "G.931"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="DownloadableSoftwareSupport">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
      <rdfs:domain rdf:resource="#SoftwarePlatform"/>
      <rdfs:comment>
         Description:  List of executable content types which the device
                       supports and which it is willing to accept from the
                       network. The property value is a list of MIME types,
                       where each item in the list is a content type
                       descriptor as specified by RFC 2045. Items in the
                       list are separated by white space.
         Type:         Literal
         Resolution:   Locked
         Examples:     "application/x-msdos-exe"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="JVMVersion">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
      <rdfs:domain rdf:resource="#SoftwarePlatform"/>
      <rdfs:comment>
         Description:  List of the Java virtual machines installed on the
                       device. Each item in the list is a name token describing
                       the vendor and version of the VM.
         Type:         Literal
         Resolution:   Append
         Examples:     "SunJRE1.2", "MSJVM1.0"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="MexeClassmark">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdfs:domain rdf:resource="#SoftwarePlatform"/>
      <rdfs:comment>
         Description:  ETSI MExE classmark. Value "1" means the MExE device
                       supports WAP. Value "2" means MExE device supports
                       WAP and Java. All other values should be considered
                       reserved for use by MExE.
         Type:         Number
         Resolution:   Locked
         Examples:     "1", "2"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="MexeSpec">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdfs:domain rdf:resource="#SoftwarePlatform"/>
      <rdfs:comment>
         Description:  Class mark specialization. Refers to the first two
                       digits of the version of the MExE Stage 2 spec.
         Type:         Literal
         Resolution:   Locked
         Examples:     "7.02"
```

```
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="OSName">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  Name of the device's operating system.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Mac OS", "Windows NT"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="OSVendor">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  Vendor of the device's operating system.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Apple", "Microsoft"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="OSVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  Version of the device's operating system.
      Type:         Literal
      Resolution:   Locked
      Examples:     "6.0", "4.5"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="RecipientAppAgent">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  User agent associated with the current request. Value
                    should match the name of one of the components in the
                    profile. A component name is specified by the ID
                    attribute on the prf:Component element containing the
                    properties of that component.
      Type:         Literal
      Resolution:   Locked
      Examples:     "SpeedyMail"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID=”SoftwareNumber”>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource=”#SoftwarePlatform”/>
   <rdfs:comment>
      Description:  Version of the device-specific software (firmware) to
                    which the device's low-level software conforms.
      Type:         Literal
      Resolution:   Locked
      Examples:     "2"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="VideoInputEncoder">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
```

```
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  List of video input encoders supported by the device.
      Type:         Literal
      Resolution:   Append
      Examples:     "MPEG-1", "MPEG-2", "H.261"
   </rdfs:comment>
</rdf:Description>



<!-- ************************************************************** -->
<!-- ***** Component: NetworkCharacteristics ***** -->

<rdf:Description ID="CurrentBearerService">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  The bearer on which the current session was opened.
      Type:         Literal
      Resolution:   Locked
      Examples:     "OneWaySMS", "GUTS", "TwoWayPacket"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="SecuritySupport">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  Type of security or encryption mechanism supported.
      Type:         Literal
      Resolution:   Locked
      Example:      "PPTP"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID=”SupportedBearers”>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  List of bearers supported by the device.
      Type:         Literal
      Resolution:   Locked
      Examples:     "GPRS", "GUTS", "TwowaySMS", CSD", "USSD"
   </rdfs:comment>
</rdf:description>



<!-- ************************************************************** -->
<!-- ***** Component: BrowserUA ***** -->

<rdf:Description ID="BrowserName">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  Name of the browser user agent associated with the
                    current request.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Mozilla", "MSIE"
   </rdfs:comment>
</rdf:Description>
```

```
<rdf:Description ID="BrowserVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  Version of the browser.
      Type:         Literal
      Resolution:   Locked
      Examples:     "1.0"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="CcppAccept">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  List of content types the device supports. Property
                    value is a list of MIME types, where each item in the
                    list is a content type descriptor as specified by
                    RFC 2045.
      Type:         Literal
      Resolution:   Append
      Examples:     "text/html", "text/plain", "text/html", "image/gif"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="CcppAccept-Charset">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  List of character sets the device supports. Property
                    value is a list of character sets, where each item in
                    the list is a character set name registered with IANA.
      Type:         Literal
      Resolution:   Append
      Examples:     "US-ASCII", "ISO-8859-1", "Shift_JIS"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="CcppAccept-Encoding">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  List of transfer encodings the device supports.
                    Property value is a list of transfer encodings, where
                    each item in the list is a transfer encoding name as
                    specified by RFC 2045 and registered with IANA.
      Type:         Literal
      Resolution:   Append
      Examples:     "base64", "quoted-printable"
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="CcppAccept-Language">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Seq"/>
   <rdfs:domain rdf:resource="#SoftwarePlatform"/>
   <rdfs:comment>
      Description:  List of preferred document languages. If a resource is
                    available in more than one natural language, the server
                    can use this property to determine which version of the
                    resource to send to the device. The first item in the
```

```
                    list should be considered the user's first choice, the
                    second the second choice, and so on. Property value is
                    a list of natural languages, where each item in the list
                    is the name of a language as defined by RFC 1766.
      Type:         Literal
      Resolution:   Append
      Examples:     "zh-CN", "en", "fr"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="DownloadableBrowserApps">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  List of executable content types which the browser
                    supports and which it is willing to accept from the
                    network. The property value is a list of MIME types,
                    where each item in the list is a content type
                    descriptor as specified by RFC 2045.
      Type:         Literal
      Resolution:   Append
      Examples:     "application/java-applet" "application/javascript"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="FramesCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  Indicates whether the browser is capable of displaying
                    HTML frames.
      Type:         Boolean
      Resolution:   Override
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="HtmlVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  Version of HyperText Markup Language (HTML) supported
                    by the browser.
      Type:         Literal
      Resolution:   Locked
      Examples:     "2.0", "3.2", "4.0"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="JavaScriptVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
   <rdfs:comment>
      Description:  Version of the JavaScript language supported by the
                    browser.
      Type:         Literal
      Resolution:   Locked
      Examples:     "1.4"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="PreferenceForFrames">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#BrowserUA"/>
```

```
      <rdfs:comment>
         Description:  Indicates the user's preference for receiving HTML
                       content that contains frames.
         Type:         Boolean
         Resolution:   Locked
         Examples:     "Yes", "No"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="TablesCapable">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdfs:domain rdf:resource="#BrowserUA"/>
      <rdfs:comment>
         Description:  Indicates whether the browser is capable of displaying
                       HTML tables.
         Type:         Boolean
         Resolution:   Locked
         Examples:     "Yes", "No"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="XhtmlVersion">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdfs:domain rdf:resource="#BrowserUA"/>
      <rdfs:comment>
         Description:  Version of XHTML supported by the browser.
         Type:         Literal
         Resolution:   Locked
         Examples:     "1.0"
      </rdfs:comment>
   </rdf:Description>

   <rdf:Description ID="XhtmlModules">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
      <rdfs:domain rdf:resource="#BrowserUA"/>
      <rdfs:comment>
         Description:  List of XHTML modules supported by the browser. Property
                       value is a list of module names, where each item in the
                       list is the name of an XHTML module as defined by the
                       W3C document "Modularization of XHTML", Section 4. Note that
                       the referenced document is a work in progress.  Any subsequent
                       changes to the module naming conventions should be
                       reflected in the values of this property.
         Type:         Literal
         Resolution:   Append
         Examples:     "XHTML1-struct", "XHTML1-blkstruct", "XHTML1-frames"
      </rdfs:comment>
   </rdf:Description>



   <!-- ************************************************************** -->
   <!-- ***** Component: WapCharacteristics ***** -->

   <rdf:Description ID="WapDeviceClass">
      <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
      <rdfs:domain rdf:resource="#WapCharacteristics"/>
      <rdfs:comment>
         Description:  Classification of the device based on capabilities as
                       identified in the WAP 1.1 specifications. Current
                       values are "A", "B" and "C".
         Type:         Literal
         Resolution:   Locked
         Examples:     "A"
```

```
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WapPushMsgPriority">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  User's preference on the priority of incoming push
                    messages.
      Type:         Literal
      Resolution:   Locked
      Examples:     "critical", "low", "none", "all"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WapPushMsgSize">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  Maximum size of a push message that the device can
                    handle. Value is number of bytes.
      Type:         Number
      Resolution:   Locked
      Examples:     "1024", "1400"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WapVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  Version of WAP supported.
      Type:         Literal
      Resolution:   Locked
      Examples:     "1.1", "1.2", "2.0"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WmlDeckSize">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  Maximum size of a WML deck that can be downloaded to
                    the device. This may be an estimate of the maximum size
                    if the true maximum size is not known. Value is number
                    of bytes.
      Type:         Number
      Resolution:   Locked
      Examples:     "4096"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WmlScriptLibraries">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  List of mandatory and optional libraries supported in
                    the device's WMLScript VM.
      Type:         Literal
      Resolution:   Locked
      Examples:     "Lang", "Float", "String", "URL", "WMLBrowser", "Dialogs"
   </rdfs:comment>
</rdf:Description>
```

```
<rdf:Description ID="WmlScriptVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource=" http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  List of WMLScript version numbers supported by the device.
      Type:         Literal
      Resolution:   Append
      Examples:     "1.1", "1.0"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WmlVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  List of WML language version numbers supported by the device.
      Type:         Literal
      Resolution:   Append
      Examples:     "1.1", "1.0"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WtaiLibraries">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdf:type rdf:resource=" http://www.w3.org/TR/PR-rdf-schema#Bag"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  List of WTAI network common and network specific
                    libraries supported by the device that are URI accessible.
                    Property value is a list of WTA library names, where each
                    item in the list is a library name as specified by
                    "WAP WTAI" and its addendums. Any future addendums to "WAP WTAI"
                    should be reflected in the values of this property.
      Type:         Literal
      Resolution:   Locked
      Examples:     "WTAVoiceCall", "WTANetText", "WTAPhoneBook",
                    "WTACallLog", "WTAMisc", "WTAGSM", "WTAIS136", "WTAPDC"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="WtaVersion">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#WapCharacteristics"/>
   <rdfs:comment>
      Description:  Version of WTA user agent.
      Type:         Literal
      Resolution:   Locked
      Examples:     "1.1"
   </rdfs:comment>
</rdf:Description>



</rdf:RDF>
```

7.6 Profile Example in RDF

This section is intended to help the reader better understand the implementation and use of the specified schema and vocabulary for User Agent Profiles. The reader is expected to be familiar with RDF specifications [RDF], [RDF-Schema].

Consider a fictitious mobile device having most of the capabilities identified in the vocabulary. A sample RDF graph for the profile is shown in Figures 7.1, 7.2, 7.3, 7.4, and 7.5. Note that these are directly translatable to RDF triples, each describing resource, attribute, and value**.**

These figures are followed by an XML implementation of the profile for this fictitious device. This profile conforms to the schema described in Section 7.4. When compiled using an RDF parser (such as SiRPAC [SiRPAC]), it generates the necessary tuples corresponding to the graph. Note that the profile does not have to contain all attributes specified in the vocabulary.

In this example, the indirect referencing scheme has been used for default resources located at the vendor site. The long serialized syntax has been used only to improve readability, with the expectation that machine readable RDF parsers will also be able to interpret and understand the compact syntax. See Appendix 2 for an example comparison of the serialized syntax versus the abbreviated syntax for RDF documents.

Note: The enclosed XML implementation has been verified with RDF parser SiRPAC.

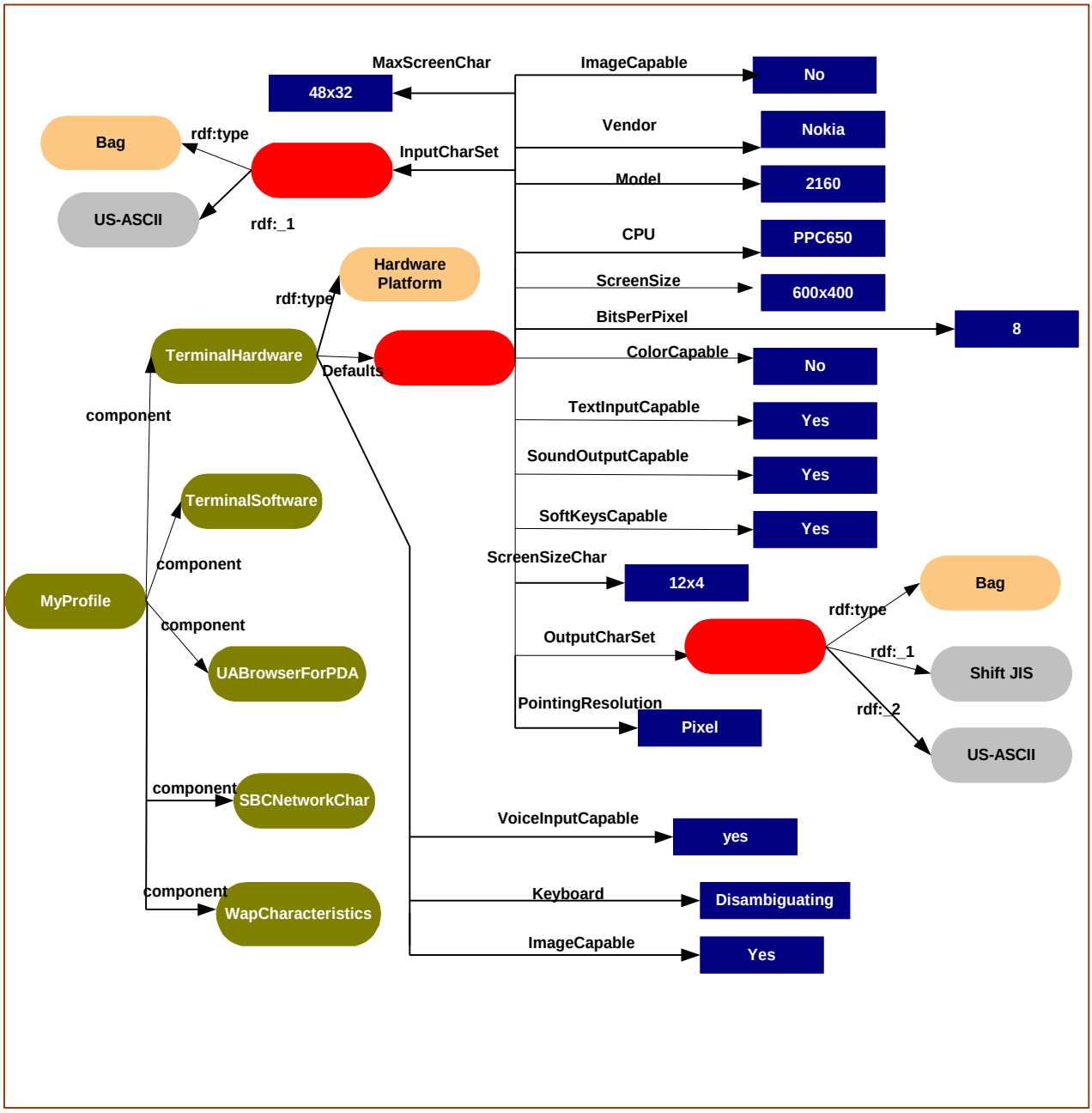


Figure 7.1: Root graph for Profile and Sub-graph for Hardware characteristics

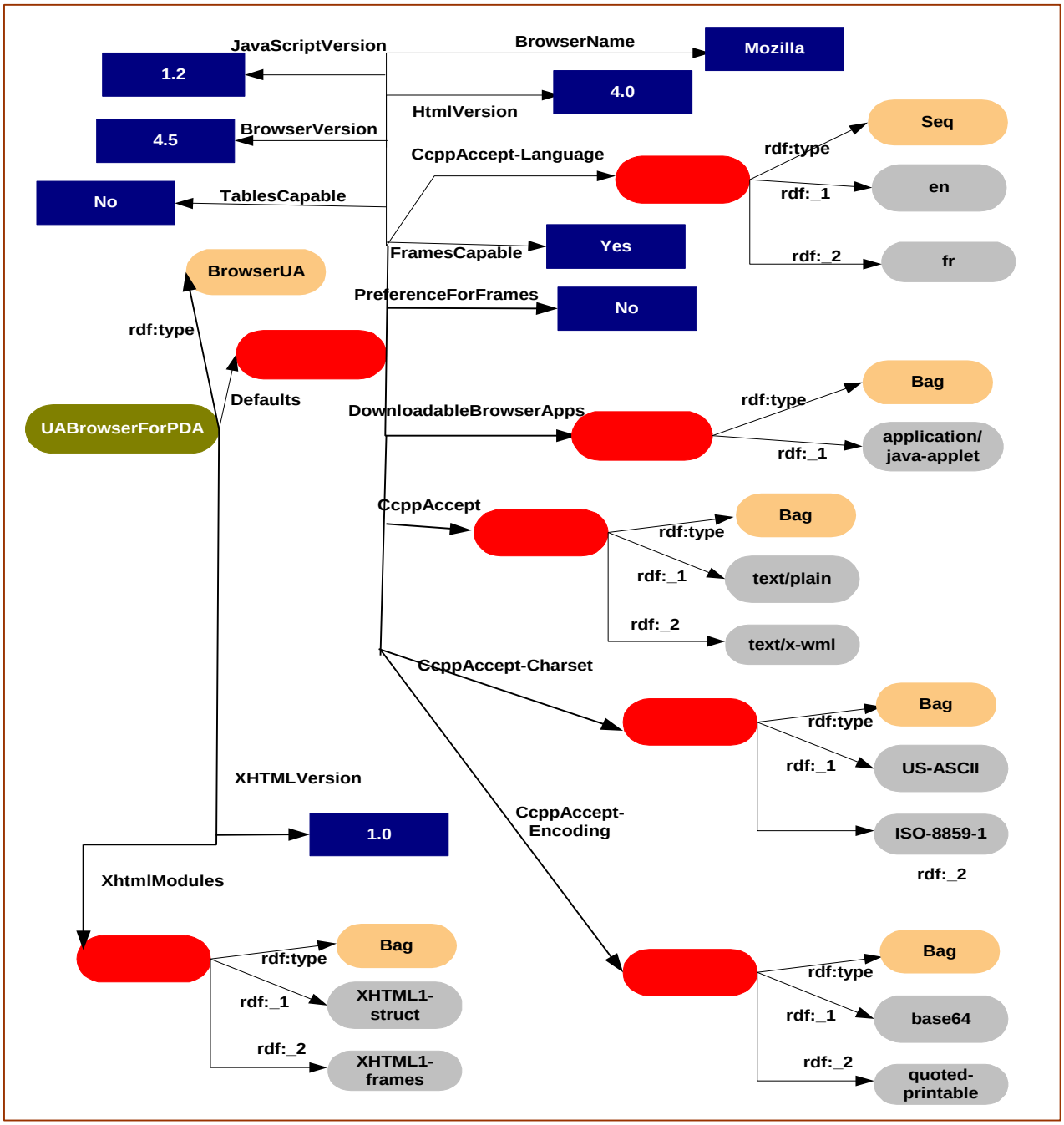


Figure 7.2: Sub-graph for User Agent - Browser

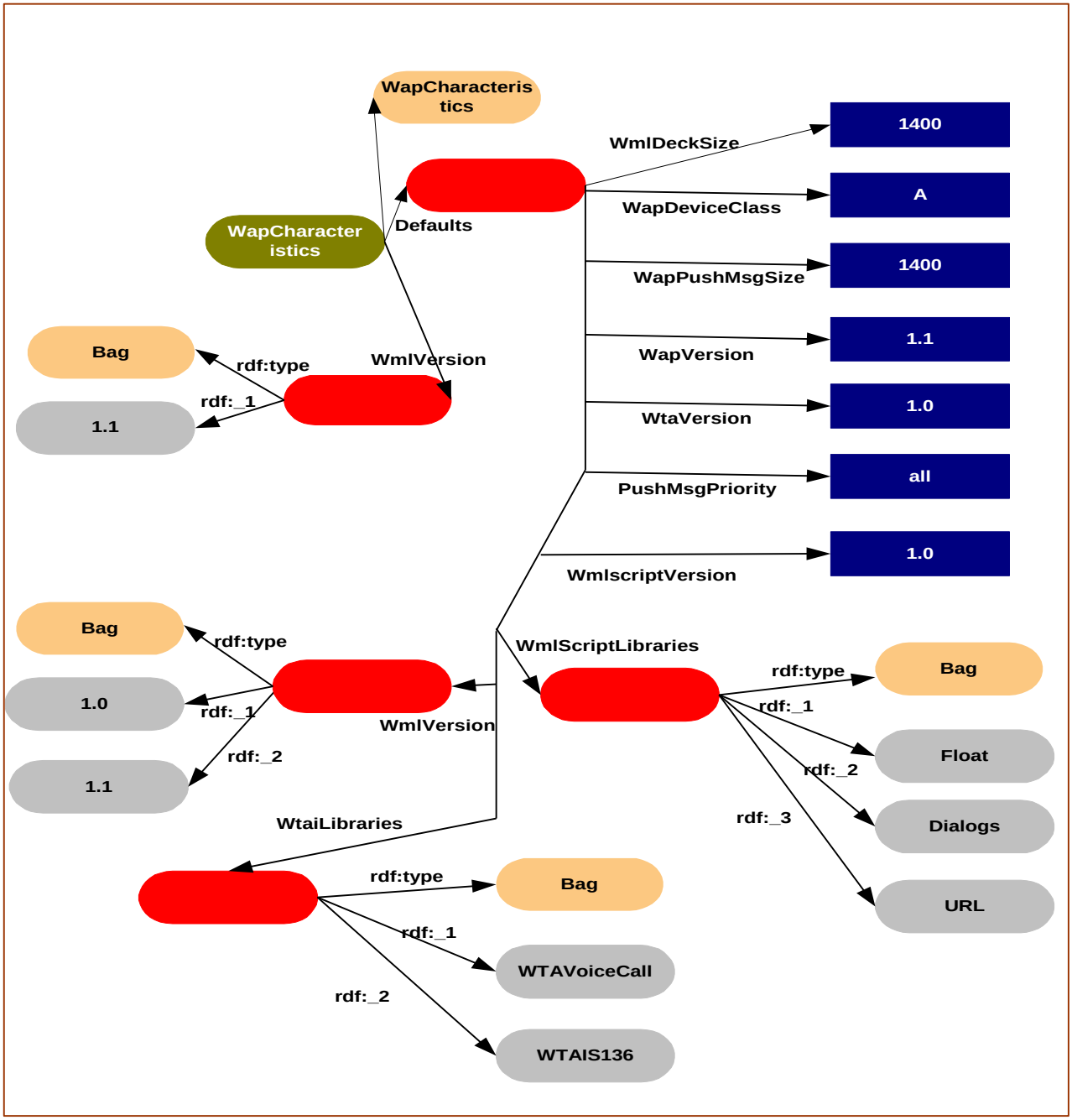


Figure 7.3: Sub-graph for WAP Characteristics

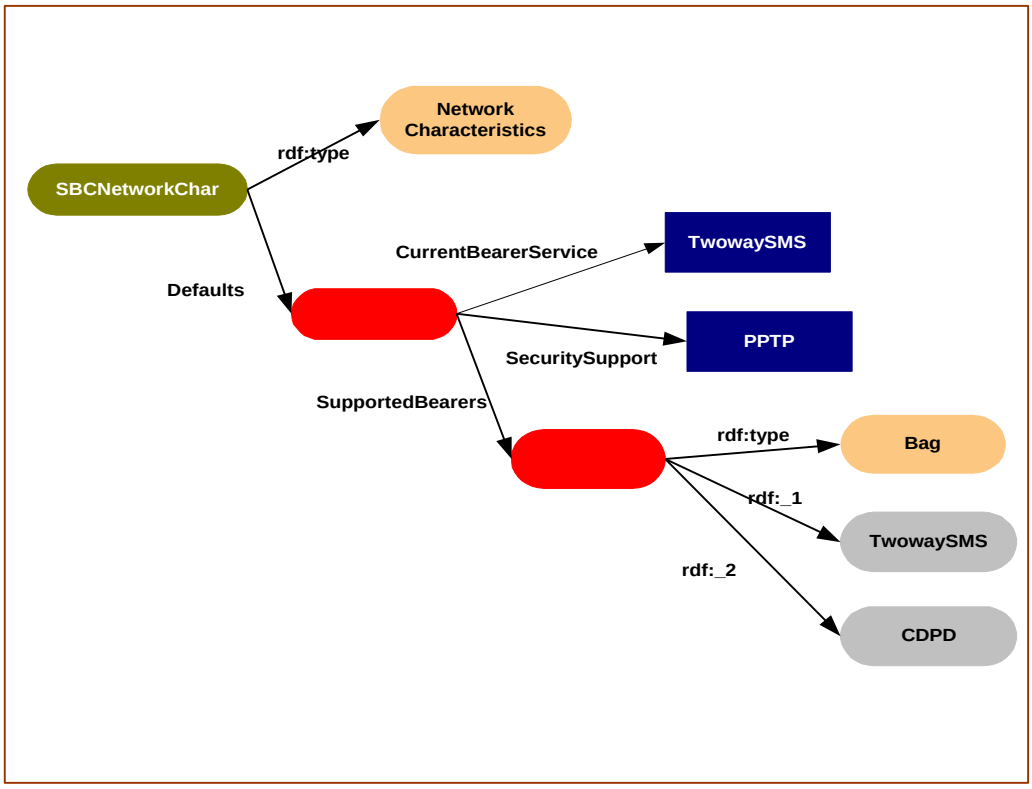


Figure 7.4: Sub-graph for Network Characteristics

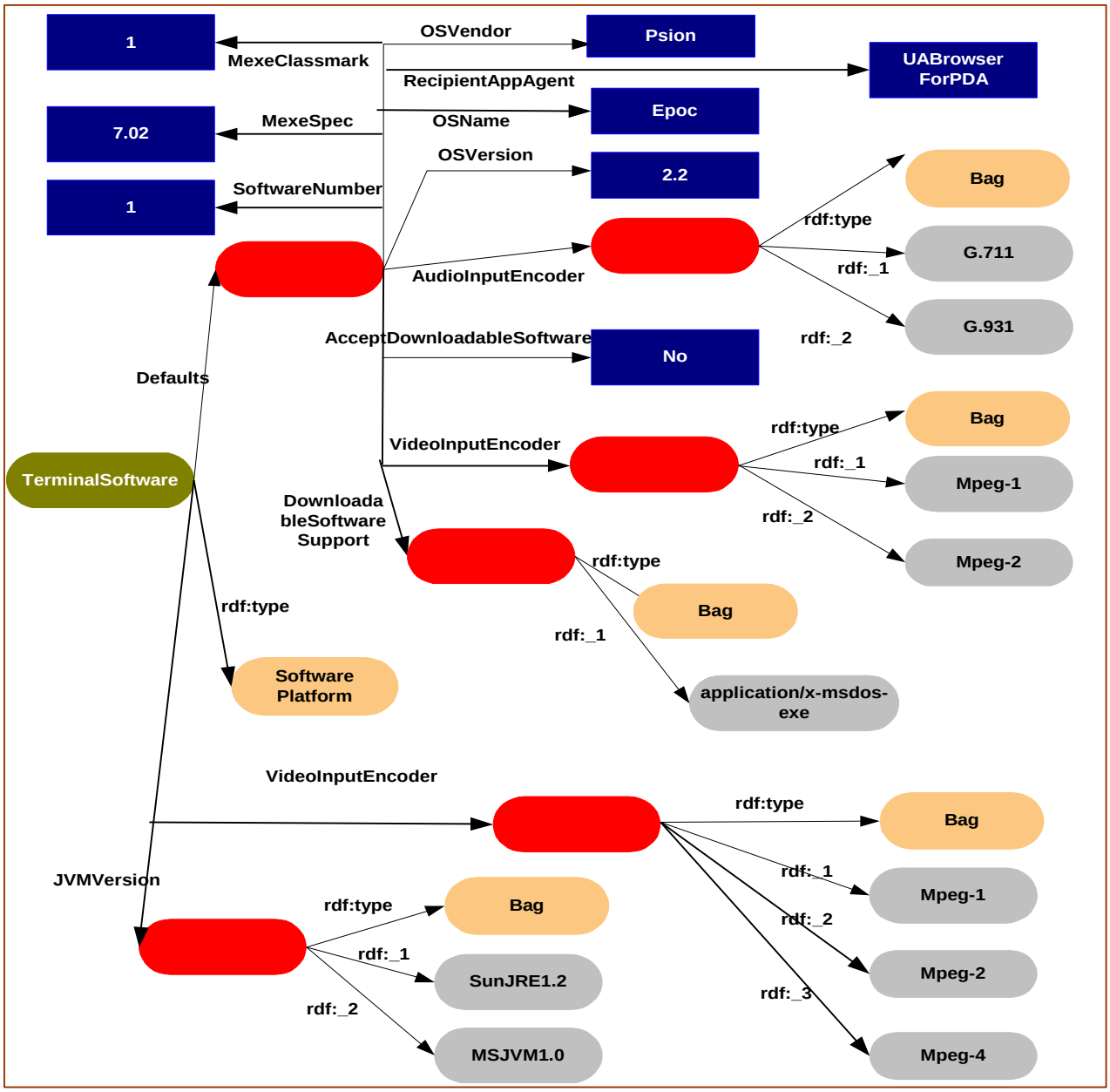


Figure 7.5: Software Characteristics

XML encoded profile that maps the graph for the fictitious device (verified with SiRPAC[8]):

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014991014#">

<rdf:Description ID="MyProfile">
```

```
<prf:component>
   <rdf:Description ID="TerminalHardware">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#HardwarePlatform" />
      <prf:Defaults rdf:resource="http://www.nokia.com/profiles/2160" />
         <!-- override the ImageCapable property, and add VoiceInputCapable
              and Keyboard properties  -->
      <prf:Imagecapable>Yes</prf:Imagecapable>
      <prf:Keyboard>Disambiguating</prf:Keyboard>
      <prf:VoiceInputCapable>Yes</prf:VoiceInputCapable>
   </rdf:Description>
</prf:component>

<prf:component>
   <rdf:Description ID="TerminalSoftware">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#SoftwarePlatform" />
      <prf:Defaults rdf:resource="http://www.symbian.com/profiles/pda/epoc" />
         <!--Override VideoInputEncoder property and add JVMVersion property -->
      <prf:JVMVersion>
         <rdf:Bag>
            <rdf:_l1>SunJRE1.2</rdf:_l1>
            <rdf:_l2>MSJVM1.0</rdf:_l2>
         </rdf:Bag>
      </prf:JVMVersion>
      <prf:VideoInputEncoder>
        <rdf:Bag>
            <rdf:_l1>Mpeg-1</rdf:_l1>
            <rdf:_l2>Mpeg-2</rdf:_l2>
            <rdf:_l3>Mpeg-4</rdf:_l3>
         </rdf:Bag>
      </prf:VideoInputEncoder>
   </rdf:Description>
</prf:component>

<prf:component>
   <rdf:Description ID="UABrowserForPDA">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#BrowserUA" />
      <prf:Defaults rdf:resource ="http://www.netscape.com/Navigator/4.5/PDA" />
         <!-- Add property regarding XHTML version and XHTML modules -->
      <prf:XHTMLVersion>1.0</prf:XHTMLVersion>
      <prf:XhtmlModules>
         <rdf:Bag>
            <rdf:_l1>XHTML1-tables</rdf:_l1>
            <rdf:_l2>XHTML1-frames</rdf:_l2>
         </rdf:Bag>
      </prf:XhtmlModules>
   </rdf:Description>
</prf:component>

<prf:component>
   <rdf:Description ID="SBCNetworkChar">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#NetworkCharacteristics" />
      <prf:Defaults rdf:resource="http://www.sbcwireless.com/texas/profiles/sms-
service"/>
         <!-- no overrides. -->
   </rdf:Description>
</prf:component>

<prf:component>
   <rdf:Description ID="WapCharacteristics">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#WAPCharacteristics" />
```

```
      <prf:Defaults rdf:resource="http://www.phone.com/PDA/WAP1.1" />
         <!--override WmlVersion property; no addition of new property descriptions -->
      <prf:WmlVersion>
         <rdf:Bag>
            <rdf:li>1.0</rdf:li>
         </rdf:Bag>
      </prf:WmlVersion>
   </rdf:Description>
</prf:component>



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

Hardware Platform: Defaults at http://www.nokia.com/profiles/2160

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
         <!--hardware vendor site: Default description of properties -->
   <rdf:Description>
      <prf:Vendor>Nokia</prf:Vendor>
      <prf:Model>2160</prf:Model>
      <prf:CPU>PPC650</prf:CPU>
      <prf:TextInputCapable>Yes</prf:TextInputCapable>
      <prf:ImageCapable>No</prf:ImageCapable>
      <prf:SoftKeysCapable>Yes</prf:SoftKeysCapable>
      <prf:SoundOutputCapable>Yes</prf:SoundOutputCapable>
      <prf:PointingResolution>Pixel</prf:PointingResolution>
      <prf:ColorCapable>No</prf:ColorCapable>
      <prf:ScreenSize>600x400</prf:ScreenSize>
      <prf:ScreenSizeChar>12x4</prf:ScreenSizeChar>
      <prf:MaxScreenChar>48x32</prf:MaxScreenChar>
      <prf:InputCharSet>
         <rdf:Bag>
            <rdf:li>US-ASCII</rdf:li>
         </rdf:Bag>
      </prf:InputCharSet>
      <prf:BitsPerPixel>8</prf:BitsPerPixel>
      <prf:OutputCharSet>
         <rdf:Bag>
            <rdf:li>US-ASCII</rdf:li>
            <rdf:li>Shift_JIS</rdf:li>
         </rdf:Bag>
      </prf:OutputCharSet>
   </rdf:Description>
</rdf:RDF>
```

SoftwarePlatform Default properties at http://www.symbian.com/profiles/pda/epoc

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
         <!--software vendor site: Default description of software properties -->
   <rdf:Description>
      <prf:OSVendor>Psion</prf:OSVendor>
      <prf:OSName>Epoc</prf:OSName>
      <prf:OSVersion>2.2</prf:OSVersion>
      <prf:AcceptDownloadableSoftware>No</prf:AcceptDownloadableSoftware>
      <prf:RecipientAgent>UABrowserForPDA</prf:RecipientAgent>
      <prf:MexeClassmark>1</prf:MexeClassmark>
      <prf:MexeSpec>7.02</prf:MexeSpec>
```

```
      <prf:SoftwareNumber>1</prf:SoftwareNumber>
      <prf:AudioInputEncoder>
         <rdf:Bag>
            <rdf:li>G.711</rdf:li>
            <rdf:li>G.931</rdf:li>
         </rdf:Bag>
      </prf:AudioInputEncoder>
      <prf:DownloadableSoftwareSupport>
         <rdf:Bag>
            <rdf:li>application/x-msdos-exe</rdf:li>
         </rdf:Bag>
      </prf:DownloadableSoftwareSupport>
      <prf:VideoInputEncoder>
         <rdf:Bag>
            <rdf:li>Mpeg-1</rdf:li>
            <rdf:li>Mpeg-2</rdf:li>
         </rdf:Bag>
      </prf:VideoInputEncoder>
   </rdf:Description>
</rdf:RDF>
```

Browser User Agent Default Properties at http://www.netscape.com/Navigator/4.5/PDA

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
         <!-- browser vendor site: Default description of properties -->
   <rdf:Description>
      <prf:BrowserName>Mozilla</prf:BrowserName>
      <prf:BrowserVersion>4.5</prf:BrowserVersion>
      <prf:HtmlVersion>4.0</prf:HtmlVersion>
      <prf:JavaScriptVersion>1.2</prf:JavaScriptVersion>
      <prf:FramesCapable>Yes</prf:FramesCapable>
      <prf:PreferenceForFrames>Yes</prf: PreferenceForFrames>
      <prf:TablesCapable>No</prf:TablesCapable>
      <prf:DownloadableBrowserApps>
         <rdf:Bag>
            <rdf:li>application/java-applet</rdf:li>
         </rdf:Bag>
      </prf:DownloadableBrowserApps>
      <prf:CcppAccept>
         <rdf:Bag>
            <rdf:li>text/plain</rdf:li>
            <rdf:li>text/x-wml</rdf:li>
         </rdf:Bag>
      </prf:CcppAccept>
      <prf:CcppAccept-Language>
         <rdf:Seq>
            <rdf:li>en</rdf:li>
            <rdf:li>fr</rdf:li>
         </rdf:Seq>
      </prf:CcppAccept-Language>
      <prf:CcppAccept-Encoding>
         <rdf:Bag>
            <rdf:li>base64</rdf:li>
            <rdf:li>quoted-printable</rdf:li>
         </rdf:Bag>
      </prf:CcppAccept-Encoding>
      <prf:CcppAccept-Charset>
         <rdf:Bag>
            <rdf:li>US-ASCII </rdf:li>
            <rdf:li>ISO-8859-1</rdf:li>
         </rdf:Bag>
      </prf:CcppAccept-Charset>
```

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

Network characteristics located at Carrier web site: http://www.sbcwireless.com/texas/profiles/sms-service

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
         <!-- carrier site: Default description of Network related properties
              for SMS subscribers -->
   <rdf:Description>
      <prf:CurrentBearerService>SMS</prf:CurrentBearerService>
      <prf:SecuritySupport>PPTP</prf:SecuritySupport>
      <prf:SupportedBearers>
         <rdf:Bag>
            <rdf:li>SMS</rdf:li>
            <rdf:li>CDPD</rdf:li>
         </rdf:Bag>
      </prf:SupportedBearers>
   </rdf:Description>
</rdf:RDF>
```

WAP Default properties at http://www.phone.com/PDA/WAP1.1

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
         <!--WAP Browser vendor site: Default description of WAP properties -->
   <rdf:Description>
      <prf:WapVersion>1.1</prf:WapVersion>
      <prf:WmlDeckSize>1400</prf:WmlDeckSize>
      <prf:WapDeviceClass>A</prf:WapDeviceClass>
      <prf:WapPushMsgSize>1400 octets</prf:WapPushMsgSize>
      <prf:WapPushMsgPriority>all</prf:WapPushMsgPriority>
      <prf:WtaVersion>1.0</prf:WtaVersion>
      <prf:WmlScriptVersion>1.1</prf:WmlScriptVersion>
      <prf:WmlscriptLibraries>
         <rdf:Bag>
            <rdf:li>Float</rdf:li>
            <rdf:li>Dialogs</rdf:li>
            <rdf:li>URL</rdf:li>
         </rdf:Bag>
      </prf:WmlscriptLibraries>
      <prf:WtaiLibraries>
         <rdf:Bag>
            <rdf:li>WTAVoiceCall</rdf:li>
            <rdf:li>WTAIS136</rdf:li>
         </rdf:Bag>
      </prf:WtaiLibraries>
      <prf:WmlVersion>
         <rdf:Bag>
            <rdf:li>1.0</rdf:li>
            <rdf:li>1.1</rdf:li>
         </rdf:Bag>
      </prf:WmlVersion>
   </rdf:Description>
</rdf:RDF>
```

7.7 Extensions to the Schema/Vocabulary

While this specification provides a base vocabulary of property descriptions for User Agent Profiles, it is anticipated that implementers of new applications or device capabilities may, in the future, have a need for expressing new or additional attributes in the profile. The use of RDF enables an extensibility mechanism for CC/PP-based schemas that addresses the evolution of new types of devices, applications, or hardware/software. The vocabulary extensions will constitute a different RDF schema and will have a corresponding RDF model and syntactic representation. A particular profile may compose attributes from multiple schemas, namely the base vocabulary schema and the vocabulary extension schema.

The new schema/vocabulary MUST adhere to the following guidelines and recommendations for compliance with this specification and the CC/PP framework:

- New schemas and vocabularies MUST be uniquely identified using well-defined XML namespaces.
- Since RDF supports interoperability of schemas, the new schemas MUST NOT contain the same attributes (name and semantics) as specified in the base vocabulary. *Note that profiles can be composed from attributes from multiple schemas*. Moreover, extended attributes SHOULD NOT use attribute names that are reserved for future use by the base vocabulary, as listed in Appendix A.5.
- The schema vocabulary designer SHOULD have a mental model of the RDF document as an RDF graph, and should take care to express and verify the intuitions against common RDF tools. *This will eliminate potential ambiguity regarding the schema and its semantics, and interpretation by tools.*
- To reduce the verbosity of the profile and therefore the size of the HTTP headers, the CC/PP document SHOULD be expressed using the abbreviated syntax.
- Control characters and binary bytes MUST be encoded in conformance with XML syntax specifications.
- In addition to the naming conventions specified in Appendix C of the RDF Specification [RDF], the following naming conventions MUST be adhered to while defining new profile attributes and components:

  All components and attributes MUST start with an upper case letter.

  Multiple word names MUST be described as one word, with the first letter in the second word in upper case. There MUST NOT be any separator or underscore between the words. For example, Software Platform must be identified as *SoftwarePlatform.*

  Boolean values MUST be described as "Yes" and "No".

- Cyclic references (URIs) MUST NOT be used during schema design.

7.7.1 Addition of Components

In defining components within the new schema, the designer MUST apply the following rules:

- The components used in a standardized base vocabulary MUST not only contain enough information to meet the needs of a majority of current content providers but also allow for the meaningful introduction of future device capabilities into the profile.
- As long as the new attributes fall within the realm of one of the base profile components (*HardwarePlatform*, *SoftwarePlatform*, *NetworkCharacteristics*, *WAPCharacteristics*, and *BrowserUA*), the designers of new schemas must add those attributes to these defined components (instead of creating new components).
- New applications or user agents may need to assert their capabilities and preferences in a new component. The schema designer MUST uniquely define the attributes to be included in the vocabulary for the component. Attribute definition includes identifying the semantic description, attribute type (resource/ literal), the type of resolution rule and sample values.

- The schema for the new components MUST follow the general schema layout for the core profile components. Therefore, the nested description for each component MUST incorporate the following structure:

  A subordinate description block to identify default attributes if any, preferably referenced by a resource URI

  Any overrides/modifications to the default descriptions outside the Default subordinate block

7.7.2 Addition of Attributes

In defining attributes within the new schema, the designer MUST apply the following rules:

- The attributes described in the vocabulary MUST be atomic and semantically unambiguous. The names used to define/represent the attributes MUST be unique within the namespace.
- To preserve the simplicity of the schema, the use of complex data types such as containers as well as schema validation conditions such as value ranges, constraints and units SHOULD be minimized.
- A value of an attribute SHOULD be expressed as a string of characters (literal) or an RDF resource. For values that are complex data types (such as lists), RDF containers MUST be used. The **rdf:resource** construct MUST be used, where appropriate, to indicate to the RDF parser that the value of an attribute is indeed a resource and not a literal.
- The rule for resolving multiple descriptions of an attribute MUST be specified as part of the semantics. The rule must specify a *Locked*, *Override*, or *Append* treatment for value resolution. If no rule is specified, a default rule of *Override* is assumed for the attribute.
- Escape control characters and binary bytes MUST be in conformance with XML syntax [XML].

8 Binary Encoding of User Agent Profiles

This section is normative. Examples are not normative.

A User Agent Profile that is constructed in accordance with the CC/PP syntax [CC/PP] and the schema of Section 7 MAY be encoded using a compact binary representation. This compact representation is based upon the WAP Binary XML (WBXML) Content Format [WBXML]. This section defines how to use WBXML 1.1 to encode a Profile document.

8.1 Token Description

8.1.1 Global Extension Tokens

This specification does not require the use of the [WBXML] global extension tokens.

8.1.2 Tag Tokens

Section 7 defines properties for conformant User Agent Profiles, which are structured according to the CC/PP note [CC/PP], and correspondingly use RDF XML serialization syntax. This section defines single-byte tokens corresponding to the elements of the RDF serialization syntax. These tokens are distributed among three code pages. Code page zero (0) defines tokens for RDF serialization elements. Code page one (1) defines tokens for properties in the core profile schema. Code page two (2) defines tokens for properties in the Browser user-agent component of the schema.

User agents or applications other than the browser may define additional tag code pages for their own properties.

8.1.3 Attribute Tokens

Section 7 defines properties for conformant User Agent Profiles, which are structured according to the CC/PP note [CC/PP], and correspondingly use RDF XML serialization syntax. This section defines a set of single-byte tokens corresponding to the attribute names and values of the RDF serialization syntax. These tokens are distributed among three code pages. Code page zero (0) defines tokens for RDF serialization attributes. Code page one (1) defines tokens for properties in the core profile schema. Code page two (2) defines tokens for properties in the Browser user-agent component of the schema.

User agents or applications other than the browser may define additional attribute code pages for their own properties.

8.1.4 Additional Tokens

Each user agent or application that wants to define properties for use in the user agent profile MUST first define a component to hold its properties. The name of the component MUST be globally unique. Each such user agent component is considered to have a unique namespace, so that, for example, the property BackgroundColor for User Agent A is a property distinct from the property BackgroundColor for User Agent B. See Section 7for more information.

In addition, each user agent or application SHOULD define a series of token table code pages containing the properties from its component. If it chooses to define code pages, then it SHOULD define at least two: one in the "Tag" space and one in the "Attribute" space. The property names SHOULD be inserted into each page. Any well-known values for the properties should be inserted into the "Attribute" page. Additional pages may be required if the component contains a large number of properties.

A default user agent has been defined as part of the schema: the browser. Tables 8.4, 8.7 and 8.10 define the code page, two, for the browser user agent. Table 8.4 defines "Tag" code space code page two, and Tables 8.7 and 8.10 together define "Attribute" code space code page two.

8.2 Encoding Semantics

8.2.1 XML Namespaces

WBXML does not currently support XML namespaces. As User Agent Profiles make use of namespaces, special measures must be taken to encode Profile documents using WBXML. Specifically, the XML namespaces that can be used to construct Profile documents are defined and fixed by this specification. In addition, the prefixes that correspond to these available namespaces are fixed. The namespaces and their prefixes are defined in Table 8.1.

If the WBXML encoder encounters a namespace declaration that matches one of the allowed namespaces, and if its prefix does not match the defined prefix, the encoder MUST convert the document's chosen prefix to the prefix defined for that namespace in Table 8.1.

If the encoder encounters a namespace other than those defined here, it MUST encode all elements from that namespace using literal tags and MUST encode all attribute names for those elements using literal names. The namespace prefix MUST be preserved in the encoding of the tags and attribute names, and all namespace declarations (instances of the attribute xmlns) MUST be preserved.

Namespace	***URI***	***Prefix***
Resource Description Framework	http://www.w3.org/1999/02/22-rdf-syntax-ns#	rdf
Composite Capability/Preferences Profiles	http://www.w3.org/TR/WD-profile-vocabulary-ns#	prf

Table 8.1: Namespace Defined for User Agent Profiles

8.2.2 Document Validation

The process of tokenizing a Profile document MUST verify that the document is well-formed according to [XML]. If it is not well-formed, encoding MUST NOT be performed. An error condition SHOULD be raised in this case.

8.2.3 Decoder Behavior

A decoder processing a Profile document encoded using WBXML MUST NOT consider significant the encoding means applied to a markup construct. It MUST treat a tag or attribute name encoded with a single-byte token and a tag or attribute name encoded using the LITERAL global token as equivalent if the resulting strings are equivalent.

Example: Consider the start-tag `<FavoriteColor>` and the following token table:

Tag Name	***Token***
FavoriteColor	25

Example Tag Table, Code Page 0

According to this table, the start-tag should be encoded as a single-byte tag token, 0x25.

If the above token table entry is not available to the encoder, the start-tag would be encoded using the LITERAL global token (0x04), with an argument that points to the string "FavoriteColor" in the string table.

A decoder processing WBXML-encoded Profile documents must treat the two encodings as equivalent.

8.3 Numeric Constants

8.3.1 Tag Tokens

8.3.1.1 RDF

The following tokens represent tags in code page zero (0). All numbers are in hexadecimal.

Tag Name	***Token***
rdf:RDF	5
rdf:Description	6
rdf:Alt	7
rdf:Bag	8
rdf:Seq	9
rdf:li	A

Tag Name	***Token***
rdf:type	B
rdf:value	C
rdf:subject	D
rdf:predicate	E
rdf:object	F

Table 8.2: Tag Tokens, Code Page 0

8.3.1.2 Core Vocabulary

The following tokens represent tags in code page one (1). All numbers are in hexadecimal.

Tag Name	***Token***
rdf:Description	6
rdf:Alt	7
rdf:Bag	8
rdf:Seq	9
rdf:li	A
rdf:type	B

Tag Name	***Token***
prf:component	C
prf:Defaults	D
prf:BitsPerPixel	E
prf:ColorCapable	F
prf:CPU	10
prf:ImageCapable	11

Tag Name	***Token***
prf:InputCharSet	12
prf:Keyboard	13
prf:MaxScreenChar	14
prf:Model	15
prf:OutputCharSet	16
prf:PointingResolution	17
prf:ScreenSize	18
prf:ScreenSizeChar	19
prf:SoftKeysCapable	1A
prf:SoundOutputCapable	1B
prf:TextInputCapable	1C
prf:Vendor	1D
prf:VoiceInputCapable	1E
prf:AcceptDownloadableSoftware	1F
prf:AudioInputEncoder	20
prf:DownloadableSoftwareSupport	21
prf:JVMVersion	23
prf:MexeClassmark	24
prf:MexeSpec	25

Tag Name	***Token***
prf:OSName	26
prf:OSVendor	27
prf:OSVersion	28
prf:RecipientAppAgent	29
prf:SoftwareNumber	2A
prf:VideoInputEncoder	2B
prf:CurrentBearerService	2C
prf:SecuritySupport	2D
prf:SupportedBearers	2E
prf:WapDeviceClass	2F
prf:WapPushMsgPriority	30
prf:WapPushMsgSize	31
prf:WapVersion	32
prf:WmlDeckSize	33
prf:WmlScriptLibraries	34
prf:WmlScriptVersion	35
prf:WmlVersion	36
prf:WtaiLibraries	37
prf:WtaVersion	38

Table 8.3: Tag Tokens, Code Page 1

8.3.1.3 Browser User-Agent

The following tokens represent tags in code page two (2). All numbers are in hexadecimal.

Tag Name	***Token***
rdf:Description	5
rdf:Alt	6
rdf:Bag	7
rdf:Seq	8
rdf:li	9
rdf:type	A
prf:component	B
prf:Defaults	C
prf:BrowserName	D
prf:BrowserVersion	E
prf:CcppAccept	F

Tag Name	***Token***
prf:CcppAccept-Charset	10
prf:CcppAccept-Encoding	11
prf:CcppAccept-Language	12
prf:DownloadableBrowserApps	13
prf:FramesCapable	14
prf:HtmlVersion	15
prf:JavaScriptVersion	18
prf:PreferenceForFrames	19
prf:TablesCapable	1A
Prf:XhtmlVersion	1B
prf:XhtmlModules	1C

Table 8.4: Tag Tokens, Code Page 2

8.3.2 Attribute Start Tokens

8.3.2.1 RDF

The following tokens represent the start of an attribute in code page zero (0). All numbers are in hexadecimal.

Attribute Name	***Attribute Value Prefix***	***Token***
ID		5
rdf:about		6
rdf:aboutEach		7
rdf:aboutEachPrefix		8
rdf:bagID		9
rdf:type		A

Attribute Name	***Attribute Value Prefix***	***Token***
rdf:resource		B
rdf:parseType	Literal	C
rdf:parseType	Resource	D
xml:lang		E
xmlns:prf		F
xmlns:rdf		10

Table 8.5: Attribute Start Tokens, Code Page 0

8.3.2.2 Core Vocabulary

The following tokens represent the start of an attribute in code page one (1). All numbers are in hexadecimal.

Attribute Name	***Attribute Value Prefix***	***Token***
rdf:resource		5
rdf:resource	http://www.wapforum.org/UAPROF/ccppschema-19991014#HardwarePlatform	6
rdf:resource	http://www.wapforum.org/UAPROF/ccppschema-19991014#SoftwarePlatform	7
rdf:resource	http://www.wapforum.org/UAPROF/ccppschema-19991014#NetworkCharacteristics	8
rdf:resource	http://www.wapforum.org/UAPROF/ccppschema-19991014#WAPCharacteristics	9
rdf:resource	http://www.wapforum.org/UAPROF/ccppschema-19991014#BrowserUA	A
prf:BitsPerPixel		10
prf:ColorCapable	Yes	11
prf:ColorCapable	No	12
prf:CPU		13
prf:ImageCapable	Yes	14
prf:ImageCapable	No	15
prf:InputCharSet		16
prf:Keyboard		17
prf:MaxScreenChar		18
prf:Model		19
prf:OutputCharSet		1A
prf:PointingResolution		1B
prf:ScreenSize		1C
prf:ScreenSizeChar		1D
prf:SoftKeysCapable	Yes	1E
prf:SoftKeysCapable	No	1F

Attribute Name	***Attribute Value Prefix***	***Token***
prf:SoundOutputCapable	Yes	20
prf:SoundOutputCapable	No	21
prf:TextInputCapble	Yes	22
prf:TextInputCapble	No	23
prf:Vendor		24
prf:VoiceInputCapable	Yes	25
prf:VoiceInputCapable	No	26
prf:AcceptDownloadableSoftware	Yes	30
prf:AcceptDownloadableSoftware	No	31
prf:AudioInputEncoder		32
prf:DownloadableSoftwareSupport		33
prf:JVMVersion		37
prf:MexeClassmark		38
prf:MexeSpec		39
prf:OSName		3A
prf:OSVendor		3B
prf:OSVersion		3C
prf:RecipientAppAgent		3D
prf:SoftwareNumber		3E
prf:VideoInputEncoder		3F
prf:CurrentBearerService		50
prf:SecuritySupport		51
prf:SupportedBearers		52
prf:WapDeviceClass		60
prf:WapPushMsgPriority		61
prf:WapPushMsgSize		62
prf:WapVersion		63
prf:WmlDeckSize		64
prf:WmlScriptLibraries		65
prf:WmlScriptVersion		66
prf:WmlVersion		67
prf:WtaiLibraries		68
prf:WtaVersion		69

Table 8.6: Attribute Start Tokens, Code Page 1

8.3.2.3 Browser User-Agent

The following tokens represent the start of an attribute in code page two (2). All numbers are in hexadecimal.

Attribute Name	***Attribute Value Prefix***	***Token***
prf:CcppAccept		5
prf:CcppAccept-Charset		6
prf:CcppAccept-Encoding		7
prf:CcppAccept-Language		8
prf:DownloadableBrowserApps		9
prf:FramesCapable	Yes	A
prf:FramesCapable	No	B
prf:HtmlVersion	3.2	C
prf:HtmlVersion	4.0	D
prf:JavaScriptVersion		12
prf:PreferenceForFrames	Yes	13
prf:PreferenceForFrames	No	14
prf:TablesCapable	Yes	15
prf:TablesCapable	No	16
Prf:XhtmlVersion		17
prf:XhtmlModules		18

Table 8.7: Attribute Start Tokens, Code Page 2

8.3.3 Attribute Value Tokens

8.3.3.1 RDF

The following tokens represent attribute values in code page zero (0). All numbers are in hexadecimal.

Attribute Value	***Token***
rdf:Statement	85
http://	86
http://www.	87
https://	88
https://www.	89
www.	8A
.com/	8B
.edu/	8C
.net/	8D
.org/	8E

Table 8.8: Attribute Value Tokens, Code Page 0

8.3.3.2 Core Vocabulary

The following tokens represent attribute values in code page one (1). All numbers are in hexadecimal.

Attribute Value	***Token***
No	85
Yes	86

Table 8.9: Attribute Value Tokens, Code Page 1

8.3.3.3 Browser User-Agent

The following tokens represent attribute values in code page two (2). All numbers are in hexadecimal.

Attribute Value	***Token***
No	85
Yes	86

Table 8.10: Attribute Value Tokens, Code Page 2

9 User Agent Profile Transport Over WSP

This document specifies how Profiles get transported over WSP. Section 9.1 is a an introduction and is not normative. The CC/PP Exchange Protocol over WSP, referred to as CC/PP-WSP, is specified in the normative Sections 9.2 and 9.3. Section 9.4 specifies static conformance requirements. Examples are not normative.

9.1 Introduction

9.1.1 The CC/PP Framework and the CC/PP Exchange Protocol Over HTTP

The Composite Capability and Preference Profiles (CCPP) defines a framework for content negotiation [CC/PP]. Section 7 defines a vocabulary of categories and attributes for WAP-enabled devices, and Section 8 describes how the CC/PP document is encoded using the WAP Binary XML [WBXML] standard.

To transport CC/PP documents, or references to such profiles, over the Internet, the CC/PP Exchange Protocol Over HTTP has been specified in [CCPPEx]. The CC/PP Exchange Protocol Over HTTP, referred to as CC/PP-HTTP, is used over HTTP [HTTP] and uses the HTTP Extension Framework [HTTPex]. The mapping onto WSP is specified in this section and is sometimes referred to as CC/PP-WSP.

The CC/PP Exchange Protocol specifies two new request header fields (*Profile*, *Profile-Diff*) and one new response header field (*Profile-Warning).* The *Profile* header is used to transport one or many Profile identities, URIs, from the client to the server. This set of Profiles is used to construct the Composite Profile Information (CPI). The *Profile-diff* header is used to transport changes to the CPI. This means that the *Profile-Diff* header must always be used together with and referenced by the *Profile* header. The *Profile-Warning* header is used by the server to notify the client whether the request to use Profiles was fulfilled, partly fulfilled, or not fulfilled. To extend the HTTP protocol with the new header in a structured way, the HTTP Extension Framework has been used [HTTPext].

9.1.2 Using WSP to Transport CCPP Profiles

The WSP protocol has some features that can not be found in HTTP. To reduce the size of request messages the WSP client can cache headers in the server for the lifetime of a WSP session. The cached headers are called session headers and are sent to the server during session establishment. The client can use the **Resume** operation of WSP to update the session headers during the session. At any time the client or the server can terminate the session and establish a new one, with new session headers.

The WAP gateway combines request headers with session headers to create HTTP requests [WAE]. The following list summarizes the WSP header management:

- If one or many *Profile* headers are cached in the server, the client can override all of them within the scope of a particular request by sending other *Profile* headers in a single request message. The same is true for the *Profile-Diff* header.
- If one or many *Profile* headers, but no *Profile-Diff* headers, are cached in the server, the client can append one or many *Profile-Diff* headers within the scope of a particular request by including them in a request message.
- If one or many *Profile-Diff* headers, but no *Profile* headers, are cached in the server, the client can append one or many *Profile* headers within the scope of a particular request by including them in a request message.

- By using the Resume function the client can update the session headers, without establishing a new session[3].

9.1.3 Differences Between CCPP/HTTP and CCPP/WSP

The following are the differences between CC/PP-HTTP and CC/PP-WSP:

- In the CC/PP-WSP *Profile-Warning* response header, the warning text is not included.
- In CC/PP-HTTP, multiple profile references are transmitted in one *Profile* header. In WSP, a *Profile* header can only transmit one profile reference, but multiple *Profile* headers can be transmitted in the same WSP header. Consequently, no functionality is lost. In the CC/PP-WSP *Profile-Diff* header, the profile section must be encoded using WBXML as specified in Section 8. In the CC/PP-HTTP *Profile-Diff* header, the profile section must be sent as XML text.

9.2 Structure and Encoding of Header Fields

9.2.1 The Profile Header

The syntax of the Profile header MUST conform to the production of [**1.**].

[1.] Profile = Profile-field-name Profile-field-value

[2.] Profile-field-name = Short-integer [WSP]

[3.] Profile-field-value = Uri-value

Example:

0xb5 http://anyco.com/anypda 0x00

In the above example the profile URI is http://anyco.com/anypda. Note that 0x00 signals the end of the Uri-value string.

9.2.2 The Profile-Diff Header

The syntax of the Profile-Diff header MUST conform to the production of [**4.**]. The CC/PP-profile in production [**7.**] MUST be encoded using WBXML as specified in Section 8.

[4.] Profile-diff = Profile-diff-field-name Profile-diff-field-value

[5.] Profile-diff-field-name = Short-integer [WSP]

[6.] Profile-diff-field-value = Value-length CCPP-profile

[7.] CCPP-profile = *Octet; encoded using WBXML [WBXML]

Example:

0xb6 0x0A 0x01 0x05 0x04 ...

In the above example:

- The length is 10 octets: 0x0A;

[3] This is not possible in WSP 1.0.

- The binary XML version is 1.1: 0x01;
- The public identifier has been assigned the value of 5: 0x05; and
- The character set is iso-8859-1 : 0x04

9.2.3 The Profile-Warning Header

The syntax of the Profile-Warning header MUST conform to the production of [**8.**].

[8.] Profile-warning = Profile-warning-field-name Profile-warning-value

[9.] Profile-warning-field-name = Short-integer [WSP]

[10.] Profile-warning-value = Warn-code | (Value-length Warn-code Warn-target *Warn-date)

[11.] Warn-code = Short-integer

Status codes (and corresponding Short-integer values) are 100 (0x90) | 101 (0x91) | 102 (0x92) | 200 (0xa0) | 201 (0xa1) | 202 (0xa2) | 203 (0xa3) [CC/PPEx]

[12.] Warn-target = Uri-value

[13.] Warn-date = Date-value

Example 1:

0xb7 0x16 0x92 http://anyco.com/pda 0x00

In the above example:

- The length is 22 octets (0x16)
- The profile warning code is 102 (encoded as 0x92); and
- The profile URI is http://anyco.com/pda
- No date is provided

Example 2:

0xb7 0x92

In the above example, only the warning code is sent.

9.3 Protocol Procedures

9.3.1 Session Establishment

The following procedure is used to establish a WSP session that uses User Agent Profiles:

[14.] To indicate support for User Agent Profiles, the client MUST include the *Profile* header in the connect message.

[15.] If the server supports User Agent Profiles, it MUST respond with the *Profile-Warning* header with the warning code set to 100 ("OK").

[16.] If the client does not receive the *Profile-Warning* header with the warning code 100, it MUST NOT send any additional User Agent Profile headers during the WSP session.

During the session the client can use the Resume procedure to update the session header [WAE][4].

Session headers are cached in the server for the duration of the WSP session [WSP]. When the gateway generates HTTP request messages it combines the headers from the request message with the headers in the session's cache, according to the rules in section 9.3.2.

9.3.2 Combining Session and Request Headers

Upon reception of a WSP request, the WAP gateway combines the request headers with the cached WSP session headers. The result is a list of WSP headers that gets translated into one HTTP request, see section 9.3.3. The following procedure is used to combine the request headers with the cached session headers into one list of headers:

[4] This is not possible in WSP version 1.0.

[17.] Headers from the WSP request MUST override cached WSP session headers according to the rules for client headers defined in [WAE].

[18.] If, after the previous operation, both request headers and session headers are present in the list of headers, request headers MUST follow after session.

Since request headers are appended to the list after the session headers, request headers will take precedence over session headers if the headers are applied to the profile in the same order as they are transported. **Example 1:**

In this example, both request and session headers remain after the WSP request and the WSP session have been combined. In the result, the request header comes after the session headers in the list.

CCPP cached session headers:

```
Profile: URIx
Profile: URIy
```

CCPP request header:

```
Profile-Diff: 0x01 0x05 0x04 0xBB
```

Will result in:

```
Profile: URIx
Profile: URIy
Profile-Diff: 0x01 0x05 0x04 0xBB
```

Example 2:

In this example, the request headers override all session headers.

CCPP cached session headers:

```
Profile: URIx
Profile: URIy
```

CCPP request header:

```
Profile: URIz
Profile-Diff: 0x01 0x05 0x04 0xBB
```

Will result in:

```
Profile: URIz
Profile-Diff: 0x01 0x05 0x04 0xBB
```

9.3.3 Header Translation Between CC/PP-WSP and CC/PP-HTTP

The WAP gateway MUST forward WSP requests as HTTP 1.1 requests [WAE]. In forwarding the request, the gateway MUST forward all CC/PP-WSP headers (defined in Section 9.2 and resolved at the gateway according to the rules of Section 9.3.2) as CC/PP-HTTP headers (defined in [CCPPEx]) according to these rules:

[19.] Each CC/PP-WSP Profile-Diff header field is translated into exactly one CC/PP-HTTP Profile-Diff header field. The CC/PP-HTTP Profile-Diff headers are generated dynamically as specified in [CC/PPEx]. WBXML encoding of the profile section MUST be undone, leaving the profile section as XML text.

[20.] A single CC/PP-HTTP Profile header field is generated as specified in [CC/Ppex] by listing the values of each CC/PP-WSP *Profile* header and the values of each dynamically generated CC/PP-HTTP *Profile-Diff* header.

[21.] One CC/PP-HTTP *Profile-Warning* response header is translated into exactly one CC/PP-WSP *Profile-Warning* response header. The warning text from the HTTP header is not translated.

[22.] One CC/PP-HTTP *Profile-Warning* response header is translated into exactly one CC/PP-WSP *Profile-Warning* response header. The warning text from the HTTP header is not translated.

In forwarding the HTTP request and generating the CC/PP-HTTP Profile and Profile-Diff headers, the gateway MAY insert additional profile information into the request. If provided, this additional information MUST be presented by appending to the end of the *Profile* header either a URI or a dynamically generated *Profile-Diff* header identifier in accordance with the CC/PP Exchange Protocol over HTTP [CC/PPex]. Accordingly, a compliant gateway MAY therefore introduce a *Profile* (and, if necessary, *Profile-Diff* header) on behalf of a client whose WSP session has no cached *Profile* or *Profile-Diff* headers. This support enables a gateway to support User Agent Profiles on behalf of client devices that are otherwise unable to convey profile information.

Example:

The WSP headers:

```
Profile: URIx
Profile-Diff: 0x01 0x05 0x04 0xAA
Profile: URIy
Profile-Diff: 0x01 0x05 0x04 0xBB
```

Are translated into the following HTTP headers:[5]

```
99-Profile: URIx, 1-uKdjJHuhjHUuj, URIy, 2-jdsjhUHjsuHjU
99-Profile-Diff-1: <?xml version="1.0"?><RDF>AA...
99-Profile-Diff-2: <?xml version="1.0"?><RDF>BB...
```

9.4 Static Conformance Requirements

A conformant *Profile* header MUST match the production rule [**1.**].

A conformant *Profile-Diff* header MUST match the production rule [**4.**].

A conformant *Profile-Warning* header MUST match the production rule [**8.**].

A conformant CC/PP-WSP gateway MUST comply to the rules in [**15.**], [**17.**], [**18.**], [**19.**], [**7.**] and [**7.**].

A conformant CC/PP-WSP client MUST comply to the rules in [**14.**] and [**16.**].

[5] The value of the *Profile-diff* digest is not real, see [CC/PPEx].

10 Origin Server Behavior

This section is informative.

From the gateway to the origin server, the User Agent Profile is transported over the Internet. In the WAP architecture specification, and as specified in Section 9.3.3, HTTP is assumed to be the transport mechanism for Internet-related information, using the HTTP 1.1 mechanisms for header management and caching [WAE].

The transmission of profiles, overrides, and associated mechanisms is managed by the CC/PP Exchange Protocol over HTTP [CC/PPex]. This protocol was developed by the W3C as an application of the HTTP 1.1 Extension Framework [HTTPext].

The following components are necessary to implement an Internet server capable of receiving User Agent Profile information:

- A HTTP 1.1 server [HTTP]
- The HTTP 1.1 Extension Framework [HTTPext][6]
- The CC/PP Exchange Protocol [CC/PPex]

Upon receiving a User Agent Profile, an origin server MAY do the following:

- Parse the profile
- Validate the syntax of the profile
- Resolve attribute values by applying overriding rules and default values
- Validate the attribute value types
- Customize content according to the information contained within the profile

11 Deployment Considerations

This section is informative.

End-to-end systems supporting User Agent Profiles will depend upon the support of many of the elements in the WAP architecture:

- Client devices may need to store, generate, and transmit profile information
- Repositories may provide profile persistence or caching services
- Gateways need to forward profile information to proxies or servers
- Proxies may need to modify some of the profile information according to services that they provide, or they may need to tailor their provided services according to the client's profile information
- Servers may need to utilize the profile information to help adapt content that is to be provided to the client device

A variety of schemes may be used to provision and maintain the profile information. Designation of these methods and approaches is beyond the scope of this specification. However, this section outlines some concepts that may need to be taken into consideration in the preparation and deployment of a UAProf capable system.

[6] It should be noted that an HTTP 1.0 or HTTP 1.1 origin server may still honor User Agent Profiles and the CC/PP Exchange Protocol without implementing the full HTTP 1.1 Extension Framework. To do this, the origin server must specifically parse the HTTP request and seek those headers that apply to the CC/PP Exchange Protocol. However, in a pure sense, these origin servers are respectively non-conformant with the HTTP 1.0 or HTTP 1.1 specifications.

11.1 Client Support

It is clear that this specification will not be applied to initial WAP products which conform to the WAP 1.1 specifications. For backward compatibility, therefore, future systems will need to support clients and gateways that are unaware of User Agent Profiles. Similarly, support for one-way and broadcast service paradigms will create service models different than that for web browsing. Generally, therefore, client device support for User Agent Profiles will take various forms and will need to be supported in a variety of ways.

11.1.1 Client Devices Not Supporting User Agent Profiles

For those devices that do not directly support or cannot transmit UAProf information, indirect support may be provided by the gateway. It may be possible to have a static profile provisioned at the gateway by a carrier or service provider. This profile would be presented to gateways, proxies, and servers on behalf of the device(s) involved. The client device in this case does not require any special provisioning or support for this service.

The gateway may also support dynamic, customized profiles on behalf of these legacy devices and one-way devices. For example, the gateway may apply a dynamically-generated profile according to the particular device ID or subscriber invoking the service.

More information about the indirect support scheme is provided in Section 9.3.3.

11.1.2 Client Devices Supporting User Agent Profiles

For those devices that provide UAProf capabilities, the support may take different forms and therefore demand various provisioning schemes. Some devices may only support a static profile header which is transmitted during session invocation. Other devices may also be able to collect and send user preference information during the session invocation. Yet other devices may additionally support interactive, or request-specific, preference reporting.

11.1.2.1 Static Header Support

For some client devices, a fixed header may be utilized to support the UAProf *Profile* header reported during WSP session creation. This fixed header may be common for all devices of the same model or may be somewhat individualized for each device.

A profile could be loaded into non-volatile memory by the manufacturer, network operator, or service provider. Typically, this profile would simply be a URI reference to a shared set of preference data stored on some repository on behalf of all such devices. This shared set of profile information would likely be deployed by the manufacturer, network operator, or service provider to cover the service characteristics that they wish to enable.

Depending on the number of devices that share the static profile, the supplied URI may itself become a de-facto preference indicator for proxies or origin servers. For example, a reference to a profile at http://www.foo.com/dev123 that would be invoked by millions of client devices eventually could lead to content shaping operations based upon the URI itself rather than upon the particular profile attributes that the URI references. This could lead to a misuse of the UAProf capability because, strictly speaking, the profile referenced by the URI could change arbitrarily at its server, thereby leading to incorrect behavior at the proxies or origin servers. Consequently, once a URI emerges as a moniker to represent a default profile, care must be taken not to modify the profile that the URI references.

An individualized URI for the static profile could be stored on each device. This would provide the opportunity to have some user-specific information stored within the profile.

Alternatively, the entire static profile may be stored on the client. The definition of the values in such a scheme may be performed in several ways: preset by the manufacturer during production; hardcoded in the user agent software; programmed by the network operator or service provider at time of subscriber lease/purchase; and/or, programmed over-the-air by the network operator or service provider. The methods that may be developed are beyond the scope of this specification.

11.1.2.2 Session-Constant Support

The client device may have the capability to dynamically construct its profile and present that profile during the WSP session creation. The profile would be used throughout the session lifetime.

The profile attributes delivered by the device may be user alterable or may be pre-set by the device manufacturer, network operator, or service provider. For example, a device may permit the user to select the default language (e.g. English, French, or German) but may prevent the user from altering certain other profile attributes, including the URI representing the basic characteristics of the device or network. The techniques for changing these attributes may differ: a user interface or form may be used for the user-modifiable values, and an over-the-air scheme may be used for the server-managed values.

11.1.2.3 Request-Specific Support

For more dynamic profile updates, the client device needs to support the *Profile-Diff* header scheme.

For example, a user agent, like a browser, may allow the subscriber to indicate a preference for images. By changing the preference, the user agent would have to send an update over the WSP session indicating the change so that it could be included in the cached profile at the WAP gateway. Depending on the user interface model of the user agent, the change may be temporary (for the current WSP request) or more long term (service toggle). These would be reflected in whether the new attribute value is sent within a *Profile-Diff* included in a WSP request or a WSP **Resume** operation.

The device may update the cached session profile without direct user intervention. For example, a phone could be designed to block a user agent's access to the audio services while a call is in progress. When a telephone call is made, the device may send an updated Profile-Diff to the WAP gateway to indicate that the user agent no longer has access to audio capabilities.

11.2 Repository Support

A profile repository typically would be an HTTP server that provides UAProf CPI elements upon request. UAProf profiles may reference data stored in repositories provided and operated by the subscriber; network operator; gateway operator, device manufacturer; or service provider. It is expected that there will be a variety of mechanisms put into place to create and manage the data in such repositories:

- Manufacturers will likely provide static profile resources that describe the client devices that they produce. Such profiles will describe the hardware and standard software elements that exist on the client devices.
- Network operators will nominally provide profile information that includes network characteristics, and gateway operators may provide profile information that defines permitted service levels through the communications infrastructure. Some of these operators may provide additional services enabling the storage of user preference information.
- Subscribers may look to other entities to store preference information.
- Service providers, including enterprise IT managers, may create profile repositoriesthat reflect the needs of their hosted applications or services.

Independent of the content of the stored profile, policy controls may be imposed that limit what profile information is delivered to a particular requester. Any such limits, or methods employed to ensure compliance, are beyond the scope of this specification.

As noted in Section 4.2, support for third-party queries and updates to profile repositories is left as an area of future work.

11.3 Gateway Support

Gateway systems bridge between the CC/PP-WSP and CC/PP-HTTP representations of the User Agent Profile. In many situations, gateways may need to augment the profile (if any) provided by the client with additional attributes supplied by the network operator, gateway operator, or service provider. This situation arises

- To enable legacy devices which do not directly support the UAProf conventions (Sections 9.3.3 and 11.1.1)
- To enable one-way devices that cannot directly establish WSP sessions (Sections 9.3.3 and 11.1.1)
- To support specification and overriding of operational characteristics controlled by the network or gateway operator. For example, the gateway may override the device profile to reflect the current network bearer characteristics or services available to roaming customers.
- To allow the device profile to be augmented with user preferences that are stored at a repository in the network (Section 11.2)

The CC/PP Exchange Protocol [CC/PPex] defines several mechanisms by which profiles may be transmitted, namely

- Mandatory and hop-by-hop
- Optional and hop-by-hop
- Mandatory and end-to-end
- Optional and end-to-end

Given the relative novelty of the CC/PP mechanism, initial WAP gateway deployments should use the optional and hop-by-hop forwarding mechanism.

The WAP Push Access Protocol [WAP-PAP] defines a service whereby a third party may request and obtain User Agent Profile Information from a gateway for the purposes of implementing PUSH services. As discussed in Section 4.2, further support for PUSH services is beyond the scope of the current specification and is left as an item for future work.

As an alternative for situations in which the Push Access Protocol is unavailable, PUSH services may be designed to include an explicit subscription request initiated by the client to the PUSH origin server. In making this subscription, the client conveys its CPI profile, which, in turn, can be cached by the PUSH origin server and be used when generating content to be pushed to the client. This approach reduces the PUSH process to a conformant implementation of the PULL model that is supported by this specification.

11.4 Interim Proxy Support

This specification relies on mechanisms (such as the HTTP 1.1 Extension Framework [HTTPext] and the CC/PP Exchange Protocol over HTTP [CC/PPex]) that are not currently deployed widely over the Internet. Over time, it is expected that content servers will support these new conventions. In the meantime, it is expected that a class of proxy services will be deployed to support the conversion of content based upon the information available in the User Agent Profiles. These proxies will utilize information from content sources that are unaware of profile information and adapt it based upon profile information contained within the request.

In general, a request to an adapting proxy service will look like a request to a content server having the requisite profile support. These proxies will, in turn, issue requests to content servers or other adapting proxies. These proxies will need to be flexible in forming the outbound request so as to best support the capabilities of the target server or proxy. For example, if a request is directed to a a content server that does not support the HTTP 1.1 Extension Framework, the proxy would need to adapt the request to a standard HTTP 1.1 form (e.g. using HTTP *Accept* and *User-Agent* headers).

A chain of such adapting proxy services can be envisioned, with each service performing a specialized set of adaptations to the content based upon the profile information presented to it. Each link in such a chain would be able to revise the profile information to account for the services that it supports. For example, a proxy that can convert

graphic images of various formats to a standard bitmap format for WAP devices would advertise this capability by extending the request profile (that is, by adding a new *Profile-Diff*) with the image types it supported.

Within such an adapting proxy chain, each supporting proxy will determine whether to adapt the content being returned by examining the content, the profile contained in the received request, and the profile forwarded to the content server. In particular, if a server returns a data type that was enabled as a result of a *Profile-Diff* introduced by the adapting proxy, then the proxy would be expected to adapt the content to a format that is supported by the received request.

11.5 Origin Server Support

Initially, there will be many origin servers that will not support User Agent Profiles. Consequently, it is expected that data delivered by these servers will not match the capabilities of the device making the request. Proxies and gateways can be used to adapt content provided by these servers.

The origin servers that do support User Agent Profiles will be consumers of profile information and will therefore be able to generate targeted content that is appropriate for the particular device, network, and user.

A.1 Summary of User Agent Profile Schema

The table below summarizes the components and attributes defined within the WAP User Agent Profile schema (see Section 7.4). This section is informative.

Attribute	Description	Resolution Rule	Type	Sample Values
Component: HardwarePlatform				
BitsPerPixel	The number of bits of color or grayscale information per pixel	Override	Number	"2", "8"
ColorCapable	Whether the device display supports color	Override	Boolean	"Yes", "No"
CPU	Name and model number of device CPU	Locked	Literal	"Pentium III", "PowerPC 750"
ImageCapable	Whether the device supports the display of images	Locked	Boolean	"Yes", "No"
InputCharSet	List of character sets supported by the device for text entry	Locked	Literal (bag)	"US-ASCII", "ISO-8859-1", "Shift_JIS"
Keyboard	Type of keyboard upported by the device	Locked	Literal	"Disambiguating", "Qwerty", "PhoneKeypad"
MaxScreenChar	Size of the virtual page onto which a document is rendered, in units of characters	Locked	Dimension	"16x80", "48x32"
Model	Model number assigned to the terminal device by the vendor or manufacturer	Locked	Literal	"Mustang GT", "Q30"
OutputCharSet	List of character sets supported by the device for output to the display	Append	Literal (bag)	"US-ASCII", "ISO-8859-1", "Shift_JIS"
PointingResolution	Type of resolution of the pointing accessory supported by the device	Locked	Literal	"Character", "Line", "Pixel"
ScreenSize	The size of the device's screen in units of pixels	Locked	Dimension	"160x160", "640x480"
ScreenSizeChar	Size of the device's screen in units of characters	Locked	Dimension	"12x4", "16x8"
SoftKeysCapable	Indicates whether the device supports programmable soft keys	Locked	Boolean	"Yes", "No"
SoundOutputCapable	Indicates whether the device supports sound output	Locked	Boolean	"Yes", "No"
TextInputCapable	Indicates whether the device supports alpha-numeric text entry	Locked	Boolean	"Yes", "No"
Vendor	Name of the vendor manufacturing the terminal device	Locked	Literal	"Ford", "Lexus"
VoiceInputCapable	Indicates whether the	Locked	Boolean	"Yes", "No"

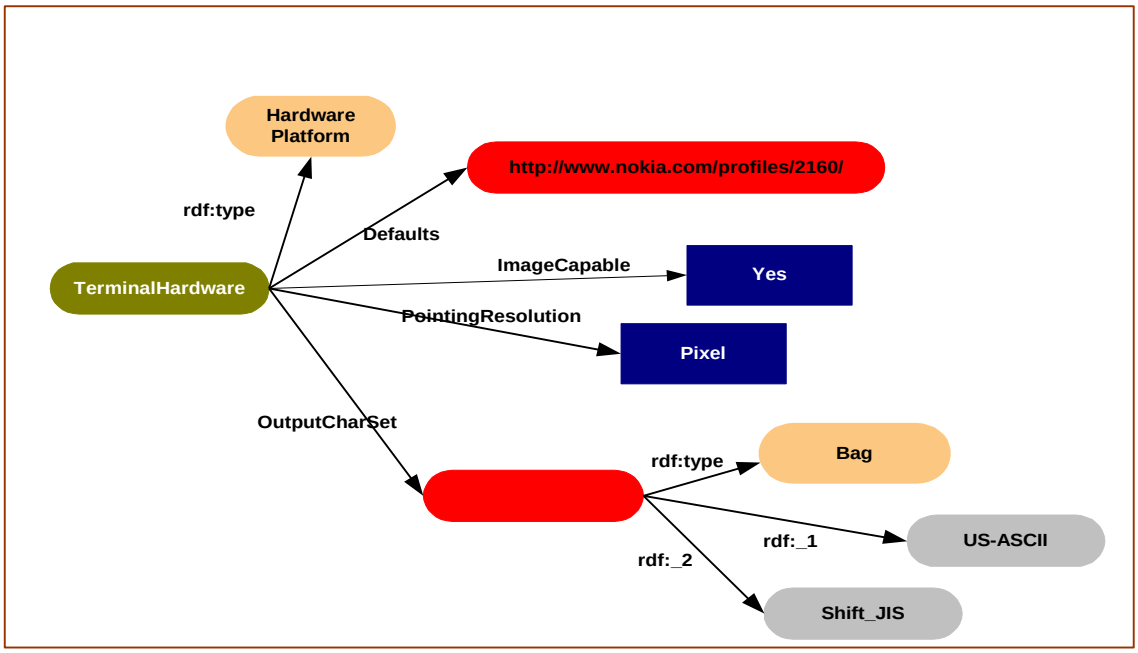
	device supports any form of voice input, including speech recognition			
Component: SoftwarePlatform				
AcceptDownloadableSoftware	Indicates the user's preference on whether to accept downloadable software	Locked	Boolean	"Yes", "No"
AudioInputEncoder	List of audio input encoders supported by the device	Append	Literal (bag)	"G.711", "G.931"
DownloadableSoftwareSupport	List of executable content types which the device supports and which it is willing to accept from the network	Locked	Literal (bag)	"application/x-msdos-exe"
JVMVersion	List of the Java virtual machines installed on the device	Append	Literal (bag)	"SunJRE1.2", "MSJVM1.0"
MexeClassmark	ETSI MExE classmark	Locked	Number	"1", "2"
MexeSpec	Class mark specialization	Locked	Literal	"7.02"
OSName	Name of the device's operating system	Locked	Literal	"Mac OS", "Windows NT"
OSVendor	Vendor of the device's operating system	Locked	Literal	"Apple", "Microsoft"
OSVersion	Version of the device's operating system	Locked	Literal	"6.0", "4.5"
RecipientAppAgent	User agent associated with the current request	Locked	Literal	"SpeedyMail"
SoftwareNumber	Version of the device-specific software (firmware) to which the device's low-level software conforms	Locked	Literal	"2"
VideoInputEncoder	List of video input encoders supported by the device	Append	Literal (bag)	"MPEG-1", "MPEG-2", "H.261"
Component: NetworkCharacteristics				
CurrentBearerService	The bearer on which the current session was opened	Locked	Literal	"OneWaySMS", "GUTS", "TwoWayPacket"
SecuritySupport	Type of security or encryption mechanism supported	Locked	Literal	"PPTP"
SupportedBearers	List of bearers supported by the device	Locked	Literal (bag)	"GPRS", "GUTS", "TwowaySMS", CSD", "USSD"
Component: BrowserUA				
BrowserName	Name of the browser user agent associated with the current request	Locked	Literal	"Mozilla", "MSIE"
BrowserVersion	Version of the browser	Locked	Literal	"1.0"
CcppAccept	List of content types the device supports	Append	Literal (bag)	"text/html", "text/plain", "text/html", "image/gif"
CcppAccept-Charset	List of character sets the device supports	Append	Literal (bag)	"US-ASCII", "ISO-8859-1",

				"Shift_JIS"
CcppAccept-Encoding	List of transfer encodings the device supports	Append	Literal (bag)	"base64", "quoted-printable"
CcppAccept-Language	List of preferred document languages	Append	Literal (sequence)	"zh-CN", "en", "fr"
DownloadableBrowserApps	List of executable content types which the browser supports and which it is willing to accept from the network	Append	Literal (bag)	"application/ava-applet" "application/javascript"
FramesCapable	Indicates whether the browser is capable of displaying HTML frames	Override	Boolean	"Yes", "No"
HtmlVersion	Version of HyperText Markup Language (HTML) supported by the browser	Locked	Literal	"2.0", "3.2", "4.0"
JavaScriptVersion	Version of the JavaScript language supported by the browser	Locked	Literal	"1.4"
PreferenceForFrames	Indicates the user's preference for receiving HTML content that contains frames	Locked	Boolean	"Yes", "No"
TablesCapable	Indicates whether the browser is capable of displaying HTML tables	Locked	Boolean	"Yes", "No"
XhtmlVersion	Version of XHTML supported by the browser	Locked	Literal	"1.0"
XhtmlModules	List of XHTML modules supported by the browser	Append	Literal	"XHTML1-struct", "XHTML1-blkstruct", "XHTML1-frames"
Component: WapCharacteristics				
WapDeviceClass	Classification of the device based on capabilities as identified in the WAP 1.1 specifications	Locked	Literal	"A"
WapPushMsgPriority	User's preference on the priority of incoming push messages	Locked	Literal	"critical", "low", "none", "all"
WapPushMsgSize	Maximum size of a push message that the device can handle	Locked	Number	"1024", "1400"
WapVersion	Version of WAP supported	Locked	Literal	"1.1", "1.2", "2.0"
WmlDeckSize	Maximum size of a WML deck that can be downloaded to the device	Locked	Number	"4096"
WmlScriptLibraries	List of mandatory and optional libraries supported in the device's WMLScript VM	Locked	Literal (bag)	"Lang", "Float", "String", "URL", "WMLBrowser", "Dialogs"
WmlScriptVersion	List of WMLScript version numbers supported by the device	Append	Literal (bag)	"1.1", "1.2"
WmlVersion	List of WML language	Append	Literal	"1.1", "1.0"

	version numbers supported by the device		(bag)	
WtaiLibraries	List of WTAI network common and network specific libraries supported by the device that are URI accessible	Locked	Literal (bag)	"WTAVoiceCall", "WTANetText", "WTAPhoneBook", "WTACallLog", "WTAMisc", "WTAGSM", "WTAIS136", "WTAPDC"
WtaVersion	Version of WTA user agent	Locked	Literal	"1.1"

A.2 Serialized and Abbreviated Syntaxes

This section provides a comparison of the long and abbreviated syntax of XML encoded RDF using an example indicated in the following RDF graph (verified with SiRPAC [SiRPAC]). This section is informative.



1) The long or serialized syntax:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
   <rdf:Description ID="TerminalHardware">
      <rdf:type resource="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#HardwarePlatform"/>
      <prf:Defaults rdf:resource="http://www.nokia.com/profiles/2160"/>
      <prf:OutputCharset>
         <rdf:Bag>
            <rdf:li>US-ASCII</rdf:li>
            <rdf:li>Shift_JIS</rdf:li>
         </rdf:Bag>
      </prf:OutputCharset>
      <prf:ImageCapable>Yes</prf:ImageCapable>
      <prf:PointingResolution>pixel</prf:PointingResolution>
   </rdf:Description>
</rdf:RDF>
```

2) The equivalent compact or abbreviated syntax:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema-19991014#">
   <rdf:Description ID="TerminalHardware"
```

```
         rdf:type="http://www.wapforum.org/UAPROF/ccppschema-
19991014/#HardwarePlatform"
         prf:ImageCapable="Yes"
         prf:PointingResolution="pixel">
      <prf:Defaults rdf:resource="http://www.nokia.com/profiles/2160"/>
      <prf:OutputCharset>
         <rdf:Bag rdf:_1="US-ASCII"
                  rdf:_2="Shift_JIS"/>
      </prf:OutputCharset>
   </rdf:Description>
</rdf:RDF>
```

A.3 Conneg Vocabulary

The following attributes are taken from the vocabulary defined in the IETF Content Negotiation Group [need reference to the RFC]. This section is informative, and is provided as a guide to those who may be defining a UAProf schema based upon the Conneg vocabulary.

Attribute	Definition	Resolution Rule	Sample Value	Notes
dpi	Resolution in display or print, in pixels per inch	L	300	Can this be mapped to screensize?
paper-size	Size of the stationery	L	a4, letter, a3, b4, legal	Modify UAProf Screensize to UaMediaSize? How would we handle MaxScreenChar and ScreenSizeChar attributes?
color	Indicates a gross level of capability to represent (or need for) for handling of colour, out of a limited set of choices.	L	Grey, binary, limited, mapped, full	Replace with ColorCapable and BitsPerPixel?

A.4 Salutation Vocabulary

The following attributes are taken from the Salutation [Salutation] vocabulary. These attributes are considered as extensions to the base vocabulary for User Agent Profile. This section is informative, and is provided as a guide to those who may be defining a UAProf schema based upon the Salutation vocabulary.

Attribute	Definition	Resolution Rule	Sample Value
major version	Version of the salutation protocol		2
minor version			0
Contact Person name	Manager of the client device		"John Doe"
Authentication Flavors	How authentication can be performed		null|userIdAndPassword
Printer Language	Supported printer languages		PS|ESC/P|...
documentFormat			ms53a2|tiff|bmp|…
imageCompAlgorithm	Image compression algorithm. Applicable for TIFF images		raw|mh|mhb|…
imageResolution	Image resolution		800x600
printPaperDirection			Portrait|landscape
printPaperOutputSelect	Paper source		standard|collatedSort|staple
printOutputBinSelect	Output bin		0|1|2…
dataTransferTimeOutSettable	Boolean, is the time-out configurable?		Yes|No
dataTransferTimeOutLength	Transfer time-out length in seconds		value in seconds
faxProtocol	Fax protocol version		g3|g4|auto

dataTransferTimeOutSettable	Boolean, is the time-out configurable?		Yes|No
dataTransferTimeOutLength	Transfer time-out length		value in seconds
dataTransferTimeOutSettable	Boolean, is the time-out configurable?		Yes|No
dataTransferTimeOutLength	Transfer time-out length		value in seconds
maxDuration	Maximum duration of message in seconds		60
maxReceiversPlay	Maximum number of receivers		1
maxRecipientsSend	Maximum number of recipients		1
voiceSpeed	Voice speed (1-10)		5
voiceVolume	Volume (1-10)		5
synthesize	Boolean		Yes|No
synthesizeVoiceSpeed	Voice speed (1-10)		6
synthesizeVoiceVolume	Volume (1-10)		4
synthesizeVoiceType	Voice type		male|female
synthesizeTextLanguage	Encoding language. Language tag which is defined in RFC 1766.		x-klingon
encoding	Speech encoding method, sampling rate		(pcm,r8k)
conversionWithLossProhibited	Is the use of lossy conversions allowed?		Yes|No
implicitConversionProhibited	Are format conversions allowed?		Yes|No
latestDeliveryTime	date and time by which the message must be delivered to the Receiver(s)		
Formatted Name			
Name			
Photograph			
Birth date			
Address			
Address delivery label			
Telephone Number			
Electronic mail			
Mailer			
Time Zone			
Location			
Title			
Business category			
Logo			
Organization			
Comment			
Pronounciation			
Uniform Resource Locator			
Unique Identifyer			
Public key			
personalityProtocol			
supportedCommand			

exchangeDataFormatSupport			
characterSetSupport			
SearchSupport			
SortSupport			

A.5 Reserved Attributes

This section is normative.

The following attributes are not currently included in the base vocabulary for User Agent Profiles, but are reserved for possible incorporation at a future date. Extended vocabularies SHOULD NOT make use of these attribute names.

```
<rdf:Description ID="LocationCapable">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  Indicates whether the device supports determination and
                    transmission of its location.
      Type:         Boolean
      Resolution:   Override
      Examples:     "Yes", "No"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="LocationEncoding">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  Description of the encoding of the device's location.
      Type:         Literal
      Resolution:   Locked
      Examples:     "lat-long", "cell-id"
   </rdfs:comment>
</rdf:Description>

<rdf:Description ID="LocationValue">
   <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property"/>
   <rdfs:domain rdf:resource="#NetworkCharacteristics"/>
   <rdfs:comment>
      Description:  Description of the device's location.
      Type:         Literal
      Resolution:   Override
   </rdfs:comment>
</rdf:Description>
```

A.6 Requirements

This section contains the approved requirements document used to guide development of this specification. This Section is informative.

Introduction

The purpose of this document is to define requirements for the UAPROF Drafting Committee. These requirements are based on WAP and W3C input papers, discussions in meetings and mailing list, and UAPROF Drafting Committee Charter.

This document also guides the Drafting Committee in their specification work. Ultimately, the Drafting Committee will address all of the requirements here but the Committee reserves the right to prioritise requirements in the final specification schedule.

The requirements presented here are in no particular order.

Note: This is a DRAFT document, which represents work in progress. The requirements identified here are not final and are subject to change.

Definitions, Abbreviations, and Acronyms

UA	*User Agent: A program that acts on a user's behalf.*
UAPROF	*User Agent Profile: A set of attributes and associated data elements that describe the end user device hardware and software characteristics (metadata) as well as the user's personal preferences.*
WAP gateway	*A network element that is responsible for encoding and decoding data transferred from and to the mobile client via the WAP protocol (WDP/WTLS/WTP/WSP) stack. [REF???]*
CC/PP	*Composite Capabilities / Preferences Profile: A user side framework for content negotiation*
OTA	*Over The Air*
P3P	*Platform for Privacy Preferences Project: An architecture that enables trusted and informed online interactions between a user agent and a service.*
CPI	*capability and preference information*

References

[WSP]	*"Wireless Session Protocol", WAP Forum 1.0 Specifications, 04/98, http://www.wapforum.org/docs/technical.htm*
[UAPM]	"UAPROF Working Group Meeting minutes", Malmo, 10/98 http://www.wapforum.org/member/archives/wap-uaprof.html
[P3P]	"Platform for Privacy Preferences Project," W3C, 11/98, http://www.w3.org/TR/WD-P3P/
[P3Pa]	Johan Hjelm, "P3P for Position Dependent Privacy", Input Paper Malmo Meeting, 10/98, http://www.wapforum.org/member/input/archive/ARTICLE-P3P-location.htm
[CCPP]	Franklin Reynolds, et. al. "Composite Capability/ Preferences Profiles: A user side framework for content negotiation", W3C Note, 11/98, URL:http://www.w3.org/TR/NOTE-CCPP/

[HTTPX] H. Frystyk Nielsen "HTTP Extension Framework for Mandatory and Optional Extensions", Internet Draft 08/98

[CHTR] "Wireless Application Group: UAPROF Drafting Committee Charter", WAP Forum, 12/98, http://www.wapforum.org/member/wg/wag/Activities/ActivityUAPROF/UAPCharter.htm

[MExE] include reference to the ETSI/ MExE documentation

Design Assumptions

UAPA: 4-1

- Implicit in the requirements and the architecture is an assumption of the existence of a WAP gateway function in the network.

UAPA: 4-2

- CCPP [CCPP] will support the WAP UAPROF efforts.

UAPA: 4-3

- The WTLS layer of the WAP stack shall be used if secure transmission of UAPROF is desired.
- Each WSP session is bound to a single WAP gateway.

UAPA: 4-3

- A client may participate in multiple WSP sessions simultaneously, each possibly with a different WAP gateway.
  - ➢ e.g. Secure/insecure sessions
  - ➢ e.g. Multiple bearers

Design Goals

UAPG: 5-1

- The User Agent Profile (UAPROF) framework shall ultimately enable delivery of content in a format tailored to the specific device characteristics, application settings, operating environment and user preferences.

UAPG: 5-1-1

➢ In doing the above, the UAPROF should enable reduction of the network resources otherwise required to deliver content to the device.

UAPG: 5-2

- For this purpose, the UAPROF data model shall adequately represent the CPI of the WAP client device, operating and network environment, user agent/application settings, as well as the user's preferences.

UAPG: 5-3

- The framework for UAPROF shall provide the ability to transmit the CPI across the wireless network (from the device to the WAP gateway) in a flexible, yet efficient and optimum manner that minimises round-trip delays, network bandwidth, and number of messages exchanged.

UAPG: 5-3-1

➢ The content of the CPI shall be composed with the above objectives in mind.

UAPG: 5-3-2

➢ OTA transmission mechanism and encoding of the CPI shall be optimized as mentioned above.

UAPG: 5-4

- The CPI may change during the course of a communication session between a client and the wireless network and the Internet, and those changes will be made effective for the remainder of that session.

UAPG: 5-5

- Information in the UAPROF framework shall be semantically compatible with established capability description vocabularies, to enable speedy adoption and ease of use by developers.

UAPG: 5-6

- The format and communication mechanism of the UAPROF information shall support and be compatible with existing and emerging Internet standards for the desktop environment.

Requirements

UAPR: 6-1

- The UAPROF shall support any client application that uses the WSP network stack..

UAPR: 6-2

- The CPI generated by the device will be communicable to an Internet-based origin server.

UAPR: 6-2-1

➢ To the extent that the OTA CPI is encoded, a WAP gateway will be able to map that CPI to a standard representation for Internet transmission.

UAPR: 6-2-2

➢ When transmitted over the Internet, the CPI will be represented within HTTP headers, and this transmission will be HTTP1.1-compatible.

UAPR: 6-3

- CPI shall be available to each gateway with which the device communicates, though each WSP session may be associated with a different CPI.

UAPR: 6-4

- An initial set of elements that comprise the CPI shall be defined. Additionally, the UAPROF data model shall be extensible to allow for rapid and easy adoption of new features and capabilities of the client and preferences of the user.

UAPR: 6-5

- The UAPROF framework shall support the ability to reference capability information stored separately on various systems and databases. Specifically,

UAPR: 6-5-1

➢ The UAPROF shall be extensible to dynamically compose CPI located at several sites in the network, as well as information transmitted by the device.

UAPR: 6-5-2

➢ The UAPROF framework shall support an indirect addressing scheme based on RFC 2396 for referencing profile information.

UAPR: 6-6

- The UAPROF framework will enable the WAP gateway to receive and cache the CPI for every user agent on every WAP device, across requests within a particular communication session.

UAPR: 6-6-1

➢ For situations where the WAP gateway is caching the CPI, the UAPROF framework will provide a lightweight exchange mechanism that permits the client to avoid resending the elements of the CPI that have not changed since the last time the information was transmitted.

UAPR: 6-6-1-1

- Addressing this requirement may require changes to WSP to handle incremental update of a header.

UAPR: 6-6-1-2

- Alternatively, addressing this requirement may require mechanisms to explicitly post incremental changes to the cached CPI.

UAPR: 6-6-2

➢ For situations where the WAP gateway is caching the CPI, the UAPROF framework shall ensure consistency of the cached information under all device states and modes of a WSP session (e.g. based on the WSP state transitions). Specifically when,

UAPR: 6-6-2-1

- The device is power cycled.

UAPR: 6-6-2-2

- The device is temporarily out of coverage (roaming-> out of coverage -> back into coverage scenario).

UAPR: 6-6-2-3

- There is no currently active session or the active session has been lost.

UAPR: 6-6-2-4

- The session is in a suspended state

UAPR: 6-6-2-5

- The current session is active

UAPR: 6-6-2-6

- During session set-up

UAPR: 6-6-2-7

- During session resumption

UAPR: 6-7

- The UAPROF framework shall support client-initiated changes to the CPI during a communication session between a client and the wireless network and the Internet.

UAPR: 6-8

- The UAPROF framework shall support the addition, deletion, or modification of the CPI by intermediate network elements and proxies within the client-to-origin server path.

UAPR: 6-8-1

➢ The UAPROF framework shall include a set of recommended CPI modification policies.

UAPR: 6-9

- To enable communication of UAPROF information to the WAP gateway, the device must,

UAPR: 6-9-1

➢ Generate CPI.

UAPR: 6-9-2

➢ Encode the CPI into a format that satisfies the requirements of the low bandwidth/high latency bearer networks currently supported by the WAP specifications.

UAPR: 6-10

- The UAPROF framework shall support internationalisation as required.

UAPR: 6-11

- A client will be able to embed CPI elements in a request to be applied to that request (in addition to information derived from the cache, if any, in the WAP gateway), without affecting the contents of the cache, if any, at the WAP gateway.

UAPR: 6-12

- The UAPROF specification will explicitly list the assumed features provided by external standards, including, but not limited to, CCPP, RDF, and HTTP.

Desirable Features

UAPF: 7-1

- Expand WBXML to support encoding of RDF based CPI.

UAPF: 7-2

- A trust model for network elements, gateways, and proxies.

Open Questions

1) Network may change CPI

➢ Notify client or simply override client?

- e.g. network operator turns off sound preferences on behalf of user

2) Impact of UAPROF on content caching

➢ Introduce new “cache-able independent of uaprof” tag in response

3) Separation of capabilities and preferences within CPI schema

A.7 Acknowledgements

This specification benefited from the participation of many participants of the WAP Forum's UAProf working group and various reviewers in both the W3C and WAP Forum communities. We are particuarly indebted to the following individuals:

- Nigel Byrnes, Philips; lead on scope and limitation
- Doug Dominiak, Motorola; lead on WBXML encoding
- Johan Hjelm, Ericsson/W3C; co-lead on gateway behavior and Internet interoperability
- Mikael Nilsson, Ericsson; lead on definitions and Static Conformance Requirements (SCRs)
- Hidetaka Ohto, W3C; co-lead on gateway behavior and Internet interoperability
- Franklin Reynolds, Nokia, CC/PP and RDF compliance
- Sandeep Singhal, IBM; document editor, end-to-end architecture, and Chair of WAP UAProf drafting committee
- Dwight Smith, Motorola; lead on deployment
- Peter Stark, Phone.com; lead on CC/PP-WSP protocol
- Lalitha Suryanarayana, SBC Technology Resources; lead on requirements and profile schema and vocabulary definition
- Yaacov Weingarten, Converse; liaison to Push group
- Steve Wingard, Spyglass, working group secretary, assistance with schema vocabulary, and liaison to Protocols group