EMPOWERING WIRELESS UPNP DEVICES WITH WEBPROFILES

JUAN IGNACIO VAZQUEZ AND DIEGO LOPEZ DE IPIÑA

Deusto University,
Avda. Universidades, 24,
48007 Bilbao, Spain
E-mail: {*ivazquez,dipina*}*@eside.deusto.es*

Our surrounding environment is changing day after day. Almost in an unperceivable way, even though steadily, more and more little computing and communicating devices are populating our homes, workplaces, clothes, streets or cars. All these devices need a common architecture to communicate, self-organise and cooperate, being one of such architectures Universal Plug and Play (UPnP). Wireless UPnP is the appropriate technology for mobile devices that roam around, creating and partaking in ad-hoc networks emerging everywhere. But UPnP still uses an interaction model with the environment not suitable for present users' needs that require more intelligence around them, as stated in the concept of Ambient Intelligence (AmI). Issuing commands continually such as open the door, turn on the light, or play the movie using some kind of universal controller such as a PDA or mobile phone, only relieves the user from having physical contact with the device, but not from completely free him from directing and coordinating the action. In this paper we propose the use of the WebProfiles model to extend UPnP capabilities enabling wireless UPnP devices to act in response to user's preferences, adapting the environment without being explicitly commanded, and so, getting closer to the new, more subtle interaction model with the activated world.

1. Wireless UPnP

Universal Plug and Play [1] is a standard that describes an architecture for connecting and communicating devices, most of them wireless-enabled. It is strongly based on TCP/IP and Web technologies, mainly HTTP, XML and derived protocols such as SSDP (Simple Service Discovery Protocol), GENA (General Event Notification Protocol) and SOAP (Simple Object Access Protocol). HTTP over UDP, both in unicast (HTTPU) and multicast (HTTPMU) flavours, is also used as a substrate for SSDP communication.

UPnP relies on a zero-configuration auto-descriptive model where no drivers are needed to interact with the devices, but discovering and standard interaction mechanisms are applied to achieve a universal "invisible" networking system. Wireless technologies are at the core of UPnP since they are the selected communication alternative for user-agents (PDA, mobile phones) and for many other UPnP powered devices.

1.1. *Adapting the UPnP environment*

In a wireless UPnP scenario, a user can make use of a control point powered wireless PDA to discover surrounding devices and interact with them switching the TV channels, checking the heating, validating the identity at the door and so on. The control point acts as a user-agent or proxy that represents the actual user when interacting with an UPnP powered environment. The main UPnP mechanism perceived by the user when performing these tasks is control: a very active control where the user has to command the actions via the PDA interface, graphical or voice-sensitive. This kind of interaction only relieves the user from physically performing the task over the involved devices, but all the previous phases of thinking what to do, which devices are involved, selecting them and invoking the actions must be performed both mentally and physically over the PDA interface.

The outcome is that the whole process of adapting the environment for user's preferences slows down making highly undesirable to use UPnP wireless technology when he enters home and want to have the present devices (heating, TV channel, lights) configured for his profile. Of course, for concrete actions the user must interact and invoke concrete operations over devices explicitly, but it would be desirable to find a way for automatically configure the environment, and thus, achieving a true Ambient Intelligence (AmI) scenario: interactions become invisible and unperceivable for the user, but they exist and tasks are performed silently. In an AmI scenario, the user enters a room and is identified, heating is automatically configured for his preferences and, if present, the TV switches to his preferred show at this time. No action has been explicitly commanded but adaptation has been performed.

2. Passive Influence and Context-Aware Scenarios

The previous examples illustrate how a concrete agent can influence the environment, and thus, its constituent agents' state (devices), via active or passive methods. Active methods are those in which the agent explicitly commands other agents to change their state or perform an action. Example: as a user enters the building, a sensor identifies him and commands the elevator to come and get him in. When the user stops at the room door his mobile phone commands the electric lock to open.

Active methods can be implemented using any of the well-known distributed computing technologies such as CORBA [2], SOAP (Simple Object Access Protocol) [3], OBEX, etc. In UPnP, strongly based on XML technologies, SOAP over HTTP is used for representing invocations back and forth between control points and devices.

Passive methods to influence the environment are those in which an agent disseminates certain information, expecting that other agents change their state or perform an action at their discretion to create a more adapted environment. Using passive methods an agent does not command the target agents to do anything concrete, it simply publishes/broadcasts information preferences expecting the others react changing their state in a positive way. We can state that passive mechanisms are not intrusive, but they are less predictable. The particular set of information to disseminate by the agent is dependant of the configuration of the environment in which is going to be published. Example: a user behavioral profile can be formed by thousands of different parameters, but only a subset of those are required to adapt an hotel room (with TV set, telephone, temperature and lights) to his preferences.

Anyway, an agent must be aware of the surrounding environment to identify and disseminate the proper information that can influence the neighbor devices in the desired way. Active and passive methods are complementary. Active methods perform in a master-slave way, where advanced smart features in agents are not required except for authorization processes. Usually, smart environments are based only on "command and control" mechanisms that centralize intelligence in only one or few agents that control a greater number of "dummy" entities.

2.1. *UPnP Passive Interaction*

UPnP covers quite well the active methods functionality using SOAP over HTTP to implement active control for devices and adaptation. No passive alternatives are provided, which in most of cases would simplify user's behavior, without worrying about how to interact. Passive methods can be also coordinated with active ones to provide additional information for the device when performing a task. That additional information creates some kind of background for performing the desired process, not forcing but suggesting.

For example, an active invocation such as "switch on the TV" can be complemented by passively disseminated information representing "these are my favorite shows". The active command is the former and the passive suggestion for a better adaptation of the task is the latter. Passive interaction mechanisms allow devices to know user's profile when carrying out an action, probably performing it in a more adapted way. In our research we have found that UPnP can be extended by passive mechanisms enriching its features, without interfering with the existing behavior and creating and interaction model fully compatible with traditional UPnP devices.

3. Wireless UPnP Extended With WebProfiles

3.1. *Introduction to WebProfiles*

In order to add passive influence capabilities to the HTTP protocol we have developed the WebProfiles interaction model. It is a non-intrusive mechanism that enriches HTTP with passive interaction capabilities if supported by the communicating entities.

The goal of the WebProfiles model is to provide an HTTP-based mechanism to negotiate and exchange contextual information that can be used for the client to obtain more adapted web results. The client is the unique entity that manages the contextual information repository, providing the authorized services with the appropriate subset to generate adaptation. The client repository stores user-related profiles on different knowledge domains, being several profiles allowed and applicable to different scenarios. The point with the WebProfiles model is that the context information is not statically structured and composed, but it is dynamically generated depending on the situation by selecting and grouping the convenient profiles and forwarding them to the service provider. The elements that define the situation and, thus, influence the selection of profiles are the profiles themselves, the service provider data, and the user's established permissions about profile information access.

All these entities' data serve as criteria to negotiate and exchange the context information with the service provider, and so, set up the environment for further services execution. The main involved structure is the WebProfile: an XML document representing user preferences on some domain(s) under certain conditions from the same or other domain(s), via a language called WPML (WebProfiles Markup Language).

It is out of the scope of this paper to detail thoroughly the conformation of WebProfiles and how they are generated or changed either by the client (user) or the server (service). It is quite evident that they constitute the context information about user preferences available to authorized services for adaptation.

For example, in a simple Ambient Intelligence scenario, the client disseminates appropriate WebProfiles to present servers, which use them to adapt different services (light and temperature control, TV programs presentation, presence information, among others) in order to provide a more suitable user experience. The servers probably follow a passive influence model as detailed in [4]. The information contained in the WebProfiles is very dependent on the services involved, since they represent the context information understandable by those services, but it is expected to be standardized via XML Schemas or Semantic Web technologies.

3.2. *WebProfiles Negotiation*

The WebProfiles model defines an HTTP-based negotiation mechanism that allows both client and service providers to set up the context in which further interactions can be performed. The most remarkable phases within this negotiation process involve notification of negotiation capabilities, knowledge domains, WebProfiles selection and delivery, and service adaptation. The figure 1 illustrates the negotiation process at a higher level, stressing the sequence of tasks each party must accomplish.

The detailed description of each step is: (1) The client issues a normal request to get some resource from the service provider. (2) The service provider processes the request and sends back the resource along with information about the types of adaptation available for this and future requests, indicating the supported domains structures about which preferences can be processed to generate a more adapted response. Service Credentials are sent, so the client can verify whether the service provider is authorized to receive the WebProfiles information in order to perform adaptation. If the client does not support WebProfiles, or it does not validate credentials or it does not require adaptation for this service, the negotiation process ends at this point as if it was a normal finalization without WebProfiles. (3) If the client demands service adaptation, it checks the presence of suitable WebProfiles with preferences about the declared domains, in order to create a candidate list of WebProfiles for the service. (4) The client filters the list of candidate WebProfiles against the Service Credentials supplied by the service provider, and thus obtaining the final list of validated WebProfiles suitable for that concrete service adaptation. (5) The client issues the

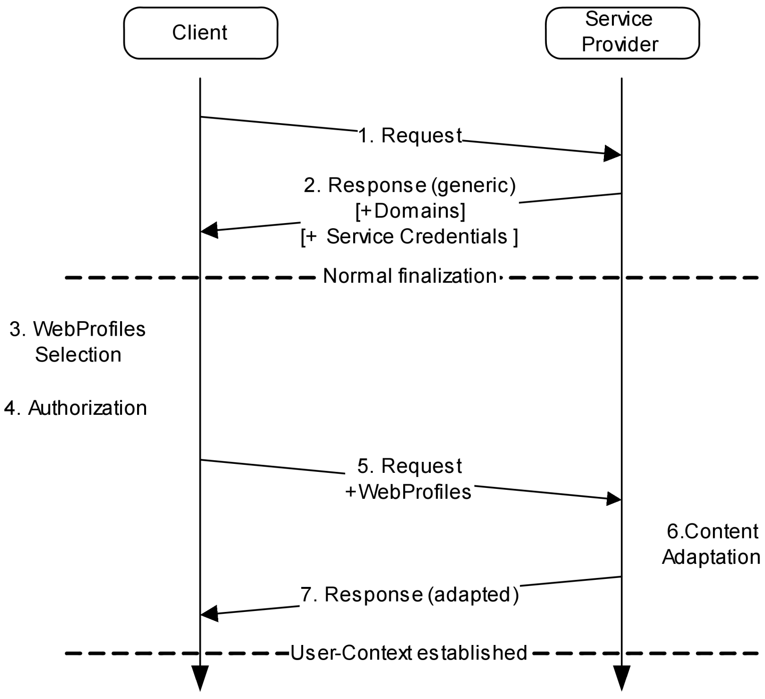

Figure 1. The WebProfiles negotiation process.

original request adding the validated WebProfiles. (6) The service provider uses the information conveyed in the received WebProfiles to better know the client and adapt the responses. (7) The service provider generates the corresponding response to the request, conveniently adapted by means of the WebProfiles. Now, the contextual information between the client and the service provider is established for further interactions, allowing even dynamic modification by sending WebProfiles updates.

This interaction model illustrates the process of contextualization via WebProfiles. In the case context information is not needed or WebProfiles are not supported either by the client or the service provider, the interaction finishes at step 2 and the overload is minimal in relation to the normal process.

Only if WebProfiles are applicable and agreed by both parties, a further interaction is required where WebProfiles are exchanged in an overall process that resembles HTTP Basic Authentication [5], in the sense that the client is the responsible for resending the original request extended with additional information to obtain a preferred response (client-driven negotiation). In fact, this resemblance is not casual. The WebProfiles model has been designed in such a way that shares many similarities with existing HTTP mechanisms in order to be easily integrated within the hypertext protocol. Nevertheless, the WebProfiles negotiation model does not follow an strict client-driven or server-driven negotiation model as specified in [6], but it shares hybrid characteristics with both of them.

3.3. *UPnP Messages with WebProfiles*

WebProfiles can be applied mainly during two different processes of the UPnP interaction: description and control. During description, the user agent acting as a WebProfiles client can suggest adaptation from the device when obtaining its description information. The interaction follows the general WebProfiles negotiation process described previously and the interaction is illustrated by the following (not complete) messages:

```
UPnP User Agent (Control Point)                              UPnP Device
Request →                                                     ← Response
--------------------------------------------------------------------------

GET description_uri HTTP/1.0
WP-Version: 1.0

                                                          HTTP/1.0 200 OK
                                                          WP-Version: 1.0
                                WP-Accept: text/vnd.webprofiles.wpml+xml;
                cnf-1="http://www.webprofiles.org/dataschemas/ambient"

      <!-- Description XML with generic content: Temperature control
                                                        device ready -->

POST description_uri HTTP/1.0
WP-Version: 1.0
WP-Activate: urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

--multipart_separator
Content-Type: text/vnd.webprofiles.wpml+xml
WP-Content-URI: urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

<!-- Content of the WebProfile with
urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6 -->
--multipart_separator--
```

```
                                                 HTTP/1.0 200 OK
                                                 WP-Version: 1.0
    WP-Collection: urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6;
                                                     max-age=300

<!-- Description XML with adapted content: Temperature configured
      between 20°C and 30°C. Temperature control device ready -->
```

The user agent asks for the device description fulfilling the UPnP description process. The device generates the HTTP response normally, but it adds a WP-Accept header indicating the list of WebProfile configuration domains from which WebProfiles are accepted (in this example, it accepts adaptation of ambient conditions: temperature, music, lights, ...). If the user agent keeps WebProfiles with information about those domains in the repository and user's permissions allow delivering them to the involved device, a second interaction is carried out, where the user agent re-sends the original request along with the appropriate WebProfiles to perform adaptation. The unique identifiers of those WebProfiles are listed in the WP-Activate header and the WebProfiles contents are embedded in the body of the HTTP POST request (in this case it contains preferences about temperature). Finally, the device sends back the response again confirming that adaptation has been performed using the WebProfiles whose unique identifiers are listed in the WP-Collection header, and it will last for 300 seconds (max-age parameter), allowing renewal. Example: when the user enters a room, his PDA (user agent) acting as an UPnP control point discovers a surrounding UPnP heating service using the UPnP defined mechanism and obtaining the description for it. During this process, the PDA negotiates adaptation with the heating service re-sending the description request with the appropriate WebProfiles containing user preferences about temperature. The perceived result is that the heating service meets user preferences with neither explicit human intervention nor action invocation.

During control, the user-agent can invoke an operation on the device supplying WebProfiles in a second message if supported by the device. Again the mechanism is similar to that on the description phase, but now the adaptation scope (the request URI) is only the invoked service and not the overall device state as with description. Example: the user interacts with his PDA in order to turn on the TV invoking the appropriate action. The PDA (user agent) acting as an UPnP control point negotiates with the UPnP device (TV set) the WebProfiles that can contextualize the action, maybe sending WebProfiles with user's TV preferences information. The perceived result is that the TV turns on at the appropriate channel to meet user preferences, without human explicit intervention.

4. Conclusions and Future Work

WebProfiles are a suitable mechanism to extend the wireless UPnP architecture in order to create passive influence scenarios, fully compatible with traditional UPnP mechanisms. The user interacts with the environment in a completely free manner, while his user agent adapts the surrounding devices in an unperceivable way, preparing them for further active invocations.

The WebProfiles model is an extension to HTTP with minimal interference with traditional HTTP parties, since the whole negotiation is carried out by the means of added HTTP headers that can be silently ignored by no-supporting entities. In this way WebProfiles enabled control points can interact with traditional UPnP devices as well as traditional control points can communicate with WebProfiles enabled devices.

Wireless UPnP architecture can incorporate WebProfiles as a passive influence mechanism, creating smart and adaptable environments that take advantage of the flexibility provided by wireless communications extending user influence around him in an invisible way.

Acknowledgements

This work has been partially supported by the Cathedra of Telefonica Moviles at Deusto University, Bilbao, Spain.

References

1. UPnP Forum. *UPnP Device Architecture1.0.* UPnP Forum (2003).
2. Object Management Group. *Common Object Request Broker Architecture (CORBA/IIOP). Version 3.0.2.* Object Management Group (2002).
3. World Wide Web Consortium. *Simple Object Access Protocol (SOAP) 1.1.* World Wide Web Consortium (2000).
4. J. I. Vazquez and D. Lopez de Ipiña. *An Interaction Model for Passively Influencing the Environment.* Adjunct Proceedings of the 2nd European Symposium on Ambient Intelligence, Eindhoven, The Netherlands (2004).
5. J. Franks et al. *RFC 2617: HTTP Authentication: Basic and Digest Access Authentication.* IETF RFC (1999).
6. R. Fielding et al. *RFC 2616: Hypertext Transfer Protocol – HTTP/1.1.* IETF RFC (1999).