SOAM: An Environment Adaptation Model for the Pervasive Semantic Web

Juan Ignacio Vazquez, Diego López de Ipiña, and Iñigo Sedano

MoreLab - Mobility Research Lab
University of Deusto
Avda. Universidades 24, 48007 Bilbao, Spain
{ivazquez, dipina}@eside.deusto.es, isedano@tecnologico.deusto.es

Abstract. Nowadays, there is a major interest in applying Web and Semantic Web techniques for the creation of pervasive computing scenarios, where devices and objects communicate using these technologies. The Web model has largely proved validity both in Internet-wide and intranet scenarios, but it is starting to be applied in personal area networks as a communication and knowledge reasoning system.
In this paper we present SOAM, an experimental model for the creation of pervasive smart objects that use Web and Semantic Web technologies in new ways – resulting in the novel concept of Pervasive Semantic Web – for enabling personal area semantic communication and reasoning processes in order to provide environment adaptation to user preferences.

1 Introduction

The ultimate goal of Ambient Intelligence [1] is to create intelligent spaces to empower users in everyday tasks at home, work, street, vehicle and others. In this vision, environments are proactive perceiving users' surrounding information, often referred as context, and reacting in the appropriate way to facilitate user's activities. In fact, more and more designers are starting to think that the most valuable resource in such environments are not computing power, communication or storage capabilities, but user interaction [2].

Since intelligent objects are not isolated and should not act in their own, some kind of infrastructure, communication and reasoning model must be provided to guarantee that coordination and organisation activities among those objects are performed to facilitate users' experience.

Traditional Web technologies, such as HTTP, HTML and XML, have been used for providing presentation and control mechanisms in pervasive computing environments, creating a sort of Personal Area Web for interacting with devices (e.g. UPnP [8]). We think that these models can be augmented with Semantic Web technologies to provide reasoning processes at the devices and at the environment itself.

In this way, full intelligent environments can be created where reasoning processes are automatically performed, communicated and agreed upon between intelligent objects, based on users' context perceptions and other available inputs. The goal is to adapt the environment without user intervention [9] [11] (maximizing intelligence by minimizing explicit user interaction).

In this paper, we present SOAM – Smart Objects Awareness and Adaptation Model –, a proposal joining the forces of well-known Web-based communication mechanisms with intelligent capabilities provided by Semantic Web technologies in a structured manner, so that smart objects, or *smobjects* the term we coined, can be easily designed to create user-aware adaptive intelligent spaces following simple principles. SOAM is a preferences-based environment adaptation model in which part of user's context is built up by semantic preferences about environmental conditions. These preferences lead to adaptation on smobjects without explicit user intervention, yet allowing automatic reasoning over those preferences.

There are a number of technologies and initiatives related to this field that constituted the background for our work, such as Universal Plug and Play (UPnP) [8], Task Computing [4] and SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) [3].

In section 2, the smobject concept is detailed as well as the exchanged information structures, while in section 3 the SOAM adaptation model and involved entities are described. Finally, in section 4 some future research directions are given.

2 Pervasive Smart Objects

2.1 The Pervasive Semantic Web Vision

We coin the term *Pervasive Semantic Web* to designate the result of applying Semantic Web technologies to Pervasive Computing scenarios in order to perform reasoning processes. The main representatives of those technologies are RDF (Resource Description Framework) [6] and OWL(Ontology Web Language) [7].

These scenarios are populated by different kinds of devices with a number of capabilities such as temperature sensing, video capturing, door opening, and so on. Our strategy is based on using ontologies to represent knowledge about different domains, so that we use appropriate ontologies for temperature, physical access control, location and so forth.

In our vision, we pretend to create some kind of Personal Area Web, where devices are interconnected, hosting knowledge about environmental perceived conditions and using references to link resources inside and outside this space. This vision determines the creation of a new type of logical environment in ubiquitous computing scenarios: a personal area semantic web with information flows back and forth among communicating devices, sharing their knowledge about users inside the environment and coordinating their tasks via distributed reasoning procedures in order to provide an ambient intelligence experience.

This point of view about future living and working environments is shared with other research groups that provided similar viewpoints [5], but until now there are no practical results or architectures developed and tested.

SOAM – Smart Object Awareness and Adaptation Model – is intended to fill this gap by providing a comprehensive architecture, easily replicable, to test automatic environment adaptation scenarios by applying semantic annotation techniques.

2.2 Smobjects: Smart Objects

In SOAM a major entity of the architecture is what we denominate *smobject* as a short for "smart object". A smobject is an agent in the form of a piece of software, representing an intelligent device, several devices or event part of a device. Smobjects capture a subset of environmental conditions, provide that perceived context information under request, and act upon the same or other subset of environmental conditions in order to modify them and adapt them as needed. Smobjects need to access built-in sensors, effectors, communication ports, maybe storage facilities if available on the device, and so forth.

The real strength of the smobject-based agent architecture in SOAM is that standardizes the way sensors and effectors information is represented and accessed. Smobjects manage three different types of information structures that illustrate the functions a smobject can carry out:

- **Context Information**: smobjects provide information about perceived state of the environment via semantically annotated data under request. Context Information is gathered through built-in sensors in the bounded device and provided by the smobject to requesting parties.
- **Capabilities**: a smobject is capable of perceiving only some concrete environmental conditions depending on the bounded-device built-in sensors, and it is also capable of operating over some (same or other) conditions depending on the bounded-device built-in effectors. Perception and operation capabilities are provided by the smobject to requesting parties.
- **Constraints**: smobjects can be influenced by other entities using some data constructions called constraints, which declare valid ranges on the desired state of the environment, so that the smobject is in charge of driving adaptation honouring them. Smobject's behaviour is defined by active constraints, which represent existing influences over the smobject, and have a limited lifespan. Smobjects can provide information about their active constraints to requesting parties, as well as accept constraints from other entities that desire to influence the smobject's behaviour.

In SOAM, these data entities are exchanged through smobject standard communication interfaces as shown in the figure 1. We can also notice how the smobject interacts with the host device, using their built-in sensors and effectors.

Capabilities and Constraints are represented and exchanged in XML using structures declared in a grammar called SOAM Datatypes XML Schema, while Context Information is represented using RDF and domain ontologies honouring the OWL specification. Standard HTTP is used for transport and negotiation purposes between other entities and the smobjects in order to retrieve and store

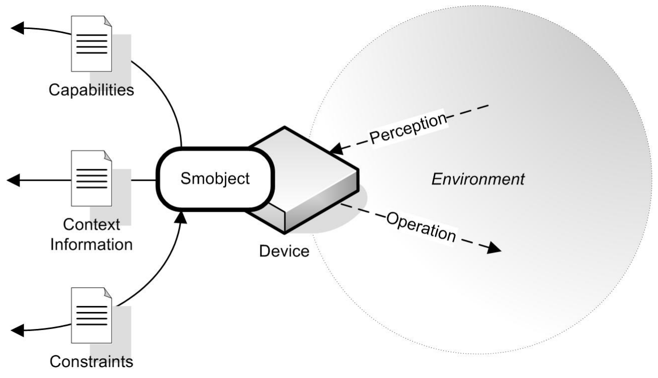

Fig. 1. Smobject communication interfaces.

these information structures in SOAM (HTTPS could be used to facilitate a secure communication channel, providing every smobject has a valid and trustable X.509 certificate).

2.3 Context Information

Context Information is probably the most important data a smobject can provide. Context Information is constructed using RDF, serialized in the form of XML. It conveys perceived information captured through device's sensors, annotated via RDF and OWL. Captured data semantics is highly knowledge domain dependent, for example temperature measures, an item location or an elevator's present position.

Since devices are specialized in domains (TV, temperature control system, light), smobjects act as control processes deeply associated to the concrete device to act upon built-in sensors and effectors and programmed to semantically annotate the perceived data using the most appropriate and standard ontology for that purpose. It is up to devices' and smobjects' designers to select suitable ontologies among available ones.

An example of a Context Information message conveying knowledge about luminance is shown if figure 2. Probably, the smobject is installed in a lighting device called `light1`, and it provides information about `light1`'s state upon request (luminance, light color, ...).

As we can notice, a smobject normally does not only provide information about perceptions obtained by sensors, but also the device identification and type, that is, the full semantic description of available data. This annotation is particularly useful to automate processes depending on device identification, type or other device parameters.

```
<rdf:Description rdf:about="urn:uuid:light1">
  <lit:luminance rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
    30
  </lit:luminance>
  <lit:color rdf:resource="http://www.awareit.com/onto/light#White"/>
  <rdf:type rdf:resource="http://www.awareit.com/onto/light#Light"/>
</rdf:Description>
```

Fig. 2. An example Information structure provided by a smobject.

2.4 Capabilities

A smobject can exhibit perception capabilities on some domains and operation capabilities on the same or different domains. Perception capabilities represent sensing mechanisms the smobject is able to access on the host device about some domains (for example, lighting conditions), while operation capabilities represent control mechanisms the smobject features about some domains.

For example, a light sensor has perception capabilities about the "lighting domain" in a room, while a switch has operation capabilities about the "lighting domain" (via a lamp o light bulb). Usually, both devices would be modeled using the same "lighting control system" smobject, thus featuring at the same time perception and operation capabilities about the "lighting domain".

Capabilities are generally not only bounded to a knowledge domain, but also to concrete elements to which the information is related. For instance, a smobject can perceive lighting conditions, but only those related to `light1`. Or maybe, the electronic thermometer smobject measures the existing temperature, but only in `room1`.

Of course, some smobjects can measure conditions related to undefined actors, or unbounded at all. The SOAM Datatypes XML Schema defines data structures to declare perception and operation capabilities, using even wildcards to denote the "any" concept.

Figure 3 is an example of the capabilities file of the previous lighting smobject with both perception and operation capabilities on a concrete light.

Basically, this document means that the smobject can perceive the state of `light1` in the "light" domain (with all the predicates included in the ontology) and it can also adapt `light1` dynamically in the same domain.

2.5 Constraints

Smobjects receive requests to perform environment adaptation through effectors. These requests come in the form of Constraints, represented by statement patterns in the desired behaviour. A smobject can receive a number of this kind of constraints over the time, so its behaviour is influenced and driven by them. In fact, a smobject is in charge of managing the active Constraints and trying to perform in such a way that Constraints are honoured.

```
<capabilitiesCollection>
  <perceptionCapability id="urn:uuid:light1_pcap1">
    <subject resource="urn:uuid:light1"/>
    <ontology resource="http://www.awareit.com/onto/light"/>
  </perceptionCapability>
  <operationCapability id="urn:uuid:light1_ocap1">
    <subject resource="urn:uuid:light1"/>
    <ontology resource="http://www.awareit.com/onto/light"/>
  </operationCapability>
</capabilitiesCollection>
```

Fig. 3. An example Capabilities structure provided by a smobject.

The SOAM Datatypes XML Schema provides a way to represent and exchange constraints. Those Constraints are generated by initial configuration settings and/or adaptation requests sent by other actors.

Figure 4 illustrates an example of active Constraints on `light1`.

```
<constraintsCollection>
  <constraint expires="PT1M"
      subject="urn:uuid:light1"
      predicate="http://www.awareit.com/onto/light#luminance">
    <objectLiteral datatype="http://www.w3.org/2001/XMLSchema#int">
      10
    </objectLiteral>
  </constraint>
  <constraint expires="PT1M"
      subject="urn:uuid:light1"
      predicate="http://www.awareit.com/onto/light#color">
    <objectResource ref="http://www.awareit.com/onto/light#Yellow"/>
  </constraint>
</constraintsCollection>
```

Fig. 4. An example Constraints information provided by a smobject.

The previous Constraint could be read as "*light1 must have a luminance of 10 and yellow color*"

Constraints are the unique out of the three data entities (Context Information, Capabilities and Constraints) that can be also injected into smobjects and not only requested from them. As explained previously, in SOAM, any actor can retrieve Constraints from the smobject to find out how its behaviour is being driven, but also existing actors can send Constraints to the smobject to constrain its behaviour and have the environment conveniently adapted.

Since HTTP messages are used in SOAM to negotiate information exchange, HTTP Basic Authentication [10] or other standard web mechanisms can be used for identification and authentication purposes if needed.

3 Environment Adaptation

3.1 Adaptation Profiles

The goal of SOAM is to achieve a comprehensive model for automatic adaptation of the environment to user preferences, needs and behavioural patterns. As shown, smobjects are the entities in charge of performing the final operations to achieve adaptation.

Adaptation Profiles are the information elements that conveys user's adaptation requirements that eventually drive smobjects behaviour. Adaptation Profiles are stored and exchanged with the environment via the user's personal device, which contains an Adaptation User-Agent in charge of negotiating Adaptation Profiles with surrounding entities as explained below.

An Adaptation Profile is a conditional preference or environment adaptation requirement that contains two different sections:

– **Preconditions**: represent existing requirements about the environment's present state, that must be met for the Adaptation Profile to activate. It makes the adaptation to have a conditional nature. Often, adaptation requirements are not fixed, e.g. a user does not need his preferred temperature to be always 22^{o}C, but maybe only when he is at the car.
– **Postconditions**: represent desired patterns in the environment's future state that must be met for the adaptation to be considered as honoured. Postconditions eventually generate constraints.

Variable substitution in Adaptation Profiles is possible to allow postcondition elements to be bounded to precondition elements as shown in figure 5.

This Adaptation Profile can be read as "*whatever the location Alice is in with an ambient light, that ambient light should have a luminance of 90*", which is a very simple but powerful mechanism to force every location's lights to adjust automatically as Alice gets in.

3.2 Other SOAM Entities

Despite smobjects play a fundamental role in the SOAM architecture, they just act as intermediates with the associated device to fulfil adaptation requests. There are some other entities needed in SOAM in charge of generating those requests on behalf of the user and instructing smobjects to adapt the environment in a coordinated way:

```
<adaptationProfile id="urn:uuid:prof1" expires="PT2M">
  <variable id="x"/>
  <variable id="y"/>
  <precondition subject="urn:uuid:Alice"
      predicate="http://www.awareit.com/onto/location#isLocatedIn">
    <objectVariable ref="x"/>
  </precondition>
  <precondition subject="x"
      predicate="http://www.awareit.com/onto/light#hasAmbientLight">
    <objectVariable ref="y"/>
  </precondition>
  <postcondition subject="y"
      predicate="http://www.awareit.com/onto/light#luminance">
    <objectLiteral datatype="http://www.w3.org/2001/XMLSchema#int">
      90
    </objectLiteral>
  </postcondition>
</adaptationProfile>
```

Fig. 5. An example Adaptation Profile with bounded variables.

- **Adaptation User-Agent**: a piece of software acting on behalf of a user that is aware of the user's Adaptation Profiles and negotiates with surrounding Orchestrators the adaptation process to exchange those profiles.
- **Orchestrator**: an entity that perceives and orchestrates existing smobjects in the environment to perform the adaptation process following Adaptation Profiles. Orchestrators feature semantic information reasoning and a rule engine in order to generate Constraints from Context Information and Adaptation Profiles.

Adaptation User-Agents act generally on behalf of a user, silently starting the process of adapting the environment by finding an available Orchestrator to which they send user's Adaptation Profiles.

4 Conclusions and Future Work

SOAM is an effort to create a comprehensive model for automatic environment adaptation to user's preferences, based on existing well-proven technologies such as SSDP, HTTP and XML, as well as the Semantic Web (RDF, OWL) for knowledge representation and reasoning. SOAM is based in a special kind of pervasive agents called smobjects that interface with real devices via a standard interface for exchanging information.

Our current prototype implementation is based in a single board computer in the role of Orchestrator, using Jena libraries for semantic information processing, and ARM9 embedded processors (UNC20) for smobjects. Adaptation

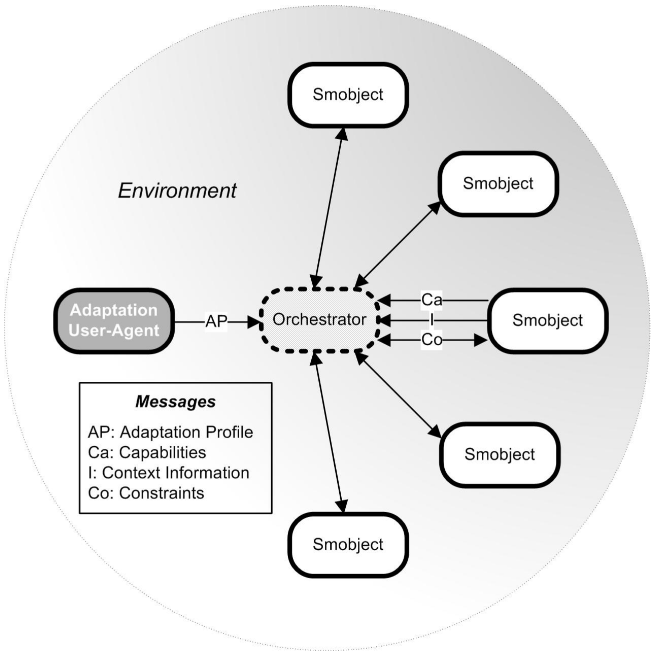

Fig. 6. Diagram illustrating SOAM architecture.

User-Agents can be implemented in PocketPC o cellular phones. Some experimental scenarios, related to home and intelligent workplace environments are being created to test SOAM feasibility and capabilities. SOAM can take advantage of standard ontologies, such as SOUPA, for concrete domain knowledge representation.

SOAM illustrates the possibilities of the new paradigm emerging from the joint forces of the Web and Semantic Web technologies applied to Pervasive Computing scenarios, creating Pervasive Semantic Webs everywhere and augmenting intelligence in environments.

There are some open research issues related to SOAM architecture that still need to be studied such as conflict resolution with multiple users' disjoint requirements, usage of standard orchestration languages, or the possibility of removing the Orchestrator element of the architecture to create true decentralized choreography among smobjects.

5 Acknowledgements

This work has been partially supported by the Department of Industry, Commerce and Tourism of the Basque Government under the SAIOTEK grant S-OD04UD02, and the Cathedra of Telefonica Moviles at Deusto University, Bilbao, Spain.

References

1. K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten and J-C. Burgelman. *Scenarios for Ambient Intelligence in 2010. Final Report.* IST Advisory Group. EC (2001).
2. Project Aura. http://www.cs.cmu.edu/ aura/
3. H. Chen et al. *SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications.* Proceedings of Mobiquitous 2004: International Conference on Mobile and Ubiquitous Systems: Networking and Services, Boston, USA (2004).
4. R. Masuoka and Y. Labrou. *Task Computing - Semantic-web enabled, user-driven, interactive environments.* WWW Based Communities For Knowledge Presentation, Sharing, Mining and Protection (The PSMP workshop) within CIC 2003, Las Vegas, USA (2003)
5. Ora Lassila. *Using the Semantic Web in Mobile and Ubiquitous Computing.* Proceedings of the 1st IFIP WG12.5 Working Conference on Industrial Applications of Semantic Web, pp. 19-25. Springer (2005).
6. World Wide Web Consortium. *RDF Primer. W3C Recommendation.* World Wide Web Consortium (2004).
7. World Wide Web Consortium. *OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation.* World Wide Web Consortium (2004).
8. UPnP Forum. *UPnP Device Architecture1.0.* UPnP Forum (2003).
9. 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).
10. J. Franks et al. *RFC 2617: HTTP Authentication: Basic and Digest Access Authentication.* IETF RFC (1999).
11. J. I. Vazquez and D. Lopez de Ipina. *A language for expressing user-context preferences in the web.* WWW 2005: Special interest Tracks and Posters of the 14th international Conference on World Wide Web (Chiba, Japan) pp. 904-905. ACM Press (2005).