Defining and Using Profiles to Personalize and Manage Reconfigurable Services

G. Bartolomeo, N. Blefari Melazzi, F. Martire , S. Salsano

DIE, Univ. of Rome "Tor Vergata", Rome, Italy

***Abstract*— As ICT services are becoming more ubiquitous, mobile and personalized, we are witnessing an increasing importance of two related needs: i) the necessity of defining, managing and storing user-related information, e.g. preferences, personal data, characteristics of terminals and devices, settings of network parameters and of application parameters, etc.; ii) the necessity of defining procedures and architectures allowing the users to exploit all this information, known on the whole as profile, in the most simple and intuitive way, without forgoing security requirements.**

The enhanced capabilities of this framework can be exploited: i) on the user side, to personalize services, to improve the portability of services over heterogeneous terminals and devices, to adapt services to available networking and terminal technologies; ii) on the network side, to give to operators more powerful tools to define new solutions for distributed, technology-independent, self-organizing, and autonomic networking systems. Such systems could be designed so as to be able to react autonomously to changing contexts and environments.

In this paper, we first review the current activities in this field, then we focus on our proposed approach for the definition of user profiles, device profiles and network profiles. Our solution is not limited to a "static" definition of profile information, but includes the procedures to use such profiles, logically organized in a suitable architecture. Some challenges deriving from security and privacy issues are also discussed.

***Index Terms*— service personalization, user profile, device profiles, network profiles, service reconfiguration.**

I. INTRODUCTION

Services, applications and operating systems need to be configured according to user preferences and may need to know information about the user. This results in the creation of user related data or "user profiles" specific for each service, for each application and for each operating system. It is a common experience, when accessing services on the web, to be prompted for the creation or the updating of a "user profile" for example with demographic information, addresses, phone numbers... This is already annoying when accessing services from the desktop PC. The advances in computing and communication technology are extending the number of devices where services can be accessed towards the so called "ubiquitous computing" or "pervasive services" scenario. The goal of pervasive/ubiquitous computing includes "anywhere, anytime" computing which allows users to access the same applications and information on their workstation in the office, on their computing devices at home and on a variety of mobile devices, when the users are on the move. Pervasive computing infrastructures should allow users to easily move their computational tasks from one computing environment to another, allowing at the same time to fully exploit capabilities and resources of their current environment, tailoring the services to the devices and networking facilities currently available.

In this pervasive computing scenario the need for profiles, and the complexity for the user to manage them is even more compelling. When services are invoked from and provided to different terminal devices, the service provisioning procedures may need to know details about the device currently being used; this could be done by means of a suitable device profile. In the same way, it could be useful to know the characteristics of the networking environment being used when providing a given service; also in this case, this necessity could be fulfilled by means of a suitable network profile. All in all, this means that it is necessary to deal with several user profiles, device profiles and network profiles.

In addition, this scenario is demanding also for operators, who have to devise and deploy tools and procedures to engineer and manage their networks, efficiently.

The availability of profile information is instrumental also in the definition and in the deploying of new solutions for distributed, technology-independent, self-organizing, and autonomic networking systems. Such systems could react autonomously to changing contexts and environments and be much more easy to install and to manage.

The operators' possible interest in the solution addressed in this paper could be twofold: simplify service fruition, on the user side, and service management, on the network side.

A key attribute of this approach is re-configurability, at various levels. Re-configurability was traditionally understood as operating at lower layers (e.g., software defined radio). However, to integrate different paradigms from the user point of view, it is necessary to break, as much as possible, the logical wires that still tie mobile users to networks and services, also at upper layers. This way, heterogeneous and mobile access networks can be really integrated, as IP has glued heterogeneous networks.

To go towards this full re-configurability, the Authors of this paper proposed a personalization approach, based on a user profile, in the framework of an EU co-funded project, named Simplicity [1].

In the same line, and with a broader perspective, another EU

co-funded project, named E2R [2], started to pursue the objective of devising and developing reconfigurable devices and supporting system functions to offer an extensive set of operational choices to users, application and service providers, operators, and regulators in the context of heterogeneous systems, from an end-to-end perspective.

The aim of this paper is to build on the user profile defined in the context of the Simplicity project, to expand its role, and to define a suitable architecture that can enrich its functionality. An important improvement is the exploitation of the profile concept in an autonomic perspective, enlarging the focus from an user-centric viewpoint to a more holistic and network-inclusive standpoint. This activity is being framed within a recently started EU co-funded project, E2R II [2].

We also stress that both the complete profile definition and its exploitation within an autonomic perspective and the related architecture are unpublished works.

As for the organization of this paper, we first introduce the main literature approaches for defining and using user profiles (Section II) and then we describe our own proposal for the definition of user profiles, device profiles and network profiles (Section III). An overview on the architecture and the procedures to use such profiles architecture in given in Section IV). We also discuss (Section V) the main challenges deriving from security and privacy issues, presenting both the state of the art on this matter and our ideas.

II. CURRENT ACTIVITIES ON PROFILE DEFINITION

A. *Single sign-on architectures*

An answer to the need of simplifying the handling of user profile is represented by the so-called "single sign-on" systems. Single sign-on is a session/user authentication process that allows a user to enter only one name and one password to access and use multiple applications and services.

Microsoft's "Passport" [3] single sign-on service is an example of the trend towards the use of Web-based single sign-ons. Microsoft Passport is a centralized service. According to the Microsoft Passport Web site, a consumer can use one name and password to sign in to all .NET Passport-participating sites and services. In essence, this is a centralized corporate identity system run by Microsoft and used by Microsoft customers and Microsoft business partners or other affiliates.

On the other hand, Liberty Alliance [4] is a consortium of vendors working on the development, deployment and evolution of an open, interoperable standard for network identity, where privacy, security and trust are preserved.

Differently from the logically centralized approach of Microsoft Passport, the Liberty Alliance Project relies on the concept of "federated network identity". Federated identity allows users to "link" elements of their identity between accounts without centrally storing all of their personal information. The Liberty Alliance has produced the Identity Services Interface Specifications – Personal Profile (ID-SIS PP), which defines schemas for profile information of a user.

B. *Device profiles: CC/PP and UAProf*

Work is ongoing in the World-Wide Web Consortium (W3C) to define a standard 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) [5][6] specification defines a high-level structured framework for describing this information. CC/PP provides the rules of how to construct a vocabulary that describes capabilities and preferences, but does not specify the actual attribute names and values.

The User Agent Profile (UAProf) [7] is a specific variant of CC/PP proposed by the Open Mobile Alliance. It is an application of CC/PP and therefore it inherits the syntax and semantics of CC/PP. The UAProf specification is concerned with capturing classes of device capabilities and preference information but is distinct from a user preference profile because is used only for content formatting purposes.

C. *The Generic User Profile defined by 3GPP*

The objective of the 3GPP Generic User Profile (GUP) [8] is to provide a conceptual description to enable harmonized usage of the user-related information located in different entities. The 3GPP Generic User Profile is a collection of user related data which inherits the way in which an user can exploit services in a standardized manner. The 3GPP Generic User Profile will be accessed either in a centralized or de-centralized way by user, subscriber, value added service provider and network operator with a standardized access mechanism. The 3GPP Generic User Profile will help to create an harmonized use of user data on one hand, while on the other hand will help to make it easier to find all user related data as a whole.

The GUP specifies the description of- and access of data in a standardized way. Actual content and format of subscription data, as well as the places (repositories) where subscription data are stored, may be different for different new services.

D. *Some details on the GUP architecture*

The GUP reference architecture, as shown in Figure 1, consists of:

- GUP Server;
- Repository Access Function (RAF);
- GUP Data Repositories;
- Rg and Rp reference points;
- Applications.

According to [8], the GUP Server is a functional entity that provides a single point of access to the Generic User Profile data of a particular subscriber. The GUP Server stores information about the GUP Components and the locations of data repositories of GUP Components related to each subscriber. The Repository Access Function (RAF) realizes the harmonized access interface. It hides the implementation details of the data repositories from the GUP infrastructure. The RAF performs protocol and data transformation where needed. The 3GPP GUP consists of a number of independent GUP Components and it uses the "Data Description Method"

[9] to describe in a detailed way how different profile components are specified, based on a XML Schema. Anyway, we stress that the data contents themselves are not described within the Generic User Profile, but only the data model and the schema is defined.

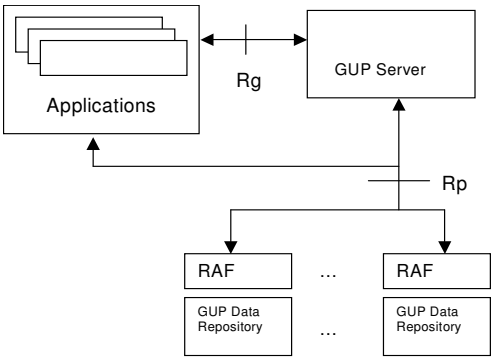

Figure 1: GUP reference architecture

III. Our Profile

In this section, we present our solution for profile definition. We start from the work carried out in 3GPP on the GUP but we extend that approach introducing three main requirements/characteristic:

1. we introduced a more distributed approach to handle the profile, in which an application that needs to retrieve (and to modify) profile information can have a more direct access to the profile itself, rather than having to pass through a centralized GUP server;
2. we tied the profile to a user's personal device, whose task is to store users' personal profiles and preferences, enabling transparent, easy and convenient customization of devices and services. We call this device Personal Identification Device (PID). Examples of PIDs may include a USB memory stick, a CompactFlash card, a Java card or a Bluetooth-enabled mobile phone. The PID interacts directly with terminals, via physical interfaces, and virtually with networks, through the terminals.
3. we identified a number of general concepts related to the user and her environment (the so called "Simplicity information model" see [11]) and began to fill out some profile contents; as a matter of fact, the abstract nature of the GUP is certainly useful and extensible, but a profile with no content at all risks to be too theoretic and impractical, and not useful for showcasing, testing and evaluation.

The resulting profile, named Simplicity User Profile (SUP) has been defined in the context of the Simplicity project, but it is general enough to be applied to a more general context and enriched with new and more advance features.

The proposed SUP includes five components (see Figure 2): user profile, device profile, network profile, service profile and PID profile. The idea is that the SUP is a "User Level" representation of the user himself and of his surrounding "information and communication technology" world. The SUP provides a logically unified representation of the information related to the user and of the "ICT" context in which the user is "embedded": the devices that he is using and that he owns, the services he has subscribed to, the network he is accessing or he could access. Figure 2 shows also that there is another level of representation, in addition to the "User Level" the "Simplicity Universe Level", which contains the description of all existing devices, access networks, services and simplicity devices. Suitable XML schemas have been defined in order to describe each component [10]. For each component we investigated existing proposals and standards and integrated them in our proposal whenever possible.

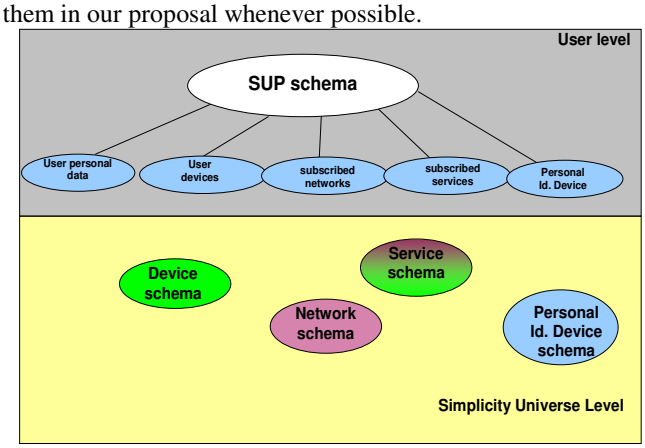

Figure 2: User Profile – Abstract view (xml schemas)

It is important to clarify that the SUP provides a definition of the profile information and of its representation, but this is not directly related to how the information is actually stored. Different parts of the profile information can be stored in the PID or in an external distributed storage system or databases or provided as external XML files referenced via URLs. Suitable entities called Profile Managers (see section IV) and residing both in the user's terminal and in the network nodes "export" the profile information towards components and applications needing those information by using the unified representation provided by the SUP schema.

A. *First component: user personal data*

The "user personal data" component includes information such as identity, biographical information, language, user's interests, hobbies and so on. This component also includes a free area that can be personalized and handled by external applications that want to store and retrieve personalization/configuration information related to the user and exploiting the facilities offered by the PID.

The part of user profile component that holds personal information data is defined taking as reference the Liberty Alliance Project Personal Profile (PP) [4]. The PP defines several data which are either "traits" (i.e. identities issued by governments and companies and also individual biometric characteristics) or "attributes" and "preferences" (which are those characteristics associated with an individual, e.g., musical preference, medical history…).

B. *Second component: user devices*

The "user device" component provides information on the devices owned or leased by the user (device type, device capabilities).

The device profile is based on a UAProf Schema provided by the WAP Forum [7], which is an RDF schema document describing the different hardware elements (e.g., mobile devices, presentation devices, terminals, etc.). In order to port the UAProf into a SUP component, the RDF schema has been translated into an XML Schema.
The user device component is made up of two sub-components: *DeviceList* and *PrefsPolicies*. The former is a list of devices which the user owns or is able to use; for each device entry there is a reference to the Device Component instance in the Simplicity Universe Level. The latter contains data useful to personalize the devices itself, such as preferences and policies related to a given device.

C. *Third component: Subscribed Networks*

The "subscribed networks" component contains connection preferences, policies, network parameters and accounts for a specific network.
The "subscribed networks" component should consider at least the following aspects: network type, access devices, network QoS, network storage space (if any), security at network level (e.g., IPsec) and AAA services; note that some items may be common to the Network and Service components
Within this component we can find information about user's accounts and the way the user has personalized his connection accounts.

D. *Fourth component: Subscribed Services*

The "subscribed services" component, like the "subscribed networks", is not currently defined in any standard, at least not in a compact and comprehensive way.
For instance, the Web Service Description Language (WSDL) is useful for describing a profile for Web services, but we think that the notion of "service" should have a more comprehensive meaning, including not only Web services but also "local" and "embedded" services, i.e., offered either by the local network or by the OS which is hosting a user's terminal. For instance, in the Simplicity framework, we had the necessity of taking into account specific "Simplicity middleware services".
Therefore, we defined a suitable "subscribed services" component. Within this component we find information about user's subscribed services and the way the user wants to personalize them. The "subscribed services" component is divided in three sub-components: *ServiceList*, *SessionList* and *PrefsPolicies*.
ServiceList contains the list of subscribed services. For each service entry there is a reference to the corresponding description, which is a Service Component instance in the Simplicity Universe Level. In addition, for each service, we specify the user data related to that specific service.
SessionList contains the list of suspended sessions, in order to be able to resume them. For each suspended session we specify session related data, which may be either raw data or structured data. *PrefsPolicies* includes all user's preferences and policies, which may be applied to all services (e.g. cost and QoS criteria).

E. *Fifth component: PID*

The PID component includes the information about the type of the PID owned by the user and its hardware/software capabilities.

IV. ARCHITECTURAL ASPECTS

In this section, we present the proposed architecture for profile handling, shown in Figure 3. The entities that manages profile information are called Profile Managers (PM). A PM is able to retrieve (and store) profile information about a user, on request of another entity which is generically denoted as Profile Requester Entities (PRE). The PMs retrieve the user profile information starting from the user's PID. In particular, there will be one PM entity called "controlling PM" (cPM) that will be directly linked to the user's PID. Other PMs, denoted as remote PMs (rPM) can access the parts of user's profile stored in different distributed repositories. cPM and rPMs cooperate to offer the profile access to PREs. As you can see in Figure 3, a PRE can reside in the same terminal where the cPM is running, or being located in a different host. In the latter case, the PRE will communicate over the network with a PM (either cPM or rPM) to request profile information. In this architecture, we can identify two relevant interfaces: the PM-PM interface and the PRE-PM interface. Both interfaces are defined using WSDL (Web Services Description Language). The implementation of the communication can rely on different solutions (e.g. SOAP, middleware like CORBA or JXTA, agent communication platforms like JADE). We are currently implementing the solution using the JXTA platform [18].

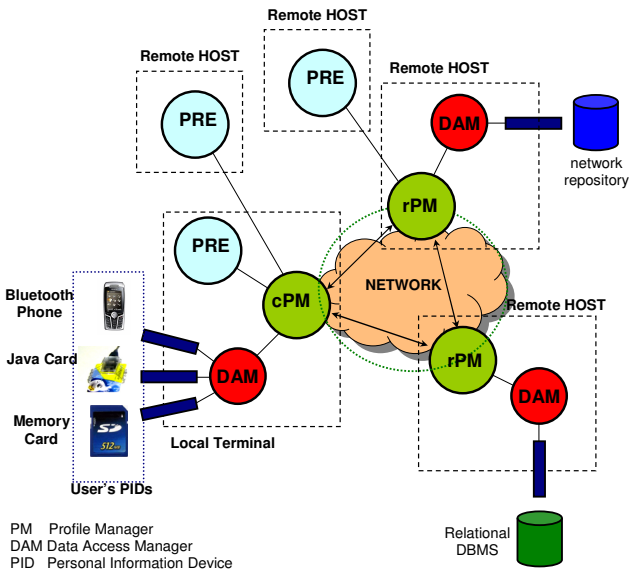

Figure 3: Proposed architecture for handling profile information.

To access the profile information, the Profile Managers exploits the Data Access Manager (DAM). This element translates the requests coming from the PM into a form suitable for the native communication mechanism used by the data repository (e.g. the PID, a database, a distributed storage system). The Profile Manager communicates with the DAM by

using an interface based on the XPath/XQuery specification [12]. This allows to achieve a total control over the XML representation of the SUP, enabling a flexible handling of every possible type of data and metadata contained in, linked to or referenced by the SUP. In our opinion, this choice has a very important impact in a system which aims to manage the widest possible set of types of user profile information. In addition, this feature makes our prototype implementation one of the first system employing XML databases in a direct way, as opposite to the use of traditional relational database.

A. *Comparison with the 3GPP GUP architecture*

If we look at the main architectural entities which play a role in managing, handling and storing the User Profile we can notice that the components specified in the GUP architecture have a counterpart in our approach. In fact, the role played by the GUP server corresponds to the one of the Profile Manager and the DAM kept together. On the network side, the counterpart of the GUP server is the Profile Manager on the network side. Furthermore, the GUP's RAFs are very similar to the "Communication Controllers", as they both provide a common interface which hides the implementation details of different data repositories.

The difference between the two models lies in the way user's data are handled: unlike the GUP architecture, which use a single entity in the network to manage user profile data (the GUP server), we use multiple peer entities to accomplish the same task. This approach could be useful for example if we want to keep sensitive data accessible only by contacting the Profile Manager on the terminal side.

V. SECURITY, TRUST AND PRIVACY ASPECTS

This Section faces the issue of protection of personal data in the proposed architecture. Due to the distribute nature of the proposed access to profile information, trust relationships must be established among the involved entities (PMs, DAMs, user PID). In particular, mechanisms are needed to provide differentiated access rights to different parts of the profile information. For example a security key for banking transaction must never be available outside the user PID, while a user can choose to make freely available his sport interests or the list of his favourite films. The terminal data needed to make a reconfiguration of the radio access capability of a device needs to be accessed only by authorized elements and may be even the user is not allowed to access it.

Hereafter, we first identify existing solutions and then we introduce our approach. In principle, the component "*common properties"*, defined in the 3GPP GUP, could be employed to define a mechanism for data access control. However, probably because of the abstract nature of the GUP, the current specification does not give indication on how the control mechanism works and how the access right are stored in the user profile. In a later specification, 3GPP stated that the GUP related security will be based on the work performed in the Liberty Alliance Project. In fact, the Liberty framework foresees support for privacy policy and preferences. Rather than defining a semantic for this purpose, the Liberty proposal is focused on defining a way to reach an agreement between a user (agent) ("Principal") and a Service Provider (SP) based on a comparison between the "level of privacy" offered by the SP and the one desired by the Principal. But what about the semantic? The Liberty's proposal, named PPEL [13], is an abstract way of defining privacy rules, and, according to the specifications, may use W3C's Platform for Privacy Preferences (P3P) [14] as a concrete syntax.

P3P is an ongoing W3C standard for Service Provider to describe in xml format the privacy practices a Service Provider conforms to. One or more policies can be associated to any resource (e.g. a web form asking user's data or retrieving a cookie) pointed by an URI which the user agent is going to access. Each policy describes which kind of data the resource will access, the purpose of the data collection, who will make use of these data and so on. Furthermore, P3P is complemented by another W3C standard, "A P3P Preference Exchange Language" (APPEL). APPEL is a xml-based language allowing a user to express her preferences about privacy in terms of a set of rules ("ruleset"). Use of P3P/APPEL is intended mainly for interaction between user agents and web servers, (Netscape, Internet Explorer and server side IBM Tivoli have already been provided). Also some public Internet sites already support P3P (a list of them can be found at [17]). In addition, there exist already organizations like TRUSTe or BBB Online which check the compliance of online services with the privacy policies that they declare.

The overhead due to verifying the compliance of policies to user's preferences may lead to bad performances, in terms of time required to access a resource. The Cobricks project [15] solves this problem by defining a so called Access Ticket (AT), which is the result of the negotiation performed between the user agent and the Service Provider in order to agree on accessing user's data.

The AT is a proprietary solution tailored for user profile data access, but it is very similar to other XML based access control approaches such as Oasis' XML Access Control Markup Language (XACML, [16]), which defines both a policy language and an access control decision request/response language and appears therefore suitable to define access rules to user profile data.

A. *Integrating security and privacy features in the SUP*

The concepts and the solutions for security and privacy discussed above can be rather easily integrated in the SUP. User's preferences about privacy can be expressed for each subscribed service, network and owned devices. Negotiation is performed between the Profile Requesting Entity and a Profile Manager and allows the requester to access and hold user's data for a given purpose and for a limited amount of time. Note that the same negotiation and trust establishment needs to be performed among two interacting Profile Managers.

At a "lower" level, the DAM is able to control the access to a physical data repository, encoding data access information as appropriate "metadata" (e.g., like the GUP common properties). In this approach, the Profile Manager acts as a Policy Enforcement Point (PEP), while the DAM acts as a Policy Decision Point (PDP), answering about whether the access should be granted.

This process is explained in the UML Sequence Diagram depicted in Figure 4, where a PRE (Requester in the figure) contacts the controlling Profile Manager to retrieve profile data residing on a remote repository. If this information is not yet cached in the local host, remote Profile Managers are contacted in order to retrieve the desired data, providing the requester credentials. After verifying access rights, data are finally returned to the requester, which takes control over it and can edit them (if authorized to do so). The local Profile Manager takes care of notifying remote Profile Managers about changes.

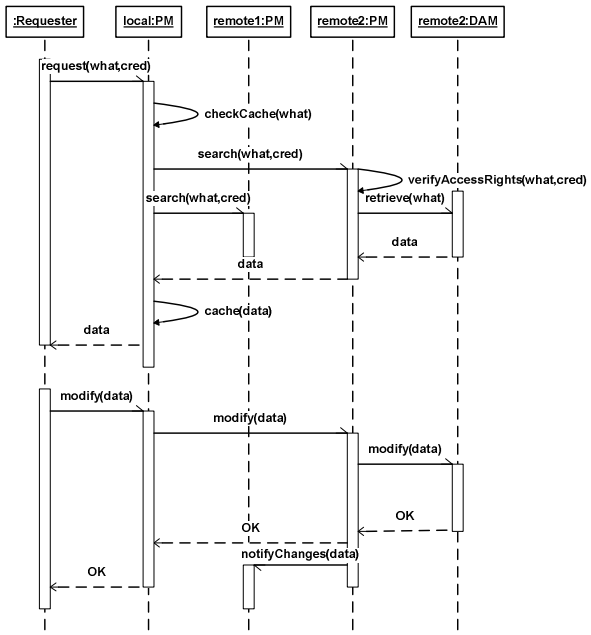

Figure 4: A scenario in which a local requester accesses data on a remote repository in a transparent way.

VI. ACKNOWLEDGMENT

This work has been performed in the framework of the EU funded project E2RII. The authors would like to acknowledge the contributions of their colleagues from the E2RII project.

VII. REFERENCES

[1] The Simplicity project web site: http://www.ist-simplicity.org
[2] The E2R and E2R II web site: http://e2r2.motlabs.com/
[3] Microsoft Passport URL:www.microsoft.com
[4] Project Liberty Alliance Web site: http://projectliberty.org
[5] G. Klyne, F..Reynolds, C. Woodrow, H. Ohto, "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies", http://www.w3.org/TR/CCPP-struct-vocab/
[6] J. Indulska, R. Robinson, A. Rakotonirainy, and K. Henricksen "Experiences in Using CC/PP in Context-Aware Systems", M.-S. Chen et al. (Eds.): MDM 2003, LNCS 2574, pp. 247–261, 2003.
[7] "UAProf", http://www.openmobilealliance.org/
[8] 3rd Generation Partnership Project. 3GPP Generic User Profile (GUP) 3GPP TS 29.240.
[9] 3rd Generation Partnership Project. Data Description Method (DDM) - 3GPP Generic User Profile (GUP) 3GPP TS 23.941.
[10] XML schemas for Simplicity User Profile: http://netgroup.uniroma2.it/SUP
[11] E. Rukzio, G. N. Prezerakos, G. Cortese, E. Koutsoloukas, S. Kapellaki, "Context for Simplicity: A Basis for Context-aware Systems Based on the 3GPP Generic User Profile", International Journal of Computational Intelligence Volume 1 Number 2 2004
[12] The Xquery project Home Page, http://www.w3.org/XML/Query
[13] L. Cranor, R. Wenning (ed.), et al., "The Platform for Privacy Preferences 1.1 (P3P1.1) Specification", W3C Working Draft 4 January 2005, http://www.w3.org/TR/P3P11/
[14] L. Cranor, M. Langheinrich (ed.), et al., "A P3P Preference Exchange Language 1.0 (APPEL1.0)", W3C Working Draft 15 April 2002, http://www.w3.org/TR/P3P-preferences
[15] Cobricks Home page, http://www.cobricks.de/
[16] XACML home page, http://www.oasis-open.org/committees/xacml/
[17] http://www.w3.org/P3P/compliant_sites
[18] Sun's JXTA platform for peer-to-peer networking http://www.jxta.org