Integrated Profile Management for Mobile Computing*

Alessandra Agostini†, Claudio Bettini†, Nicolò Cesa-Bianchi‡, Dario Maggiorini†, Daniele Riboni†

†Dipartimento di Informatica e Comunicazione
‡Dipartimento di Tecnologie dell'Informazione
Università di Milano, Milan, Italy
email:{agostini, bettini, dario, riboni}@dico.unimi.it, cesa-bianchi@dti.unimi.it

Abstract

In this paper we identify the main requisites of an infrastructure for the integrated management of profile data and propose a high-level description of its implementation. The main goal of our system is to offer to mobile users contents targeted to their needs, using a presentation suited to their device. This task can be accomplished by offering to content providers every information that can be useful for identifying the content and the presentation that best fit the user's expectations and the device capabilities. We identify the entities that are more feasible for providing profile data, and propose a mechanism for retrieving and integrating information from the different sources in this distributed environment. The presentation and content directives to be applied are determined by evaluating rules declared by both users and content providers, resolving possible conflicts.

1 Introduction

The proliferation of mobile devices, such as cellular phones, PDAs and car appliances, has made the anytime-anywhere information access paradigm a widespread reality. In order to offer many different services to a growing variety of devices, providers must perform an extensive adaptation of both content (to meet the user's interests) and presentation (to meet the user device characteristics) [Gimson, 2002]. Consider the case of a typical internet service request: currently, providers choose content based on information obtained from HTTP headers, induced by tracking the user's behavior on the site, or explicitely provided in case of subscribed users. Content is then formatted to meet device capabilities based on device information extracted from generic HTTP headers or from CC/PP profiles [Klyne *et al.*, 2002].

As the variety of devices and personalized services keeps on growing, a much richer user profile information is needed. This information may combine, for example, the specific content of the requested service with the location of the user and the action context in which the user is involved at the time of the request. To this end, we identify as *profile* any information that can be used to offer a "better" response to a request; i.e., the information that characterizes the user, the device, the infrastructure, the context and the content involved in a service request.

The information that compose a profile are highly distributed as they can be supplied by different entities. For example, personal data is provided by the user, whereas the information about the user's current location is usually provided by the network operator. As a consequence, different entities should manage distinct parts of the profile, and content providers should build the complete profile by querying these entities. This raises the problem of determining access rules for the entities and the different parts of the profile together with a protocol for applying those rules in a user-transparent manner.

Another method for addressing the problem of sharing profile information consists in implementing a distributed storage system on user devices (see for example [Riché and Brebner, 2003]). This method is better suited for preserving the privacy of data, but leads to a number of limitations in collecting and managing profiles. The architecture required for managing the different actions of our system is composed by various software elements that are exposed in the next sections. A similar infrastructure was developed in [Efstratiou *et al.*, 2001] for the real-time adaptation of applications running in a mobile environment. The focus of that work is on the adaptation of applications behaviour depending on unpredictable changes in the state of devices (e.g., bandwidth or battery availability). Our approach is different in that adaptation is performed server-side, and both users and content providers can define policies that are evaluated for determining the content presentation. Moreover, our policies take into account not only device capabilities but also user preferences, interests and action context.

There are several projects addressing the problems associated with content formatting, rendering, and delivery in a distributed multi-device environment [W3C Delivery Context Workshop, 2002]. In particular, our proposal goes along the same lines of DELI [Butler *et al.*, 2002], an initiative of HP Labs for designing an application that supports CC/PP and bridges the gap between the CC/PP-aware device and the server's delivery of device-appropriate content. With respect to DELI, the characterizing aspects of our architecture

*This work has been partially supported by Italian MIUR (FIRB "Web-Minds" project).

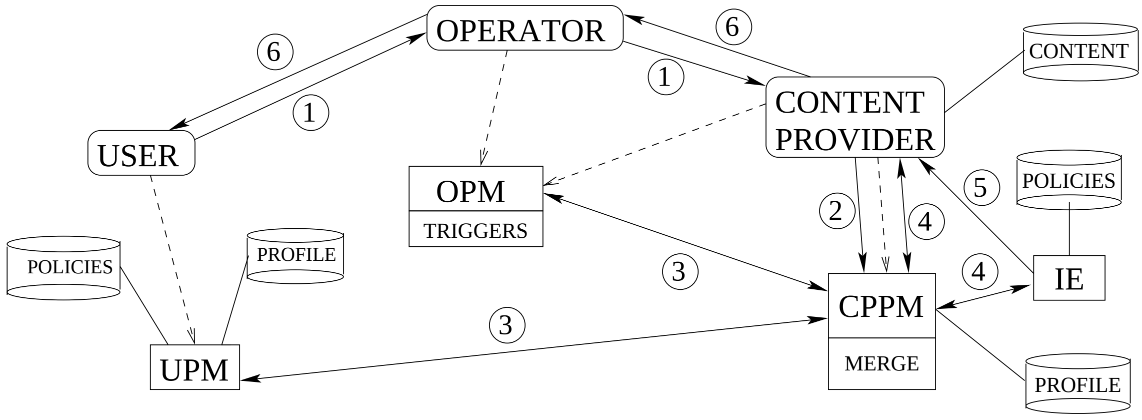

Figure 1: Information flow upon user request

are: (*i*) the presence of a rich language for user and content-provider policy specification, (*ii*) the asyncronous notification of changes in profile data via a trigger mechanism, and (*iii*) the presence of reasoning algorithms to evaluate policies against profile data and to solve possible conflicts.

2 Architectural View

As mentioned in the introduction, the goal of the system is to create, upon each user request, a set of directives for content selection and formatting. These directives are obtained combining user information, content provider information, and network operator information. The information combined can either be profile data (i.e., a hierarchical structure of attribute-value pairs) or policies (e.g., logical rules to derive presentation directives from profile data). Before describing the architecture of our profile management system, we give a high-level description of the system behavior (see Figure 1).

1. The user posts a request for a service to a content provider.
2. The content provider sends a request to the local profile manager (`CPPM`) to obtain the directives needed to handle the request.
3. The `CPPM` module issues requests to the User Profile Manager (`UPM`), and to the Operator Profile Manager (`OPM`) to retrieve remote profiles and policies.
4. The `MERGE` component of `CPPM` then combines this data with the proprietary profile information, resolving possible conflicts. The resulting combined profile and user policies become available to the content provider application logic and to the Inference Engine (`IE`).
5. The `IE` module evaluates content provider and user policies on the combined profile data, possibly resolving conflicts, and producing a set of presentation directives.
6. The content provider applies the presentation directives to the content selected by the application logic, sending the result to the user.

The dashed lines in Figure 1 represent the management of profile and policies inside a specific profile manager. In particular, the content provider has access to the `OPM` for defining triggers' behaviors, as it will be illustrated below.

Let us now describe units involved in profiles information management and discuss association with data subsets they are responsible for. In fact, heterogeneity and multi-party management of profiles information call for a strict definition about who is responsible to manage a certain data segment.

2.1 User Profile Manager

The `UPM` is responsible to manage attributes describing user-related information. These data are stored at user side and comprise personal profile (e.g., name and address), interests (e.g., politics and sport) and device capabilities (e.g., connection bandwidth and display type). User-related information does also include policies user may define with respect to the content and the presentation he wants to receive under particular conditions. Policies are stored in a different repository.

Since users do not want every entity in the framework to gain access to their complete profile, a specific `UPM` module called Access Control List (`ACL` in Figure 2) takes care of access control. Access control should also be performed on user policies in order to be accessible only by trusted service providers.

A `UPM` module called Semantic Profile Assistant (`SPA` in Figure2) is included in our framework to analyze user behavior and personal data (e.g., bookmarks and contacts), with the purpose of improving user interest profiling.

2.2 Operator Profile Manager

The `OPM` is responsible to manage attributes describing the user inside the network context (e.g., location, bandwidth and

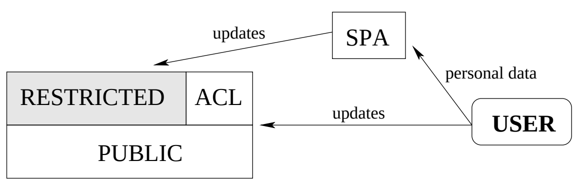

Figure 2: User Profile Manager architecture

latency). A change in these attributes may require the content provider to change the content and/or its presentation. As an example, if the user is looking for a restaurant while traveling in a car, his current position should be available to the content provider in order to select nearby restaurants only. Changes in information managed by an OPM may be frequent and unpredictable. Content providers are required to specify conditions for notifications, otherwise they may receive a number of unneeded signals. For example, a content provider based on location could specify to the OPM to notify a location changes only when greater than 3 miles.

This feature will be implemented using a triggering mechanism. Upon firing of one of the triggers, the OPM notifies the content provider and changes in profile attributes are propagated to the CPPM module. A similar mechanism is adopted by [Wu and Chao, 2001] for the notification of changes in device capabilities, network state, and location in a mobile computing environment.
Figure 3 depicts the management of OPM data by the operator and the content provider.

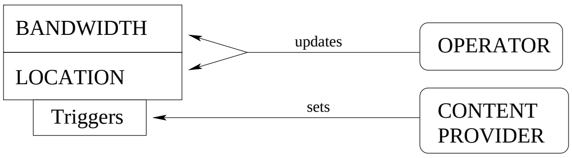

Figure 3: Operator Profile Manager architecture

2.3 Content Provider Profile Manager

The CPPM is responsible for management of content provider proprietary data that are not intended to be shared with other entities due to privacy (e.g., billing data and credit card number) or business reasons (e.g., user behavior information).

Let us consider the case of an online computer shop. Analyzing the web pages browsed by a user, the system can infer that he is especially interested in laser printers and cellular phones. This information will be useful when advertising to that specific user. This task is usually performed by CRM systems. A CRM can be easily integrated in our architecture; in addition to having access to proprietary profile data it will have access to the whole integrated profile through the CPPM module.

Additional features can be envisioned for the CPPM module; for instance, when the user switches to a different device, an image of the current session can be stored as part of the integrated profile, so that freeze and resume can be efficiently supported over multiple devices.

2.4 Inference Engine

The IE is provided in order to evaluate user and content provider policies, possibly resolving conflicts. Indeed, a provider may not always be able or want to agree with user policies. On the contrary, a provider usually wants to follow its own policies which, for example, distinguish various services related to different user categories. Further considerations on the internal structure of this module will be made in Section 4.

3 Integrated Profiles at Work

In this section we illustrate, through a simple example, the main features and the potential benefits of using our integrated approach to profile management.

Let us imagine that Will is one of the people in charge of the marketing at "O Sole Mio", an Italian company which core business is exporting pasta. Will, as every morning, is analyzing via internet the latest business news via its preferred portals. When in his office, he is used to keep an Instant Messaging System (IMS) active on his desktop to chat with colleagues and friends.

Context 1: Will is working from his office machine.

Let us examine what happens every time Will starts a network application, like the IMS, which takes advantage of the services of our Integrated Profile Manager. The IMS client application sends to the provider (via the operator infrastructure) the request to open a session, together with user login and password and some attached meta-information like the address of its UPM and OPM. The provider's application authenticates the user and asks its CPPM module to retrieve Will's local and remote profile information. This information is then integrated by the Merge component, and becomes available to the application logic and to the Inference Engine. Indeed, among these data, the application logic exploits the Will's buddies list, his location, and availability (*location='office'*; *availability='available'*). As a consequence, for example, Will's buddies are informed appropriately. With respect to presentation, the IE module communicates back to the provider's application a directive imposing the use of a text-only formatting. Indeed, the module has evaluated a CP policy, which states an HTML default format unless a user preference is given. Thus, the IE evaluates a personal Will's policy retrieved by the CPPM module, which reads as follows:

"**If** content='chat message' **and** location='office' **and** action-context='at work' **and** device='workstation' **Then** *set preferred-media='plain-text'*.

Therefore, till some changes are notified (e.g., by the firing of an OPM trigger), all messages coming from the IMS application will be sent in textual mode.

Context 2: Will explicitly changes context and sets temporary filtering preferences.

Will is going to receive the visit of his boss, Jill, at 3 PM in his office. Will has to report him the results obtained in the last quarter. Unlucky enough, he is still waiting for an e-mail message from an important customer —Mr. Mac Burghi, CEO of an international chain of fast food restaurants— regarding his decision on a long-term contract. If it were the case, Will really would like to include this success in the discussion with his boss. Therefore, he decides to set his current action context to "important meeting", specifying that Mr. Mac Burghi is the only person allowed to disturb him. This is done through an appropriate web interface to Will's UPM; in particular, the value of the *action-context* attribute in the profile is changed. The CPPM module is notified of the change and a new integrated profile is produced by the

Merge component. The change in that attribute disables, for example, the IMS and all other data/voice applications except for email, which is however filtered accordingly to Will's preferences. In particular, the following user policy is applied:

"**If** content='e-mail message' **and** location='office' **and** action-context='important meeting' **and** device='workstation' **Then** *allow-only-messages-from='(Mr. Mac Burghi)'*".

Exploiting the above policy, the application logic applies a filter regarding the senders of messages. On the contrary, the IE communicates to the provider that there are no changes regarding presentation directives.

Context 3: Will changes location and device.
During the meeting with his boss, Will received a positive answer regarding the contract and the meeting was better than expected. Will wants to share his happiness with his friend Jim, and, after resetting his action context, he starts chatting with him. At 6:30 PM Will leaves his office in a very good mood, to go home. To keep chatting with Jim while on the way to his car, he resumes the messaging session from his cellular phone application. Due to the resume request, the provider has to manage a complex roaming for session and device. In particular, it collects all previous session information and restores the relevant status of the session (e.g., the current application, the people involved in the chat, current messages) to the new device. Regarding presentation, the IE module retrieves from the CPPM the new device capabilities, and issues appropriate directives. Moreover, applying the actual Will's profile and policies, it discovers that the following policy applies:

"**If** content='chat message' **and** device='cellular phone' **Then** *set preferred-media='audio'*".

Therefore, Will is able to continue his chat in vocal mode exploiting the "text-to-speech" and "speech-to-text" interfaces of the provider.

Context 4: Will activates location-based services.
Will reaches his car, turns off his cellular phone, and turns on his car appliance. He switches off the IMS but turns on a push service about entertainment. After a while, a message alerts him that his friend Alina is having a drink in a pub in the surroundings, and he decides to join her at the pub.

Let us examine the system behavior in this case. Will has been notified because the CP application logic has retrieved from the CPPM module Will's preferences about news to be pushed, including the list of people whose location he is interested in. Note that the Merge component will integrate, according to some rules, the preferences stored in the proprietary profile repository and the preferences retrieved from the UPM. The Content provider will have set appropriate triggers on the (possibly multiple) OPM to be periodically informed of location changes regarding people on the list. Of course, a privacy policy will ensure that location data is revealed only to authorized users, and authorizations in our framework are actually part of user profile data. In this case, we assume Alina explicitly authorized Will in the past, by updating her profile data. This made possible for the content provider to notify Will of Alina's proximity. Note that, thanks to our framework, every third party can offer location-based services.

Context 5: Profile-based recommendation systems help Alina and Will.
Alina and Will decide to go to the movies. They have no idea about movies that are worth to be watched, so they ask for help to a dedicated web service. In a few moments, they receive a list of movie theaters, which are close to them, and which show movies that fulfill their tastes.

At the system level, the CPPM module has gathered and integrated profile information on movie preferences for Alina and Will. In the content provider proprietary profile a list of movies for which they have bought tickets in the past has been retrieved. From the UPM, explicit or derived profile data on general interests has also been retrieved. The application logic of the recommendation system has used the combined profile information for each user to identify a relevance ranked list of movies that both of them can enjoy. Then, thanks to the location information (again retrieved from the CPPM which is regularly updated by the OPM), the application has updated the ranking of the movies taking into account also the distance between Will and the movie theaters. The IE module, considering that the current device Will is using is a large screen WAP enabled cellular phone, has instructed the application to format the message with the movie list using the WML markup language, and a screen optimized interface which will allow Will to easily buy a ticket for the show.

4 Research Issues

Several research issues are involved in the representation and management of profile and policy information accordingly to the framework we are proposing. In particular, the *representational* issue concerns the identification of:

- **A language for describing the profile information.** Profile data is not simply a list of attribute/value pairs but it has a hierarchical structure, and complex data types may be involved. We are currently oriented towards the adoption of the CC/PP framework [Klyne *et al.*, 2002], as proposed in the W3C working group on Composite Capabilities/Preference Profiles. Hence profile data will be represented using the RDF syntax over a vocabulary defined by an RDF schema. CC/PP is currently being used mostly for representing device capabilities; UAProf [WAP Forum, 2000], for example, is a CC/PP compliant specification proposed by the Open Mobile Alliance to describe capabilities of Wap enabled devices. We plan to work within the CC/PP framework, possibly extending it and defining new vocabularies for expressing new features, such as users' context and interests. Moreover, an access control model needs to be defined over the profile data, and we are considering some of the recent proposals for XML access control.

- **A language for describing policies.** We are working at the definition of a simple language for expressing user and content provider policies. In its simplest form, a policy can be formally specified as a set of logical rules (like a datalog program[1]) whose antecedent is a set of conditions on profile data (interpreted as a conjunction), while the consequent is either new profile data (dynamically derived) or a presentation/content directive. The policies used in the example illustrated in the previous section follow this syntax, even if conditions on profile data are simplified in order to be more intuitive. Formally, since profile data is stored in an RDF structured document, each term in the logical rule is defined through a query on the RDF document which identifies an attribute value, a comparison operator, and a value, either identified by another query, a constant or a regular expression. Other more expressive languages for policy specification (e.g., [Damianou *et al.*, 2001; Lobo *et al.*, 1999]) may also be taken into account.

A second major issue concerns *reasoning* algorithms. In particular we need to investigate mechanisms for:

- **Evaluating content provider policies**, possibly resolving conflicts, and generating presentation/content directives. This will be the core algorithm of the `IE`. One possibility is to apply well-known techniques to evaluate datalog programs [Ceri *et al.*, 1990], but more sophisticated approaches like the one proposed in [Bettini *et al.*, 2002] may also be investigated.
- **Merging profile data** extracted from the content provider proprietary repository and user defined profile data, possibly resolving conflicts. This functionality must be integrated in the `CPPM`; it is based on a set of merging rules establishing overriding criteria in the case different values are found for equivalent attributes in profile data obtained from different sources.
- **Deriving new profile data** by induction on the basis of user behavior and user personal data (e.g., bookmarks, contacts, emails). Despite this is not a crucial aspect in the architecture, it can highly increase the value of profile data for service personalization. Data mining techniques can be exploited for this purpose. Preliminary results on personal user bookmarks classification can be found in [Bettini and Cesa-Bianchi, 2001].

5 Conclusions and future work

In this paper we proposed an infrastructure for the distributed management and retrieval of profile information, targeted to facilitate the provision of highly personalized services to the end user. We are currently investigating various approaches for implementing the inference engine with particular attention to performance issues, and we are working at the development of a first prototype of our system.

[1]Datalog programs are function-free Horn-clause programs.

References

[Bettini and Cesa-Bianchi, 2001] Claudio Bettini and Nicolò Cesa-Bianchi. An Architecture for Advanced Bookmark Management. In *Proceedings of WS-DAAL 2001*, Como, Italy, September 10-12, 2001. *http://tizero.usr.dico.unimi.it/papers*.

[Bettini *et al.*, 2002] Claudio Bettini, Sushil Jajodia, X. Sean Wang, and Duminda Wijesekera. Provisions and Obligations in Policy Rule Management and Security Applications. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 2002)*, pages 502-513, Hong Kong, China, August 20-23, 2002.

[Butler *et al.*, 2002] Mark Butler, Fabio Giannetti, Roger Gimson, and Tony Wiley. Device Independence and the Web. *IEEE Internet Computing*, 6(5):81–86, September-October 2002.

[Ceri *et al.*, 1990] Stefano Ceri, Georg Gottlob, and Letizia Tanca. *Logic Programming and Databases*. Springer-Verlag, June 1990.

[Damianou *et al.*, 2001] Nicodemos Damianou, Naranker Dulay, Emil Lupu, and Morris Sloman. The Ponder Policy Specification Language. In *Proceedings of Policy Workshop 2001*, pages 18-39, Bristol, UK, January 29-31, 2001.

[Efstratiou *et al.*, 2001] Christos Efstratiou, Keith Cheverst, Nigel Davies, and Adrian Friday. An Architecture for the Effective Support of Adaptive Context-Aware Applications. In *Proceedings of the Second International Conference on Mobile Data Management (MDM 2001)*, pages 15-26, Hong Kong, China, January 8-10, 2001.

[Gimson, 2002] Roger Gimson. Delivery Context Overview for Device Independence. *http://www.w3.org/2001/di/public/dco/dco-draft-20020912/*

[Klyne *et al.*, 2002] Graham Klyne, Franklin Reynolds, Chris Woodrow, and Hidetaka Ohto. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies, November 8, 2002. *http://www.w3.org/TR/CCPP-struct-vocab*.

[Lobo *et al.*, 1999] Jorge Lobo, Randeep Bhatia, and Shamim Naqvi. A Policy Description Language. In *Proceedings of the National Conference of the American Association for Artificial Intelligence*, pages 291-298, Orlando, FL, USA, July 18-22, 1999.

[Riché and Brebner, 2003] Stépanie Riché and Gavin Brebner. Storing and Accessing User Context. In *Proceedings of the 4th International Conference on Mobile Data Management (MDM 2003)*, pages 1-12, Melbourne, Australia, January 21-24, 2003.

[W3C Delivery Context Workshop, 2002] W3C Delivery Context Workshop. Sophia-Antipolis, France, March 4-5, 2002. *http://www.w3.org/2002/02/DIWS/final.html*

[WAP Forum, 2000] WAP Forum. User Agent Profile Specification. WAP-248-UAProf. *http://www.wapforum.org/*.

[Wu and Chao, 2001] Shiow-yang Wu and H. S. Cinatit Chao. Event Engine for Adaptive Mobile Computing. In *Proceedings of the Second International Conference on Mobile Data Management (MDM 2001)*, pages 27-38, Hong Kong, China, January 8-10, 2001.