Dynamic Discovery and Monitoring in Migratory Interactive Services

Renata Bandelloni, Fabio Paternò, Zigor Salvador
ISTI-CNR
{renata.bandelloni, fabio.paterno, zigor.salvador}@isti.cnr.it

Abstract

Migratory interactive services play an important role in supporting mobile users because they allow them to continue their activities across interaction with different devices. When designing architectures and infrastructures for supporting such services, one important issue is how to manage the dynamic set of interactive devices that can be involved. In this paper, we present a solution able to flexibly manage dynamic discovery of interactive devices and their monitoring and exploit such information.

1. Introduction

One important aspect in pervasive environments is the possibility for users to freely move about and continue to interact with the potential services present in the environment through a variety of interactive devices (cell phones, PDAs, desktop computers, Digital TVs, intelligent watches, and so on). However, this implies the need for services which are able to follow users and adapt to the changing context. In practise, it is sufficient having only the part of a service interacting with the users migrate to different devices. However, effective migration requires that the service user interfaces adapt to the interaction resources of the hosting device in such a way as to preserve usability.

Little work has been carried out so far in this area. In Aura [4] support for services able to follow users has been considered, but the envisioned solution does not allow obtaining user interface adaptation within a given application. The PUC [2] solution within the Pebbles project supports user interface adaptation starting with some conceptual description, but only limited to domestic appliances. ICrafter [4] provides support for adapting user interfaces in ubiquitous environments with the possibility to dynamically compose services, but migration support is not provided.

2. Migratory Interactive Services

In previous work we have presented a solution for obtaining a service supporting migratory user interfaces in the case of Web applications. The service was based on a client-server architecture that supports migration requests, which are managed at the server side [1].

The core of the migration service is the migration server. The server automatically retrieves and collects information about the devices discovered in the environment. Such information is necessary in order to select the proper migration target device in case of an automatically triggered migration. Independent of which side (the user or the system) starts the migration process, the server retrieves the original version of the Web page that has to be migrated and builds the corresponding logical description through a reverse engineering process. Such semantic information is used to perform a redesign for the target platform, which is followed by a forward engineering process to generate the adapted Web page(s). Moreover, in order to support task continuity, the state resulting from the user interactions with the source device (entered data fields, selected items, …) is gathered through a dynamic access to the DOM of the pages of the source device. Then, through an intelligent analysis such information is associated to the corresponding elements of the generated pages adapted to the features of the target device, which are then uploaded to the device.

In this paper, we present a solution aiming at overcoming the limitations of the previous version of the migration service, such as deficiencies in handling dynamic environments with devices joining and departing from them, and insufficient monitoring of the state of the surrounding client devices. In addition, we introduce improvements obtained with the introduction of service discovery mechanisms as well as the use of additional technology, such as

RFID tags, to sense device proximity. A further innovation is the separation of the system's intelligence from the code. Instead of using hard-coded rules for migration, as in the previous version of the system, herein we introduce the concept of external XML rules taken as input by the migration server and used to make decisions concerning migration. This is the case, for example, of target device selection rules in the event of migration activated by the system. Using these techniques and tools improves the ease of use and enables the possibility of natural interaction for the user as well as expandability of the system itself.

3. The New Solution

Migration can be triggered by different conditions. Both the system and the user have the possibility to initiate the migration process, depending on the surrounding environmental conditions and context.

User-initiated migration

Users have two different ways to issue migration requests. The first one is to select the icon corresponding to the desired target device, which is already present in the environment. Users only have the possibility of choosing those devices that they are allowed to use and are available for migration in that precise moment. The second possibility for issuing migration requests occurs when the user is interacting with the system through a device equipped with an RFID reader. In this case, users can move their device near a tagged migration target and keep it close for a number of seconds in order to trigger migration to that device. In this case a time threshold has been defined in order to avoid accidental migration, for example when the user is just passing by a tagged device. This second choice offers users a chance to naturally interact with the system, requesting a migration just by moving their personal device close to the desired migration target, in a straightforward manner.

Automatic migration

Migration can also be initiated by the system skipping explicit user intervention in critical situations when the user session could accidentally be interrupted by external factors. For example, we can have a user interacting with a mobile device that is shutting down because its battery power is getting too low. Such situation can be recognised by the system and a migration is automatically started to allow the user to continue the task from a different device.

Mixed-initiative migration.

Alternatively, the server can provide users with migration suggestions when migrating to another device in order to improve the overall user experience. This happens when the system detects that in the current environment there are other devices that can better support the task being performed by the user. For example, if the user is watching a video on a PDA and in the same room a wide wall-mounted screen is detected and available, the system will prompt the user to migrate to that device, as it could improve his performance. The user would obviously retain the freedom to continue working in the current device and refuse to migrate. Receiving undesired migration suggestions can be annoying for the user, thus users who want to receive such suggestions when better devices are available must explicitly subscribe for this service. In general, some rules for triggering suggestions include:

- If the user is interacting with a small screen device for graphical interaction (PDA, mobile phone) and a bigger screen device becomes available (desktop computer, wall mounted screen), then suggest migration to the device with the bigger screen to improve information visualisation.
- If the user interacts with a device having limited interaction capabilities (voice interaction, touch screen) and a device offering better interaction capabilities is available (graphic/multimodal interaction, mouse and keyboard) then suggest migration to the device offering better interaction capabilities in order to increase comfort.
- If the user is interacting with a mobile device and a stationary device is available then suggest migration to the stationary device to save mobile device power.

4. Device and Service Discovery

We have designed a service-oriented architecture (SOA), in which a service is basically a container of functionality and all of the main functional modules are offered as separate services (the grain of the system being rather coarse for the sake of performance). This approach is regarded as a good design choice in pervasive distributed systems in which several platforms and several functional modules are to coexist because of its loosely coupled nature. The technology that enables this discovery in our migration architecture is a custom discovery protocol explicitly created to handle service discovery, service description, and service state monitoring tasks. The design of this protocol provides multicast mechanisms for service discovery combined with reliable unicast service description and service monitoring primitives. The implementation and use of this protocol provides means for the system to gather a rich set of information from each of the devices that are present in the environment, regarding their interaction and communication capabilities as well as their general computational capabilities. It also offers an efficient way to monitor the state of the devices available in the environment, by subscribing to receive events when the device state changes occur. All this has allowed us to improve the support for automatic migration through richer and more accurate information and monitoring of the environment.

In the new discovery-enabled environment, users no longer need to manually specify the IP address of the migration server, the middleware will automatically discover it for them. Neither do they need to login their personal interaction device into the migration environment, as their devices will automatically be detected by the system both when connecting to it and when disconnecting from it. Thus, the new migration architecture and support offers increased robustness, better consistency and a greatly improved ease of use.

5. Migration Target Identification

When a migration request is issued by the user, the target device is already specified as a request parameter. Conversely, when the migration is automatic, the target device has to be chosen by the system. Both for automatic and mixed-initiative migration, the server applies a sorting algorithm that assesses the device features (display capabilities, supported interaction modalities, existing communication interfaces, RFID related features and other platform specific data) depending on their actual values. These metrics are used to obtain a sorted list of the devices available for migration, ordered in accordance to their relative suitability for the migration in progress. Some of the considered features can assume different levels of importance depending on the device from which the Web page is migrating, that is, the source device. This means that for each possible source device, the evaluation and ordering of all the available targets is reconsidered, by assigning different weights to the different features involved.

Both mixed-initiative and automatic migration require two sets of rules: one of them is useful for deciding when to trigger/suggest migration, and the other is useful in order to determine the most suitable target device. An automatic migration is usually triggered by events that indicate that the current device is going to be unusable, such as if the user device has a battery and the battery is getting too low or if the device is using a GSM connection and the signal is getting too low. In the case of automatic migration, it is often necessary to find a target device as quickly as possible. Therefore, possible migration targets will be evaluated, firstly for their suitability, then for their availability, until the best available choice is selected.

The possibility of suggesting a migration is considered when a new device becomes available for the migration service or one of the devices already in the migration environment changes one of its dynamic attributes, such as its availability. In this case, the system will re-calculate the suitability value for that given device and re-position it in the dynamically generated ordered list for the migration in progress.

6. Example Application

The presented system for the dynamic discovery and monitoring in migratory interactive services is particularly well suited to be applied in organizational environments in order to support advanced ways of interaction between the users and such environments. We are testing it in the context of this type of application. In order to illustrate its suitability for supporting mobile users, in the following paragraphs we describe a scenario which is made possible by our migration environment.

Claudia is a project leader in a software development organization. Today, just as every other day, she is commuting to work by train. During the ride, she accesses her personal Web calendar through her UMTS enabled (PDA) and receives the list of her commitments for today, which include an early meeting with her project team-mates. As soon as she arrives to her office, the software running on her PDA detects the availability of a wireless corporate network, and automatically terminates the UMTS connection and switches to it (step 1 in Figure 1).

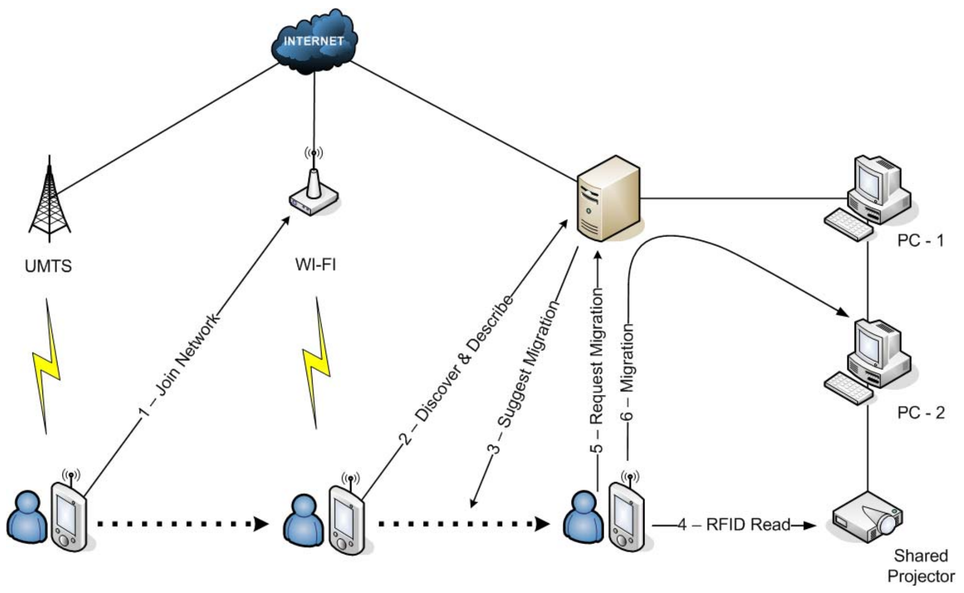

Fig. 1 – An Example scenario.

Once connected to the corporate network, a migration server active in the environment detects Claudia’s presence through the PDA, and taking into account the user’s past behaviour and the available devices’ features, suggests a migration to Claudia’s desktop computer (step 3 in Figure 1). Any other day, the suggestion that just popped up in Claudia’s PDA would have been accepted, but today she is in a hurry as the meeting is about to start, and she chooses not to migrate and to take the PDA with her to the conference room instead.

Once she arrives at the conference room, and after greeting everyone, she access her PDA’s Web browser and, in particular, the corporate project management Web application they have been using to track the progress of the current project. After some interaction, she realises that she needs a larger screen and with a simple gesture puts her PDA, which is equipped with a RFID reader, close to the video projector of the conference room, thus triggering a migration from the PDA to the computer hosting the projector (step 6 in Figure 1). After the migration everyone can see the project’s current status and the meeting can start without further delay.

7. Conclusions

In this paper we have presented the improvements that are being implemented in order to enhance our architecture and infrastructure for supporting migratory user interfaces creating a new version of our migration service.

The enhancements provide a framework for more natural interaction between users and the pervasive system, thanks, on the one hand, to the introduction of RFID-enabled devices, and to the new automatic discovery features, on the other. Combined, the improvements allow users to better handle changing, dynamic environments in a way that is more intuitive and natural, thereby improving usability to a degree which will be measured in future evaluation work.

References

[1] Bandelloni, R. Mori, G. Paternò, F., “Dynamic Generation of Migratory Interfaces”. Proceedings of Mobile HCI 2005, ACM Press, Salzburg, September 2005.
[2] Nichols, J. Myers B. A., Higgins M., Hughes J., Harris T. K., Rosenfeld R., Pignol M.. “Generating remote control interfaces for complex appliances”. Proceedings of ACM UIST’02, pp.161-170, 2002..
[3] Ponnekanti, S. R. Lee, B. Fox, A. Hanrahan, P. and Winograd T. “ICrafter: A service framework for ubiquitous computing environments”. Proceedings of UBICOMP 2001. (Atlanta, Georgia, USA., 2001). LNCS 2201, ISBN:3-540-42614-0, Springer Verlag London UK. pp 56-75.
[4] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P. “Project Aura: Toward Distraction-Free Pervasive Computing”. Proceedings of IEEE Pervasive Computing, Vol 21, No 2 (April-June 2002), pp.22-31.