Exploiting MARIA Models at Runtime in Ubiquitous Environments

GIUSEPPE GHIANI, FABIO PATERNO', CARMEN SANTORO

ISTI-CNR, HIIS Laboratory, Via Giuseppe Moruzzi, 1 56124 Pisa (ITALY)

One important evolution in software applications is the spread of service-oriented architectures in ubiquitous environments. Such environments are characterized by a wide set of interactive devices, with interactive applications that exploit a number of functionalities developed beforehand and encapsulated in Web services. In this paper, we discuss how a novel model-based UIDL can provide useful support at run time for these types of applications. The language is exploited to support dynamic generation of user interfaces adapted to the different devices at hand during the user interface migration process, which is particularly important in ubiquitous environments.

Model-based UIDLs are utilized at design time to help the user interface designer cope with the increasing complexity of today's applications and contexts. The underlying user interface models are mostly used to generate a final user interface code, which is then executed at run time. Nevertheless, approaches utilizing the models at run time are receiving increasing attention. We agree with [Sottet et al. 2007], who call for keeping the models alive at run time to make the design rationale available. We are convinced that the utilization of models at run time can provide useful results, such as support of migratory user interfaces. Migratory user interfaces are interactive applications that can transfer among different devices while preserving the state and therefore giving the sense of a non-interrupted activity. The basic idea is that devices that can be involved in the migration process should be able to run a migration client, which is used to allow the migration infrastructure to find such devices and know their features. Such client is also able to send the trigger event to the migration server, when it is activated by the user. At that point the state of the source interface will be transmitted to the server in order to be adapted and associated to the new user interface automatically generated for the target device (see Figure 1).

Fig. 1. The Migration Approach

We have designed and developed a software architecture able to support the main phases in the migration, based on previous experiences in the field [Bandelloni et al., 2005]. The first phase is Device Discovery (step 1 in Figure 2). It concerns the identification of the devices that can be involved in the migration process and the attributes that can be relevant for migration (private or public device, their connectivity, their interaction resources, …). The Device Discovery has to be activated in every device that is involved in the migration (including the Migration Server). Its purpose is allowing the user to trigger the migration by selecting the migration target device. In order to do this, the Device Discovery module has to notify the presence of the associated device to a known multicast group. The list of the devices currently subscribed to such a group defines the list of devices that are available and could be involved in a migration process. In order to do carry out this notification, the Device Discovery/Migration Client module use multicast datagrams communications using UDP/IP protocol.

After the device discovery phase, the user requests a page access from the current device (2), the request is transmitted to the application server (3), which provides the page (4). Such page is downloaded by the Migration server (which works also as a proxy server) and then annotated by the Migration Server before delivering it to the requesting device (5). The annotation is the automatically insertion of a script that is then used to transmit the state of the user interface. Thus, when the migration is triggered and the target device is identified (6), the current page along with its state is transmitted through an Ajax Script (7) to the migration server. At that point a version of the page adapted for the target device is created, with associated the state of the source version, and uploaded so that the user can immediately continue with all the data entered beforehand.

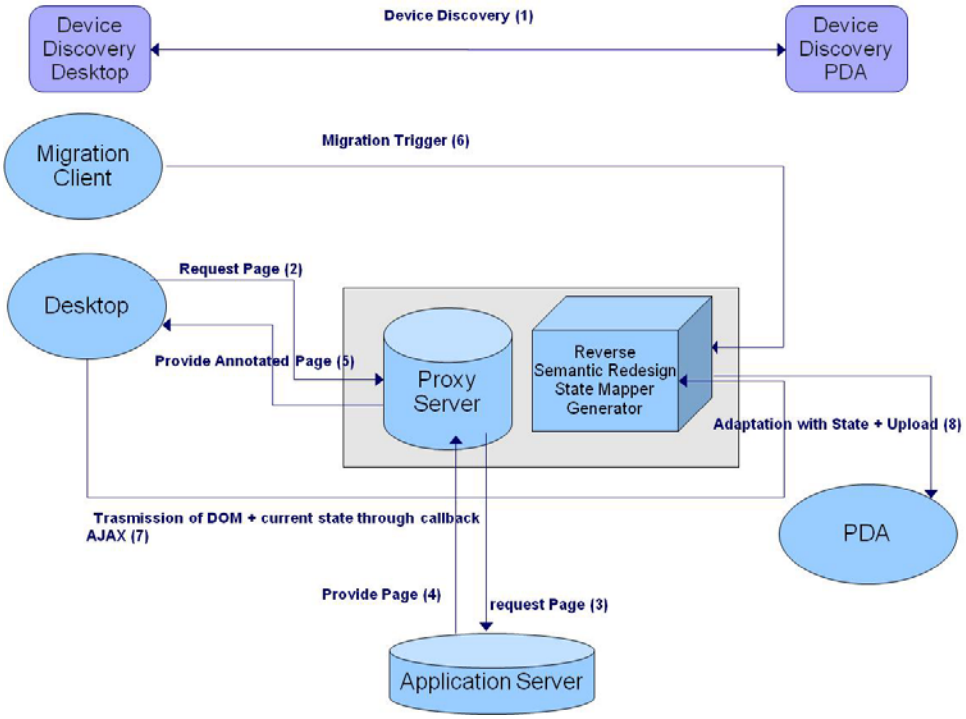

Fig. 2. The steps of the Migration

Figure 3 shows how the abstraction layers are exploited to support migratory user interfaces, by showing the various activities that are done by the Migration Server. First of all the migration approach supposes that various UI models at different abstraction levels are associated to the various devices involved in a migration: such UI models are stored/manipulated centrally, in the Migration Server.

The current architecture assumes that a desktop Web version of the application front-end exists and it is available in the corresponding Application Server: this seems a reasonable assumption given the wide availability of this type of applications. Then, from such final UI version for the desktop platform, the Migration Server automatically generates a logical, concrete UI description for the desktop platform through a reverse-engineering process. After having obtained such a concrete UI description for the desktop platform, the Migration server performs a semantic redesign of such CUI [Paternò et al., 2008] for creating a new, concrete, logical description of the user interface, adapted to the target device. The purpose of the semantic redesign is to preserve the semantics of the user interactions that should be available for the user but to adapt the structure of the user interface to the resources available in the target device. It may happen that some task is not supported by the target device (e.g. a long video cannot be rendered with a limited mobile phone). For all the tasks that can be supported the semantic redesign identifies concrete techniques that preserve the semantics of the interaction but supports it with techniques most suitable for the new device (for example in mobile devices it will replace interactors with others that provide the same type of input but occupying less screen space). In a similar way also page splitting is supported: when there are pages too heavy for the target device they are split taking into account their logical structure so that

elements logically connected remain in the same page. Thus, the groupings and relations are identified and some of them are allocated to newly created presentations so that the corresponding page can be sustainable by the target devices.

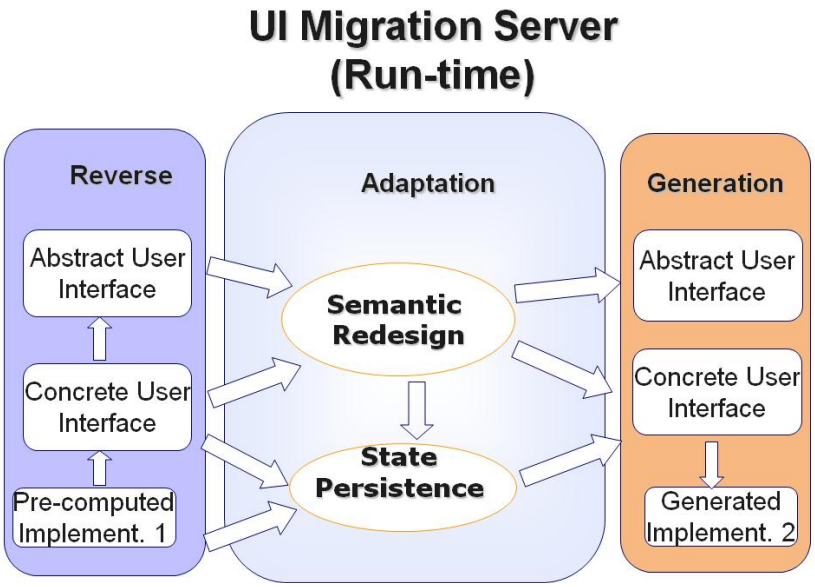

Fig. 3. The relationships among abstraction layers supporting migration

The state is extracted through specific JavaScripts, which are automatically included in the Web pages when they are accessed through the intermediate migration server. When the migration is triggered the state is transmitted to the server where there is a module (State Mapper) whose purpose is to associate the state with the concrete description for the target device, which is used for the generation of the final user interface.

CONCLUSIONS

In this paper we have discussed the use of the MARIA language in a software architecture able to support migratory user interfaces. Migratory user interfaces are able to exploit ubiquitous environments and thereby follow mobile users as they change devices while maintaining the interactive application state.

ACKNOWLEDGMENTS

This work has been supported by the OPEN (http://www.ict-open.eu) ICT EU Projects .

REFERENCES

BANDELLONI R., MORI G., PATERNÒ F. 2005. Dynamic Generation of Migratory Interfaces, in *Proceedings Mobile HCI 2005*, Salzburg, Austria, 83-90.

PATERNÒ F., SANTORO C., SCORCIA A. 2008. Automatically adapting Web sites for mobile access through logical descriptions and dynamic analysis of interaction resources, in *Proceedings of the working conference on Advanced visual interfaces(AVI*), Napoli, Italy, 260-267

SOTTET, J., CALVARY, G., COUTAZ, J., FAVRE J.2007. A Model-Driven Engineering Approach for the Usability of Plastic User Interfaces, in *Proceedings of the Working Conference on Engineering Interactive Systems*, Adelaide, Australia, 140-157.