APPLICATION-TRANSPARENT ADAPTATION IN WIRELESS SYSTEMS BEYOND 3G

Nikos Houssos, Vangelis Gazis, Athanassia Alonistioti
Communication Networks Laboratory, Department of Informatics & Telecommunications,
National and Kapodistrian University of Athens,
157 84, Athens, Greece,
email:{nhoussos, gazis, nancy}@di.uoa.gr

ABSTRACT - The forthcoming wireless systems beyond 3G are heralded to trigger dramatic changes in the way that mobile services are being developed and delivered. User expectations are expected to raise to a significantly higher level, towards the demand for terminal-, network- and location-aware provision of ubiquitous, personalized multimedia services. Adaptability, a feature that did not attract particular attention from system and service developers of services in the pre-3G era, is widely recognized as a critical factor for realizing this vision. The present paper discusses adaptation of end-user services provided over next generation mobile systems, identifies relevant important issues and introduces a mechanism for deployment–time, application-transparent adaptation.

1. INTRODUCTION

The evolution of mobile networks and systems to 3rd generation and beyond is expected to bring about substantial changes to telecommunication service provision, which has been based until 2G on rigid, operator-centric paradigms. In the envisioned new era, where customer expectations as well as investments and revenue requirements are raised to a new level, a plethora of functionality-rich, profit-creating applications, typically contributed by many different business players, should be available to mobile users [1]. Moreover, services should be accessible in an unprecedented variety of contexts [2], which could not be predicted or catered for during service design and development. These extensively demanding requirements create the need for adaptability of applications to highly diverse environments, a feature that has not yet attracted adequate attention in the research regarding design and engineering of mobile services. In this paper, we introduce a mechanism that addresses a specific case of service adaptation and we present its successful incorporation in a 3G service provision platform.

The rest of this document is organized as follows: In Section 2 we discuss the notion of service in mobile systems and introduce a high-level model for the service software architecture. Next, we define service adaptation and provide a taxonomy of relevant significant issues that come up in the design of the corresponding functionality. In Section 4, we present the environment where the proposed adaptation mechanism was integrated and applied, namely a software platform for provision of services over 3G mobile networks. In Section 5 we elaborate on the introduced mechanism for application adaptation, including explanation/benefits of the design choices made and presentation of the required interaction sequences for the accomplishment of a service adaptation operation in our prototype. The last sections of this paper are dedicated to summary, conclusions and acknowledgements.

2. THE NOTION OF SERVICE IN MOBILE NETWORKS

The concept of service is undoubtedly hard to define. Anything a service provider can sell that is not hardware equipment can be considered a service. In the present paper the term "service" (used interchangeably with the terms "application", "value-added service", "VAS") refers to an information technology product which is directly accessible and perceptible to the end user and whose value resides mostly in functionality and content, rather than transport or connectivity. Thus, the notion of service includes almost anything imaginable from plain messaging and information retrieval to intelligent, personalised navigation guides and shopping assistants.

Although the concept of service is generic, we will try in the following to identify some common characteristics in the architecture of applications offered to users of next generation networks. Notably, the above definition as well as the service model described below do not apply to 2G mobile services whose development and delivery followed the traditional paradigms of the telecommunications world [3], where services were mostly call-related and their logic resided in core network nodes and was tightly coupled with the underlying equipment.

A service will typically consist of code that runs in the end-user terminal (service client) and optionally a server (stationary) part. The former includes, besides other functionality, the service user interface. It can be pre-installed in the terminal, or downloaded and executed on demand. The latter case is aligned with the dynamic nature of service provision desired for 3G systems. Service execution normally, but not necessarily, includes interaction with the server part. For example, a game could be a standalone application, while a streaming audio/video service involves communication with remote servers. If available, the application server part is located in the premises of the service provider (e.g., web server). Multi-tier architectures will be also commonplace, where the server part comprises many interacting components. These components may be located in the administrative domains of entities other than the service provider, like content providers marketing access to their databases, or network operators providing interfaces to network functionality (e.g., OSA, Parlay, JAIN).

Services will in general come in many versions (editions), so that it can be available to users accessing them with various types of terminals and having diverse preferences. In particular, the client part comes in multiple predefined versions or it can be even dynamically composed (in a context dependent manner) by application components on demand. Notably, the difference between distinct versions lies in functionality, and not how the latter is implemented. Furthermore, different implementations of the service client part or each component, targeting different contexts, could be available. Essentially, in a service there is some core functionality, which is present in all versions/configurations and an optional/adaptable part that is tailored to the environmental conditions. This generic model is depicted in Figure 1.

3. ADAPTATION OF MOBILE SERVICES – IMPORTANT ISSUES

Adaptability/adaptivity is commonly defined as the ease with which an entity can be dynamically modified according to its context. The dynamic modification (adaptation) can be considered as a transition to another state of the entity and could include an alteration in the entity's behavior. Context refers to any information that can be used to characterize the situation of the entity[1] [4].

In the present section we list and briefly discuss certain significant issues that a designer/implementer of service adaptation functionality has to address.

[1] Many definitions of the notion of context exist in the literature. The one we have adopted has been introduced in [4].

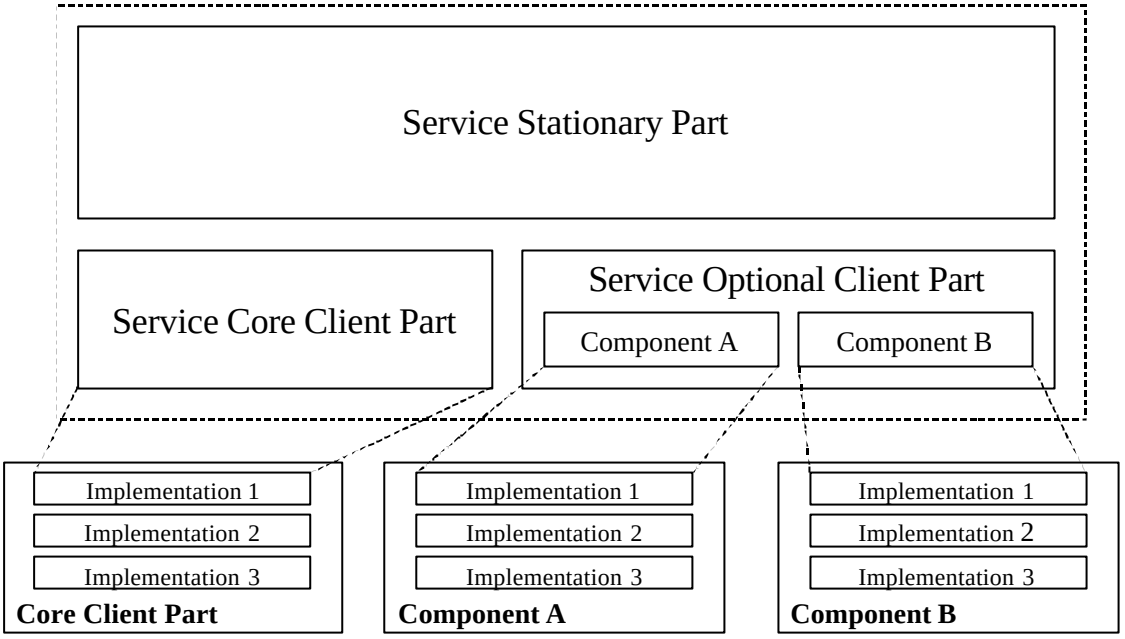

Figure 1. Generic service architecture.

3.1 What is the actual adaptation task?

An apparent question relates to what is actually meant by service adaptation, and more specifically which tasks/operations applicable on a service do we consider that adapt it. A list of potential answers is the following:

- Appropriate setting/modification of operational parameters (e.g., the update/refresh rate in a stock exchange prices notification application) without a change in the actual service logic.
- Determination/modification of the physical distribution of the application components (i.e., which components belong to the server/stationary and which to the client/downloadable part, according to the terminology of Section 2).
- Determination/modification/switching of the functional entities that should constitute the application in each particular case. Different configurations of a service could differ in the supported functionality (e.g., inclusion/exclusion of an encryption/decryption module) and/or in the functionality's implementation (e.g., a lightweight implementation could run in a PDA, while a "heavier" version with the same functionality could be the best choice for a laptop computer).

3.2 When does adaptation take place?

Another issue concerns the time that adaptation takes place. Two rough categories can be identified, according to that criterion: installation/deployment-time and run-time adaptation. The former occurs during application installation (a procedure which for mobile services typically includes among others the on-demand downloading of the service client part) and the latter during the actual service execution. Apparently, run-time adaptation introduces a higher degree of flexibility, albeit together with increased implementation complexity. Deployment-time adaptation is a useful approach, which may be also easier to realize. Notably, all the adaptation actions that are listed in Section 3.1 can be performed during installation as well as execution. Combinations of the two alternatives are also possible.

3.3 Where is the required functionality incorporated?

Another taxonomy pertains to whether the logic/intelligence that actually carries out the adaptation task resides inside the service itself or not. One could identify three cases for where adaptation logic resides:

- Exclusively within the application.
- Exclusively outside the application, which is totally agnostic/independent of the existence of adaptation functionality. "Outside" here could mean operating system, middleware, service management application, etc.
- Within the application as well as outside it (this is obviously a combination of the two above alternatives).

Notably, these three cases were called *laissez-faire adaptation*, *application-transparent adaptation* and *application-aware adaptation*, respectively in [5]. In the rest of this paper, we adopt those terms.

3.4 What is the adaptation decision model employed?

The way that adaptation decision is made can follow several different models, namely *selection*, *transformation* [11] and *hybrid adaptation*.

- **Selection** is the process of choosing between a number of distinct, predefined alternatives. It requires that all the possible outputs of the adaptation function can be exactly predicted and statically defined a priori.
- **Transformation** is used when the possible output of the adaptation function cannot be predicted with a degree of accuracy that enables its classification in distinct categories. Thus, when this approach is adopted, the adaptation function forms its output by dynamically combining appropriate entities on the fly.
- **Hybrid adaptation** is a combination of the two paradigms described above. As in selection, there is a pre-existing number of alternatives. These alternatives are, however, highly customizable, so that the final output of the adaptation function is a result of transforming and combining them with suitable entities.

4. SERVICE PROVISION PLATFORM OVERVIEW

The proposed scheme for service adaptation has been validated in the context of a distributed software platform for the provision and management of value-added services over mobile networks in 3G and beyond[2] [6] [7]. The platform, called *Reconfiguration Control and Service Provision Platform (RCSPP)*, aims to address major issues regarding the deployment and management of services offered to users of next generation mobile networks. These applications are typically provided by third-party software vendors, commonly termed Value-Added Service Providers (VASPs). The platform's functionality comprises automated procedures for service deployment that include appropriate reconfiguration of the underlying network for optimal service delivery. In addition to that, an intelligent context-aware mobile portal is offered to the end-user, where procedures like service discovery, downloading and adaptation are fully tailored to terminal capabilities, user preferences and network characteristics.

[2] An earier version of this platform has been developed in the frame of project IST-10206 MOBIVAS (MOBIle Value-Added Services), http://mobivas.cnl.di.uoa.gr.

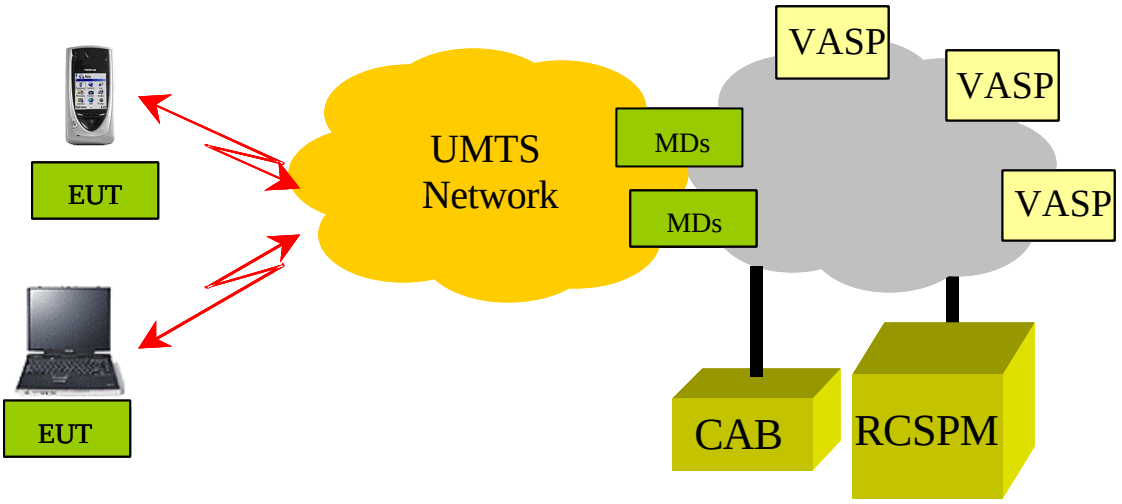

Figure 2. Architecture for flexible service provision in 3G networks.

The architecture of the platform is depicted in Figure 2. The main components of this architecture are the following:

The *Reconfiguration Control and Service Provision Manager (RCSPM)* is the central platform component in that it co-ordinates the entire service provision and management process. It includes modules that undertake on-line service deployment by VASPs, network reconfiguration, maintenance of service- and user-related data in suitable databases and repositories, as well as customized service discovery, downloading and adaptation.

The *Charging, Accounting and Billing (CAB)* [9] system is responsible for producing a single user bill for service access and apportioning the resulting revenue between the involved business players. This is accomplished by collecting metering data from standard core network elements as well as MDs.

The *End User Terminal Platform (EUT)* [10] resides in the user terminal and incorporates an execution environment for applications and functionality such as service downloading management, GUI clients for service discovery and selection and capturing of event notifications. The EUT, depending on the equipment available at the terminal (e.g., GPS devices, sensors), is able to communicate to the RCSPM information useful to the adaptation mechanism, like terminal capabilities and user status and location.

In our platform prototype we have also used *Metering Devices (MDs)*, which are essentially DiffServ-enabled IP routers with the additional capability of generating traffic monitoring records that are subsequently used in the charging, billing and accounting process.

As mentioned in the preceding paragraph, the RCSPM is the main component of the proposed architecture. It is responsible for a number of important tasks including mechanisms for application deployment and data management, service discovery by end-users, VASP and user authentication and adaptation of service provision to user preferences and terminal characteristics. The architecture of the RCSPM is depicted in Figure 3.

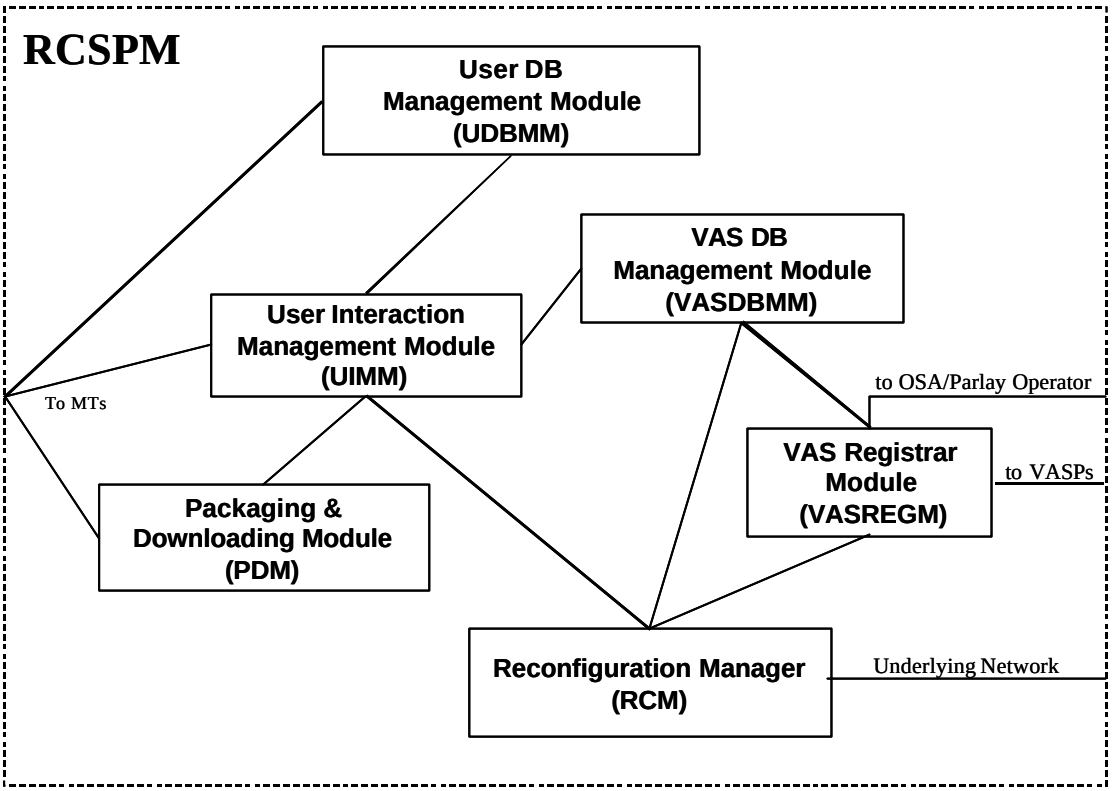

Figure 3. RCSPM internal architecture.

In the following paragraphs, the functionality of the main modules of the RCSPM is briefly presented.

The ***User Interaction Management Module (UIMM)*** is responsible for providing the user with a highly personalizable, context-aware mobile portal. It co-ordinates user-related operations like service discovery, selection, adaptation and downloading. To accomplish these tasks the UIMM keeps track of user session information, such as user context data.

The ***VAS Registrar Module (VASREGM***) is responsible for interacting with 3rd party service providers. Through the VASREGM the platform operator provides VASPs with a way to automatically deploy their services. The VASP compiles a descriptor, encoded in XML, of service attributes. Based on these attributes, the VASREGM co-ordinates service deployment, including various actions like reconfiguration of the underlying infrastructure and uploading of service components to the RCSPM (see Section 5 for details). The service provider is able to manage (add/delete/update) its services via a convenient web interface.

The ***Reconfiguration Manager (RCM)*** undertakes network, platform and service reconfigurability. The RCM is responsible for executing the appropriate reconfiguration actions on the underlying network during VAS management procedures (registration/de-registration/update), triggered by the VASP. The RCM also comprises a generic adaptation module [17] [18] that is used for supporting service adaptation through functions like intelligent profile matching.

The ***Packaging and Downloading Module (PDM)*** is responsible for dynamically creating a single bundle that contains all the software components and other supporting resources (e.g., images, etc.) required for executing a service and for making it available for download to the mobile client. The single archive

produced is dynamically tailored to the context of the particular VAS selection request (see Section 5 for a more elaborate discussion).

The ***Value-Added Service Database Module (VASDBMM)*** and the ***User Database Management Module (UDBMM)*** are essentially front-ends to databases that maintain profile information about service and users, respectively.

5. REALISING APPLICATION ADAPTATION

In this section we elaborate on a service adaptation mechanism that has been incorporated in the RCSPM. First we provide an overview of this particular case of service adaptation functionality and the corresponding design choices that we made during its design and implementation. Next, we describe the internal architecture of the PDM module, which plays a vital role in our mechanism. Finally, we illustrate the service adaptation procedure in the form of a sequence of interactions between RCSPM components.

5.1 Overview and important design choices

In the proposed mechanism, adaptation affects two features of the application:

- The constituent components of the application, which may vary for different user requests. These could be variations in functionality between distinct service versions (e.g., a multimedia player could come in an audio/video as well as a plain audio version) or different implementations of the same functional entities (e.g., more efficient codec implementations could be used over low capacity wireless links).
- The physical location of the application components and in particular the split between the part that is downloaded and runs on the terminal and the (possibly multi-tiered) part that resides in the server-side, fixed infrastruc ture. Given the wide range of terminals with diverse capabilities that are expected to be used in systems beyond 3G, highly dissimilar separations in service stationary and movable parts can be optimal in different situations.

Individual decisions regarding both the above aspects are made and applied for every service selection request, based on the current service provision context (e.g., terminal/network characteristics, user preferences and status information).

The above adaptation tasks are performed each time a user selects a particular service for downloading and do not further intervene with the application during execution. Thus, this is a case of *deployment-time* adaptation.

Regarding the adaptation decision model employed, we have chosen hybrid adaptation. One can identify the inefficiency of the pure selection approach and the obvious need for dynamically enriching and customizing the offered services with components appropriate to the requesting user and the underlying terminal and network. On the other hand, the creation of the customized service version that will be downloaded and executed to the terminal typically requires application-specific input from the human user that cannot be derived from the user profile. However, the user interface should be as simple as possible and not overwhelm the user with a multitude of choices and parameters. Thus, we propose the existence of predefined application versions, which can be dynamically tailored to user preferences by choosing an appropriate implementation and adding appropriate service components.

It is worth noting that the above procedures are orthogonal to the service; the implementation of its components is totally agnostic of them. Thus, this is an example of *application-transparent* adaptation. Even in view of this fact, an important design issue, specific to service adaptation, is where (outside the

service) this application adaptation and bundling function is performed. In the wired Internet world, where terminal connectivity constantly becomes cheaper, higher data rate and more reliable, while the environmental conditions are relatively static, current approaches (see [12] [13] [14]) propose that individual components constituting or supporting an application are downloaded separately by the client (Figure 4 (a)). In a wireless network, however, this leads to significant overhead as far as the usage of the scarce resource of bandwidth is concerned. Moreover, the frequently changing context data is collected from distributed sources and cannot be easily tracked solely by the terminal. Thus, we have chosen to place this function at the server-side (Figure 4 (b)), where all executables, libraries and supporting software that are required to run an application are packaged in a single file and downloaded by the terminal over a single network connection. Moreover, this task is handled by the service provision platform, which already incorporates the required sophisticated mechanisms for context retrieval and therefore does not put the corresponding burden on the VASP. The major advantage of this approach is that a client is able to download the archive over a single network connection, thus avoiding superfluous network traffic and excessive signaling, without any need for the application to be enhanced with extra functionality.

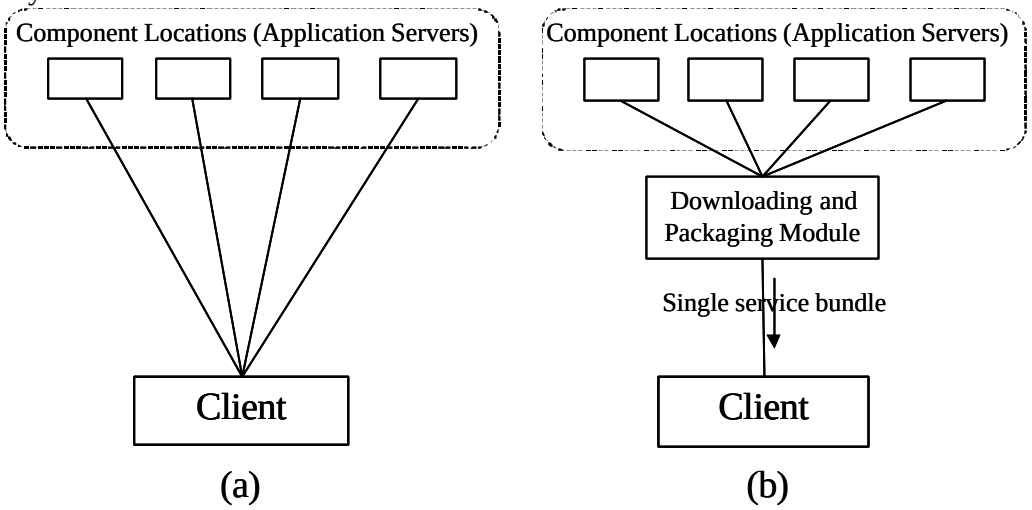

Figure 4. The different alternatives for the location of the component downloading and application packaging functions: (a) Client-based, like in existing approaches in the Internet, (b) The proposed, mediator-based approach.

5.2 PDM internal architecture

The core module for the service adaptation mechanism presented in this paper is the PDM, whose high-level internal architecture and interfaces is depicted in Figure 5.

The main internal PDM components are the following:

- *Repository*. This component acts as a local store for the various packages and resources that are useful for the packaging of one or more services. The PDM-internal maintenance of this repository enables faster retrieval and packaging of application components.
- *Registrar*. This component enables VASPs to register their packages and service components to a repository maintained by the PDM. This procedure is part of the VAS registration operation that is offered to the VASPs by the RCSPP [7].

- *Pre-processor*, which transforms, if necessary, service software descriptions of various formats (e.g., OSD, JNLP, CIM) to the internal application meta-data format used by the PDM.
- *Packager*. This component creates a single downloadable service bundle according to the package format desired (e.g., JAR file). This could require interaction with other RCSPM components for making intelligent adaptation decisions. Note that this procedure is independent of the interfacing (CORBA IDL, WSDL) and communication (CORBA, SOAP/XML) mechanisms employed within the application itself.
- *Retriever*, which is employed for the efficient retrieval of executables and resources from the local repository and remote locations.
- *Dispatcher*, which forwards incoming messages to the appropriate internal component.

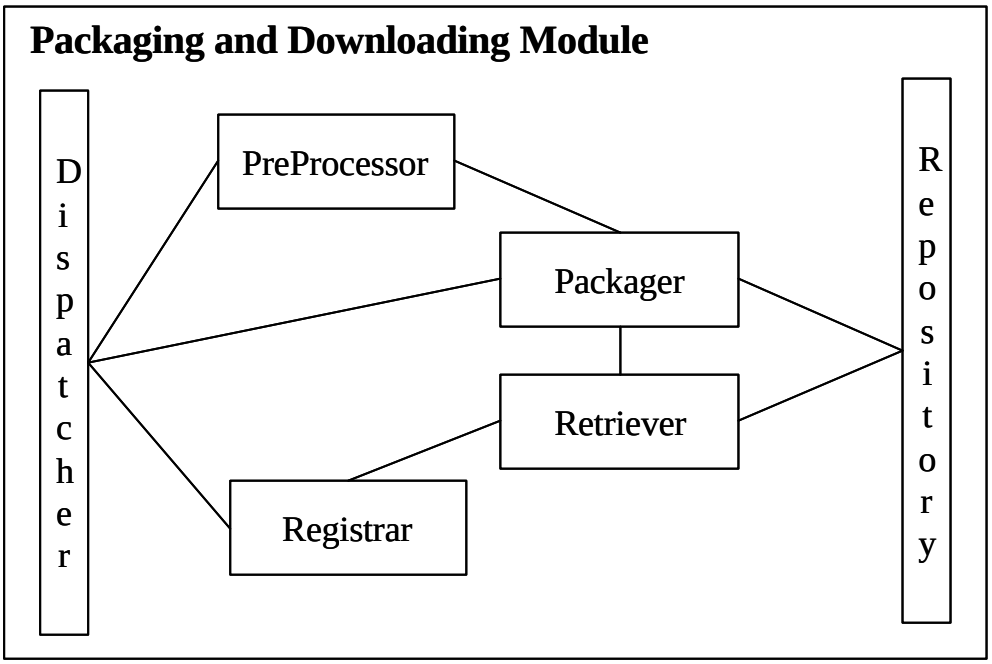

Figure 5. PDM internal architecture.

5.3 Example interaction sequence

In the following, we describe in detail the sequence of steps through which service adaptation is accomplished. This is illustrated in Figure 6, where a sequence diagram is presented, including all the interactions between the user terminal, the service provision platform and the VASP, required for user login/logoff, service discovery, selection, adaptation and downloading. We assume that before the beginning of the depicted interaction, the VASP has registered the selected service with the platform and has provided service metadata, including the description of the service decomposition into components and the network locations from where those can be downloaded. Optionally, the VASP may upload the service components to an appropriate component repository maintained at the PDM.

The sequence is performed through the following steps:

1. The user logs in the platform, an operation that includes user authentication.

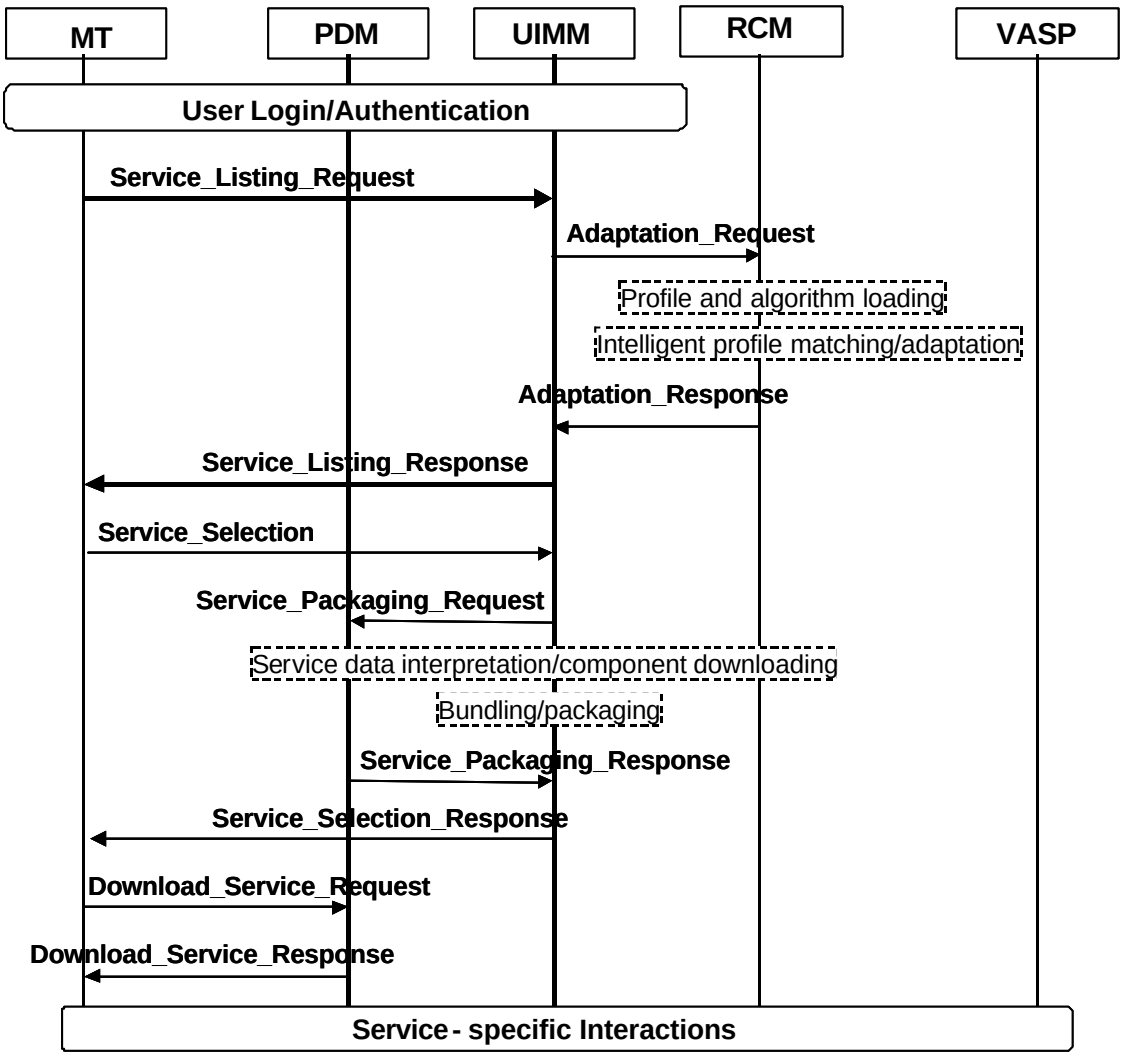

Figure 6. Sequence diagram for service discovery, selection and downloading.

2. The user issues a request for a service listing, with criteria like service category and keywords. Terminal capabilities and possibly also network characteristics are sent to the UIMM as parameters of the request.
3. The UIMM retrieves from the service database the profiles of applications that match the user query and formulates the service listing according to terminal capabilities, network characteristics and user preferences. To achieve that, it collects all the required profile data and forwards it to the adaptation module in the RCM, which performs the necessary matching, by identifying the services/service versions that are suitable for the specific user/terminal/network and also determining the appropriate configuration (e.g., inclusion of optional components or not) of each service version. The customized listing is returned to the user.
4. The user selects a service from the list.
5. The UIMM triggers the PDM to perform packaging of the adapted service and after completion of this

task, returns the URL of the corresponding customized package to the user.

6. The user downloads and executes the customized service. Service execution typically results in interactions between the downloadable and stationary service parts.
7. The steps 2-6 can be repeated for as many times the user desires, for discovering and accessing other services. At the point where the user does not wish to use any other applications, he logs off the platform.

6. SUMMARY – CONCLUSIONS

Application adaptation has been widely recognized as a critical enabler of flexible service provision over wireless systems beyond 3G. The present contribution discussed the concept of service as it is shaped by the recent and anticipated developments in mobile communication networks, defined service adaptation, presented a taxonomy of relevant important technical issues and elaborated on a proposed mechanism for application-transparent, deployment-time service adaptation that has been successfully incorporated and tested in a distributed service provision platform.

7. ACKNOWLEDGEMENTS

This work has been partially performed in the framework of the project "ANWIRE" (www.anwire.org), which is funded by the European Community under the contract IST-2001-38835.

REFERENCES

[1] UMTS Forum Reports No. 9, 10. 11, available from http://www.umts -forum.org/.
[2] N. Houssos, M. Koutsopoulou, S. Schaller, "A VHE architecture for advanced value-added service provision in 3rd generation mobile communication networks", Proceedings of *the Globecom 2000 Workshop on Service Portability and Customer Services Environments*, San Francisco, USA, November 2000, pp. 69-79.
[3] A. Lazar, "Programming Telecommunication Networks", *IEEE Network Magazine*, October 1997.
[4] A K. Dey, "Providing Architectural Support for Building Context -Aware Applications", PhD thesis, College of Computing, Georgia Institute of Technology, December 2000, available at http://www.cc.gatech.edu/fce/ctk/pubs/dey-thesis.pdf
[5] M. Satyanarayanan, "Mobile Information Access", *IEEE Personal Communications Magazine*, February 1996.
[6] N. Houssos, et al., "Value-Added Service Management in 3G networks", *IEEE/IFIP Networks Operations and Management Symposium (NOMS 2002)*, 15-19 April 2002, Florence, Italy, pp. 529-545.
[7] N. Alonistioti, N. Houssos, "The need for network reconfigurability", in Markus Dillinger et al. (Eds.), "Software Defined Radio: Architectures, Systems and Functions", John Wiley & Co., May 2003, pp. 47-72.
[8] T. Harbaum, et al., "Layer 4+ switching with QoS support for RTP and HTTP", Proceedings of *Globecom 1999*, Rio de Janeiro, Brazil, December 1999.
[9] M.Koutsopoulou, C. Farmakis, E. Gazis, "Subscription Management and Charging for Value Added Services in UMTS Networks", *IEEE Semiannual Vehicular Technology Conference VTC2001*, May 2001, Rhodes, Greece.
[10] O. Fouial. K. A. Fadel, I. Demeure, "Adaptive Service Provision in Mobile Computing Environments", *4th IEEE International Conference on Mobile Wireless Communication Networks*

(MWCN 2002), Stockholm, Sweden, 9-11 September 2002.
[11] Kynn Bartlett, CC/PP, http://www.ccpp.org.
[12] K. Hoeltman, A. Mutz, "Transparent Content Negotiation in HTTP", IETF RFC 2295.
[13] Java Web Start technology, Java Network Launching Protocol (JNLP), http://java.sun.com/products/javawebstart/.
[14] The Open Software Description Format Specification, W3C Note, available at
[15] http://www.w3.org/TR/NOTE-OSD.html.
[16] Erich Gamma et al., "Design Patterns: Elements of Reusable Object Oriented Software", Addison Wesley Longman, Inc.
[17] N. Houssos, S. Pantazis, A. Alonistioti, "Generic adaptation mechanism for the support of context-aware service provision in 3G networks*", 4th IEEE Conference on Mobile and Wireless Communication Networks (MWCN 2002)*, Stockholm, Sweden, 9-11 September 2002.
[18] N. Houssos, et al., "Advanced adaptability and profile management framework for the support of flexible mobile service provision", *IEEE Wireless Communication Magazine*, August 2003, in press.