A REFERENCE MODEL for ADAPTABLE SERVICE PROVISIONING

I. Demeure[1], S. Gessler[2], N. Alonistioti[3], N. Houssos[3]

[1] *Ecole Nationale Supérieure des Télécommunications, CNRS UMR 5141 – LTCI, France,*
isabelle.demeure@enst.fr
[2] *NEC Europe Ltd, Network Laboratories*
gessler@netlab.nec.de
[3] *University of Athens, Communication Networks Laboratory*
{nancy | nhoussos}@di.uoa.gr

Abstract. This paper reports on an activity, conducted as part of the European IST Project ANWIRE, towards the identification of requirements and the main functional building blocks for middleware architectures for personalized, context-aware and adaptable services in wireless networks. It introduces a generic service adaptation loop. It then present an adaptation reference model involving the service elements derived from the analysis of the adaptation loop.

1 INTRODUCTION

This paper reports on an activity towards the identification of requirements and the main functional building blocks for middleware architectures for personalized, context-aware and adaptable services in wireless networks. We introduce a generic *service adaptation loop* from which we derive a collection of adaptation service elements. We then present an adaptation reference model involving the identified service elements.

The results presented here were obtained as part of the European IST Project ANWIRE (Academic Network for Wireless Internet Research in Europe, IST-2001-38835 ANWIRE) [1]. It aims at organizing and coordinating parallel actions in key research areas of Wireless Internet and Reconfigurability and promoting and disseminating Wireless Internet and Reconfigurability solutions, in order to make them available to the research and industrial community. One of those key research areas is dedicated to "Architectures for Adaptable Services" [2].

2 The SERVICE ADAPTATION PROCESS

The definition of adaptation, which denotes *adaptation as* "*change in the system to accommodate change in its environment"*, clearly shows the requirement for a service adaptation middleware to consider the actual environment (or context) in which a service is used.

Adaptation operations may include a wide variety of actions such as modification of operational parameters, update/replacement and physical re-allocation of service components. Another adaptation realm is the *content* of the service: *syntactic* adaptation concerns the format and syntax of the content provided to the user. It can be done through QoS adaptation of streams and by data transcoding. *Semantic* adaptation relates to the meaning of the adapted content (e.g., localization/personalization, filtering for security / privacy issues). It is done, for example, by a simple re-parameterization of the data, by data filtering and also by a major replacement of content.

An analysis of typical adaptation processes show that they all follow the same scheme, independent at which time of the service lifecycle, design time, deployment time or execution time, they are performed. This overall scheme for a service adaptation process can be modeled by an *adaptation loop* represented in Figure 1. Note that the adaptation loop captures the adaptation process as performed both at initialization time and at runtime.

In the figure, "box 1" depicts the service and its execution environment. In this paper the term "execution environment" is purposely used in a broad sense covering:

- User preferences, location, identity, presence information, history.
- Physical and social environment such as presence of other people, other devices, indoor or outdoor operation, date and time.
- Technological environment such as terminal capabilities and, network infrastructure capabilities.
- Service characteristics, configuration and settings.

Prior to any adaptation action, the system in charge of adaptation must be aware of the value of various execution environment parameters and of any change in this execution environment. For example, if a user decides to continue executing a service on a device that is different from the device on which the service was initially started, then there is a significant change in the environment corresponding to that of the operating device. A variation in user's connectivity due for example to the user mobility is an other example of modification of the execution environment.

Some changes in the environment may not be relevant to the adaptation process and should be ignored. Some changes may only be relevant if correlated to other changes. This is why the parameters to be monitored must be identified, and all changes perceived in the execution environment must be analyzed. This

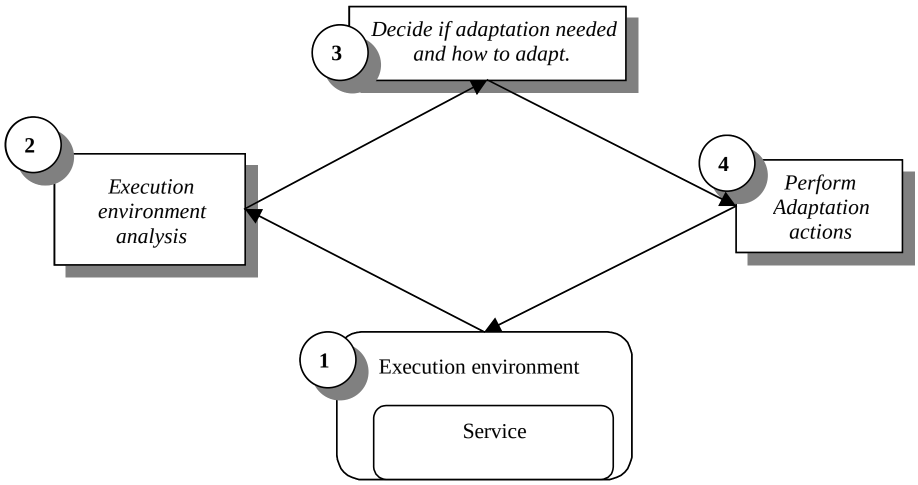

Figure 1-Service Adaptation Loop

intelligent environment monitoring and analysis functionality is captured by "box 2" in Figure 1 that materializes the need for the adaptation logic to always be able to capture and interpret the changes in the execution environment.

If the analysis concludes that an observed context change should lead to service adaptation, then the system must decide of the nature of the adaptation to be performed and how they should be performed. This is depicted by "box 3". Notably, the environment analysis phase is closely related to the adaptation decision-making and could be considered as a pre-processing function for it. The decision of whether or not an adaptation is needed is usually made with respect to clearly defined adaptation policies, which may originate from various stakeholders of the overall service provision process (service developers/providers, infrastructure operators, users). Adaptation policies are generally expressed in the form of a collection of "if *condition* then *action*" statements. The decision box may therefore prescribe a series of actions that correspond to service adaptation.

The actual execution of the adaptation actions is depicted by "box 4". Adaptation may be relative to the service behavior, presentation or content. Examples of adaptation actions are changing execution parameters, service replacement, service component addition, removal or replacement, service component migration, service composition, media stream adaptation, data filtering and transcoding.

3 SERVICES ELEMENTS REQUIRED by the ADAPTATION PROCESS

The adaptation process that we just described assumes that a number of basic services or *service elements* are available from the *service adaptation middleware*. Furthermore, the adaptation process cannot be carried out without a number of important supporting services that are complementary to the main adaptation functionality. In this section, we provide the present results of our attempts at identifying these services.

3.1 MAIN ADAPTATION SERVICES

The main adaptation services presented here implement the basic functionality of the adaptation loop introduced in section 2 and form a service adaptation middleware that is able to implement service adaptation management based on policies by various stakeholders. It is notable that all these components are administered and configured by the *Administration and Configuration* function that enables the interaction of the adaptation middleware with its owner as well as third parties (e.g., application developers).

3.1.1 EXECUTION ENVIRONMENT ANALYSIS

Facilities are required to monitor, filter and analyze the changes in the execution environment that are significant for adaptation. The service adaptation middleware uses existing infrastructure and capabilities to retrieve environmental information (e.g., open network APIs, context servers). From the middleware point of view these functions act as service enablers for the adaptation service. The added functionality related to execution environment handling concerns processing of environment data. This processing may include:

Preprocessing, that is transformation from various interchange formats (e.g., XML, RDF) to a form that facilitates unified handling. This is particularly important, since the environmental information originates from a variety of sources and is unavoidably represented in diverse formats [Hou03].

Aggregation. This function refers to summarising, consolidating and transforming the context information to a higher level of abstraction. For example, the fact that a user is located at a specific set of coordinates can be translated to the fact that he is at home or at work. This way the environment data becomes easier to use and exploit by the adaptation decision function.

Inference/Reasoning. This function refers to infering new contextual information on the basis of available environmental data. For example, the co-existence of an employee in a company room with many of his colleagues together with the fact that the room projector is activated, may lead to the conclusion that the employee is in a meeting and cannot answer incoming voice calls.

3.1.2 POLICY-BASED ADAPTATION DECISION-MAKING

Adaptation decisions are based on environmental information (e.g, terminal capabilities, user preferences and status, network characteristics, ambience). Their main output is the specification of the optimal new state of the adaptable entity, which is reflected in the entity's *profile* (metadata).

An important issue that should be taken into consideration relates to the fact that in general the adaptation decision logic is not monolithic; it typically consists of small, autonomous adaptation decision algorithms [1]. Thus, a crucial task for the co-ordinating part of the adaptation decision logic is loading the appropriate algorithms for serving every decision request. The algorithms can be loaded from appropriate repositories. These context-dependent algorithms, which actually comprise the *policies* of the owner/administrative authority of the adaptable entity (e.g., service developers or providers in the case of mobile applications). Thus, mechanisms to describe, deploy and maintain adaptation policies for various parties (e.g., service providers) need to be established [2]. In our architecture, this function is accomplished through the administration and configuration component of the adaptation middleware.

Adaptation policies are generally expressed in the form of a collection of "if *condition* then *action*" statements. The adaptation decision logic should ensure that the applied adaptation policies do not compromise the stability of the system (e.g., the system does not oscillate too quickly between a number of transient states).

3.1.3 POLICY-BASED ADAPTATION INITIATION and CONTROL

The control of the adaptable service enables the service adaptation middleware to perform a variety of adaptation operations, as prescribed by the adaptation decision logic. Examples of such operations are: modification of operational parameters, update/replacement and physical re-allocation of service components. These operations are accomplished through *"adaptation enablers"*.

A portion of this functionality may be highly dependent on the type of adaptation (e.g., adaptation at run-time, content adaptation). It comprises the following functional elements:

Actuation co-ordination. This function, based on the adaptation decision and the type of adaptation required, triggers and co-ordinates the appropriate adaptation components.

Adaptation actuation. This function is dependent on the type of adaptation. Separate functionalities for this task may be implemented for each type of adaptation. The adaptation actuation function is further divided into two parts:

- Actuation decision. This relates to *how* adaptation actions shall be activated/performed.
- Adaptation entity control. The control of the adaptable entity (e.g., a service) enables the service adaptation middleware to perform a variety of adaptation operations, as they are prescribed by the adaptation decision logic. Such operations may include a wide variety of actions such as modification of operational parameters, update/replacement and physical re-allocation of service components. These operations are accomplished through appropriate functions aka "adaptation enablers". The availability of these enablers is taken into consideration during the adaptation decision-making process as well as the actuation decision.

3.2 SUPPORTING SERVICES

This section relates to services that may not be part of the main adaptation logic, but are crucial for enabling adaptation.

3.2.1 EXECUTION ENVIRONMENT MONITORING and DATA RETRIEVAL ELEMENTS

Facilities are required to retrieve data relative to the execution environment that will be subsequently used for its analysis. Among these facilities is the information relative to the infrastructure, e.g. the network QoS or the terminal capability, and such things as a location service that gives the location of the user and a presence service that provides information about other users and devices in the environment; further information sources which provide physical conditions of the environment like weather, daylight, temperature or even biometric information (blood pressure etc) may also be available.

3.2.2 PROFILING

Facilities are required to capture and manage various profiles including: user profile, role and preferences (for personalization); terminal capabilities; network infrastructure capabilities; service description. In ANWIRE, as of today, we define five profiles: terminal profile, user profile, network profile, user access profile (relative to access network) and service profile.

We are currently discussing the introduction of an additional profile relative to information on the ambience. A unified way to process and interpret the various profiles taken into consideration is an important issue.
Various profile representation formats and mechanisms shall be supported. An ontology scheme could be used for profile representation. In such a scheme, resources like user behaviour and preferences, terminal equipment capabilities, and network module features can be represented in a combined graph model using the Ontology Web Language (OWL) and be seamlessly invoked by the Web Service infrastructure[6]. OWL has more facilities for expressing meaning and semantics than XML and RDF. Thus, OWL goes beyond these languages in its ability to represent machine interpretable content on the Web. It is equivalent to the Description Logic mathematical model. Hence, OWL is very powerful in that it can describe anything that can be expressed in description logic. This way, lower-level entities can be illustrated in the same way as higher-level entities.

3.2.3 SERVICE MANAGEMENT

The service management requires the ability to *describe* services, service components and service compositions, to *discover* services and service components, to *create*, to *compose* and to *deploy* services (components), and last but not least to *control* the service (component) execution.

3.2.4 SERVICE ADAPTATION ELEMENTS

The Service Adaptation Elements represent the 'Point of Service' for the actual service adaptation functions.

3.2.5 OPERATION SUPPORT

Adaptation processes must be coordinated with the operation of the services, since service adaptation may require re-synchronization with the service operation itself. This includes aspects such as QoS provision, charging/billing and accounting services, provisioning of privacy and security on various levels, support for interdomain roaming and also with session management issues.

4 An ARCHITECTURAL VIEW

Having identified the main service elements required for an adaptation service middleware, we designed a reference model for adaptation service architecture, with the scope for the support of mobile services delivery. As a starting point we took the view proposed by the WWRF (Wireless World Research Forum) and worked the “adaptation enabler” aspect into the WWRF service framework (also known as “cake”) [4]. The result is shown in Figure 2.
The core adaptation functionality in our model is contained in an Adaptation Kernel, which represents the center of the model. and is made up by the *main service enablers* identified in section 3. It decides on the necessity of an adaptation action, plans it and finally initiates and controls the desired adaptation process. It also provides the logic and technology for the necessary interaction with the service operation.
The Kernel interfaces with generic Service Elements, which provide either access to environment monitoring systems (e.g., User profiles, Location Services, Network monitoring, ambience sensors) or to adaptation service enablers (e.g., Network Management, Media Transcoder, Terminal/UI Configuration Management). Notably, these Service Elements are independent from the Kernel and not dedicated to adaptation. However, they enable the Kernel to perform the adaptation service and their availability is therefore required by any adaptation system. Figure 2 also depicts the importance of a variety of profiles for a service adaptation process.
This architecture reference model was first presented at the last WWRF meeting in Beijing in February 2004 and is now under refinement [5].

5 ACKNOWLEDGEMENTS

This work has been produced in the framework of the project “ANWIRE” (www.anwire.org), which is funded by the European Community under the contract IST-2001-38835. We are grateful to all task force 4 participants whose contributions have inspired parts of this paper.

6 REFERENCES

[1] ANWIRE (Academic Network for Wireless Internet Research in Europe), IST-2001-38835, http://www.anwire.org/

[2] ANWIRE TF4, Isabelle Demeure (ed.) et al., UK. IST-2001-38835 ANWIRE Deliverable D1.4.1, Adaptable Service Architectures. September 2003.

[3] N. Houssos, A. Alonistioti, L. Merakos, “A scheme for the introduction of 3rd party application-specific adaptation features in mobile service provision”, in J. Stefani, I. Demeure, D. Hagimont, Proceedings of the 4th IFIP Conference on Distributed Applications and Interoperable Systems (DAIS 2003), Paris, France, Springer, Lecture Notes in Computer Science 2893, November 2003, pp. 15-28.

[4] F. Carrez, “Generic Service Element White Paper”, Presentation at WWRF #9, Zurich. 1.-2. July 2003.

[5] I. Demeure, S. Gessler, N. Alonistioti, N. Houssos. "ANWIRE visions of architectures for adaptable services in the wireless Internet", WWRF Meeting, Beijing, February 2004.

[6] D. L. McGuinness, F. van Harmelen (eds.), "OWL Web Ontology Language Overview", W3C

Recommendation, 10 February 2004, available at http://www.w3.org/TR/owl-features/.

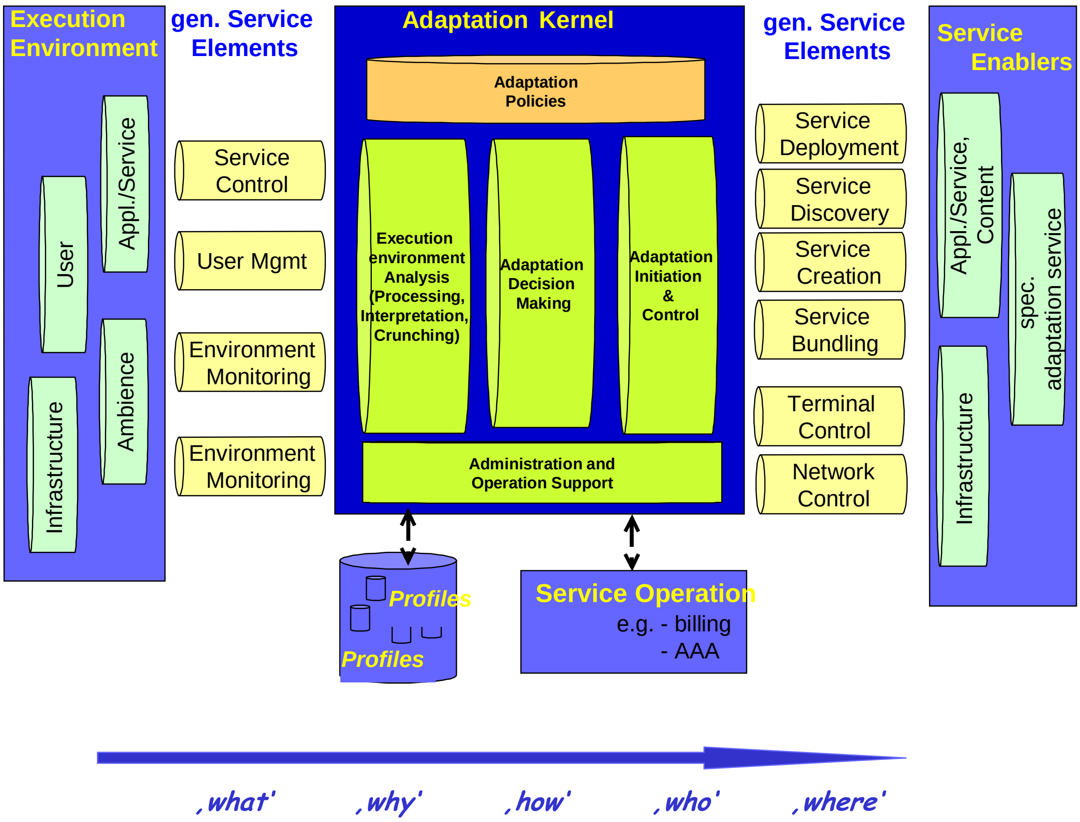

Figure 2 ANWIRE Adaptation Reference Model based on the WWRF Framework