A Service Continuity Layer for Mobile Services

Ivar Jørstad, Do van Thanh, and Schahram Dustdar

***Abstract*—This paper presents and discusses models of generic mobile services. The primary goal is to gain understanding of the challenges in designing, developing and deploying advanced mobile data services. Two types of models are introduced. First, a composition model describing the components of a generic mobile service and the components relationships are given. Second, distribution models describing the distributions of the components in the former model across hosts, networks and domains are presented. A brief mobility analysis is carried out, followed by a discussion of mobility and service continuity dependency on the service distribution. The functions necessary to provide service continuity are identified and incorporated in a service continuity layer.**

***Index Terms*—Service continuity, service composition, service distribution, generic mobile services, personalization, UML**

I. Introduction

TILL now, the focus in mobile communications has been on providing service continuity of communication services when a mobile terminal is roaming between networks, i.e. avoiding abrupt access. The underlying mechanism for achieving service continuity is called handover or handoff. Handover was first implemented between networks of same type, e.g. GSM and is gradually extended to networks of different types, e.g. handover between GSM and WLAN. With the increasing number of devices that users have at disposition, it is quite relevant to provide service continuity across heterogeneous devices. A user having multiple devices at disposition may wish to move a service from one device to another one to have better user interface or simply to reduce the usage costs. For example, a user, when arriving to the office, may want to transfer the conversation session from the mobile phone to the multimedia PC acting as an IP phone. The goal of this paper is examine how service continuity can be provided. The paper adopts a formal and analytical approach. It starts with an analysis of current services and derives the functions and capabilities that are necessary to achieve service continuity.

Jørstad, I. is with the Norwegian University of Science and Technology, Department of Telematics, O.S. Bragstads Plass, 7491 Trondheim, Norway (e-mail: ivar@ongx.org)

Do, T.v. is with Telenor R&D, Snarøyveien, Fornebu, Norway (e-mail: thanh-van.do@telenor.com)

Dustdar, S. is with Vienna University of Technology (dustdar@infosys.tuwien.ac.at)

II. Modeling Mobile Services

A high degree of mobility is a requirement for mobile services; they should by definition be available at any time, any place using any device with communication capabilities (thus supporting many types of mobility [1]). However, it is more and more important that access to mobile services can be moved from one device to another with least possible interruption. To be able to address these requirements and further study them, more formal definitions of the composition and architecture of mobile services are needed. This section starts with some definitions.

It is possible to model mobile services according to their composition or their distribution. Whereas the compositional model is concerned with the division of a service into discrete components according to their nature and role, the distribution model is concerned with the distribution of components of a service across devices, networks and domains.

A. Composition Model

A generic mobile service is commonly modeled as consisting of two basic components:

- Service Logic
- Service State/Data

Fig. 1 displays a UML (Unified Modeling Language [2]) Class Diagram showing the composition of a mobile service.

Service logic is the program code that constitutes the dynamic behavior and provides the functions of a service. Usually, this does not only consist of one autonomous unit, but in this model the service logic represents the collection of program code for a given service.

Service state contains data used in the execution of the service logic and reflects the state of it. They are for example variable values, temporal parameters, register values, stack values, counting parameters, etc. In order to provide service continuity and personalisation we propose to introduce two additional service components as follows:

- Service Content
- Service Profile

Service content refers to data that are the product of service usage. For example it can be a document written in a word

processor or the entries in a calendar. Service content can be produced or consumed by the user.

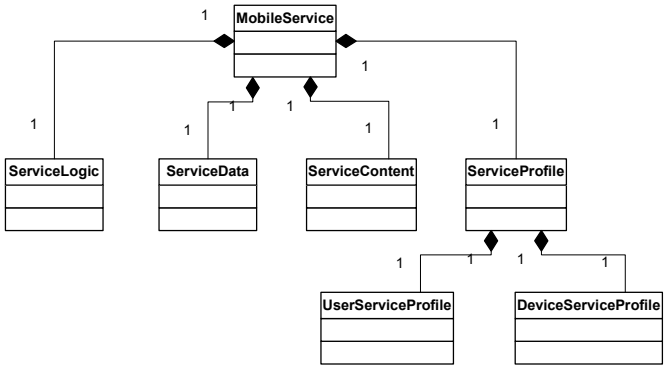

Fig. 1. Composition model of MobileService

A *Service profile* contains the settings that are related to the user or/and the accessing device. A service profile can further be divided into a *User Service Profile* and a *Device Service Profile*. The User Service Profile contains the user's settings that state how a service should behave for a specific user (e.g. what functions should be available), but it can also include personal information about a user that is used in accordance with a service (e.g. as input). The Device Service Profile states the properties and qualities of a particular device (e.g. form factor), so that a service can adapt (e.g. the presentation layer) to this device. The User Service Profile is stored or linked to the *User profile*, whereas the Device Service Profile is part of the *Device profile*.
The *User profile* can either be directly modified by the user or indirectly through the usage of the service. Alternatively, it can be implicitly altered by a separate service in the network according to the continuous usage pattern of a specific user.
All of the components of a mobile service as defined above can be subject to various distributions, as in any distributed system [3]. The most common models to describe the distribution of services are:

- Standalone (hereafter also called monolithic)
- Client-Server
- Peer-to-Peer
- Multiple distributed components

For each of these models it is possible to define finer grained, specialized models, but the above mentioned models suffice for this discussion.

B. Distribution Model

According to the Distributed Computing paradigm, the distribution of a service/application should be hidden and the mechanisms to support distribution are incorporated in the Distributed Computing middleware such that the developer can concentrate on the core functions of the application [4][5]. However, for mobile services, distribution plays a crucial role that must be considered at service design. Indeed, when the user is moving and is accessing services from different places, the location of a service and its components relative to the user's location will have great influence on its availability, quality, continuity, and personalization offerings.

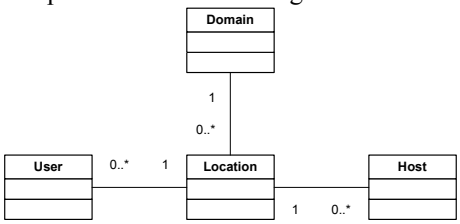

Fig. 2. Relationship between notions used in the distribution models

In order to model the distribution, we introduce the notions that are depicted in Fig 2. A user can be at one location at a time but one location may be visited by zero or more users. A location may have zero or more hosts. One or more locations belong to a domain. A domain is defined as a set of locations containing a number of hosts that are controlled by an actor. The access to the domain is controlled by this actor e.g. enterprise, home user, telecom operator, etc.

1) Monolithic services

The first distribution model in Fig.3 depicts a system where all components of the mobile service, i.e ServiceLogic, ServiceData, ServiceProfile and ServiceContent are installed in the same host which is located at the same location as the end-user. Such services can be called *monolithic*, as they constitute a single, autonomous unit.

Examples of this service type are word processors, spreadsheets, stand-alone games, calculator, etc. As shown in Fig. 3, if the user is at the same location as the host containing all the service components, he will have access to the service

2) Thin-client/Server services

The previous model is very restricted. This is partly remedied by the second model (Fig. 4) which splits the service logic into two parts; one generic part (GenericServiceLogic) and one specialized part (ServiceLogic). Service content, data and profiles are all co-located with the specialized part. The generic part is a thin-client presentation layer. Typical examples of the GenericServiceLogic are Telnet or rlogin.

3) Client/Server services

In the third model (Fig. 5), a model similar to the previous is defined. The difference is that while in the previous model, the client application (GenericServiceLogic) was generic and used for a lot of different services. In the client/server model, ServiceLogic1 is a client application specialized for a particular service.

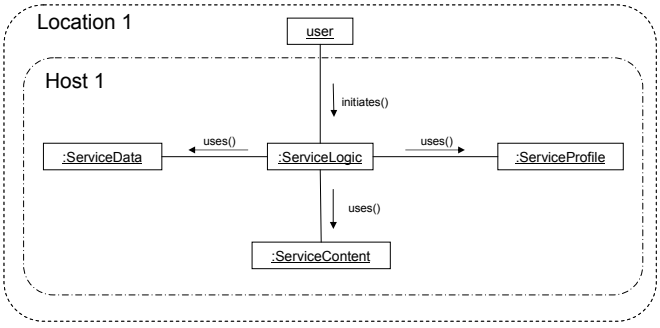

Fig. 3. Distribution model for monolithic mobile services

Both components have their own service data, and ServiceLogic2 has access to a service profile and the service content as well. As an option, ServiceLogic1 can also keep its own service profile and service content.

4) Multi-component services

A refinement, or rather specialization, of the previous model is the multi-component service defined in the model in Fig. 6. In addition to employing a client/server distribution, the server is further divide into two components. Such specialisation of the client/server model can proceed further, and therefore, such services can also be called client/composite-server services.

5) Collaborative services

The last model defined here, is the model for collaborative mobile services in Fig. 7. The special feature of this model, is that several users exist in different locations, accessing components on hosts in these locations and the service logic components communicate across locations to support collaborative work between the users. This model applies to typical peer-to-peer services.

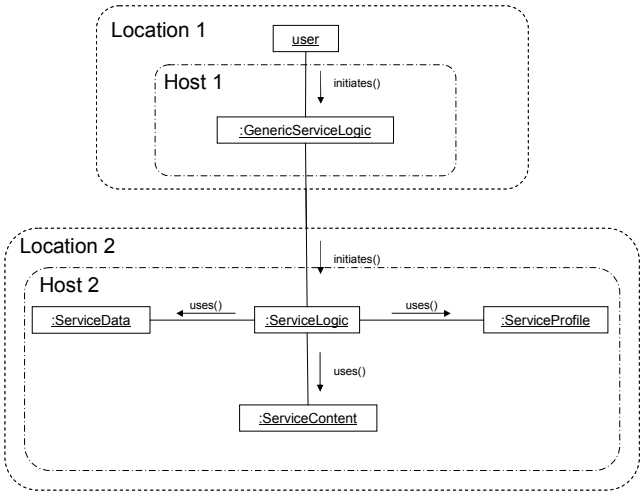

Fig. 4. Distribution model for thin-client/server mobile services

It is worth emphasizing that a service type is characterized by both the service composition and the service distribution.

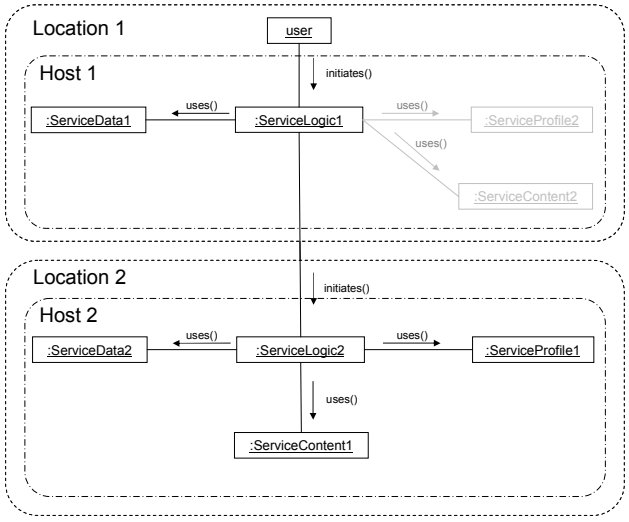

Fig. 5. Distribution model for client-server mobile services

C. Mobility Analysis

All the presented models are static and represent common views of distributed systems that do not regard user mobility as an issue. It is thus important to add and discuss the notion of movement of the user between different Locations, and to consider what effects these movements have on services of various types. Two possible effects are that the service is not available anymore in the new Location or that the service must take on another distribution. For clarity, we introduce the following axiom:

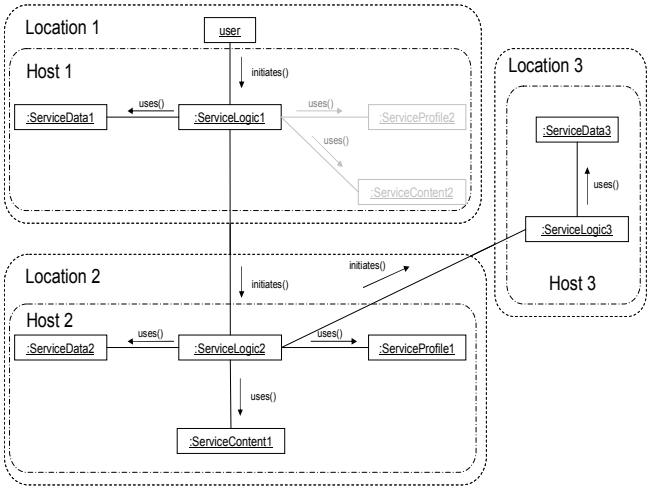

Fig. 6. Distribution model for multi-component services

Axiom 1: *"Mobility does not have any impact on service availability and continuity as long as the user moves together with all discrete components of the service."*

From this axiom, it follows that: *"Only changes in relative Location between user and parts of, or all discrete components of a service have impact on service continuity in that particular service."*

This means that the movements of the user will have different impact on each service type due to their distribution.

Whereas concepts like personal mobility and device mobility is usually concerned with the communication service at network layer (OSI layer 3), service continuity is a concept that supports for generic, data based services. Although mobility in the network layer is not entirely settled, other facets of services must also be considered. It is not given that the availability of a network service (IP) automatically allows service continuity for any service on a higher layer. Mobility and service continuity should thus be considered as supplementary to each other.

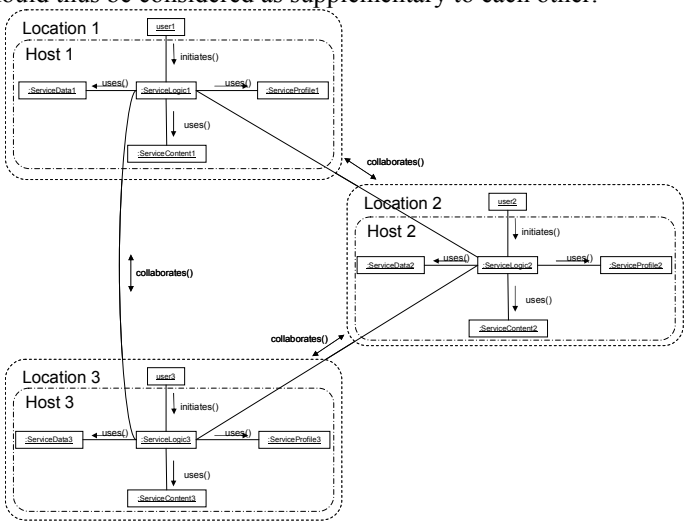

Fig. 7. Distribution model for collaborative mobile services

Service continuity is a composite concept. Initially, the concept can be broken into two types; seamless service continuity and non-seamless service continuity. We define seamless service continuity as:

"…the ability to pick up a service at a new Location, where service abruptness is bounded by the time it takes the user to move between the two Locations."

Non-seamless service continuity is defined as:

"…the ability to pick up a service at a new Location, where service usage can proceed from the 'point'/state where it was left at the previous Location, but where additional disruption is introduced due to a required reorganization of the service composition."

1) The Notion of Movement in UML

The basis for the notation used in the following analysis is UML collaboration diagrams. However, UML does not define a notion for movement, which is one of the most critical aspects in this analysis. The notion of movement is thus introduced using the stereotype <<moves>> along with a unidirectional association defining the direction of movement.

2) Service Type Specific Analysis

With service continuity defined, it is time to analyze service types further to deduce the requirements for service continuity support. The remainder of this section describes a usage scenario for a specific service type, and in particular how the movement of the user impacts the already defined service continuity aspects of the service. Based on this analysis, the next section will provide an initial framework for improved service continuity support in generic mobile services. Due to space limitation, only the client/server service type defined in Fig. 5 is considered. Mobile agents and mobile code [6] are technologies that earlier have been suggested as solutions to some of the challenges with service continuity.

Consider a scenario where UserX is accessing a service S1 in Location1 through Host1, using ServiceLogic1A. At one point, UserX moves from Location1 to Location2. The question is then how to ensure service continuity. There are two alternatives:

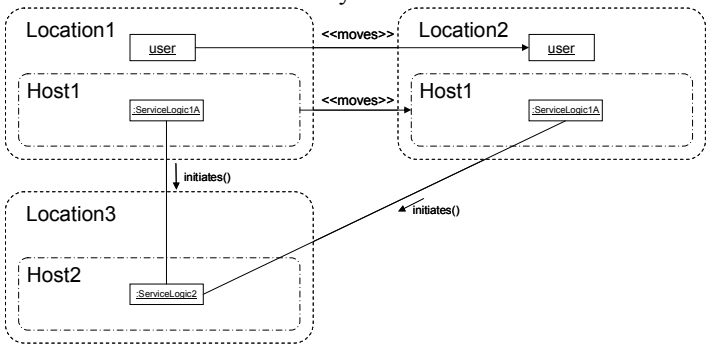

Fig. 8. User moves together with Host to a new Location

1. If Host1 moves together with the user, as depicted in Fig.8, there is no relative movement between the user and the service components that he is directly accessing. Service continuity is obtained by ensuring the continuity of communication between ServiceLogic1A's ServiceLogic2. This is the familiar case of the terminal handover that mobile communications have focused on. In these figures, other components of the service than service logic are excluded to ensure clarity and avoid cluttering. It is implicit that all the other components of the service are co-existing with the service logic as earlier described.

2. If Host1 is not moving together with the user, depicted in Fig. 9, it is obvious that it is not possible to realise seamless service continuity but only non-seamless service continuity. To be able to provide non-seamless service continuity, at Host2, there must exist a copy or an instance of the ServiceLogic1A, called ServiceLogic1B such that the user can initiate and continue the service. Typical example of such a case is a web browser, e.g. Internet Explorer which two instances are installed on two PCs.

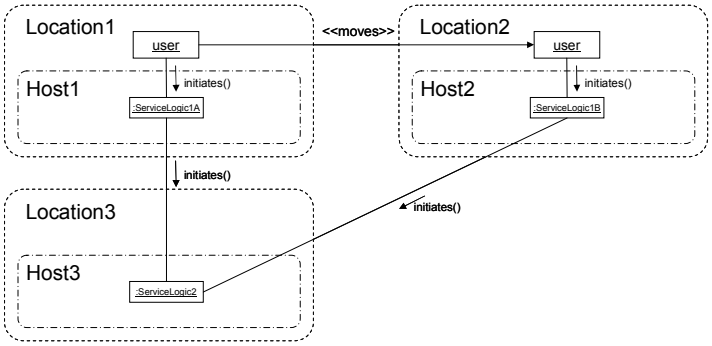

Fig. 9. User moves without Host1 to a new Location and re-initiates the service through a new Host2

Another possibility is that at Host2, an *equivalent implementation* of ServiceLogic1A is available, called ServiceLogic1C. Strictly speaking, the new implementation should be *compatible* with ServiceLogic1A to ensure complete service continuity. That means, the interface between ServiceLogic1C and ServiceLogic2 must be identical to the interface between ServiceLogic1A and ServiceLogic2 (e.g. the user changes from Internet Explorer in Location1 to Opera in Location2, and both speaks HTTP). To provide continuity, the ServiceLogic1B or ServiceLogic1C must be able to communicate with the Service Profile, Service Content and most importantly the Service State in order to resume the service from the point where the user changed from Host1 to Host2. A "Handover Manager" function is required and will be considered in the next section.

III. Support for Service Continuity

As we have seen, particularly two properties of mobile services influence service continuity:

- Service type (defined by service composition and distribution)
- Movement pattern of user

The second property that should be considered is the movement pattern of the user. Because service continuity is influenced by the user moving from one Location to another, it is important to analyze the ways of movement that are possible, which of them poses greatest requirements on the technology and what these requirements are.

- User moves between hosts
- User moves across locations
- User moves across domains

Movement between devices can further be divided into movement between devices of the same type or movement between devices of distinctive types.

A. Service Continuity Layer

As a result of the analysis performed in the previous sections, several functions that must be part of a service continuity layer can be identified. The service continuity layer can be seen as a management layer with support functions for realizing maximum service continuity and availability of generic services due to user movement relative to service components. The respective elements will now be listed and the tasks of each element will be briefly explained. The elements of the service continuity layer are:

1. Monitor: A continuously updated "map" of surrounding networks, domains and hosts
2. Handover Manager: High-level service handover
3. Interoperability Evaluator
4. Service Composition Module
5. Input/Output Redirector

Service continuity should be supported by appropriate middleware. Others have argued that the application layer is suitable for providing flexible solutions to handling service continuity and mobility issues [7]. A service continuity layer needs to have access to relevant information from the application layer (e.g. current state of service), but also from the lower layers (e.g. to infer decisions about network resources). The goal for the components mentioned above and described next, is to allow the system, not the user, to decide when to, and take care of how to, change the distribution of a service to accommodate movements of the user and maintaining service continuity and availability.

1) Monitor: Resource Map

There is a need for mechanisms and methodologies for describing surroundings of a host more detailed than only nodes in the current network (e.g. WLAN zone), which can be deduced from ARP requests on an Ethernet (i.e., neighbour discovery). There is a need for describing network boundaries, domain boundaries, restrictive elements (middleboxes [8]) etc. These must be supplied to the service continuity layer such that it can decide possible redistributions of a service to provide service continuity.

One protocol that can be part of such a protocol is defined by IETF in STUN [9] (Simple Traversal of User Datagram Protocol Through Network Address Translators).

2) Handover Manager: High-level Service Handover

High-level service handover has earlier been considered in [10]. Handover between cells in mobile telecom systems (GSM) is based on measurements of the surroundings (e.g. of received signal strength level) for the system to be able to act proactively as handover becomes necessary to provide a sustained service. Although a soft handover is not necessarily required for service continuity in data based services, the idea of monitoring surroundings could be applied here also, and performed by the service continuity layer. The monitor together with the handover manager can instruct the service composition module to design and implement a new service composition of an existing service as soon as it is recognized that a handover will be necessary, thus decreasing the abruptness time in service usage. Similarly, the

user can be warned if the service continuity layer sees no possibility to provide sustained access. Otherwise, the service handover should be automatic without any need for the user to be notified or take any action.

3) Interoperability Evaluator: Compatibility Matching

If a service is to be reorganized based on decisions by the monitor, a new service will be composed using either a replication of, or an equivalent of, each of the current components of the service. To ensure sustained service access, it must be ascertained that potential components for the new composition are interoperable with, and provide a satisfactorily equivalent interface as the current components. The task of the interoperability evaluator is thus to match compatible components based on both their *interfaces* and their *behavior*.

4) Service Composition

Based on what the Service Continuity Layer knows about the systems and the restrictions, it can dynamically compose a new service by using only components that have been validated by the element described in the previous subsection. In general, and as specified for XML Web Services, service composition can be either choreographed or orchestrated [11]. Put simple, choreography means that service components interact dynamically at runtime, without any coordinator. With orchestration a component acts as a coordinator and controls the process flow back and forth between service components. The former is more complex to realize, because it requires a lot of additional logic in each service component. The second can theoretically be applied to existing service components without changing them. Thus, orchestration seems to be the most feasible for the service continuity layer considered in this paper.

5) Input/Output Redirection between service components

Input/Output (I/O) redirection can be used between service logic components in a mobile service to avoid moving entire components around, when requirements otherwise would suggest this as a solution. A mechanism for performing the actual redirection must exist, and in addition, a generic way of representing I/O for transport between services must be defined to simplify the interfaces between service components. These challenges are already partly considered in orchestration and choreography of Web Services.

B. Conclusion

This paper initially describes the composition of generic mobile services and then provides an overview of possible distribution models for such services. The primary goal is to increase the understanding of how mobile services are organized (composed and distributed), in order to determine how support for service continuity and availability can be increased for these services. Having defined a possible composition and distributions of mobile services, the paper proceeds with a user-movement based analysis of one of the models. For the model chosen, it is suggested that service continuity can be assured if either a) a new instance of the initial service logic is available in the new location or b) a re-implementation of the initial service logic is available in the new location, and c) the communication between ServiceLogic1A and/or B and ServiceLogic2 can be reinitiated.

The result from this analysis suggests the requirements for ensuring support for service continuity in mobile services. This functionality is generalized and grouped into a Service Continuity Layer which consists of a) resource map of surroundings, b) high level service handover functionality, c) interoperability evaluation and compatibility matching, d) service composition and e) generic i/o redirection between service components (service logic).

To be able to describe and analyze mobile services, where movements of the user have great impact on the support for service continuity, a notion and a way of describing movement in the modeling phase is needed. Also, the notion of Location, Domain and Host is lacking. As part of future work, such a notion, for example for UML, should be elaborated. The user-movement based analysis should be extended, and further refinements of the Service Continuity Layer is considered future work, and so is the design and development of a prototype.

REFERENCES

[1] Jun-Zhao Sun & Jaakko Sauvola, On Fundamental Concept of Mobility for Mobile Communications, PIMRC 2002

[2] Martin Fowler, UML Distilled: A brief guide to the standard object modeling language, 3rd Edition, ISBN 0-321-19368-7, 2004

[3] George Coulouris et. al, Distributed Systems: Concepts and Design, 3rd Edition, ISBN 0-201-619-180, Addison-Wesley, Pearson Education, 2001

[4] ITU-T X.901 | ISO/IEC 10746-{1,2,3,4} Open Distributed Processing Reference Model Part 1,2,3 AND 4

[5] Kerry Raymond, Reference Model of Open Distributed Processing (RM-ODP): Introduction - kerry@dstc.edu.au - CRC for Distributed Systems Technology -Centre for Information Technology Research - University of Queensland Brisbane 4072 Australia

[6] Stefano Camapdello & Kimmo Raatikainen, Agents in Personal Mobility, Proceedings of the First International Workshop on Mobile Agents for Telecommunication Application (MATA'99). Ottawa Canada October 6-8 1999. World Scientific, pp. 359-374

[7] Proceedings of the Italian Workshop "From Objects to Agents: Intelligent Systems and Pervasive Computing" (WOA'03), Italy, ISBN 88-371-1413-3, September 10-11, 2003

[8] IETF, RFC 3303: Middlebox communication architecture and framework, August 2002

[9] IETF, RFC 3489, STUN – Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs), March 2003

[10] Thomas Strang, Claudia Linnhoff-Popien, Matthias Roeckl, Highlevel Service Handover through a Contextual Framework, MoMuC 2003, 8th International Workshop on Mobile Multimedia Communications, Munich/Germany, October 2003

[11] Chris Peltz, Web Services Orchestration: A review of emerging technologies, tools, and standards, Hewlett Packard, Co., January 2003