Capture, Reuse, and Validation of Requirements and Analysis Patterns for Mobile Systems

Rossana Maria de Castro Andrade

Thesis

Submitted in Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy in Computer Science
under the auspices of the Ottawa-Carleton Institute of Computer Science

School of Information Technology and Engineering (SITE),
University of Ottawa,
Ottawa, Ontario, Canada

In memory of my beloved grandmother,
Maria Teodora, who is an eternal source
of inspiration for her family and friends.

Abstract

In the mobile wireless communication domain, different systems apply common solutions to similar functional and architectural design problems. The recognition of these commonalities is a starting point towards reconciling differences and leading possibly to better ways to interwork these systems and to develop new ones. There is therefore a need for capturing these commonalities that can be fulfilled by the use of the pattern concept, which has been applied to document recurring problems and solutions in software engineering.

This research develops a method to extract such common problems and solutions and to document them as patterns. These behavioral and structural patterns are grouped into a pattern language in order to describe how they work together to solve recurring problems in the mobile communication domain. The examples given are related to mobility and radio resource management functions in second and third generation systems. The pattern language for **Mo**bility and **Ra**dio **R**esource management (MoRaR) makes it possible to generate different requirements and analysis models for mobile systems.

In addition, this thesis proposes an approach for reuse and validation of the MoRaR pattern language that includes a combination of different techniques at the early stages of the system development process and evolution. **U**se **C**ase **M**aps (UCMs) describe the requirements and the analysis models, which the **L**anguage **O**f **T**emporal **O**rdering **S**pecifications (LOTOS) is applied to formally specify and to validate these models. **M**essage **S**equence **C**hart**s** (MSCs) are used to represent the LOTOS validation traces at the design stage. These techniques provide confidence in the correctness of the pattern reuse. According to the system needs, mobile system designers are able to reuse and to validate each pattern individually or a set of the patterns described within the MoRaR pattern language.

Three case studies are considered in this thesis to show the application of the approach for reuse and validation: the development of a UCM framework for mobile system requirements and analysis models; the addition of a second generation feature and a generic reference model of a third generation system to the framework; and the addition of patterns to a typical wireless mobile ATM network.

Acknowledgments

The years of sacrifice in every aspect of one's life when taking a Ph.D degree, which include the writing of a thesis, are not easily explained or understood. In my case, I can confidently say that they express my strong passion for learning and teaching and I should emphasize that I was not alone in this effort.

First and foremost, I would like to thank my supervisor, Luigi Logrippo, for his support all these years. The lessons that I learned from him, specially his dedication to research and to his students, are always going to inspire me. I also extend my thanks to my committee, Dr. Tim Lethbridge, Dr. Dwight Deugo, Dr. Wilf R. LaLonde, and my external examiner for accepting to review and comment this thesis.

Then, I would like to express my eternal gratitude and love to my parents, Jesus and Andrade, to my siblings, Cássio and Tarciane, and to my fiancé, Ricardo, and to my uncles, aunties, and cousins. Their encouragement and love gave me enough strength to stay so many years abroad.

I also would like to thank the UofO's LoTOS Group for their technical support and friendship, specially Daniel Amyot, Laurent Andriansinferant, Leila Charfi, Tianbao Ding, Brahim Ghribi, Ruoshan Guan, Jennifer Fu, Masahide Nakamura, Daniela Savin, Jacques Sincennes, and Amy Yi. And many thanks to the SITE staff members, in particular, to my dear friend, Louise.

I am also indebted towards Jim Coplien for encouraging me to get close to the pattern community. John Visser, Yasser and Jim Hodges, from Nortel, owned my many thanks as well. They were particularly helpful to me on the wireless network subject, and they allowed me to get all the standard documents that I needed. Their inputs were always insightful and valuable. Todd Coram, my shepherd in the PloP'2000 conference, and Mark Bottomley, were also helpful during the development of the patterns.

Many thanks to Káthia Marçal and Nicolas Anquetil for reviewing my proposal and giving me judicious comments and lots of encouragement. And I also would like to give my thanks to my wonderful friend, Amy Yi who is like a younger sister for me, and Paulo Misaka for their help and personal support at the end.

I also would like to thank all my friends for their support. My special thanks to Riverson and Lígia Rios for their support when I arrived in Canada and to my dear friend Elisabete Malvar for the long and strong support during these 4 years (I owned her my staying in Ottawa and overcoming all troubles in my first year). I could not forget to thank Brahim and Laurent for making me laugh and enjoy life; Anita, Karina Marcus, Pierre et Perpetua; and, in particular, my gratitude to Cynthia, Daniela and Messaouda for giving me many useful advices in how to behave and conduct my studies as well as for their friendship. Many thanks to my UO colleagues, specially, Alan Williams, Jelber

Sayyad, Viviane Nastasse, and Stephane Some, and my brazilian friends in Ottawa, particularly, Jauvane e Carla Oliveira, and Igor Sales for the good times that we spent together. Last but not least, my special gratitude goes to the friends that I left in Brazil, specially, Alice, Claudiana, Janete, and Patrícia for the mails, calls, and encouragement.

Finally, I acknowledge CAPES and UFC for their financial support. In particular, I would like to give my thanks to Prof. Bigonha and Gláucia Tavares. Also thanks to my colleagues of the Department of Computer Science, particularly, Rosely, Orley Carneiro, Fernando Carvalho, Mauro Oliveira, Mauro Pequeno, Marcelino Pequeno, Tarcísio Pequeno, Vânia Vidal, Creto Vidal, and Lucy Vidal.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

This chapter presents the background that leads to the motivation of this thesis. It also outlines the objectives, the intended contributions and the relevance of this work for the mobile wireless communication domain. Furthermore, this chapter includes the strategy used to achieve the thesis objectives.

The chapter structure is described as follows. Section 1.1 presents the story behind the main motivation of this work. Section 1.2 describes its objectives and highlights its scope. Section 1.3 discusses the intended contributions and the relevance of this thesis. Section 1.4 outlines the steps followed during the development of this work. Finally, concluding remarks and the remaining chapters are summarized in Section 1.5. Related work is mentioned throughout this chapter.

1.1 Background and Motivation

In the mobile communication domain, different second generation standards provide support for mobile users. For example, the **G**lobal **S**ystem for **M**obile Communications (GSM) [139] is the European digital cellular system and the **A**merican **N**ational **S**tandards **I**nstitute – 41 (ANSI-41) [26] is one of the families of standards for cellular systems developed in the United States. Recently, ITU-T, which stands for **I**nternational **T**elecommunications **U**nion – **T**elecommunications Standardization Sector, is working on a third generation standard called the **I**nternational **M**obile **T**elecommunications Systems **2000** (IMT-2000) [112][113][114] that aims to provide global roaming capability and a seamless environment for wireless networks.

Although second generation standards have differences related to their architecture, protocols and services that make them incompatible, they have common architectural elements performing similar functions as mentioned in [35] and [86]. There is therefore a need for recognizing and capturing commonalities among these systems to allow designers to reuse them in the system development process and in the system evolution.

Meanwhile, the concept of software patterns has been used in the software engineering domain to capture existing experiences of good design and several design patterns are currently available in the literature [52][80]. These patterns have been applied to different systems and improved the overall quality of their design. In this thesis, the software pattern concept is applied to describe common functional behaviors and architectural elements among second and third generation mobile communication systems. The main motivation for capturing and documenting these patterns is to enable designers to re-use good solutions that have been currently applied to different mobile systems to solve design problems in the system development and evolution.

Attempts to document design patterns for telecommunications are presented in [184] and [3]. The former focuses on fixed telephone systems and presents patterns to detect feature interactions. The latter introduces patterns that address reliability and human factor issues for switching systems. Nevertheless, none of these authors attempts to investigate mobile communication systems or to capture patterns that express their common design problems and respective solutions.

Furthermore, the mobile communication literature does not attempt to organize a repository of the good solutions that have been recurrently used in mobile systems to solve common design problems [35][86]. Moreover, there is no attempt to show how common architectural elements and their respective functionalities work together to generate a complete scenario for a seamless mobile wireless communication system.

In addition to the GSM, ANSI-41, and IMT-2000 mobility standards mentioned earlier, the following systems are also taken in consideration for this work: the **W**ireless **I**ntelligent **N**etwork (WIN) [174][175][176], the European data extension of GSM called **G**eneral **P**acket **R**adio **S**ervices (GPRS) [84], the **U**niversal **M**obile **T**elecommunications **S**ystem (UMTS) [33][78][141], and the **W**ireless **m**obile **ATM** Networks (WmATM) [4][5].

1.2 Objectives

As mentioned in the last section, the overall objective of this thesis is the use of the software pattern concept in the domain of mobile wireless communications. To achieve this, mobility and radio resource management functions are investigated to capture common functional behaviors. Meanwhile, architectural elements associated with these functions are also examined closely to capture commonalities. Patterns are extracted from these common functional behaviors and architectural elements and they are grouped in a pattern language that shows how they relate to each other.

Furthermore, we investigate existing techniques and methodologies for the design of mobile communication systems and propose an approach that allows designers to reuse and validate pattern solutions at the early stages of the system development process and evolution.

This work also presents case studies that are built on the basis of the patterns and the pattern relationships shown in the pattern language. For instance, a framework for mobile system requirements and analysis models depicts the mapping of the patterns related to system behaviors to the patterns related to the system architectural elements.

In short, this thesis aims to:

- develop an approach to capture and document patterns that represent common functional behaviors and architectural elements among mobile wireless communication systems;

- apply this approach to second and third generation systems;
- provide a pattern language that shows the relationship among these patterns;
- propose an approach for reuse and validation of the pattern solutions;
- develop case studies to show the application of the proposed approach.

These objectives are achieved by:

- describing existing systems with a visual technique called **U**se **C**ase **M**ap**s** (UCMs) [48][50] on the basis of the documentation available in the standards or in the literature;
- identifying common functional behaviors and architectural elements among the chosen systems;
- extracting patterns from the previously identified commonalities;
- describing these patterns within a pattern language;
- applying UCMs and formal methods such as **L**anguage **O**f **T**emporal **O**rdering and **S**pecification (LOTOS) [98][100] and **M**essage **S**equence **C**harts (MSCs) [106] to better reuse and validate the pattern language. Executable system prototypes are constructed and validated with LOTOS.
- developing a UCM framework for mobile system requirements and analysis models, developing a prototype that integrates a second generation feature in a third generation systems, and showing the evolution of an existing system on the basis of the proposed approach for reuse and validation (these are the thesis case studies).

Non-objectives

Standard documents for mobile systems concentrate on the early stages of the development process. These standards leave implementation issues unspecified with the purpose of giving the designers freedom to make their own products. In addition, solutions for protocol signaling alternatives of these systems are usually described in the literature with development approaches that focus on system descriptions with text and information flows. Accordingly, the proprietary and confidential nature of telecommunication software and the competition among telecommunication industries make it difficult to capture patterns at the design and the implementation levels comparable to some of the patterns presented in the software engineering domain [52][80]. Thus, since details about the implementation of the mobile systems are not available, the patterns that are described in this work are related to the early development stages (requirements and analysis stages).

It should be also mentioned that in relation to the network communication layers presented in [173], we concentrate on the mobility and radio resource management protocols of the application layer. There is no intention to capture commonalities among the protocols of the lower layers such as physical and data link layers. The air interface protocols between the mobile station and the network are also not considered in this work.

Furthermore, telecommunication billing issues and related services such as pre-paid charging are not tackled within this thesis that mainly investigates mobility and radio resource management functions and their integration in the basic call state model of mobile systems.

Finally, infrastructureless (ad hoc) networks, which consist of mobile nodes that can be connected dynamically in an arbitrary manner [137], are also out of our scope. We feel, however, that our approach is general and applies well beyond our case studies.

1.3 Intended Contributions and Relevance

The main contributions of this thesis are the identification of problems and common solutions in the mobile communication domain and the capture as well as the documentation of these commonalities as patterns. At a high level of abstraction, this research shows that it is possible to extract common functional behaviors and architectural elements among different mobile systems. When these patterns are grouped together, they constitute a pattern language that allows designers to visualize how the patterns interact.

In order to facilitate the reuse of pattern solutions in the development of new systems or in the evolution of existing ones, this thesis presents an approach for the reuse and the validation of these pattern solutions and the pattern language relationships, which are informally described in the pattern language. This approach incorporates existing methodologies [8][10][18] that add rigor to the system development process and enable designers to detect errors, omissions, inconsistencies and ambiguities at the early stages. Besides providing a better and precise description of the system, these methodologies combine the use of different techniques at different development stages.

The proposed approach enables designers to reuse good solutions for mobility and radio resource management problems from the early stages of the mobile system development process and evolution. A designer is able to choose either an individual pattern or a set of patterns to be reused according to specific needs of the system. In the case studies presented in this thesis, we show how the pattern reuse helps the development of new systems and the evolution of existing systems. The reuse of these patterns is a promising way to overcome potential incompatibilities between new and existing mobile wireless systems, so that designers will not have to start from scratch when developing new systems.

In short, this work intends to provide means to:

- identify and describe common functional behaviors and architectural elements;
- reuse these commonalities, which are documented as patterns;
- improve mobile system development and evolution with the reuse of commonalities and the detection of protocol design errors, which in practice are often detected at the implementation stage, at the early stages of the system development process and evolution.

On the basis of the common foundation provided by this work, designers can iron out the differences in the mobile systems and produce protocols that are compatible with second generation systems. We believe that these contributions help designers to achieve a seamless network environment for mobile users and set the direction for the global roaming capability that are not presently provided due to design differences among existing systems. Solutions to offer these capabilities for third generation systems are under development in the IMT-2000 standards.

It should be mentioned at this point that the need for global roaming and seamless networks is commercially questionable. The cost of producing a single mobile terminal that not only provides voice, data, video, and Internet access capabilities but also works around the world is high. Another concern relates to the degree of acceptance that can be expected for these services. For instance, Internet users are often satisfied with portable computing and the need for global roaming comes from people that are able to get either satellite phones or overseas wireless systems. Both systems work well around the world. However, none of them provides service portability. Many features related to billing (e.g., pre-paid) will not work across networks. Although our research does not address these issues, it provides good design methods for future systems when wireless technologies become less expensive, bandwidth increases, and more users express the need for seamless services.

Another relevant aspect of this work is the possibility of applying the capture, the reuse and the validation approaches to other domains. The capture approach is briefly outlined in the next section.

1.4 Strategy for the Thesis Development

Mobile systems are often described using different techniques such as informal description, information flows, and state diagrams. As a result, the first step of this thesis is the uniform description of these systems with the same technique. The UCM notation is chosen as the visual notation to specify GSM (GPRS and UMTS are based on GSM), ANSI-41 (WIN is part of the ANSI-41-D description), WmATM, and IMT-2000 systems (as shown in Figure 1a). UCMs are able to express requirements and analysis models in such a way that the developer has a bird-eye view of the whole system. Furthermore, this

notation is easy to follow and it enables the designer to refine the problem to be solved starting from the beginning of the development process and evolution. UCMs are applied to this work as the technique to represent potential reusable units.

The second step is the identification of common functional behaviors and architectural elements. The UCM descriptions of the chosen systems are investigated in terms of common start points, responsibilities, and end points. Then, we concentrate on the system architectures that are analyzed in terms of their common architectural elements.

We adopt the behavioral and structural pattern terms as in [80] to describe the patterns that we are presenting in this thesis. In [80], the authors focus on the design stage to describe the "composition of classes and objects" (called *structural patterns*). The interaction of classes and objects and the distribution of responsibilities between them are described in the *behavioral patterns*. However, our patterns are related to functional behaviors and architectural elements at the requirements and analysis stages, respectively.

The identification of these commonalities is crucial for the capture of patterns, which constitutes the next step. Figure 1 illustrates the steps for the identification of common functional behaviors and the capture of *behavioral patterns*.

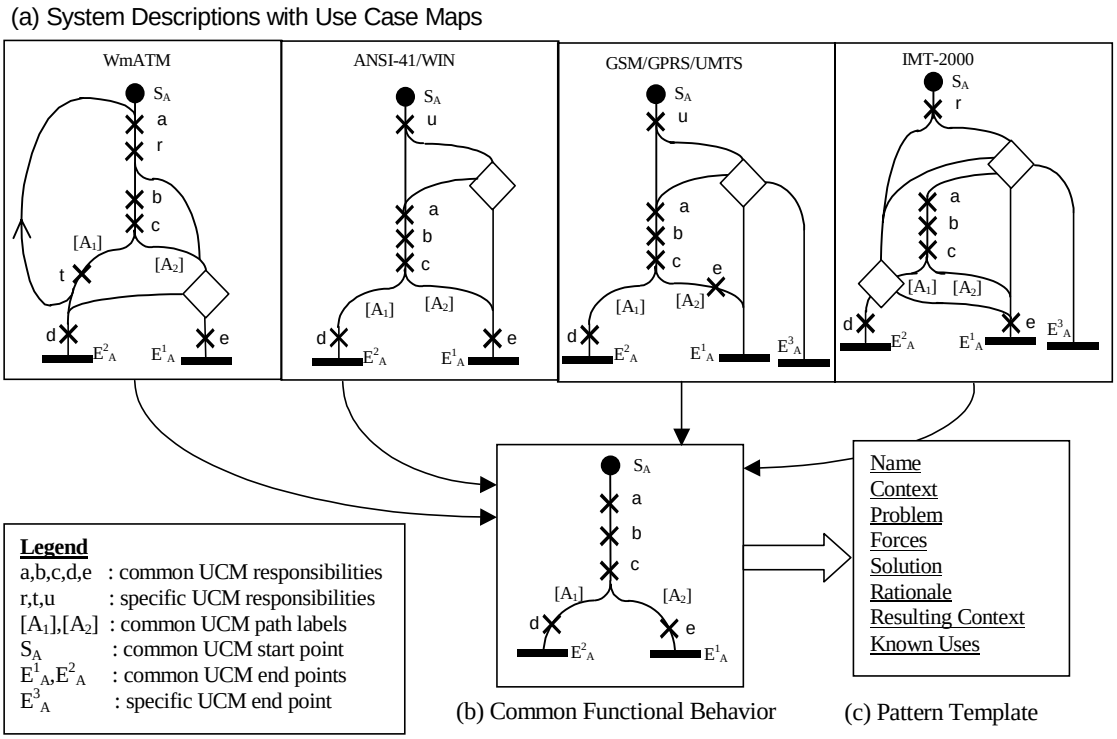


Figure 1 Steps for the Capture and the Documentation of Behavioral Patterns

Each system architecture is described with UCM components and the common elements are extracted. The common architectural elements (also called network entities in [174]) constitute a common network reference model.

In the third step, each commonality is analyzed and when the pattern concept can be applied, these commonalities are translated to the pattern template and they constitute the *behavioral* and *structural patterns*. A *pattern* is then captured and documented. As shown in the Figure 1c, each *behavioral* pattern is documented using the following pattern template: name, context, problem, forces, solution, rationale, resulting context, and known uses [59][133]. *Structural* patterns are documented in the same way (not shown in the figure). A pattern is captured and documented every time a common recurring design problem and its respective solution can be represented in this pattern template.

Figure 2 depicts the steps after the capture and the documentation of the *behavioral* and *structural patterns*. These patterns are grouped into a pattern language as illustrated in Figure 2a. In order to show how these patterns could be reused at the early stages of the development and evolution of mobile systems, an approach for reuse and validation is proposed. This approach combines UCMs, LOTOS, and MSCs techniques. On the basis of UCMs, a prototype of a mobile system can be specified and validated using the LOTOS language and its tools. The validation results are shown by means of MSCs. The proposed approach is not shown in the figure, but it is discussed in detail in Chapter 6.

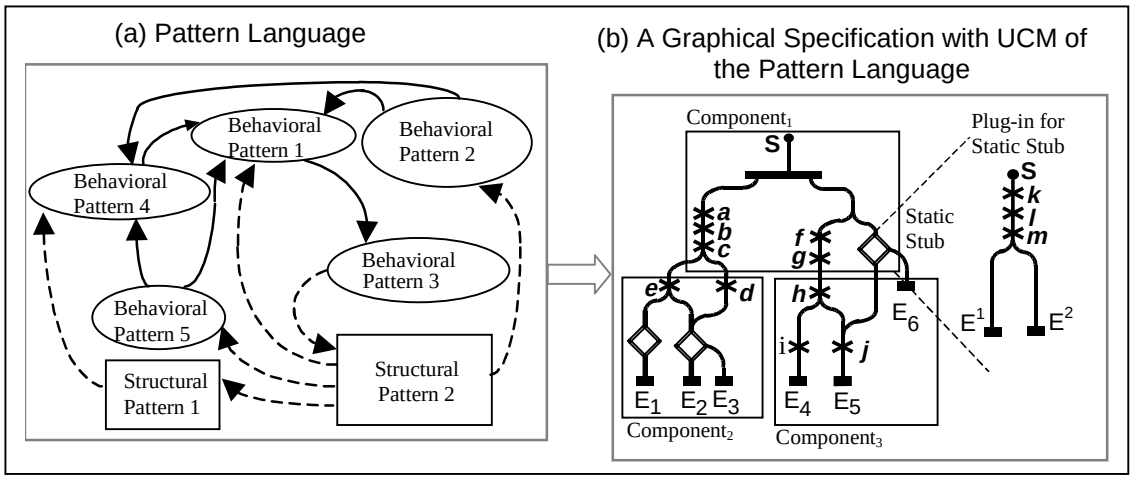


Figure 2 Steps for the Generation of a UCM Model on the basis of the Pattern Language

Figure 2b shows a simplified UCM that can represent a framework for mobile system requirements and analysis models. These scenarios are derived from the pattern solutions and pattern relationships (represented by the plain arrows in Figure 2a). The figure depicts the *structural patterns* represented by UCM components and *behavioral pattern* solutions represented by plug-ins bound to specific stubs. The mapping between the behavioral and structural patterns, whose relationships are presented by the dashed arrows, is also described in this framework. The model shows that UCMs are suitable to graphically specify the pattern language. A more complete UCM framework is presented in Chapter 7.

1.5 Summary and Structure of the Thesis

This research deals with the capture and documentation, the reuse, and the validation of requirements and analysis patterns for mobile systems. These patterns are captured among second and third generation mobile systems with focus on mobility and radio resource management functions and they are grouped into a pattern language that shows their relationships.

An approach for the reuse and validation of these patterns is also proposed. As a result, designers are able to reuse individually each pattern or a set of patterns grouped in the pattern language during the evolution of existing systems or the development of new ones. Case studies, which include a UCM framework for mobile system requirements and analysis models, are also presented in this thesis using the proposed approach.

The remaining chapters provide the details of the work that has been done and they are structured as follows:

- *Chapter 2: Survey of Mobile Wireless Communication Systems* provides a general discussion about mobile wireless communication systems with relation to well-known telecommunication standards and emerging systems that are investigated in this research. The main definitions regarding this domain are also provided in this chapter;

- *Chapter 3: Overview of Software Patterns* introduces not only the main concepts of the software pattern domain but also several types of patterns that have been presented in the literature. The pattern language concept and existing telecommunication patterns are also presented. In addition, this chapter explains how these concepts are applied to the mobile wireless communication domain;

- *Chapter 4: Capture of Common Functional Behaviors and Architectural Elements* focuses on the description of the chosen systems with the Use Case Map notation and explains the strategy used to capture common functional behaviors and architectural elements among these systems. Furthermore, the UCM notation is summarized;

- *Chapter 5*: *MoRaR: A Pattern Language for **Mo**bility and **Ra**dio **R**esource Management* presents the description of the patterns related to mobility and radio resource management as well as their interaction within the pattern language;

- *Chapter 6*: *Approach for Reuse and Validation* proposes an approach to develop new mobile systems and to help the evolution of legacy systems by reusing the pattern solutions. The combination of UCMs, LOTOS, and MSCs is included in this approach. In addition, current methodologies applied to telecommunication systems are discussed;

- *Chapter 7: Case Studies* presents the usefulness of this research by applying the proposed approach to develop mobile system prototypes. The first case study develops a UCM framework for mobile system requirements and analysis models on

the basis of the pattern language. The second case study shows the integration of new features and architectural elements to a third generation system. Finally, the third case study shows the addition of patterns to existing protocol signaling alternatives for Wireless mobile ATM networks;

- *Chapter 8: Conclusion and Future Work* presents a summary of the major and minor contributions of this thesis and the open areas of research that arise from this work.

Guide to Readers

This thesis has three main parts. The first part describes mobile communication systems and software patterns (respectively, Chapter 2 and Chapter 3). The second part of the thesis presents our main contributions, which are the capture of commonalities (Chapter 4), the documentation of requirements and analysis patterns (Chapter 5), and an approach for reuse and validation of these patterns (Chapter 6). The last part shows the application of the proposed approach to three case studies (Chapter 7).

A reader who is familiar with mobile systems and software patterns can skip the first part and go directly to the commonalities, patterns, reuse and validation topics. Case studies are useful for readers interested in the application of the proposed approach.

Yet another way to read this thesis for those who are interested in developing new systems or evolving existing systems is to use a problem-solution strategy. A reader can start with Appendix C that summarizes each pattern and then go to Chapter 5 and Chapter 6. Eventually, the remaining chapters come into play when more details are necessary.

Chapter 2 Survey of Mobile Wireless Communication Systems

This chapter focuses on the description of second generation and emerging third generation mobile communication systems, which are considered in this research. The main concepts involved in mobility, communication, and radio resource management functions as well as the evolution of mobile systems are also discussed. In addition, telecommunication standardization groups around the world are presented.

2.1 Introduction

Over the last few years, telecommunication systems, which enable users to exchange information (e.g., voice and data), have changed considerably due to the increased amount of services provided for the users. The separation of call processing and services from the signaling related to the switches has facilitated the increase of services. This separation has been achieved by the Intelligent Network standards [110][111] that are used in the fixed telecommunication networks.

In addition, mobility functions and wireless technologies have caused a rapid growth of mobile communication systems. The improvements of protocols over the air interface, the availability of network services, and the quality of wireless media have helped the high acceptance and usage of mobile systems [141][146]. Mobile communication standards have been issued to document and to provide compatibility among these improvements.

According to [22], telecommunication standards define a common point between public and private systems. For example, the **I**nternational **T**elecommunication **U**nion includes a **T**elecommunication standardization sector (ITU-T), which is the international standard organization composed of government institutions responsible for the description of laws that govern the interconnection of telecommunication and telegraphic systems. These laws also govern the new trends for telecommunication systems.

Several standardization groups have recently emerged as a result of the increased demand for services and technologies in the telecommunication domain and they are composed of different representatives of countries and companies. The following standardization groups exemplify this: the **A**merican **N**ational **S**tandard **I**nstitute (ANSI) as well as the **T**elecommunications **I**ndustry **A**ssociation (TIA) from the United States, the **T**elecommunications **T**echnology **C**ommittee (TTC) from Japan, and the **E**uropean **T**elecommunication **S**tandard **I**nstitute (ETSI) from Europe. These groups often submit their results to the ITU-T that is still the main authority of the government telecommunication standards [118].

Recently, partnership projects have been harmonizing various standard efforts in order to specify interfaces for third generation systems that rely on the integration of existing systems [179]. The first effort is on the radio interface. For example, the 3rd **G**eneration **P**artnership **P**roject (3GPP) has been working on a technical specification based on the GSM core network and the **U**niversal **T**errestrial **R**adio **A**ccess (UTRA) called **U**niversal **M**obile **T**elecommunications **S**ystem (UMTS). In addition, another group called 3GPP2 has been focusing on the ANSI-41 core networks and relevant radio access technologies. These two teams have been exchanging information regarding their results that should be submitted to the ITU-T.

This research addresses telecommunication services that are provided by second generation mobile standards (e.g., GSM and ANSI-41) and emerging systems such as Wireless mobile ATM (WmATM) networks and IMT-2000 systems. We refer to them as *mobile wireless communication systems* that represent the combination of mobile systems, which support mobility and communication management functions, and wireless systems, which provide wireless interfaces to users (both mobile and stationary) [186] using the radio resource management functions.

The next sections are organized as follows. Section 2.2 introduces the basic definitions of the mobile communication domain that are essential to understand the evolution of wireless technologies and services for mobile users, which are also discussed in this section. In order to explain the common elements, concepts and functionalities among these systems, Section 2.3 presents a functional layer scheme that is used in second generation systems. The remaining sections detail the second and the third generation systems that are investigated in this thesis. At the end of this chapter, conclusions are outlined.

2.2 Evolution of Mobile Wireless Communication Systems

Mobile communication systems are classified into different *generations* according to the wireless technologies in use and the services provided to the users [35][141]. They are also classified into cellular systems and **P**ersonal **C**ommunication **S**ystems (PCS).

On one hand, wireless technologies are used at the air interface to guarantee information exchange through the radio channels between the mobile station, the base station transceiver and the base station controller (see Figure 4). For instance, the following channel utilization techniques are often employed: **F**requency **D**ivision **M**ultiple **A**ccess (FDMA), **T**ime **D**ivision **M**ultiple **A**ccess (TDMA), and **C**ode **D**ivision **M**ultiple **A**ccess (CDMA).

The FDMA technique splits the bandwidth of the air interface into multiple analog channels (e.g., a larger frequency spectrum of 15 MHz into multiple 200kHz channels). Each channel is assigned to one user. When each direction of the transmission has its own **r**adio **f**requency (RF) channel, the technique is called **F**requency **D**ivision **D**uplex (FDD) or **F**ull-**f**ull **D**uplex (FFD).

The TDMA technique splits an analog RF channel into time slots that contain digital traffic. For instance, each user gets a digital time slot for a certain period of time in order to send traffic. **T**ime **D**ivision **D**uplex (TDD) is a channel sharing technique that allows the same channel to be used for both directions of the transmission. This technique flips the direction in a certain time to give the impression that each direction has a dedicated channel (half-duplex system with full-duplex operation). Another alternative of TDMA that is known as **e**xtended **TDMA** (E-TDMA) applies the **s**tatistical **TDM** (STDM) technique to allocate unused time slots (e.g., vacant slots belonging to an idle mobile user) to active mobile users.

Finally, the CDMA gives the same frequency spectrum to all users at the same time (spread spectrum technique). Each user is identified on the channel by a code. At the transmitter side, the traffic is encoded and sent across the frequency spectrum. At the receiver side, the same code is used to extract the user's traffic. In comparison with FDMA and TDMA, this technique provides more capacity and channel performance with the same bandwidth. In addition, more users can share the same spectrum. However, its implementation is more complex than the previous techniques.

On the other hand, several services such as voice, data, text, and multimedia have increased the user's demand for higher transmission speed in the past decade [180]. For instance, Table 1 presents a variety of user's application services within a wide range of service bit rates and different service types. Certain services are called isochronous, **c**onstant or continuous **b**it **r**ate oriented (CBR) when they allow none or very small variability in the delay of the output signal between source and destination; otherwise, the quality becomes unacceptable to the user [156]. **V**ariable **b**it **r**ate (VBR) services require the network to feed the output that is necessary to guarantee the quality of service. In the **a**vailable **b**it **r**ate (ABR) and **u**nspecified **b**it **r**ate (UBR) service capabilities, the throughput provided by the network can change after the connection is established.

Service	**Type of Service**	**Service Bit rate**
Voice/audio	Constant Bit Rate (CBR)	8-128 Kb/s
Digital Data	Available Bit Rate (ABR) Unspecified Bit Rate (UBR)	0.1-1 Kb/s
Video Telephony	CBR	3874 kb/s
Motion-Video	CBR/Variable Bit Rate (VBR)	1.5-6 Mb/s
File Transfer	ABR/UBR	1-10 Mb/s

Table 1 User's Services (adapted from Table 1 of [180])

First and second generations of mobile communication systems have been developed to provide basic and supplementary voice services. However, they also support data services at a low bit rate of 9.6-32 Kb/s. The application services shown in Table 1 have drawn the

attention to third generation systems that are able to support all types of services at a reasonable information rate.

To make the distinction among generations clearer, Figure 3 illustrates a comparison among first, second, and third generation systems in terms of mobility capability (e.g., none for fixed networks) and service bit rates (see Table 1). WmATM networks and wireless **L**ocal **A**rea **N**etworks (LANs) are also shown in this figure.

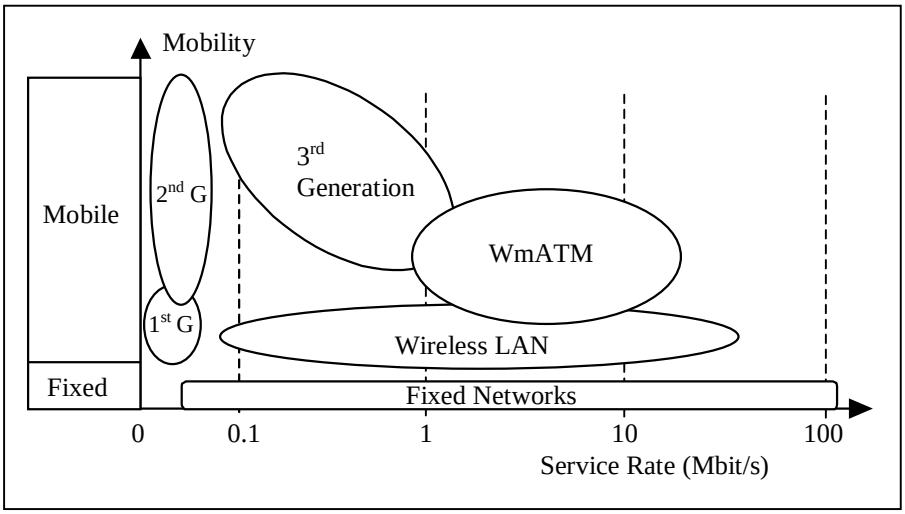


Figure 3 Comparison among Mobile Wireless Communication Systems (adapted from Figure 4 of [180])

The next sub-sections give a quick overview of first, second and third generation systems.

2.2.1 First Generation Systems

First generation systems apply FDMA analog signaling for user traffic on the air interface. Voice signals are modulated directly into the RF channel without the need for converting them into digital signals [35]. For instance, **A**dvanced **M**obile **P**hone **S**ystem (AMPS) and **T**otal **A**ccess **C**ommunication **S**ystem (TACS) are analog cellular systems that belong to the first generation.

Two of the first generation design solutions are used by second generation systems: the electronic serial number and the 30kHz RF transmission channels between the mobile station and the base station on the air interface. A disadvantage of the first generation is related to the analog aspects. For instance, these systems are not able to apply voice digitization or encryption operations, which makes them vulnerable for eavesdropping.

2.2.2 Second Generation Systems

Second generation systems employ digital signaling techniques (e.g., TDMA or CDMA). These techniques modulate user traffic as well as control traffic into analog RF channels. Voice signals are converted to digital signals using an analog to digital process and then they are modulated into a RF channel [35]. Either time slots (TDMA) or codes (CDMA) identify each user and make it possible to allocate a RF channel to more than one user. This identification also allows more flexibility in the channel allocation. For example, a channel is only assigned to a user when the user needs it. A user is in a *dedicated mode* when a traffic channel (TCH) has been assigned. Otherwise, the user is in *idle mode*. A control channel (CCH) is used to keep the network aware of the mobile station mode.

As mentioned earlier, an advantage of second generation systems in comparison with first generation systems is their ability to provide privacy and security to mobile users through encryption mechanisms that are only possible with digital signals. Another advantage is the possibility of applying error detection and correction to the digital signals. This allows a better quality of transmission by reducing the noise and fading effects on the channel. Other benefits of these systems are the international roaming capability (under one subscriber directory number), the support of low-power mobile stations, and the variety of new services as well as network facilities [160].

Digital cellular systems, such as the **G**lobal **S**ystem for **M**obile Communications (GSM), the **D**igital **AMPS** (D-AMPS), the **J**apanese **D**igital **C**ellular Systems (JDC), and **P**ersonal **C**ommunication **S**ystems (PCS) are known as second generation systems. ANSI-41 based systems also belong to the second generation, however, they support analog and digital air interfaces.

These systems employ different digital wireless technologies such as TDMA, CDMA, and **P**ersonal **D**igital **C**ellular (PDC). GSM and PCS use TDMA (e.g., PCS 1900, GSM 900 and GSM 1800), ANSI-41 can be implemented with TDMA (known as IS-136 systems) or CDMA (called IS-95 systems), and PDC, which is mostly used in Japan, employs FDMA and TDMA. D-AMPS (e.g., IS-54-B systems) defines a dual mode operation with a hybrid first generation (analog FDMA) and second generation (digital TDMA) air interfaces. GSM and ANSI-41 based systems are the most popular second generation systems among mobile users.

Second generation systems offer services such as voice, data, and short message. These services are standardized to be platform independent. This enables wireless carriers to buy equipment from different vendors. These services are provided in the *circuit-switching mode* (e.g., GSM) and in the *packet-switching mode* such as GPRS. Circuit-switching mode is a transfer mode used in the telephone networks that establishes a circuit for the complete duration of the connection between two or more stations. On the contrary, packet-switching mode is a logical connection that is established between two or more stations to provide the routing and the transfer of data in the form of packets [79][156].

Furthermore, second generation systems use a *vertical architecture*, which specifies system characteristics such as application services (see Table 1) and bearer services together. *Bearer services* refer to the data communication services that are performed by the basic capabilities (e.g., addressing and identification functions) of the transmission medium between two user-network interfaces [79][139][146]. A disadvantage of the vertical architecture is that modifications or additions of new services often generate a need for a reengineering in the existing system and it affects the network end to end. In order to overcome these limitations, the GSM and the ANSI-41 second generation standards have incorporated the Intelligent Network concepts, respectively, into the following systems: **C**ustomized **a**pplications for **m**obile network **e**nhanced **l**ogic (Camel) and **W**ireless **I**ntelligent **N**etworks (WIN).

Camel and WIN set the direction for the integration and convergence of existing mobile systems that are the main goals of the third generation systems discussed in the next sub-section. Camel is not covered in this work; however, details about WIN, which is still under development, are presented in Section 2.6.

2.2.3 Third Generation Systems

Although second generation systems are well accepted among mobile users, the standardization groups are already developing third generation systems. This new generation intends to offer better capabilities and coverage for mobile users. In addition, these systems also support circuit and packet oriented services. A better diversity of supplementary services for voice and data (low and high speed), such as multimedia and other Internet services, is also provided by these systems with the same quality as fixed networks. The main goal of third generation systems is to achieve integration and transparency of services in a seamless environment.

The international standardization committee (ITU-T) and several standardization groups (e.g., 3GPP and 3GPP2) are working on the provision of third generation systems. These systems should be first introduced in Japan, Korea and Europe.

Section 2.7 and Section 2.8 present two different standardization efforts for third generation systems, as follows: the Universal Mobile Telecommunications System (UMTS) [33][78][141] and the IMT-2000 systems [112][113][114][166]. Both systems aim at the integration of existing mobile systems in a seamless environment.

The next section presents common elements, concepts and functionalities that are identified in second and third generation systems. The identification of these commonalities is the foundation to provide a seamless mobile network environment.

2.3 Seamless Elements, Concepts and Functionalities

When mobile users are outside of their home network and try to access communication services (*roaming* capability), different wireless carriers are involved in supporting these

services due to wireless coverage limitations. These wireless carriers cannot always offer access to all services supported by the user's home network due to the diversity of service implementations. However, the concept of *seamless networks* as introduced in [86] can enable a mobile user to roam across different mobile systems and obtain the same services as they are provided in her home network. In addition, the seamless capability is necessary in order to reduce the total cost of network access and to allow the use of a single device, a single identification, and a single bill.

In order to provide a seamless environment for mobile users, which allows an increase in wireless coverage and services, it is necessary to identify common functional behaviors and architectural elements among second and third generation systems. It is also important to investigate the common concepts that are used in these systems.

This thesis presents a set of patterns on the basis of common functional behaviors and architectural elements that are investigated among second and third generation systems. The mobile systems that we have chosen to investigate in order to capture and document these commonalities are presented in Section 2.4 (GSM and GPRS), Section 2.5 (ANSI-41), Section 2.6 (WIN), Section 2.7 (UMTS), Section 2.8 (IMT-2000), and Section 2.9 (WmATM).

We use the word *seamless* to refer to architectural elements, concepts, and functional behaviors that are common among the chosen systems. The next sub-sections give an overview of these commonalities.

2.3.1 Common Elements and Concepts

Each mobile system contains a set of entities, elements or components that are named differently in the literature. At a high level of abstraction, these elements are known as *functional entities*, which are incorporated into network entities at a lower abstract level. Network entities are then mapped to the real physical entities. This research uses *architectural* or *structural elements* to describe functional and network entities.

Mobile systems that belong to second and third generations include the following common architectural elements, as [35] and [86] have identified: mobile station (MS), mobile switching center (MSC), base station transceiver (BST) or radio port, base station controller (BSC), and databases such as the home location register (HLR) and the visitor location register (VLR). The set of BST and BSC is often referred to as base station (BS). These architectural elements are depicted in Figure 4. The wireless backbone communicates with the fixed backbone (PSTN and ISDN systems) using wired access ports (not shown in the figure).

These systems also use similar concepts regarding mobility, communication, and radio resource functions. These similarities are specific to mobile systems and are different from fixed networks. In addition, these new concepts increase the complexity of mobile systems in comparison with fixed networks. For instance, as discussed in Section

2.2.3, there is a challenge for third generation systems to offer supplementary services with the same quality as fixed networks.

The term *cell* is used to define a coverage area in cellular systems. As shown in Figure 4, every mobile system split its environment into *cells*. The frequency band allocated to one cell is split among several cells that coexist next to each other. Each cell covers a geographical area with a *base station transceiver* that supports the radio resources related to the use of the allocated spectrum. A cell has different sizes depending on the operator's need. For instance, *large cells* are usually applied to remote areas and *small cells* for high-density traffic areas. In addition, radio engineers design cells with a variety of shapes (e.g., circular or sectored cells). A cell with the required base station transceiver is able to restrict transmitted power within a particular area and eliminate power from adjacent areas [160].

A *location area* contains several cells and a *base station controller* is responsible for monitoring a certain number of cells grouped in the location area. In this research, one or more base station controllers can be responsible for a location area; however, a single *mobile switching center* manages each location area. Many mobile stations share the capacity of each base station transceiver. The connection among base station transceivers, base station controllers, and mobile stations is done through radio access ports (known as *air interface*). Section 4.8 of Chapter 4 presents more information about mobile switching centers, *home location registers* and *visitor location registers*.

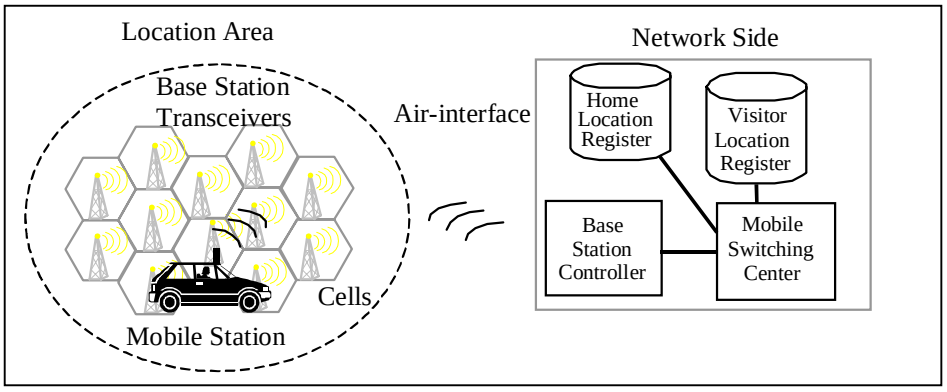


Figure 4 Common Architectural Elements for Mobile Systems

Besides cells and location areas, other concepts, such as roaming, handoff, carrier selection and call establishment, are common among mobile systems. These concepts are summarized as follows:

- *Roaming* is the capability of mobile users to move around their home location area or even to a different location area outside their own carrier. Both situations require an exchange of messages to guarantee quality of services as well as call establishment. Registration, authentication, and billing are services affected by the roaming capability. Furthermore, handoff and carrier selection are functions to be performed in order to guarantee this capability;

- *Handoff* (also known as handover in Europe) is the function that guarantees the quality of the link that allows each mobile user to roam. Mobile switching centers, base station controllers, base station transceivers and the mobile station itself are the architectural elements involved in this process. Handoff is a critical functionality for wireless networks since all communication services should be provided while the user is roaming. Without handoffs, calls are dropped as soon as the user moves far from the server base station;

- *Carrier selection* is another specific concept used by mobile wireless networks. Since users are roaming constantly, there is a need for an agreement among different carriers to guarantee the feasibility of the communication services that involve routing, billing, and provision of supplementary services;

- *Call establishment* between two mobile users is similar to call establishment in fixed networks with the exception of the information about the mobile user's location. Registration, authentication and handoff procedures are also involved in the mobile system call establishment.

2.3.2 Common Functional Behaviors

Signaling protocols are responsible for describing the exchange of information within a system or between different systems and they make sure that the provision of services is occurring with the use of defined procedures. The message exchanges often require the co-operation of distinct and distant machines.

The OSI reference model splits signaling protocols into seven different layers in order to reduce their complexity [160][173], as follows: physical, data link, network, transport, session, presentation and application. Most of the mobile systems also divide their signaling protocols into a short representation of the OSI reference model layers: physical, data link, network, transport, and application. These systems can also separate the signaling protocols in data transfer services and application services (e.g., ANSI-41). In addition, when dealing with the development of protocols for each layer, these systems can use functional layers (e.g., GSM) or procedures (e.g., WmATM networks).

GSM treats each layer complexity by dividing the signaling protocols into three distinct functional layers that rely on each other: communication management, mobility management, and radio resource management functions [139]. ANSI-41 standards describe different signaling protocol scenarios for intersystem handoff, automatic roaming, and intersystem operations, administration, and maintenance (O&M) functions [26]. Wireless mobile ATM systems split their signaling protocols in three different procedures: registration, call setup, and handoff [4].

Figure 5 shows a system, which contains signaling protocols split into layers, and its users. As depicted in the figure, these layers co-operate with each other in order to offer services to end users through the application layer. This representation is appropriate to this thesis that extracts common mobility, communication, and radio resource

management functions from the upper layers of mobile systems (as summarized in Figure 1 of Chapter 1). In this thesis, a mobile wireless communication system is composed of architectural elements (e.g., functional entities) that communicate with each other through signaling protocols. A *functional behavior* describes actions and events within each architectural element.

Communication management functions, which are in the upper layer, handle the establishment, the maintenance, and the release of the end-to-end transmission paths as well as the inter-working with external networks in order to support user to user communication. *Mobility management* functions are related to location management, paging, privacy and security of mobile users. *Radio resource management* functions deal with the management of transmission paths over the radio interface, which includes access, paging, release of resources, dynamic channel allocation, and handoff functions.

Furthermore, they handle radio properties of the transmission chain such as decisions regarding which kind of signaling is transported (e.g., voice or data) and which kind of encryption (or ciphering) is needed.

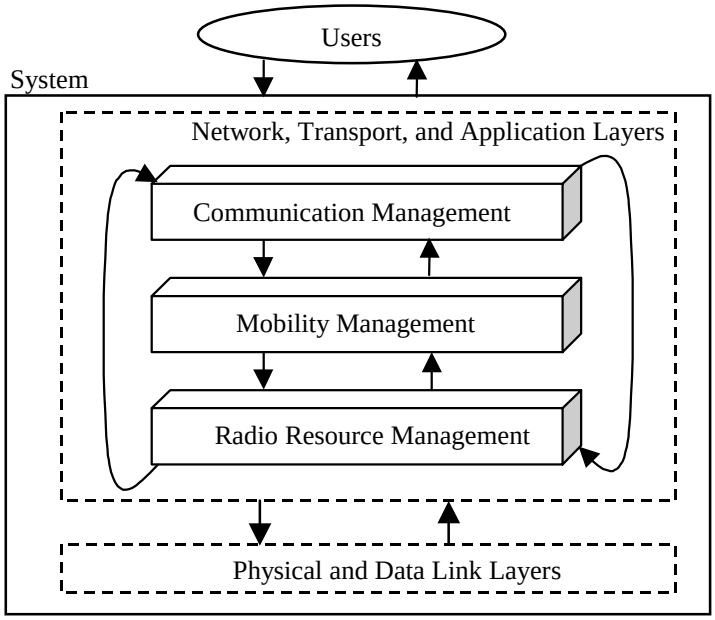


Figure 5 OSI and Functional Layers

2.3.2.1 Mobility Management Functions

Mobility brings users the freedom to move from one location area to another. Mobility management operations are responsible for keeping a record of the mobile user's location (*location registration* function) as well as for finding the correct location of a mobile station in a group of cells (*paging* function). Mobile user's privileges are also checked during mobility management in order to minimize the possibility of fraud (*ciphering* and

authentication functions). Besides mobile stations, these operations involve base stations, mobile switching centers, and databases. In this thesis, encryption and ciphering terminologies are used interchangeably. [88] describes encryption as the whole process of making all data transmitted incomprehensible to an intercepting party and ciphering as the process of substituting the data transmitted with a new alphabet.

In second generation systems, every time a powered-on mobile station crosses certain boundaries or detects a better channel in another location area, it requests a location update to the network. The network is in charge of the databases that keep information about the user. If a call is coming to the mobile user, the network requests a paging operation to locate the mobile station and establish the connection.

Authentication is applied to validate the user's identity and ciphering is used to protect the information exchanged between the mobile station and the network. Authentication and ciphering operations give the mobile user privacy and security. At a high-level of abstraction, these solutions are the same for GSM, ANSI-41, IMT-2000, and WmATM systems. However, each implementation is different. The incompatibility issue resides in the ciphering and authentication algorithms and the sequence of exchanged messages and their respective parameters that takes place between the network entities during this process.

2.3.2.2 Communication Management Functions

Communication management functions (also known as call processing and call control) are employed for all mobile systems to set up, handle, and release a call between a mobile user and another party (e.g., a mobile user or a fixed user). These functions involve routing interrogation, checking called subscriber's status, and establishing as well as releasing the call. In this research, a call is a conventional telephone call or another type of communication connection, such as data.

Communication and mobility management functions such as paging and location interrogation interact in order to establish the call. In addition, arrangements among different telephony and mobile service providers (i.e., carrier selection) are necessary to enable the call establishment.

A typical call establishment (also known as call setup) that involves a mobile user can be classified into originating calls and terminating calls, which are explained as follows.

Originating calls are originated from a mobile station to another party. The subscriber dials the digits to initiate the communication management functions with the network. When the mobile switching center (MSC) receives the digits the following procedures may be performed: authentication, digit analysis, routing information, call establishment, and, eventually, call release. The called party digits are used to route the call from the originating to the terminating party. This involves the analysis of the dialed digits and the determination of how far the current MSC can route the call. The home and visitor location register databases are responsible for keeping the current location of the mobile

station. When the network gets the routing information and checks the called subscriber's identity and status, a paging request is sent to the terminating party and the call is established. The call release (also known as call disconnection) involves the release of resources that are used to maintain and establish the call, which includes trunks, lines, and stored program control switch resources such timers, memory, and real-time software processes [79].

Terminating calls are made to a mobile station that may be in the home area or roaming. When the subscriber is in the home area, the call is completed after obtaining the MS registration status (i.e., whether the MS is available to receive a call); applying the terminating call features status (i.e., whether the MS is subscribed to call features such as incoming call screening); and paging the MS, which consists of MSC alerting MS and waiting the answer. When the subscriber is roaming outside the home area, the call is completed after obtaining the MS registration status; obtaining the routing information of the current system (i.e., the location where the subscriber is being served); applying the terminating call features status; delivering the call to the current serving MSC using call control signaling techniques (e.g., SS7 ISUP) suitable to the current network; and paging the MS.

A call may not be completed for the following reasons: the terminating party does not answer the call after a period of time, the terminating party is busy. In case of mobile-terminated calls, other reasons may affect the successful call establishment such as terminating does not respond to a page due to the subscriber's movement to an area outside the cellular coverage, or the MS is turned off and registration has not yet been cancelled, the subscriber's location is unknown, or the subscriber is inactive.

2.3.2.3 Radio Resource Management Functions

Many of the radio resource management functions are operations at the air interface that involve mobile stations (MSs), base station transceivers (BTSs) and base station controllers (see Figure 4). However, this research considers the functions that affect the application layer protocols at the network side (see Figure 5) and, thus, mobile switching centers (MSCs) are also involved.

For each mobile station engaged in a communication, the following two paths are allocated: transmission and signaling. From the point of view of a mobile station, these paths are set up when the mobile station leaves the idle mode and they are released when the mobile station goes back to the idle mode. From a network point of view, these paths can be modified mainly due to *handoffs* that are the main concern of the radio resource management functional layer. The allocation of these radio channels for the mobile station's use is also done by the radio resource functions. This lower level function is not addressed in this work.

The transmission management, which includes types of transmission modes (signaling only, speech, or data) as well as considerations about speed and terrestrial

channel management, are out of the scope of this thesis. In this work, when a channel is assigned, it allows all kinds of transmission modes and a MSC is responsible for the establishment of terrestrial circuits. Meanwhile, base stations choose the exact channel to be used and they are in charge of coordinating the quality of the radio channel as well as adapting it to the service needs.

Paging and *ciphering* are related to both mobility and radio resource management functions. As mentioned earlier, the paging function consists of finding a mobile station within a location area and involves basically base stations and mobile switching centers. The paging request is issued to the terminating party (also known as called party). When an incoming call arrives, the MSC requests the BSC to perform paging in some of the cells of the location area. The identity of the mobile station to be paged and the list of cells that at paging should address are included in this request. The decision of ciphering the transmission or not is known as cipher mode management. This decision is taken by MSC, and involves both BST and MS at the air interface.

Even though this research does not address lower layer protocols, radio resource functions that are related to paging, ciphering, and handoff are considered. The handoff procedure deals with the establishment of a new path and the release of the previous one. As depicted in Figure 6, handoffs can occur in three different ways depending on network equipment involved.

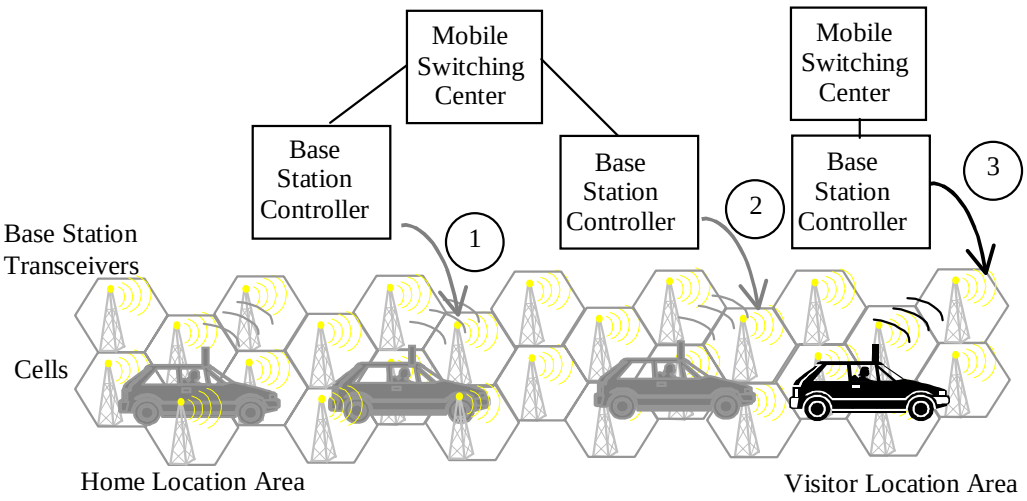


Figure 6 Type of Handoffs

Inter-base station transceiver handoff involves only the radio channel and the base station transceiver (position 1 in the figure). *Inter-base station controller* (intra-mobile switch center) handoff includes changes in the radio channel, base station transceiver, and base station controller (position 2). Finally, the *inter-mobile switching center* handoff involves the change of the previous components and the mobile switching center

(position 3). In this work, since the focus is on the upper layers, only inter-mobile switching center handoffs are investigated.

2.4 GSM and GPRS

GSM stands for Global System for Mobile Communications. The GSM standard effort was initiated in 1982. However, it was launched commercially in Europe in 1992 [139][160]. GSM is a second generation system that offers the following services [160]: teleservices (e.g., telephony speech, emergency calls, and short message services); bearer services (for instance, synchronous and asynchronous data, and alternate speech and data); and supplementary services (such as call forward, call holding, incoming calls, call waiting).

GSM specifications describe the whole infrastructure of the system as well as the air interface. Figure 7 depicts a simplified GSM infrastructure with the following entities: mobile station (MS), base station (BS) or base transceiver station (BTS), base station controller (BSC), mobile switching center (MSC), gateway mobile services switching center (GMSC), home location register (HLR), visitor location register (VLR), authentication center (AC), and equipment identity register (EIR). GSM defines a functional entity called base station subsystem (BSS) that is composed of the BSC and BTS. Furthermore, a functional entity called network switching subsystem (NSS) incorporates the GMSC, MSC, VLR, HLR, AC, and EIR architectural elements. Subscriber identity module (SIM) and operation and maintenance center (OMC) are also part of the GSM infrastructure (not shown in the figure).

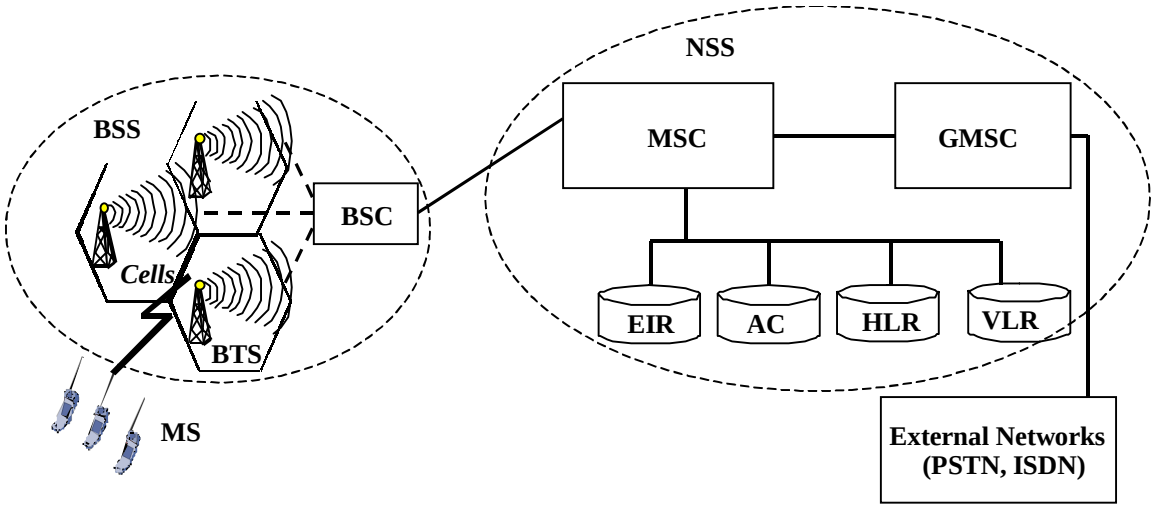


Figure 7 GSM Architecture (adapted from Figure 4-2 of [35])

GSM adopts the Signaling System no. 7 (SS7) that describes the signaling and the exchange switching for fixed networks and adds user's mobility with the Mobile Application Part (MAP).

General Packet Radio Services (GPRS) [68] is the data counter-part of GSM. It provides a packet switched architecture that allows mobile users to take advantage of the existing GSM infrastructure for running data applications [84]. GPRS aims at a fast deployment with minimum impact on the existing GSM infrastructure (see Figure 7). However, new functional and architectural elements are included, as follows. The GPRS mobile stations are composed of a mobile terminal (e.g., a handset) and terminal equipment (e.g., a laptop or a Personal Digital Assistant – PDA). Packet Control Unit (PCU) is a functional entity responsible for supporting the handling of packets at the base station system. Last, GPRS support node (GSN) is a functional entity that includes either a serving GSN (SGSN) or a gateway GSN (GGSN) responsible for supporting the handling of packets at the network switching subsystem. SGSN supports mobile stations by keeping track of their position and by controlling the data services currently in use. On the contrary, GGSN provides the interfaces to external packet data networks (PDN).

The GPRS data services include point-to-point (e.g., telnet, electronic mail, file transfer, and world wide web) and point-to-multipoint applications such as multimedia transmissions, weather and traffic reports, and conference services. Although GSM offers data services at a maximum bit rate of 14.4Kb/s, the GPRS offers bit rates around 170kb/s to make the previous applications possible.

2.4.1 Identifiers

GSM uses a set of identifiers, as follows: International Mobile Subscriber Number (IMSI), the user's phone number from the public network, International Mobile Equipment Identity (IMEI), and Temporary Mobile Subscriber Identity (TMSI). For additional addressing, a mobile station ISDN number (MSISDN) is the dialed number used to reach a called party in GSM networks [35][139][160].

IMSI is stored permanently on HLR and on the SIM card. IMSI is composed of mobile country code (MCC), mobile network code (MNC), which is the unique identification of the network provider (also known as public land mobile network – PLMN), and ten additional digits of the mobile subscriber identification code (MSIC), which is the customer identification number used to identify the subscriber's home PLMN.

IMEI contains not only the serial number of a mobile station but also the manufacturer, the country of production, and the type of approval. At the factory, when a GSM unit (e.g., mobile station) passes conformance and interoperability tests, the manufacturer gives a type approval code (TAC). In addition, a serial number (SNR) is assigned and a final assembly code (FAC) identifies the final manufacturer. This identity can be stored on EIR or HLR depending on the network operator.

TMSI is an identity alias that is used instead of the subscriber identity (the IMSI) when possible. Since sending IMSI in clear mode on the radio path would present privacy risks, TMSI is sent instead. There is a before-hand agreement between the mobile station and the network during protected (ciphered) signaling procedures. For example, TMSI is

allocated by the network (MSC/VLR) on a location area basis. TMSI is allocated to a mobile station the first time it registers in a location area, and it is released when the mobile station leaves the location. VLR stores the TMSI.

MSISDN is a directory number that routes a GSM call to a mobile station. MSISDN is composed of the country code (CC), the national destination code (NCC), and the subscriber number (SN). The CC length is different from IMSI MCC and they have different values. NDC is used to identify either a destination network or a geographical area. SN identifies the subscriber's HLR not the called party number and HLR uses this number to provide routing instructions to other network entities in order to reach the subscriber. NDC and SN are administered by each country.

Furthermore, the **M**obile **S**tation **R**outing **N**umber (MSRN) is the routing number used on an incoming call between GMSC and the visited MSC. MSRN is used only between infrastructure machines. MSRN is provided on a call per call basis, for example, MSC/VLR provides a roaming number when updating the location information in HLR. MSC/VLR chooses the roaming number from a pool of free numbers, and links it temporarily with IMSI. As soon as the call is fully established, this number is released.

2.5 ANSI-41

ANSI-41 specifications were evolved through different versions that have made them increasingly more powerful and versatile. For instance, ANSI Revision 0 was published originally in February 1988 followed by Revision A in January 1991 and Revision B in December 1991 [26]. Revision C, which was first published in February 1996, can support second generation cellular systems. These ANSI-41 specifications refer to the protocols that deal with mobile network activities that are categorized in three main functions: intersystem handoff, automatic roaming, and intersystem operations, administration, and maintenance [79]. Authentication and call processing functions are also described within these specifications.

The intersystem handoff functions allow subscribers to roam while a call is in progress. They are divided into five categories as follows: handoff measurement, handoff forward, handoff back, path minimization, and call release.

Automatic roaming enables subscribers to originate calls, receive calls, and to access supplementary services transparently while roaming. The automatic term is used in the sense of invoking each function without requiring special subscriber actions. The functions related to automatic roaming are divided into mobile station (MS) service qualification, MS location management, MS state management, and HLR and VLR fault recovery. The MS service qualification functions involve validation and service profile information related to, respectively, financial responsibility and service capabilities for the roaming MS. These issues are not tackled in this thesis. The MS location management consists of location update and location cancellation (also known as registration cancellation and de-registration).

Authentication functions contains mechanisms to verify the identity of MS and require an authentication-capable MS as well as a mobile system able to support ANSI-41 authentication operations and calculations, which involves two secret numbers: the authentication key (A-key) and the shared secret data (SSD). The A-key is the permanent key and the SSD key is the temporary key used by the authentication calculations in both the MS and the AC. The authentication functions are divided into SSD sharing, global challenge, unique challenge, SSD update, call history count update, and authentication reporting.

Intersystem OA&M, which are out of the scope of this research, describe all activities to manage and operate a mobile telecommunication network. These functions include the configuration management, the network engineering, the network change control, the circuit management (e.g., trunk maintenance between MSCs), the fault management, the performance management, and the overload control.

Figure 8 illustrates the ANSI-41 network reference model that is composed of several network entities that are quite similar to the GSM components, as follows: mobile station (MS), base station (BS), mobile switching center (MSC), home location register (HLR), visitor location register (VLR), authentication center (AC), equipment identity register (EIR), message center (MC), and short message entity (SME). The integrated services digital network (ISDN) and public switched telephone network (PSTN) are also included in the ANSI-41 network reference model. This is a conceptual model that can be implemented in several ways, for instance, a piece of physical equipment can contain one or several of these entities.

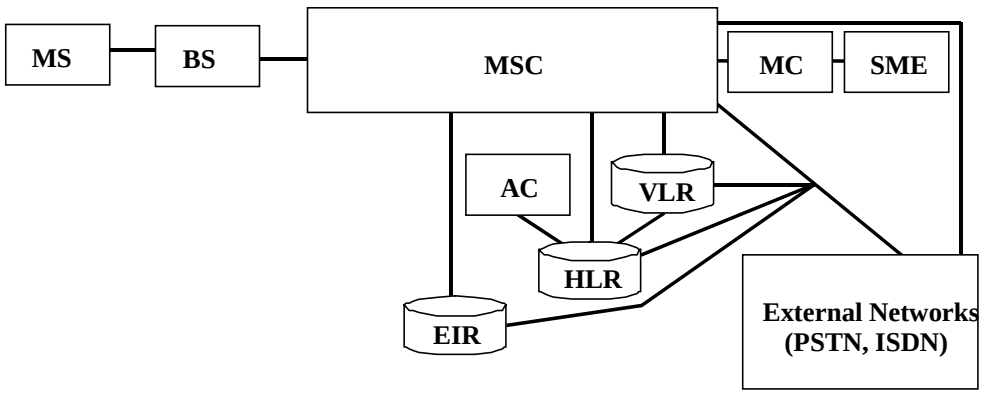


Figure 8 ANSI-41 Conceptual Model

As mentioned earlier, ANSI-41 is a protocol that operates on the network side and it is used in association with AMPS (ANSI-553), D-AMPS (IS-54-B/TDMA), IS-136 (TDMA), and IS-95 (CDMA) air interfaces [35][79]. These air interfaces define the operations between the MS and the BS. The operations between the BS and the MSC are specified, for example, by IS-634 (SS7 and Frame Relay) and IS-653 (ISDN).

As presented in [79], a high level view of the ANSI-41 protocol architecture is divided into application services and data transfer services. Application services incorporate the following OSI layers: application, presentation, session, presentation, and

transport. The ANSI-41 application services depend on the functions of SS7, which support transaction-oriented services. On the other hand, data transfer services wrap the network, data link, and physical layers. The ANSI-41 data transfer services are supported by X.25 and SS7 protocol sets.

According to [35], the ANSI-41 specifications for application services correspond to the Mobile Application Part (MAP). GSM systems also define a MAP protocol as part of their specifications. Currently, the interaction of the North American network protocols (e.g., ANSI-41 MAP) and the European network protocols (e.g., GSM MAP) is done through gateways.

2.5.1 Identifiers

In ANSI-41 networks, the subscriber identification is composed of two unique identifiers: the mobile identification number (MIN) and the electronic serial number (ESN) [79]. The former is used to identify a subscriber in many ANSI-41 operations and the combination MIN-ESN is used by the authentication and registration functions.

The cellular service provider assigns MIN to the mobile subscriber. The 10-digit MIN is composed of area code or Numbering Plan Area (NPA), the office code, and the subscriber number. The MIN functions are transmitted through the air interface during registration to inform the network of the subscriber's identity and it is the key field to access the user profile record stored in HLR. Although this number can be also implemented as the dial directory number of the mobile station, MIN is not meant to be the subscriber's directory number in ANSI-41 specifications. ANSI-41 provides a Temporary Mobile Subscriber Identity (TMSI) with the same functionality as the GSM TMSI.

ESN identifies a mobile station and it is stored permanently in the MS equipment by the manufacturer. It is composed of a manufacturer's code and a serial number. The internal circuitry that contains ESN guarantees protection against fraudulent contact and tampering.

The **T**emporary **L**ocal **D**irectory **N**umber (TLDN) is a network address that is temporarily (about 20s) assigned for call setup. This identifier is used to redirect a call from an originating MSC to a serving MSC in call delivery procedures, which are used to deliver mobile-terminated calls to roaming subscribers. For example, when a user is originating a call (call origination from external source), HLR knows whether the user is roaming or not and sends a ROUTREQ message to the appropriate VLR in order to get a TLDN. TLDN is dynamically allocated and it is released when the call is completed. This TLDN is associated with VLR and consequently with the serving MSC. ANSI-41 TLDN identifier is close to the functionality of the GSM MSRN identifier. Both subscriber identifiers are based on the subscriber's routing information. Fixed networks such as ISDN and PSTN also use these identifiers for routing purposes.

2.6 Wireless Intelligent Networks

A set of ITU-T recommendations called **I**ntelligent **N**etworks (IN) [110][111] has been applied to fixed telecommunication networks since the end of the 80's with the goal of separating call processing and services from the signaling related to the switches. In short, IN defines and standardizes a way to support the development of services independently of the signaling protocols related to switches. It is important to mention that IN is not an architecture but a framework model, which includes several capabilities. The first set of IN capabilities (capability set 1 or CS-1) was approved in 1995 by ITU-T [111] and the second set (capability set 2 or CS-2) in 1997 [110].

The IN concept is responsible for the fast growth and success of the telecommunication sector in the last decade since it facilitates the development of supplementary services such as call forward on busy, call waiting, and call name presentation that have been attracting fixed users.

The need for advanced services has been the motivation for the introduction of the IN concept in cellular systems. As stated earlier, the IN model allows the fast development of new services that can offer competitive advantages among wireless carriers. Since mobile users are willing to have the same supplementary services offered by fixed networks, the standardization community is working on the integration of the wireless technologies and the IN standard [70]. **W**ireless **I**ntelligent **N**etworks (WIN) [174][175][176] have emerged in North America as a result of this effort. IN capability sets are also the foundation for third generation systems.

The next subsections introduce general definitions related to the North America effort to integrate IN and ANSI-41. In addition, WIN services, the Distributed Functional Model (DFM), Network Reference Model (NRM), and the mapping of FEs to NEs described in [176].

2.6.1 WIN: Basic Concepts

The IN model was proposed to better define new services. Switches, databases, and peripherals are available to provide these new IN capabilities for fixed networks. The concept of Wireless Intelligent Network (WIN) has been developed by the Telecommunication Industry Association (TIA) Standards Committee TR-45.2 [174][175], which is part of ANSI, to drive the Intelligent Network capability into ANSI-41-based wireless networks [26]. WIN was the first attempt within the telecommunication standard community to integrate wireline (fixed) and wireless networks seamlessly. As discussed in Section 2.5, ANSI-41 wireless networks also use SS7 messages to exchange information, however, new messages are added to support the roaming capability. North American carriers of wireless networks that are members of the Cellular TIA (CTIA) have identified WIN as the target architecture for the existing, under development, and third generation systems for telecommunication networks.

As mentioned earlier, the three major IN principles are independence of service, separation of basic switching functions from service and application functions, and independence of applications from lower-level communication details. In order to support wireless networks, mobility and radio resource management functions have to be added to these IN principles. As a result, WIN separates call processing intelligence and feature functionality from network switches, includes mobility and radio resource management functions, and offers a diversity of enhanced services to subscribers. The benefits of WIN are the same as those of IN: a flexible network architecture and a powerful set of network capabilities. Besides this, WIN provides the functionalities of wireless networks by supporting mobility and radio resource management functions.

2.6.2 WIN Services

The first phase of the WIN standard covers the following three major services: Incoming Call Screening (ICS), Voice Controlled Services (VCS), and Calling Name Presentation (CNAP). The second phase of WIN [175] is under development and contains services related to billing such as freephone, pre-paid, and location-based services.

Incoming Call Screening (ICS) provides for alternate routing, blocking, or allowing of specified incoming calls. The incoming calls have one of five potential termination treatments that correspond to the following screening functions: terminated normally to the subscriber (with normal alerting or with distinctive alerting), forwarded to another number, forwarded to voice mail, routed to subscriber-specific announcement, and blocked. Besides these screening functions, ICS can use a number of screening factors to determine which termination action is appropriate. These factors are related to calling party characteristics like identity, speech or voice-based identification procedure, and passwords. They can also be related to called party characteristics such as location, status, date and time.

Voice Controlled Services (VCS) employ voice recognition technology to allow wireless users to control features and services using spoken commands. For instance, VCS employs speech recognition technology to allow mobile users to control features and services using speech commands, names, and numbers. There are two types of Automatic Speech Recognition (ASR). The first is user dependent and requires specific phrases spoken by the user. The second is user independent and requires the use of specific phrases for commands that are independent of the user and, in this case, the user doesn’t need to train the system.

Calling Name Presentation (CNAP) provides the name identification of the calling party (personal name, company name, “restricted”, “not available”) to the called party. CNAP offers the identification of the caller user’s name to the called user. The Calling Name Information (CNA) is derived from the Calling Number Information (CNI) that is provided by the terminating networks as part of the basic call establishment. Optionally, date and time can also be available to the called user.

FreePhone allows any user to call a freephone number that is associated with the called party. The called party is charged for the freephone call, including access, air-time, and international fees. The caller party does not pay anything.

Pre-paid services allow the mobile user to pay telecommunication services in advance. A pre-paid mobile user establishes the call with the service provider before getting access to the services. Accounting should show a positive balance to allow call establishment; otherwise, the call is denied. The call can also be interrupted as soon as the balance reaches a threshold.

Location-based charging services (LBC) allow the service provider to change the service charges or fees according to the mobile station location. This service also allows the service provider to offer or deny a specific service based on the mobile station location. LBC can be only applied if the billing party is a mobile user.

2.6.3 Distributed Functional Model

The Distributed Functional Model (DFM) is the functional plane that describes the WIN services mentioned earlier. Each function is presented in terms of Functional Entities (FEs). Different FEs exchange messages with a set of information flows (IFs) that describe the relationships between them. The IFs show the general specification of services that are implemented at the later stages in a physical platform.

Figure 9 depicts the Distributed Functional Model with FEs, and their relationships in the context of the WIN standard [174]. A grouping of actions across one or more FEs, when coordinated by information flows, provides the required WIN service execution. This functional model is non-service specific and does not imply any limitations regarding physical implementations or distribution of functions to physical platforms. It represents essentially the viewpoint of a network designer.

The FEs can be classified in relation to the execution of services and in relation to the management and creation of services. Relationships among them are based on the client-server model. Under certain circumstances, a FE performs either as a server or as a client. For example, each IF can behave either as a client request or a server reply. The roles of the FEs are summarized as follows:

- *Authentication Control Function* (ACF) provides the service logic and service data function for authentication, voice privacy and signaling message encryption.

- *Call Control Function* (CCF) provides the basic switching capabilities available in any switching system, including call and service processing and control.

- *Location Registration Functions* (LRF_V and LRF_H) provide the service logic and service data function to manage the mobility aspects for wireless users. They are respectively associated with the VLR and HLR network entities.

- *Mobile Station Access Control Function* (MACF) stores subscriber data and dynamically associates system resources with a particular set of call instance data.

- *Radio Access Control Function* (RACF) provides the service logic and service data functionality specifically related to radio link.

- *Radio Control Function* (RCF) provides radio ports and radio control.

- *Radio Terminal Function* (RTF) is an interface that provides network call control functions to wireless users.

- *Service Control Function* (SCF) handles call control functions in the processing of WIN-provided and custom service requests.

- *Service Creation Entity Function* (SCEF) provides the capability for creation, verification, and testing on WIN services.

- *Service Data Function* (SDF) contains customer and network data for real-time access by the SCF in the execution of WIN-provided services.

- *Service Management Access Function* (SMAF) provides the human interface to service management functions.

- *Service Management Function* (SMF) provides overall service management functionality for the network. The SMF may interact with any or all of the other FEs to perform service provisioning, monitoring, testing, and subscriber data management functions.

- *Service Switching Function* (SSF) is associated with CCF and provides the set of functions and the recognition of triggers required for interaction between the CCF and SCF.

- *Specialized Resource Function* (SRF) provides the specialized resources required for the execution of WIN-provided services (e.g., digit receivers, announcements, conference bridges, etc.).

Figure 9 assumes that some functional entities have links to other entities of their own type (it is the case for SCF, CCF, ACF, and RACF). The FEs related to wireless access mobility (ACF, LRF_V, LRF_H, MACF, RACF, RCF and RTF in the figure) were added in WIN since they are not part of the original IN CS-2 [111].

When a WIN service is required, this service is managed by the IF that is specified for each request, leading to the execution of a number of actions through one or more FEs. This functional model is service independent and it does not imply any limitation of either physical implementation or functional distribution of physical platforms. This model describes essentially the point of view of the designer.

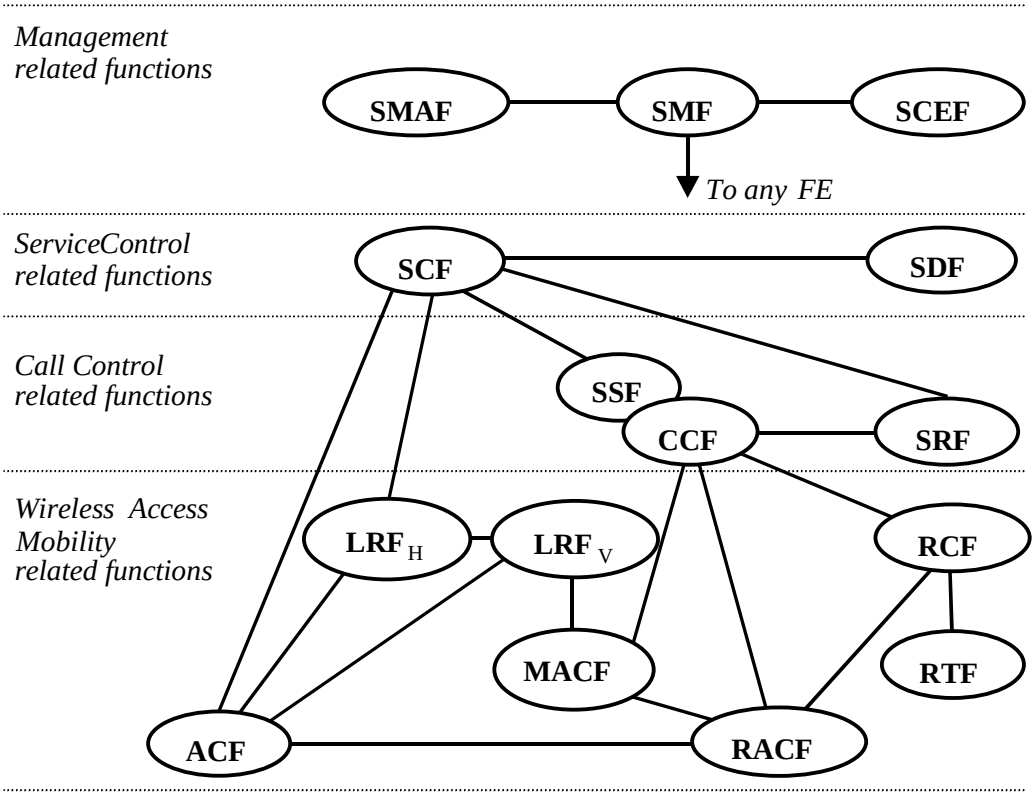


Figure 9 Wireless Distributed Functional Model (adapted from Figure 1 of ANSI-41.7-WIN [174])

This functional plane is common between WIN and IN [70] with the exception of the FEs related to mobility and radio resource management. The other FEs are also described in the IN functional plane and are complementary to CS-2. Both IN and WIN describe messages through the IF and the relationships among FEs. Each IF is presented in terms of main function, pre-condition, post-condition as well as Information Elements (IEs) such as messages that each IF carries.

2.6.4 Network Reference Model

The IN conceptual model describes a physical plane that contains two main aspects. First, physical entities (PEs) are described and they have their counterpart in WIN that are called network entities (NEs). Second, IN defines an application protocol called Intelligent Network Application Protocol (INAP), which counterpart in WIN is the Mobile Application Protocol (MAP). NEs are different from the PEs since one or more NEs can be part of a single PE. On the other hand, the concept of combining two or more entities to generate other entities that belong to a different plane is also used in WIN as follows. Two or more FEs can generate a single NE, for instance, a HLR can contain the functions described in the LRF, SCF, and SDF, as well as a MSC can contain the CCF, SSF, RACF, LRF, and SRF functions. WIN does not define PEs to offer flexibility to future implementations.

The Network Reference Model (NRM) defines network entities (NEs) and the associated interface reference points that may logically comprise a wireless network. In essence, the NRM facilitates the specification of messages and protocols within WIN stage 2 and stage 3 documents by allowing the functional entities to be mapped to network entities in the physical plane. Figure 10 depicts a simplified version of WIN NRM. Some of the NEs have links to other entities of their type (e.g., MC, MSC, SCP, SME, and VLR). All these network entities are defined in ANSI-41.1 except for IP, SCP, and SN that have been added in WIN. A short description of these network entities follows:

- *Authentication Center* (AC) manages the authentication information related to the MS.

- *Base Station* (BS) comprises a set of *Base Station Transceivers* (BST) and their *Base Station Controller* (BSC). The BS includes all radio equipment located at each cell.

- *Equipment Identity Register* (EIR) is a register to which a user's equipment identity may be assigned for record purposes.

- *Home Location Register* (HLR) is a location register to which a user's identity is assigned for record purposes such as subscriber information (e.g., profile information, current location, authorization period, etc.)

- *Intelligent Peripheral* (IP) performs specialized resource functions such as playing announcements, collecting digits, performing speech-to-text or text-to-speech conversion, recording and storing voice messages, facsimile services, data services, and so forth.

- *Message Center* (MC) stores and forwards short messages.

- *Mobile Station* (MS) is the interface used to terminate the radio path at the user side. It provides the capabilities to access network services by the user.

- *Mobile Switching Center* (MSC) constitutes the interface for user traffic between the cellular network and other public switched networks, or other MSCs in the same or other cellular networks.

- *Service Control Point* (SCP) acts as a real-time database and transaction processing system to provide service control and service data functionality.

- *Short Message Entities* (SME) compose and decompose short messages.

- *Service Node* (SN) provides service control, service data, specialized resources and call control functions to support bearer related services.

- *Visitor Location Register* (VLR) stores and retrieves information for handling calls to or from a visiting subscriber.

Real implementations of the NRM may vary with respect to how the network entities are distributed among various actual physical units. In the case of network entities, which are combined in the same physical equipment, the interface reference points become internal and do not adhere to interface standards. For instance, it is often the case that the VLR is part of the same piece of equipment as the MSC or as the HLR. The network entities are not really physical in the IN sense, but are rather a mix of IN functional and physical entities. This problem persists in much of the WIN specification, and especially in the several Message Sequence Charts based on NEs.

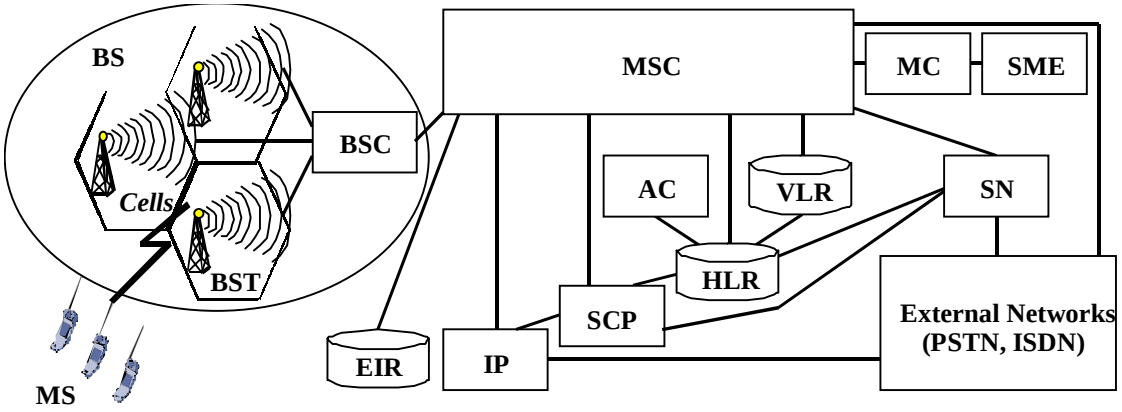


Figure 10 Wireless Network Reference Model
(adapted from Figure 2 of ANSI-41.1-WIN [174])

2.6.5 Mapping of DFM to NRM

DFM is a valuable tool for identifying the functions to be performed by network entities in NRM without restricting possible implementations. However, implementations ultimately require that FEs be allocated to specific NEs.

The WIN standard contains, as an informative annex, an example of mappings between FEs and NEs as illustrated in Figure 11. This annex is not intended to restrict or prejudice in any way other possible allocations of FEs or interfaces. The standard does not enforce this mapping in order for the industry to preserve the freedom that is necessary for the evolution of future implementations. In this figure, FEs and NEs involved in the ICS service are shaded in gray, and some minor NEs (EIR, MC and SME), which are often integrated with other NEs, have been omitted.

This mapping is similar to the one in [174], and this combined structure enables the generation of stage 2 and stage 3 information flows in the WIN standards.

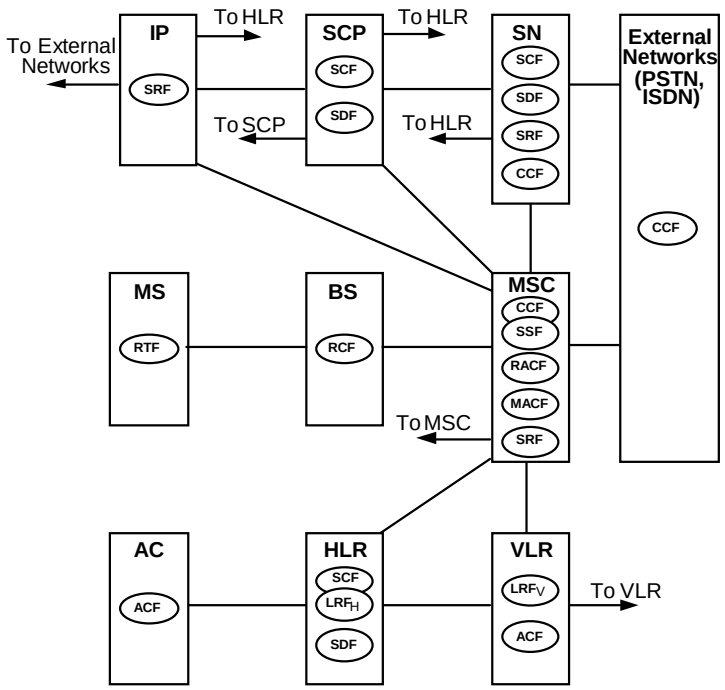


Figure 11 Possible Mapping of WIN Functional Entities to Network Entities (adapted from Figure 40 of ANSI-41.7-WIN [169]**)**

2.7 Universal Mobile Telecommunications System

ETSI had led the standardization of the Universal Mobile Telecommunications System (UMTS) [33][78][98][120][141] in Europe from 95 to 98 when the 3GPP standardization group was created. The 3GPP took over the process in order to prepare UMTS to be implemented worldwide and to avoid duplication of efforts.

The UMTS goal is to provide integration and transparency of services and to enable multimedia services for GSM and GPRS mobile users with the quality necessary to achieve a seamless environment for future broadband networks. UMTS consists of several projects that are being developed in parallel. Besides the support of multimedia services and other telecommunication services for mobile users, these projects aim to provide bearer communications for the lower layers and to provide these services in individual equipment.

In short, UMTS is a third generation cellular system that offers:

- broadband and multimedia services with quality compatible with fixed networks;
- speed of 144 kb/s for all environments with high mobility;
- speed up to 2 Mb/s for local environments with low mobility;

- a new air interface based on packets, a new spectrum, and a new network technology in addition to the existing technologies;
- capability to provide services to 50% of the European population;
- capability to create and to offer services independent of the network operation;
- interconnection of multiple networks.

The UMTS flexibility reduces the investment for the network operators as well as facilitates the migration of second generation to third generation. For instance, second generation base stations, which constitute the biggest investment for the existing mobile systems, can be reused.

Figure 12 illustrates the UMTS architecture [98], which is a simplified version of the overall architecture presented in the first release. This architecture is able to support multiple core networks such as GSM/GPRS, ISDN, PSTN, and networks based on the Internet Protocol (IP).

The air interface of the UMTS Terrestrial Radio Access Network (UTRAN) is connected to two separate UMTS core networks as follows: the circuit domain based on the enhanced GSM mobile switching center (E-MSC) and the packet domain based on the Enhanced GPRS support nodes (E-GSNs). The HLR handles subscriber data and provides mobility for both domains. The other edge nodes, which are shown in the figure, are the enhanced serving GPRS support node (E-SGSN), and the enhanced visited mobile switching center (E-VMSC). A radio network controller (RNC) is responsible for the communication between the site controller and the Core Network (CN).

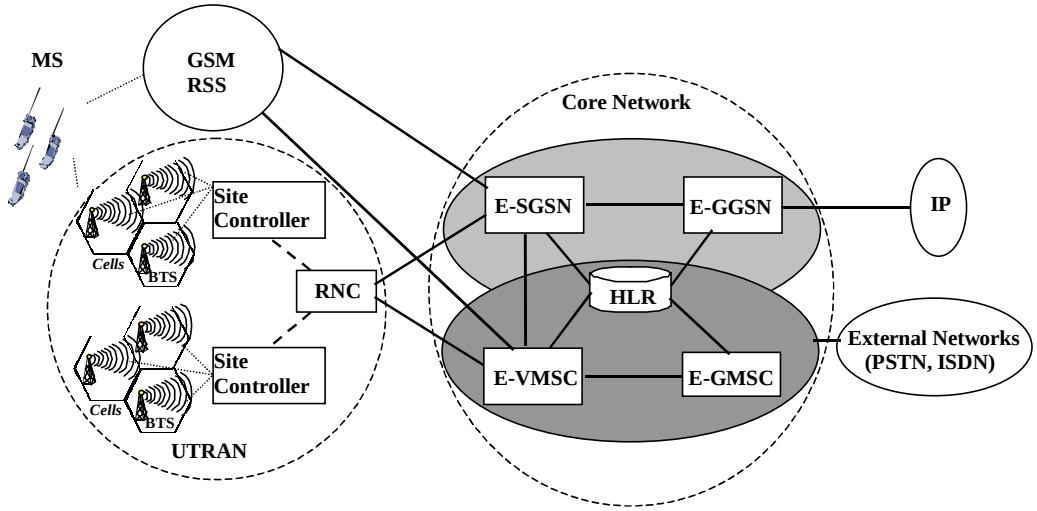


Figure 12 Simplified UMTS Architecture

(adapted from Figure 2 of [98])

Mobile stations are able to use UTRAN and GSM radio subsystems (RSS) that act as complementary access systems to the same network infrastructure (core network). The UMTS UTRAN is also able to support different wireless technologies such as Code Division Multiple Access (CDMA), Time Division Multiple Access (TDMA), and Digital European Cordless Telephone (DECT). For instance, Figure 12 depicts an interface between the mobile station and UTRAN that is able to support wideband CDMA (W-CDMA) for FDD mode and time-division/CDMA for TDD mode. In addition, UMTS is also able to support the IN principles that allow a wide range of supplementary services and value-added voice services.

UMTS is a family member of the ITU-T effort to develop a family of third generation systems called IMT-2000. This system is presented in the next sub-section.

2.8 IMT-2000 Systems

The need for providing a global roaming capability to mobile subscribers has been one of the motivations for the development of third generation systems, as mentioned earlier. In 1998, ITU-T started the development of a new international third generation standard called IMT-2000 systems [33][78][98][112][113][114][166]. Several countries are involved in IMT-2000 related activities. For instance, the following standardization groups are developing activities related to IMT-2000 systems: ITU-T Sub-Group 11; ITU Radio Communications (ITU-R) Task Group 8/1; the Association of Radio Industry and Business (ARIB) and the Telecommunications Technology Committee (TTC) in Japan; ETSI in Europe; TIA in North America; the Telecommunications Technology Association (TTA) in Korea; and the Ministry of Posts and Telecommunications (MPT) in China.

The IMT-2000 systems are under development to standardize the use of the same spectrum bandwidth for mobile users around the world (1.8 – 2.2 GHz), multiple radio environments (cellular, wireless, satellites, and LANs) as well as a variety of communication services (voice, data, and multimedia), and the integration with Internet services. The following aspects should be supported by the IMT-2000 systems: speed around 2Mb/s for LANs, maximum use of the IN capabilities, global roaming, security, high performance, and integration with satellites and terrestrial systems.

ITU-R Task Group 8/1 is responsible for the IMT-2000 radio aspects. On the other hand, the ITU-T Sub-Group 11 is responsible for the definition of the signaling and protocol requirements of the IMT-2000 interfaces. Within these ITU groups, different teams are responsible for the radio interface layer, application services, mobility management, call control, bearer communications, and security.

Figure 13 illustrates the IMT-2000 family concept that defines how family members such as UMTS should support users of other family members by offering roaming services as well as providing a set of consistent services. The IMT-2000 documents provide interfaces and a set of capabilities to achieve this goal.

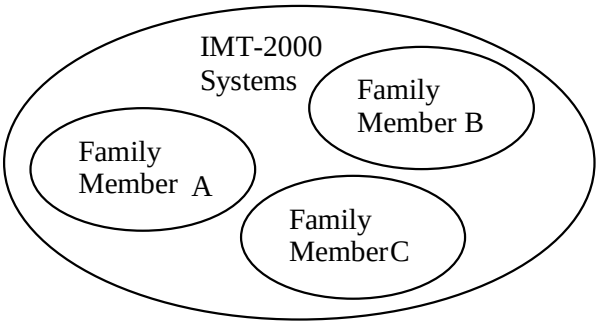


Figure 13 IMT-2000 Family Concept

(adapted from Figure 2/Q.1701 [112])

The ITU-T recommendations for IMT-2000 systems present the following order of development: the IMT-2000 framework, the signaling requirements (information flows and interfaces), and the protocols. Figure 14 illustrates the sequence among these recommendations.

Q.1701 recommendation introduces the complete framework for the IMT-2000 signaling requirements. This framework contains a description of the IMT-2000 family system concepts; the identification of services and network capabilities for the first set of IMT-2000 capabilities; the identification and description of the interfaces necessary for the standardization to support the Capability Set 1 (CS-1); and, last, the description of the IMT-2000 recommendations.

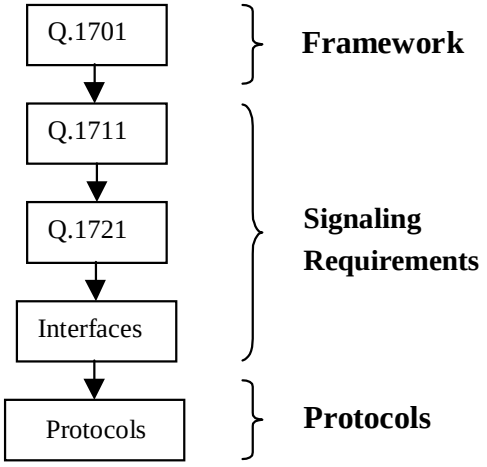


Figure 14 IMT-2000 *Recommendations*

Q.1711 identifies the specific network and interface functions for the support of IMT-2000 systems. In addition, this recommendation uses functional entities (FEs) to generate a distributed functional model (DFM) that is mapped to a network reference model (NFM) at later stages. These models were respectively proposed by the Intelligent Network (IN) and the Wireless Intelligent Network (WIN) standards (more details in Section 2.6). Q.1711 also introduces the foundation for the development of the IMT-2000

information flows and the definition of the functional entity actions (FEAs). Finally, requirements for the development of several signaling interfaces identified in the Q.1701, the global roaming, and the network interconnections are also presented.

Figure 15 depicts an alternative of the mapping of the FEs described in the IMT-2000 Distributed Functional Model (DFM), which is not shown in this thesis, to the Network entities (NEs). The following NEs incorporate one or more FEs: Authentication Center (AC), Base Station (BS), Mobile Switching Center (MSC), Drift MSC (DMSC), Gateway Location Register (GLR), Gateway MSC (GMSC), Home Location Register (HLR), Intelligent Peripheral (IP), Mobile Terminal (MT), Radio Network Controller (RNC), Packet Data Serving Node (PDSN), Packet Data Gateway Node (PDGN), Service Control Point (SCP), Service Data Point (SDP), User Identity Module (UIM), and Visitor Location Register (VLR).

These NEs are grouped in the functional subsystems presented in the Q.1701 document as follows. The User Identity Module (UIM) subsystem supports user security and services that can be implemented either in a removable physical card for a mobile terminal (MT) or can be integrated into the physical MT. The MT functional subsystem supports communication between the UIM and radio access network (RAN) as well as user services and mobility. RAN functional subsystem provides the communication between the MT and the core network (e.g., acts like a bridge, router, or gateway). Last, the Core Network (CN) subsystem supports both communication with the mobile terminal as well as user services and mobility.

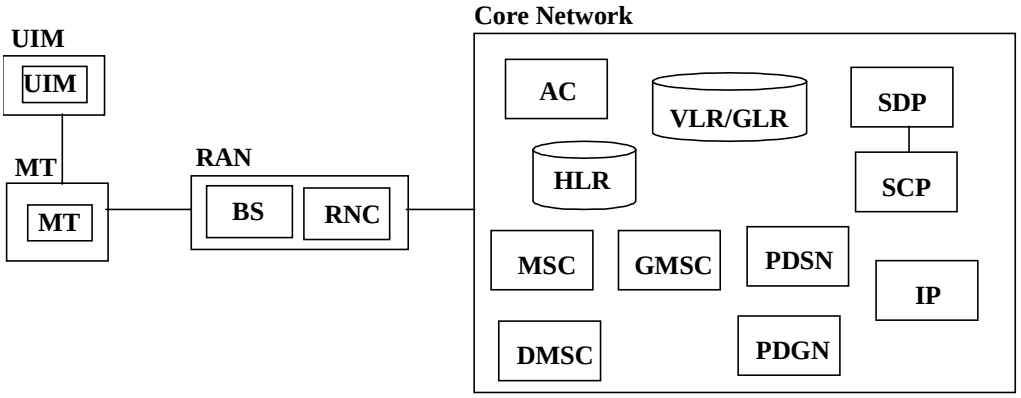


Figure 15 Generic Reference Model of IMT-2000 Systems (adapted from Figure 6A/Q.1711 [113])

Recommendation Q.1721 is under development and it offers detailed information flows for IMT-2000 services (stage 2 documents), network capabilities, and details for the establishment of signaling channels that are not visible for the end users. Interfaces and protocols are left for future development stages.

These recommendations define three types of general services that intend to solve the demand for global roaming, as follows: terminal mobility services, personal mobility

services, and advanced mobility services. Terminal mobility services allow a terminal to access services ubiquitously starting from any geographical point and going across any set of points during roaming. Personal mobility services provide the ability to end users to access the subscribed services from any terminal (fixed or mobile) in any location. Advanced mobility services offer the ability to access subscribed services ubiquitously (also known as service portability).

2.9 Wireless mobile ATM Networks

Asynchronous Transfer Mode (ATM) was developed in the 90s to support high-bandwidth multimedia applications and to provide bandwidth on demand, traffic integration, cost effectiveness, as well as flexible data networking [156]. Nowadays, ATM is viewed as a strong candidate to extend these services to portable and mobile systems using wireless technologies [1][179][185]. Accordingly, several alternatives for adding mobility to ATM signaling protocols have been presented in the literature. For example, [29], [54], and [180] present Wireless mobile ATM (WmATM) networks as a wireless extension of ATM networks with mobility and no modification in the existing ATM signaling protocols. On the contrary, [158], [185], and [187] believe that minimum changes should be done in the ATM networks to support mobility and to achieve a global WmATM network environment. In [4] and [5], the authors present two different signaling protocols that support both alternatives.

Like third generation systems, the WmATM network is a seamless alternative for providing mobile wireless networking independently of wireless access technologies such as IS-54 TDMA and IS-95 CDMA. Different from the IMT-2000 systems and the UMTS that offer transmission speeds up to 2Mb/s with high mobility capability, WmATM aims at supporting the transmission speed of 10Mb/s with limited mobility capability.

WmATM networks require the support of mobility-related functions for call establishment in addition to the ITU-T Q.2931 ATM signaling protocols [115]. For instance, these functions include mobile user authentication and registration, location management, routing of mobile connections, and handoff control.

Figure 16 illustrates a possible environment that can support the concepts involved in designing a global WmATM network.

The wireless service area is divided into cells and each cell is equipped with a base station transceiver (BST) that is responsible for the use of the allocated spectrum. A base station (BS) is responsible for a set of BSTs that are connected to the BS through wireless access ports. Several mobile stations (MSs) share the capacity of each BST. A wireless ATM network backbone is composed of WmATM switches attached through high-speed transmission links. Databases are responsible for keeping information about mobile users. In the ATM fixed networks [115], there is no need for databases that keep track of the user location since each fixed station has a user's identification that determines where the

user is and how to route a call to the user. The wireless backbone can communicate with the ATM network backbone using wired access ports.

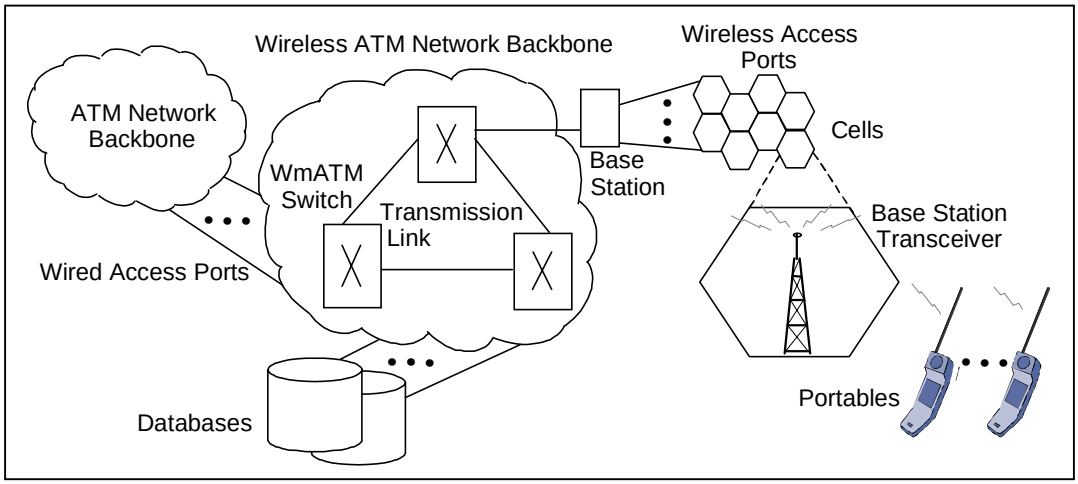


Figure 16 A Possible Wireless Mobile ATM Network Environment (adapted from Figures 2, 3 and 4 of [2], [29], and [158], respectively)

In order to design a WmATM network that supports mobility, communication and handoff procedures, changes to the current ATM network protocol layers should be done. Figure 17 illustrates three components of the reference architecture shown in Figure 16 and their corresponding WmATM and ATM protocol layers. The latter is defined by ITU-T as follows: the physical layer (PHY) transports information (bits/ATM cells); the ATM layer performs switching, routing, and multiplexing; the ATM adaptation layer (AAL) is responsible for the adaptation of the service information from the higher layer protocols to the fixed size of ATM cells [156].

The wireless physical layer consists of radio access control and Medium Access Control (MAC) and Radio sub-layers responsible for maintaining the error performance at a minimal tolerable level [54][158]. The radio access control sub-layer guarantees the WmATM high-speed physical-level transmission and reception. In a range of 100-500m, the target bit rates are around 25Mb/s. However, these bit rates are reduced to 1-10 Mb/s due to the current capability of modulation methods, such as spread spectrum CDMA. On the other hand, the MAC layer supports the shared use of the radio channels by multiple mobile stations. In addition, this layer should be able to maintain the high radio channel efficiency while providing the ATM traffic classes (available bit rate, variable bit rate, and so on) with associated quality of service (QoS) controls.

The wireless data link layer performs retransmission only for data services and guarantees error control before sending cells to the wireless network layer. The wireless ATM layer and the wireless ATM Adaptation layer guarantee high-speed reliable point-to-point links and include part of location management and handoff functions. The wireless network layer supports mobile user authentication and registration, as well as routing of mobile connections. The user application layer contains the signaling protocols for setting up the calls between users, for maintaining these calls, and for releasing them.

Supplementary services that allow users to have some control over the originated and terminated calls are also included in this layer.

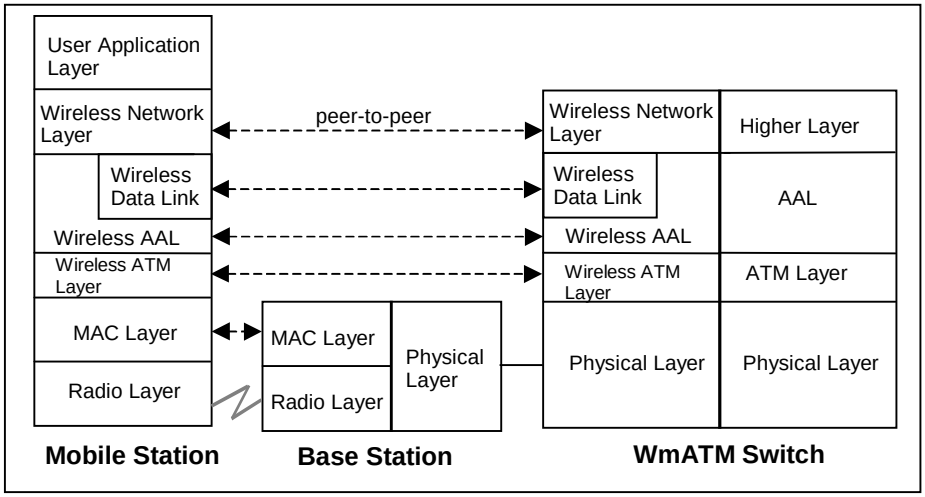


Figure 17 WmATM Network Protocol Layers

(adapted from Figure 4 of [54], Figure 4 of [158], and Figure 3 of [29])

2.10 Conclusion

The need for evolution and improvement of second generation systems is mainly due to the demand for supplementary services with the same quality as those available for fixed networks. In addition, the provision of the global roaming capability that offers transparency and integration of systems is the main motivation for the development of third generation systems.

The North-American WIN standard, which also belongs to second generation systems, aims to introduce IN concepts within the ANSI-41 cellular networks. In short, WIN intends to solve the demand for supplementary services and guarantees portability with existing ANSI-41 standards. While WIN sets the direction for third generation systems in North America, GSM and GPRS provide the foundation for third generation systems in Europe that is called UMTS.

UMTS and IMT-2000 systems are third generation systems that aims to work globally and transparently when any user requests their services anywhere at anytime. The development of the GPRS standards and the second phase of the WIN standards are occurring in parallel with the development of the ETSI UMTS, which is a family member of the ITU-T IMT-2000 systems.

This research focuses on the investigation of common mobility and radio resource management functions among second and third generation systems as discussed earlier. Patterns, which are introduced in the next chapter, are captured from these commonalities (see Chapter 4) and then documented (see Chapter 5). It is also important to mention that

the increasing complexity introduced by mobile systems has lead the standardization community to search for better development approaches. In this context, different approaches, which contain different techniques at different development stages, have been proposed in the literature [8][12][13]. These approaches are discussed in this thesis and an approach for reuse and validation of patterns is introduced in Chapter 6.

Chapter 3 Overview of Software Patterns

This chapter summarizes the main concepts of software patterns that are related to this work. First, it gives the reader a background on patterns. Then, an overview of the software pattern and the pattern language concepts is introduced. In addition, guidelines on how to write a pattern are discussed followed by an explanation of the types of patterns available in the literature and the existing formats used to describe them. Related research regarding patterns for telecommunications, distributed processing, and design process is also outlined. Finally, a discussion about the pattern concept and concluding remarks are presented.

3.1 Introduction

A *pattern* is defined in [6] as "an entity that describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you use this solution a million of times over, without ever doing it the same twice." In programming as in other areas, many problems also occur recurrently and they are solved each time they appear. In this context, the concept of patterns, which is conceived to describe buildings and towns in **[6]**, is also valid as a way of describing good solutions for specific analysis and design problems in the software engineering domain **[165]**.

The software community is applying the pattern concept to recurring problems and solutions in software development. Several software patterns have been published in the literature **[52][75][80]**. Recently, an almanac of published patterns is also available **[161]**. This almanac organizes the published patterns in a wide range of categories that goes from the categories defined by each pattern author, such as creational, behavioral, and structural in **[80]**, to categories that group patterns by their application domains (e.g., accounting, design process, distributed systems, and telecommunications).

Particularly in domains that work with complex and distributed systems such as telecommunications, the pattern concept has been applied to facilitate the development, the maintenance, the evolution, and the reuse of software. Despite the confidentiality problems in telecommunications, software developers and maintainers from different industries are working together to identify patterns in this area **[3][31][63]**.

In order to address software reusability, the next section introduces a summary about software evolution, frameworks (basic concepts and ideas behind them), and patterns. Then, Section 3.3 presents an overview of the pattern and the pattern language concepts. Section 3.4 introduces guidelines on how to write a pattern. Section 3.5 outlines a

classification of patterns and introduces different pattern formats. Section 3.6 focuses on patterns for telecommunications. First, it discusses the pattern template used in this domain and then presents a summary of published patterns as well as experience reports on reusing the existing patterns. Section 3.7 discusses the misconceptions about the pattern concept. Finally, our main conclusions and future work are outlined in Section 3.8.

3.2 Reusability: Software Evolution, Frameworks, and Patterns

With the application of the concept of modularity in the 70's **[146][197]**, software changed its design approach by becoming more structured. In addition, several programming languages were developed to support these structured software systems. Before the module-oriented programming languages, functional and top-down design decomposition, also known as stepwise refinement [196], were the trends. At that time, the main idea was the existence of one main routine that uses others for its implementation.

Various object-oriented languages have emerged after the module-oriented programming languages to improve software flexibility and, thus, reusability. These languages describe systems as *objects* and support the basic concepts of object-orientation, which are data abstraction, inheritance, polymorphism, and dynamic binding **[60][153][164]**.

Objects are entities with attributes and operations, respectively, called *instance variables* and *methods* at the implementation stage. Methods are defined for each particular object. *Classes* describe the scheme of an object (attributes and operations). Objects are instances of classes, which can be also viewed as the abstract data types described in module-oriented languages. *Inheritance* is described in **[164]** as "the mechanism of sharing attributes and operations using the relationships among classes" and in **[153]** as "the possibility of extending and adapting object descriptions and implementations without changing their source code." *Polymorphism* refers to the same operation behaving differently in distinct classes. Finally, *dynamic binding* represents a run-time determination of the proper method to be used when each class implements a different version of a method.

The concepts of inheritance and dynamic binding mentioned earlier are sufficient for constructing reusable, ready-to-use, and semi-finished building blocks that are called *frameworks* in the OO literature **[153]** (see other uses of the term framework in Section 4.1 of Chapter 4). This framework also defines the composition and the interaction among its components, which constitute the overall system architecture. The term *application framework* is used to describe a framework that constitutes a generic system for a specific application domain **[151]**.

As presented in **[116][117][152]**, these object-oriented concepts are pre-requisites for the development of reusable software architectures and provide a quality assurance

measure. These capabilities distinguish object-oriented languages from module-oriented languages. In order to reuse modules in other systems, major modifications, which can introduce design errors, are necessary. Module-oriented languages are not suitable for constructing complex project-independent modules that meet the requirements evolution of a system. However, the reuse of single software components is a benefit of both object-oriented and module-oriented languages.

Most recently, the application of software patterns to the development of systems has also contributed to software quality and reusability improvements. As discussed in the next section, several software patterns are available in the literature with focus on different development stages. Design patterns are often object-oriented driven. The relation between design patterns and frameworks is described in **[152]** as follows: "design patterns can help to adapt a framework to specific needs and construct new frameworks incorporating mature and proven designs." According to **[152]**, when software patterns are used together with frameworks, they are a valuable means for documenting existing frameworks. In addition, they are useful for developing new reusable object-oriented software architectures with the application of design approaches that have already been matured in other frameworks.

In short, software reusability and the overall software quality have improved in the past decade with the application of object-oriented programming concepts, frameworks, and patterns **[151][152]**.

3.3 Patterns and Pattern Language

The *pattern* and the *pattern language* concepts have their roots in **[6]** and **[7]** that are two halves of the same work on architecture design. These books present detailed patterns for towns, neighborhoods, houses, gardens, and rooms. As stated earlier, the authors describe a pattern as an entity that creates a uniform definition of a problem and its solution, which can be applied many times in a system. These authors also mention that the pattern solution should summarize a characteristic that is common to all possible ways of solving the stated problem. According to their work, a designer can analyze and modify these patterns without losing the essence that is central to each pattern.

Another commonly accepted vision of patterns describes them as "a piece of literature that describes a design problem and a general solution for the problem in a particular context" **[57]**. Software patterns are not only a form of documentation of the knowledge that experts have, but they also describe real design solutions recognizable in multiple systems. Furthermore, they address problems and relationships that are hidden in the code or in the analysis and design system documents, but are established practices of a given domain **[160]**.

Software patterns have essential components that constitute the *pattern template* as follows: name, context, problem, solution, forces, known uses, resulting context, and related patterns **[133]**. Templates are a good practice to document patterns, since they

guarantee a consistent format and increase the possibility to learn, compare, and reuse patterns.

A *name* is a word or a short meaningful phrase. A *context* helps to have a wide view of where a problem arises by expressing its beginning, its essence, and its body. A *problem* to be solved is presented in a clear statement. A *solution* is both "the heart of the pattern" and "a property common to all possible ways of solving the stated problem" as stated in **[6]**. The *solution* should describe very clearly what is necessary to solve the problem and is classified into the following three types: a solution that can be used in all the occurrences of the problem (the pattern *known uses*) and represents the best solution for the problem; a solution that can be improved; and a solution that will be refined at the design and the implementation levels in different ways.

In addition, *forces* are positive or negative considerations to be weighed in order to choose the best solution and also to show why a problem is difficult to solve. The *rationale* explains the importance of the pattern and how it works [57]. The *resulting context* is the conclusion of the pattern and it also shows what can happen next as a consequence of the pattern. Finally, *related patterns* explain how the pattern is connected to other patterns that refer to the same problem.

These essential components are often found in the published patterns with a clear distinction or within each pattern description. Another component called *hypotheses* is used by **[6]** to talk about the degree of satisfaction provided by the pattern solution and the authors use asterisks to picture that. However, as pointed out in **[161]**, this pattern evaluation depends on whether a pattern has worked or not for several applications. This information is a result of experience reports. Thus, unless the pattern author has reused its pattern several times, it is hard to evaluate a pattern.

The software community has also adopted the term *pattern language* that was introduced by **[6]** and **[7]** as mentioned earlier. This concept was redefined in **[162]** as "a collection of patterns that work together to solve problems in a specific domain." In a pattern language, a resulting context of one pattern becomes the starting context of other patterns. A fundamental view of a pattern language is the description of the pattern relationships that is also stressed in **[6]**, as follows: "when you build a thing you cannot merely build that thing in isolation." These relationships constitute the whole system to be designed. In other words, the notion of sequence of patterns in a pattern language is crucial to explain how the language works as pointed out in **[6]**.

Pattern languages are used to guide the development of a new system as well as the evolution of an existing system. As a result, each pattern is not described as an isolated entity. The purpose is to create something general and abstract enough to be reused, modified, and related to other patterns.

3.4 The Pattern Writing Process

The software pattern literature has discussed the pattern writing process in different ways. For instance, [188] includes seven good habits on how to write patterns and **[133]** introduces a set of patterns, which are organized in a pattern language, on how to solve recurring problems that happen when writing a new pattern or a pattern language. In addition, [52] uses the term "*pattern-mining*" to describe six steps on how to write new patterns. The authors also mention two guidelines on how to integrate a new pattern into a pattern catalogue.

This thesis also introduces guidelines on how to write a new pattern, to rewrite an existing pattern, and to join patterns in a pattern language. These guidelines are split into four steps. It starts from the *decision* to write down a recurrent problem and its respective solution in a certain domain, followed by the *capture* of the problem and of its solution within this domain. After these initial steps, there is the *search* among existing patterns to determine whether the pattern is new or can be an improvement of an existing pattern. On the basis of the search result, the last step consists of either *writing* a new pattern or *rewriting* an existing pattern.

These four steps are useful for software developers who are interested in using their experience to write patterns. The pattern writing experience acquired while writing a pattern language for mobile wireless communication systems, which is introduced in this thesis, is used to explain these steps. Furthermore, the "*seven habits of successful pattern writers*" presented in **[188]** and "*a pattern language for pattern writing*" introduced in **[133]** are incorporated into these steps and mentioned throughout the next sub-sections. The steps and guidelines presented in **[52]** will be also included before the completion of the thesis.

3.4.1 Decision and Capture

The decision and capture steps are tackled in [188] with "*Habit 1: Taking Time to Reflect*." This habit mentions the need for taking time to reflect about the software experiences the pattern writer has had or is currently going through. The writer can think about how specific analysis and design problems were solved in a system that was built. To do this, the author suggests that the designer records these experiences "incrementally" and solves them step-by-step; the designer takes a look at other related systems "designed by other people;" and the designer finds at least two examples of the same problem and its solution before trying to write a pattern for it. These examples are also related to the *known uses* element of the pattern template.

Another alternative for the pattern decision and capture steps involves the investigation of design problems over systems that were not developed by the pattern writer. This thesis introduces patterns for mobility and radio resource management functions that are extracted from different mobile systems. First, each chosen system is described with the same notation to make the recognition and the capture of commonalities among them feasible. Second, these commonalities are investigated in

order to find recurring problems and their respective solutions. In this case, the *known uses* are the chosen systems themselves. More details about the decision and capture processes are presented in Chapter 4 and about the patterns in Chapter 5.

3.4.2 Search

A pattern writer should look at existing patterns as the third step to be considered before developing a new pattern. The writer is challenged with a hard task that is to search over a wide range of application domain patterns to avoid writing a pattern that already exists. However, this search can be done easily with the help of "*The Pattern Almanac 2000*" presented in [161]. The almanac presents patterns "that stand alone" and patterns "that work within a collection." A collection is a pattern language or a pattern catalogue. A pattern catalogue comprises a set of patterns that refer to a particular domain and are independent of each other.

If the pattern is new, the hard task that remains is to think about similarities as well as differences between the new pattern and the existing patterns (e.g., how the existing patterns are interrelated with the new pattern). In other words, how to integrate single experiences over a broad number of application domain patterns. This task is also useful for describing the *related patterns* element of the pattern template. The "*Pattern language for pattern writing*" **[133]** describes how to organize related parts within a pattern called *relationship to other patterns* (see Figure 18). Furthermore, before starting to put into writing the new pattern, a designer should also think about the pattern intention and the target audience. This makes the pattern easier to write and to understand (see more details in sub-category C of Figure 18).

If the pattern already exists, a designer can decide to improve it (rewriting step) or to reuse it without major modifications (see **Chapter 6** about pattern reuse). If the designer decides to improve an existing pattern (e.g., an analysis pattern or a design pattern), questions such as what is the reason for redesigning and what should change in the specific design should be answered in advance. In the reuse case, the designer should ensure that the pattern is right for the problem. In addition, the chosen pattern should be understood individually and in relation with the other patterns mentioned in its *related patterns*. If the pattern provides code examples, the designer should also study the code and usage; choose names for the pattern and operations that have a meaning in the current context; define classes; and implement the operations [133].

As mentioned in Chapter 1, there is no attempt in the literature to identify and document patterns for mobile systems that are the focus of our research. However, we can find related patterns in **[44][46][183]** and **[184]** as shown in **Chapter 5**.

3.4.3 Writing and Rewriting

The pattern writing and rewriting step is composed of several small guidelines. These guidelines are largely discussed in [133] that, as mentioned earlier, presents "*a pattern language for pattern writing*." In Figure 18, we rearrange the patterns grouped in this

pattern language in four different categories (“context setting patterns”, “pattern structure”, “making patterns understandable”, and “pattern language structure”) instead of the original five sections introduced in [133] as follows: “context setting patterns”, “pattern structure”, “naming & referencing”, “making patterns understandable”, and “language structure”. The patterns depicted in gray in the figure are the ones applied to write the patterns and the pattern language presented in Chapter 5.

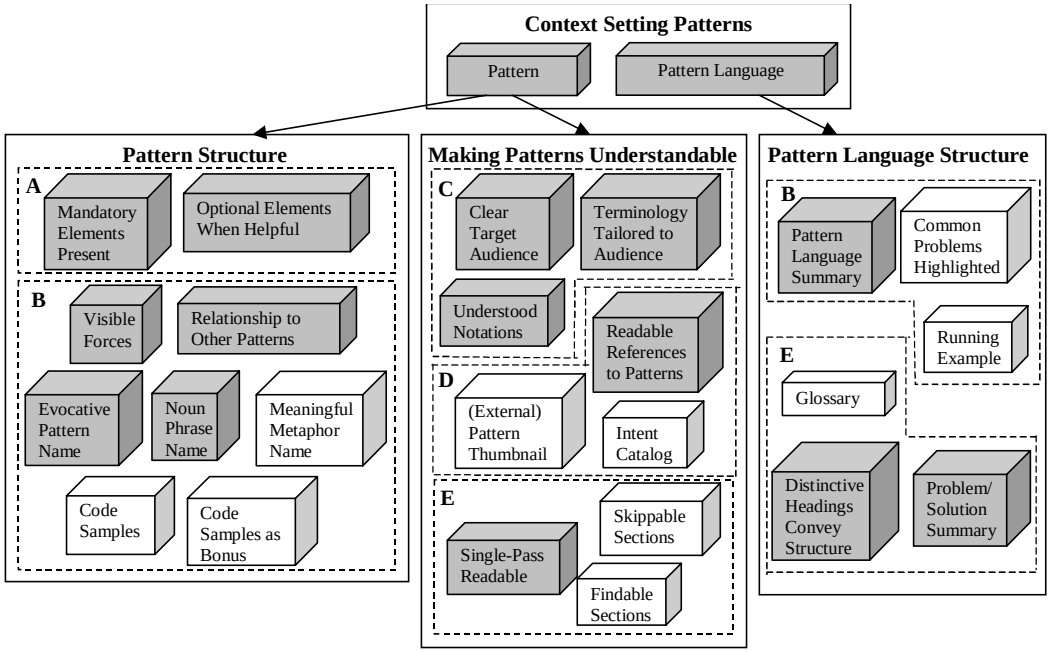


Figure 18 “*A Pattern Language for Pattern Writing*” Structure

(adapted from Figure 1 of [133]**)**

The description of *pattern* and *pattern language* that is done in the context setting patterns category (the same as the category presented in [133]) helps the reader to understand these concepts. In the remaining categories, which are discussed in the next subsections, sub-categories are introduced to organize the guidelines for writing patterns and pattern languages with the following subjects: (A) pattern template, (B) description of either the pattern or the pattern language elements, (C) audience, (D) referencing other patterns, and (E) readability.

3.4.3.1 Pattern Structure

As mentioned in [188] under the title the “Habit 2: Adhering to a Structure,” before writing a pattern down, it is important to decide on its template. The more information the pattern has, the more important the pattern template is as this habit stresses. The pattern

structure category shown in 0 includes patterns that refer to the elements that should be presented in the pattern description. The writer's decision about the pattern template can follow an existing form such as [75] and [80] (shown in Section 3.5). However, if the existing templates do not fit the writer's domain problem, a short or a new template can be defined based on *mandatory elements present* and *optional elements when helpful* (see sub-category A in the figure). For instance, some of the patterns for telecommunications, as presented in Section 3.6, are described with a template that contains only the mandatory elements. The patterns described in Chapter 5 are also presented with a short template.

After deciding about the template, the next step is the description of each pattern element. Problem and solution have been already tackled in the capture step (see Section 3.4.1). Forces, name, and related patterns are discussed on the basis of Figure 18 as follows. For instance, *visible forces* explain the forces within the pattern template and *relationship to other patterns* clarifies the related patterns. In addition, *evocative pattern name*, *noun phrase name*, and *meaningful metaphor name* give guidelines on how to name a pattern. Furthermore, this sub-category presents *code samples* and *code samples as bonus* to describe an optional pattern element called code samples. In **[133]**, while the former patterns are described in the "naming and referencing patterns" section, the latter patterns are included in the "making patterns understandable" section. However, since these patterns are related to the pattern template, we include them in the "pattern structure" category for clarity.

A good habit that is related to the pattern structure is called "Habit 3: Being Concrete Early" [188]. This habit stresses the importance of showing examples and counterexamples that illustrate key points of the pattern. These examples can be described in the motivation part as well as throughout the whole pattern description. The author also stresses that concepts are better understood when they are quickly illustrated with concrete examples. As stated earlier, a better explanation on how to include examples within a pattern can be found in the *code samples* and *code samples as bonus*.

3.4.3.2 Making Patterns Understandable

The "making patterns understandable" category focuses on readers (sub-category C), reference to other patterns (sub-category D), and readability (sub-category E) issues.

A pattern description becomes more attractive to the audience and easy to understand when it follows the guidelines contained in *clear target audience*, *terminology tailored to audience*, and *understood notations*.

The *readable references to patterns*, the *(external) pattern thumbnail*, and the *intent catalog* point out good writing practice in referencing other patterns. For this reason, these patterns are reorganized in a subcategory on "referencing other patterns" in contrast with the "naming and referencing patterns" section in which they are described in **[133]**.

Although *single-pass readable*, *skippable sections*, and *findable sections* are included in the "pattern structure" section in **[133]**, they describe ways to improve the readability of the pattern instead of being directly related to the pattern template. In Figure 18, these patterns are reallocated to the sub-category E within the "making patterns understandable" category. The pattern clarity and readability that is the subject of this subcategory are also mentioned in [188] as "Habit 5: Presenting Effectively" that expresses the effectiveness of how the patterns are described. People understand better if patterns are illustrated with drawings, good typesetting, and writing style. However, the conversational style proposed by [188] is not very well accepted in academic writing. Thus, if the pattern writer intends to get a worldwide audience throughout publications, the writing style should be academic.

It is also important to stress in this sub-section that pattern writing is an iterative process like software development itself as pointed out in "Habit 6: Iterating Tirelessly" [188]. For this reason, a pattern has to be written and re-written many times until the writer reaches a stable point that other people can "read, understand, and comment on."

As soon as the pattern writer finishes these steps, the last habit ("Habit 7: Collecting and Incorporating Feedback") introduced by [188] should be followed. This habit is related to pattern reuse that should be encouraged by publishing the pattern even within the pattern author's group.

3.4.3.3 Pattern Language Structure

The last category in Figure 18 is the "pattern language structure" that contains solutions to recurring problems that happen when grouping patterns in a pattern language. These patterns refer to the content (subcategory B) and readability (subcategory E) of the pattern language. The former is well explained within the *pattern language summary*, the *common problems highlighted*, and the *running examples*. The latter is tackled in the *distinctive headings convey structure*, the *problem/solution summary*, and the *glossary*.

For instance, *common problems highlighted* refers to patterns within the pattern language that present alternative solutions to the same problem. When referring to the description of a pattern language, the addition of *running examples* is also a good practice to describe the pattern language relationships.

The topic "Habit 4: Patterns Distinct and Complementary" also refers to the pattern language structure. This habit is about how to make patterns distinct from each other as well as to ensure that the relationship among them is clear. For example, **[188]** presents the importance of describing which patterns are complementary. In other words, a pattern language should make vivid the comparison and the contrast between patterns. As discussed in the previous sub-section, sub-category D ("referencing to other patterns") is also useful for describing the relationships among the patterns.

Finally, uniformity during the description of patterns within a pattern language or a pattern catalogue is also another concern for the pattern writer. According to **[188]**, the template should be consistent throughout every pattern description and this helps the understanding of a pattern language or catalogue. Nevertheless, if the writer realizes that the *mandatory elements present* insufficient information about the pattern, one or more *optional elements when helpful* should be included. This is a good practice as long as the writer explains the template at the beginning and stresses which parts are not presented in every pattern (see an example in **Chapter 5**). It is not necessary that the remaining patterns include these new elements as **[188]** stresses.

3.5 Classification of Patterns

The software engineering literature splits the software development life cycle into the following activities (also called steps, levels or *stages* in this thesis) **[60][116]**: requirements capture, analysis, design, implementation, and testing. Each step produces a view of the system that is more detailed than the view of the previous stage.

The *requirements capture* step describes the system objectives as well as the user's needs and encourages the thinking process in terms of generic behavior. The *analysis* step comprises the static structure, the sequence of interactions that describe the problem to be solved in terms of entities (e.g., objects or functions), and the data transformations. The overall system structure is decided during the *design* step by expressing the problem solution in terms of how the entities interact. In the *implementation* step, the design is converted into executable code. Finally, *testing* assures that the appropriate requirements are satisfied (*validation*) and proves that the implementation is correct (*verification*). According to **[116]**, a good software life cycle guarantees that the design corresponds to the code and that the correspondence is correct and traceable.

Software patterns are classified into different categories depending on several factors including their application to the software development stages as follows: requirements patterns **[44]**, analysis patterns **[77]**, design patterns **[80]**, and idioms (implementation stage) [57]. As illustrated in Figure 19, we adopted this classification to explain the software patterns available in the literature. We believe that these categories can group the wide range of software patterns available for each application domain. This thesis describes requirements and analysis patterns for mobility and radio resource management functions in **Chapter 5**.

It is also important to organize patterns in *catalogues* on the basis of different application domains, pattern languages, or software development stages. This practice helps developers to direct efforts to learn and find new patterns. These catalogues aim to help designers to find patterns that are suitable for their specific requirements, analysis or design problems. The pattern almanac **[161]** brings a collection of published patterns organized in catalogues by application domains.

As stated earlier, software patterns can provide reusable and flexible solutions for common problems (see a discussion about pattern misconception in 3.7).

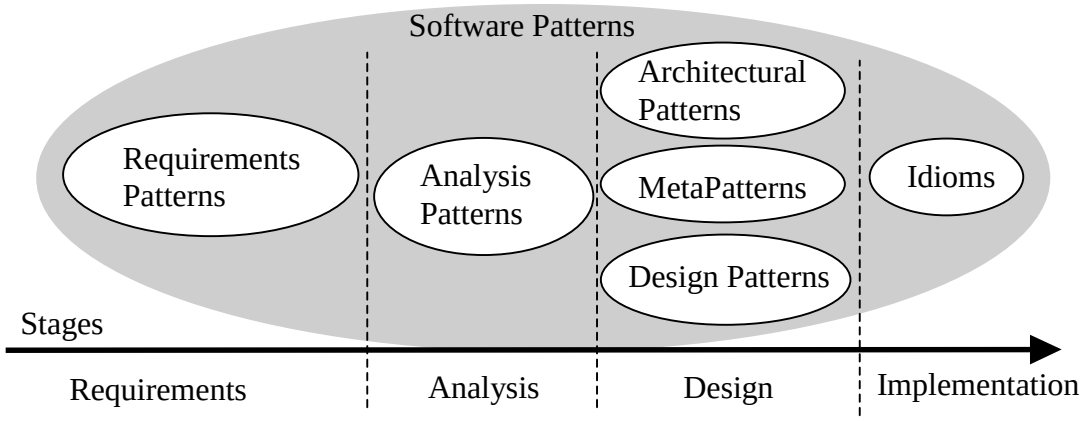


Figure 19 Classification of Software Patterns

The next subsections present more details about requirements patterns, analysis patterns, design patterns, which include architectural patterns and meta-patterns, and idioms. In addition, a discussion about anti-patterns is presented in Section 3.5.5. These patterns are documented with different templates.

3.5.1 Requirements Patterns

Requirements patterns document the user's needs and the generic system behavior at a high level of abstraction. The requirements patterns described in this research focus on the functional behavior of a system without considering its structure (see more details in **Chapter 4** and **Chapter 5**).

In order to classify the patterns currently available in the telecommunication domain (see Section 3.6), we also use the requirements pattern classification to describe generic actions that software developers can take to improve non-functional software requirements such as performance, testability, reusability, reliability, and maintainability. These actions are either related to the relationship between customers and the system or related to the relationship between operators and the system. In addition, these patterns can be related to an external event that affects either the system performance or the system analysis and design decisions. As presented in **[161]**, the "*fault-tolerant telecommunication system patterns*" **[3]** belongs to this classification (Section 3.6.1 presents more details) and "addresses reliability and human factors issues in telecommunications software."

More information about the requirements patterns is presented in Section 3.6 that gives an overview of patterns for telecommunications. We outline how the

telecommunication developers have been using the software pattern concept to describe their patterns and also how specific telecommunication companies have been reusing existing design patterns.

3.5.2 Analysis Patterns

In the late nineties, most of the software patterns presented in the literature focused on object-oriented design and implementation stages **[52][80]**. Nowadays, patterns also refer to the early development stages and other application domains than software engineering [161]. As shown in the last sub-section, software patterns are no longer tied to object oriented techniques.

Object-oriented analysis patterns were firstly presented in **[75]** to complement design patterns. These patterns contain the domain knowledge of business software such as accountancy for health care systems. The analysis pattern goal is to concentrate a step before design with an analysis model that addresses "conceptual structures of business processes rather than actual software implementations." The conceptual model as well as how "the choice of [this] model affects the flexibility and reusability of the resulting system" are the main concerns of these pattern developers. Some of these practical analysis patterns can be easily applied to the object-oriented models of other systems. For instance, the *party* and *subtype state machines* analysis patterns are also useful when modeling telecommunication systems to describe the users (e.g., called and caller) and the system component behaviors, respectively.

Although there is no apparent template associated with each analysis pattern in **[75]**, the author explains each pattern within a textual form where we can identify context, forces, problem, solution, and examples. The problem and the solution are presented with object-oriented modeling techniques such as type or interaction diagrams that vary according to the pattern. Table 2 presents an attempt at separating the textual form used by **[75]** into the pattern elements of a pattern template.

Template	**Short Description**
Context	An example that explains the situations where the problem can be found.
Problem	What are the problems that the pattern addresses?
Forces	A first attempt at modeling the problem shows the negative forces that can happen in the model.
Solution	The best solution that solve the forces.
Example	Real business systems that are known uses of the pattern.

Table 2 The Analysis Pattern Template according to **[75]**

3.5.3 Design Patterns and Idioms

Most of the patterns available in the literature fall into the analysis **[75]** stage, which is presented in the previous sub-section, and the design stages **[80]**. Meta-patterns **[153]** and architectural patterns **[52]** are also considered at the design stage as illustrated in Figure 19. On the other hand, idioms **[52]** are related to the implementation stage and they refer to the guidelines on how to implement design patterns in a given language.

Architectural patterns are related to the structural organization of software system applications that include detailed design of components and the collaboration and communication between them. These patterns are suitable for the beginning of the design stage.

Design patterns are defined as a "scheme" for the refinement of the software system components or the relationships among them. These patterns can be used during the whole design stage (see more details in Section 0). In **[80]**, design patterns are presented as "descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context." A class defines the object implementation with the object "internal data and representation" as well as "the operations that the object can perform." An object contains "both data and the procedures that operate on that data." These authors present a collection of twenty-three design patterns that aim at providing a "common vocabulary" for the software designers' discussion on their system development decisions.

These design patterns are classified into two main categories related to their purpose and scope. According to their purpose, they are divided into three types: creational patterns are related to "the process of object creation;" structural patterns are concerned with "the composition of classes and objects;" and behavioral patterns pertained to the interaction between classes and objects as well as to the distribution of responsibilities. Regarding their scope, they are classified into classes and objects as stated earlier. Class patterns describe "the relationships between classes and their subclasses," and object patterns present "object relationships." In our work, we also use the term behavioral and structural patterns to group our requirements and analysis patterns; however, we refer to functional behaviors and architectural elements instead of classes and objects.

The following four essential elements are used by **[80]** to introduce a design pattern: the *name* creates a vocabulary; the *problem* describes when to apply the pattern; the *solution* is responsible for the design, the pattern relationships, the responsibilities, and the collaborations; and the *consequences* include the results and the impact of the pattern on the system flexibility, extensibility, or portability. According to these authors, a consistent and complete format to describe patterns is the one shown in Table 3.

As presented in **[80]**, the use of design patterns has a lot of advantages as follows. First, the "common vocabulary" offers better communication, documentation, and investigation of system design alternatives. Second, each problem is described in a higher level of abstraction than the implementation that simplifies the understanding of the whole system. Furthermore, design patterns capture the experience of designer experts with the description of how to use and how to parameterize a system instead of just recording the final results. Therefore, they provide a better understanding of an existing system than the object-oriented design methods. Like any software patterns, design patterns can also be combined to build a pattern language.

Furthermore, the design pattern concept is used in **[81]** to introduce a methodology for designing distributed systems with a formal method called Specification and Description Language (SDL) **[109]**. This methodology presents SDL patterns to guide the development of an SDL specification. A successful experience report that specifies and validates a model for ATM networks at the design stage is also presented.

Template	**Short Description**
Pattern Name and Classification	The essence of the pattern that constitutes the vocabulary.
Intent	The pattern task, rationale, intention, and the problem.
Also Known As	Synonymous names if the pattern already exists.
Motivation	"A scenario" that helps the understanding of the abstract description.
Applicability	An example of pattern use.
Structure	"A graphical representation" of the classes using a notation described in a modeling language such as Object Modeling Technique (OMT).
Participants	List of the classes and objects and their responsibilities.
Collaborations	The collaboration among participants in order to perform their responsibilities.
Consequences	Advantages and disadvantages of applying the design pattern.
Implementation	The "pitfalls, hints, or techniques" in reusing the design pattern and programming language restrictions.
Sample Code	Parts of the code that illustrate the design pattern implementation in C++, Smalltalk, or Java, for example.
Known Uses	At least two examples of the design pattern in real systems.
Related Patterns	The list of other design patterns that describe similar problems and solutions.

Table 3 Consistent and Complete Design Pattern Template according to [80]

3.5.4 Metapatterns

The term *metapatterns* is introduced by [153] as "a set of design patterns that describes how to construct frameworks independent of a specific domain." In this context, *frameworks* are described as "ready-to-use and semi-finished building blocks." A framework is considered well designed when the composition and the interaction of these building blocks that constitute the overall system architecture are also defined (see more details about frameworks in Section 3.2). In order to adapt application frameworks, which refer to a specific domain, and to specific design needs, [153] defines certain parts called "*hot spots*" that have to be maintained flexible. On the contrary, parts of a framework that are not designed for adaptation are called "*frozen spots*."

At the design and implementation stages, metapatterns provide the main object-oriented constructs for developing a well-designed framework (hot spots and frozen spots included). These metapatterns are complementary to the object-oriented concepts that allow the implementation of frameworks with hot spots as well as to the software patterns that outline how to design and to implement these hot spots. The author categorizes the structural principles of design patterns for the framework development into seven metapatterns as follows:

Class/object interfaces and Interaction metapatterns

- Template and hook methods;
- Narrow inheritance interface principle;

Class/object composition metapatterns or Resulting composition metapatterns

- Unification;
- 1:1 Connection;
- 1:N Connection;
- 1:1 Recursive Connection and 1:1 Recursive Unification;
- 1:N Recursive Connection and 1:N Recursive Unification

During the development of application frameworks, template methods are used to implement frozen spots and hook methods are applied to the implementation of hot spots. These methods are metapatterns that allow flexibility in the framework. As an example, in the telecommunication domain, a method for printing the customer invoice can be a template method that calls both a hook method to calculate the rate and a hook method to get the user's name.

Metapatterns not only capture the design of an existing and mature application framework but also guarantee that any domain can develop frameworks based on

previous experiences. At the design stage, metapatterns capture an application framework design with the following steps that are mentioned in **[153]**: first, the identification of the hot spots using domain-specific knowledge; second, the development of the actual framework using a new approach called the hot-spot-driven approach.

The hot-spot-driven approach is applied to **[81][199]**. This approach is an enhancement of the design pattern catalogues and the object-oriented analysis and design (OOAD) methodologies for the development of frameworks. As mentioned earlier, in this approach, metapatterns are used to adapt existing frameworks as well as to develop new ones. Furthermore, it solves the limitation of the conventional software development such as the well-known software life-cycle models **[117]**) and the state-of-the-art OOAD methodologies (e.g., the Object Model technique (OMT) **[164]** and the Unified Modeling Language (UML) **[65]**) to deal with the intertwined design and implementation of the framework development process. First, domain experts should recognize which aspects differ from application to application and what is the desired degree of flexibility. Second, they should identify whether the flexible behavior should change at run time or not.

Although the hot-spot-driven approach has the advantages discussed previously, it has limited coverage of the development stages due to its focus on the detailed design and implementation stages. As a result, it is more implementation-driven than design-driven and it has no concern about the requirements and analysis stages.

A successful experience report on the reuse of these metapatterns is presented in **[199]**. The authors apply SDL **[109]** to model a system (i.e., specification and validation of a prototype) using metapatterns. They provide SDL constructs that are suitable to support the metapattern concepts (i.e., hot spots and frozen spots).

3.5.5 Discussion about Anti-patterns

Like the software patterns mentioned earlier, antipatterns also describe recurring solutions to common design problems; however, their use provokes consequences that are considered common mistakes during the software development [44]. An antipattern also provides ways to overcome its negative consequences with a solution that is refactored from the problem. For instance, the *requirements jeopardy* antipattern presented in [44] refers to the failure to identify properly requirements in system engineering and shows the symptoms and consequences of this failure. The refactored solution is the application of requirements analysis and configuration management processes.

The template, which is used to describe antipatterns in [44], is presented in Table 4.

Antipatterns are classified by **[44]** into requirements, testing, management, and process patterns depending on their use in the software lifecycle. The use of the term antipattern is controversial among the pattern community who consider antipatterns as re-factoring patterns. This is the reason we omit them from the classification of patterns in Figure 19.

Template	Short Description
Name	A noun phrase with a pejorative meaning.
Also Known As	Additional names that can identify the antipattern.
Most Frequent Scale	The antipattern place into the software design level model (e.g., application, enterprise).
Refactored Solution Name	"the refactored solution pattern."
Refactored Solution Type	"the type of action that results from the antipattern solution according to the software design level model."
Root Causes	General keywords that cause the antipattern.
Unbalanced Forces	"… forces that are ignored, misused, or overused in [the] antipattern."
Anecdotal Evidence	Some interesting event that is associated with the antipattern.
Background	More comments or examples.
General Form	General features.
Symptoms and Consequences	Resulting context caused by the antipattern.
Typical Causes	The list of causes.
Known Exceptions	Exceptional cases in which the antipattern does not occur.
Refactored Solution	Solution that neutralizes the antipattern.
Variations	Alternative refactored solutions.
Example	Applications of the solution.
Related Solutions	References or cross-references to other antipatterns, design patterns, or pattern languages.
Applicability to other viewpoints and scales	Interaction to other viewpoints and relevancy to the other levels.
References and Resources	References and similarities to other patterns.

Table 4 AntiPattern Template according to **[44]**

3.6 Patterns for the Telecommunication Domain

According to the literature **[3][63][90][134]**, telecommunication designers have been documenting their own patterns that are grouped into the telecommunications category in the pattern almanac **[161]**. However, patterns for telecommunications are not published as often as the patterns for other domains. The reason for this is the competitive advantages that concern the telecommunication industry. In order to keep the software implementation private, they concentrate on the early stages of the development process. Under such circumstances, most patterns for telecommunications belong to the category of requirements patterns (see Figure 19 in Section 3.5).

The almanac presented in **[161]** has collected a set of thirty-nine patterns for the telecommunication domain. Some of them are organized into pattern catalogues such as the "Fault-tolerant telecommunication system patterns" **[3]** or into pattern languages such as "A generative Pattern Language for Distributed Processing" **[63]**, "An input and output pattern language" **[90]** and "A pattern language for improving the capacity of reactive systems" **[134]**. Some patterns belong to more than one domain such as the "Fault-tolerant telecommunication system patterns" that address telecommunications and fault-tolerant systems.

This section summarizes a catalogue of telecommunication patterns ("Fault-Tolerant Telecommunication System Patterns" **[3]**) that presents patterns for non-functional requirements and analysis issues as discussed earlier. These patterns are grouped into different sub-categories within the requirements patterns and classified as follows: reliability and human (users' and operators') factors. Since different companies have been using design patterns in their telecommunication systems, an experience report entitled "Industrial Experience with Patterns" **[31]** is also outlined in this section.

In general, the patterns for telecommunications presented in **[3]** are described with the template shown in Table 5. This template is a short version of the one presented in **[80]**; however, a different order is used in these telecommunication patterns to present the pattern elements. A comparison with Table 3 is also depicted in Table 5.

Template	Relation to the Table 3
Name	Pattern Name
Context	Motivation
Problem	Intent
Solution	Applicability
Forces	Collaborations
Rationale	Intent
Resulting Context	Consequences

Table 5 A Pattern Template used in the Telecommunication Domain

3.6.1 Patterns for Fault-Tolerant Telecommunication Systems

AT&T has a catalogue for fault-tolerant telecommunication systems. These patterns form a pattern language that focuses on reliability and human factor issues in the AT&T Electronic Switching Systems (ESS) such as 4ESS and 5ESS. Eight of these patterns are presented in **[3]** with the template illustrated in Table 5.

The fault-tolerant telecommunication patterns **[3]**, which are summarized in Table 6, refer to switching systems that have specific constraints and terminology. The following constraints should be considered before trying to understand these patterns: switching systems are designed to be out of service no more than two hours in forty years; human maintenance and personnel requirements are optimized. Since these systems also require automation, they provide remote computers to monitor and control the switching

equipment. Furthermore, the following telecommunication terminology is necessary: PC is the processor configuration that deals with the common hardware and software platform; phase represents a level of system recovery escalation; and transient fault is a condition that is transitory in nature.

Although these patterns aim at the maintainability of switching systems, they are appropriate measures to be taken at the requirements development stage. These patterns are grouped into the following sub-categories: patterns related to reliability (system messages and system faults) and patterns related to human factors.

Name	Problem	Solution
Five Minutes of No Escalation Messages	How do you minimize the rolling in console messages?	Display a message when taking the first action down related to a scenario that could lead to an excessive number of messages. Then, periodically display an update message. If the abnormal condition ends, display a message that everything is back to normal.
Fool Me Once	(1) How do you avoid a loop that involves a Processor Configuration (PC) when consecutive faults occur? (2) How do you reduce user concerns regarding system faults?	Believe the PC application when it tells you "all is well" the first time and reset the configuration counter. Ignore the request during the second and subsequent times within a longer time window. Use *Five Minutes of Escalation Messages* to minimize the message displays.

Table 6 Patterns related to Reliability and Message Displays

Table 6 summarizes two patterns related to reliability in the switching systems. These patterns are also related to message displays on the users' interface. This summary improves the description of the original patterns. For instance, the problem described in the *fool me once* pattern is split in two different sub-problems. The first sub-problem is the original *fool me once* solution and the second sub-problem is solved by the *five minutes of no escalation messages*.

3.6.2 Experience Report on the Reuse of Design Patterns

As discussed earlier, the telecommunication community has been describing software patterns for this domain as well as reusing existing design patterns. As an example, **[31]** presents the experiences with design patterns in the AT&T, Motorola, BNR, and IBM companies. The authors' main goal is to discuss how the industry has been using patterns and what benefits they bring in practice. This experience report emphasizes how a catalogue of design patterns is useful for both the experienced and the novice designer by recognizing situations where design reuse can be applied. In addition, they stress that the most important lesson to be learned with the pattern community is the vision that diverse experienced software developers can exchange information about design issues using the

pattern concept. However, these authors conclude that the act of designing pattern demands experience and it is as hard as the development system process.

The authors' conclusions about the reuse of design patterns are summarized as follows. First, these patterns create a common vocabulary with *names* that convey a "precise and concise" communication medium to discuss design problems. Second, these patterns were extracted from existing and working designs that make them reliable. Since a pattern addresses the best design practices, it helps less-experienced developers to improve their design skills.

On the other hand, this experience report also mentions that a software pattern does not necessarily capture object-oriented design. The diversity of patterns presented in the almanac **[161]** has confirmed this statement. The authors also point out that patterns demand time to write and their reuse requires a careful learning process that can be improved with the use of pattern mentors in an organization.

This thesis shows how to capture, to document, to reuse, and to validate requirements and analysis patterns in the mobile wireless communication domain. The pattern characteristics discussed earlier are also observed when writing the patterns introduced in **Chapter 5** and when reusing them in the case studies presented in **Chapter 7**. Furthermore, the experience in capturing patterns from this domain involves the same recurring problems presented in **[31]**, which are: the lack of explanation on what system developers had done and why the design was the way it was. In our case, the solution for both problems is to reverse engineer the design choices from the system documentation and the literature as discussed in Chapter 4.

3.7 Discussion: Clarifying the Pattern Concept

Software patterns are evolving constantly. This is a living and a fast-moving area for software developers. Therefore, some misconceptions have been propagated in the pattern community. In order to organize and clarify them, **[189]** presents a set of ten pattern misconceptions that are summarized in this section to emphasize the correct ideas behind the pattern concept. Besides these misconceptions, we introduce some misunderstandings that were perceived when presenting the results of this research (see **Chapter 5**) to diverse audiences, which were not familiar with the pattern concept.

The first misconception presented in **[189]** is about the pattern concept. According to the author, a pattern is not only "a solution to a problem in a context" as defined in **[6]**, but also needs "recurrence, teaching, and naming." On the basis of our experience on writing patterns, we agree on the importance of describing patterns with all the mandatory elements to clarify this misconception (see Section 3.2). For example, the *known uses* element expresses the pattern recurrence within the target domain and the pattern *name* helps the audience to refer to the pattern. Teaching can be achieved by publishing the pattern (see Section 3.4.3).

The second misconception is also related to the pattern concept, which is sometimes reduced to the following definition: "Patterns are just jargons, rules, programming tricks, [or] data structures …" **[189]**. The author contradicts this definition by stressing that a good software pattern should be close to its readers. This is the reason for using terms that are known by the computer science community (see also Section 3.4.3.2). In addition, the pattern template and the teaching component should be able to counteract this definition.

Furthermore, there is a set of misconceptions in **[189]** that have been discussed earlier in Sections 3.4 and 3.6. For example, the misconception that "all patterns are created equal," which implies that a single pattern template should be used in all software patterns. On the contrary, the author explains that they are not similar throughout the literature but as different as their authors, domains, styles, and scope. In addition, the misconception that patterns needs a graphical description is pointed out. Although patterns with graphical descriptions help the readers, they "do not need tool or methodological support to be effective." However, as we stress in this research, the description of pattern solution scenarios with software techniques helps the pattern reuse. This research presents patterns with a graphical description with UCMs to help the readers to visualize the pattern solution (see **Chapter 5**).

The author refutes the following two misconceptions "patterns guarantee reusable software, [and] higher productivity …" and "patterns 'generate' whole architectures." Rather, the real goal of patterns is to improve the development system approach, not to *guarantee* solutions for everything or to *generate* a system architecture. In short, the pattern strengths are to teach their forces and resolution through discussion of consequences.

There is another misunderstanding about patterns being suitable only for object-oriented design and implementation. Nevertheless, patterns are also useful for analysis, maintenance, testing, documentation, and organizational structure as shown in the previous sections. As mentioned earlier, several patterns are emerging in different domains such as telecommunications, distributed systems, and business **[161]**.

Despite the fact that the pattern concept has been applied to distinct fields, there are not enough experience reports that patterns can help everybody. This clarifies another misconception about the range of pattern usefulness. Section 3.6 gives more details about these last two misconceptions.

The last two misconceptions talk about the pattern community that is neither a "clique of elites" nor "self-serving [and] conspiratorial." On the contrary, the community is open and heterogeneous. For instance, the pattern community is receptive to newcomers and includes a variety of people (e.g., analysts, designers, implementers, students, and professors). As an example, the patterns for mobility management presented in **Chapter 5** have been reviewed by members of the pattern community and published in **[19]**. Their comments have considerably improved these patterns.

In short, these misconceptions are refuted with the main advantages of pattern use that are mentioned throughout this chapter. They are the capture of expertise and its propagation to non-experts, the common vocabulary that improves the communication between developers, the better understanding of a system, and the facility of restructuring a system. We believe that the results of this research that presents requirements and analysis patterns for the mobile wireless communication domain clarifies some of these misconceptions. For instance, we present patterns that are suitable for the early development stages and they are not object-oriented patterns. We also propose an approach to improve reusability in the development and evolution of mobile systems with pattern reuse and validation as well as to help the generation of scenarios from the pattern language introduced in **Chapter 5**.

3.8 Conclusion

This chapter shows the advantages of applying the pattern concept in different application domains such as telecommunications and distributed computing. The level of abstraction at which these patterns are described allows designers of different systems to reuse them. Attempts to describe relationships among this diversity of published patterns can point out common characteristics among different domains that can help the system development process and evolution. As an open area of research, the reuse of existing design patterns within the pattern language presented in this thesis can be investigated (see also Section Figure 52 in Chapter 5).

Another important result of software patterns is the vocabulary that comes from the effort of describing recurring problems and solutions in a specific domain. As pointed out in **[6]**, connections among patterns can generate a rich language that puts together many patterns with different meanings. In this context, there is a need for gathering the telecommunication patterns, which are mentioned in Section 3.6, and make them available in the literature to improve their reuse by the telecommunication community. As future work, these patterns can be combined with other patterns included in the telecommunication category of **[160]**. This combination can generate a pattern language to further assist in the development of new telecommunication systems.

The considerations about software patterns presented in this chapter lead to the motivation for thinking about a pattern language for mobile wireless communication systems. The main motivation of this research is the need for capturing common behavior and architectural elements from different mobile systems (see **Chapter 2**) and documenting them as patterns (see Chapter 4 and **Chapter 5**, respectively). These patterns can help these systems to interact from the early stages of the system development process and throughout its evolution. This motivation arises from the diversity of mobile systems using different wireless technologies such as TDMA and CDMA, which allow mobile users to roam, and from the diversity of agreement among companies that are involved on making a variety of services available to the users. Thus, our main goal is to represent mobility and radio resource management functions such as authentication, location registration, and handoff as patterns. These patterns are gathered

in a pattern language that shows how the integration and the convergence of mobile wireless networks can be achieved at a high level of abstraction.

The patterns for mobility and radio resource management functions are general and abstract enough to allow telecommunication designers and implementors to reuse them. These patterns describe commonalities as mentioned above and offer the same foundation to designers of new or existing mobile systems. Since the focus is on the requirements and analysis stages, designers can adapt and make changes according to their own needs.

Chapter 4 Capture of Common Functional Behaviors and Architectural Elements

This chapter focuses on the capture of common functional behaviors and architectural elements that are related to mobility and radio resource management functions. First, the chosen mobile systems introduced in Chapter 2 are uniformly described with a visual notation called **U**se **C**ase **M**ap**s** (UCMs). After this, commonalities, which are essential for the generation of the patterns and the pattern language presented in Chapter 5, are identified and extracted from the UCM system descritions.

4.1 Introduction

This research applies the software pattern concepts outlined in Chapter 3 to the domain of mobile wireless communications. Specifically, common functional behavior and architectural elements are extracted in this chapter and documented as requirement and analysis patterns in Chapter 5 while investigating design problems and solutions of second and third generation systems.

As mentioned in Chapter 2, the following mobile systems are chosen to be investigated: ANSI-41/WIN, GSM/GPRS, IMT-2000/UMTS systems, and WmATM Networks. ANSI-41 and GSM have been chosen due to their prevalence as second generation systems in many parts of the world [35]. Furthermore, they were developed in North America and Europe, respectively, which makes them representatives of different markets. Within second generation systems, it is also important to consider WIN, which brings new advanced features to ANSI-41, and GPRS, which adds data to GSM.

Furthermore, UMTS and IMT-2000 systems are currently the most prominent initiatives of third generation systems, which aim to integrate diverse second generation systems and to incorporate Internet features. Finally, WmATM networks represent mobile systems with the possibility of exchanging voice and data over high-speed **L**ocal **A**rea **N**etworks (LANs). WmATM networks, UMTS, and IMT-2000 systems are under development.

This chapter concentrates on the uniform description of these systems with **U**se **C**ase **M**aps (UCMs) [51][48] and the capture of common functional behaviors and architectural elements among them. Figure 20 depicts these systems with the functionalities and the protocol layers that are the focus of this research. As mentioned in Chapter 2, the behaviors of mobile systems are often grouped in three different functional layers [139], as follows: mobility, communication, and radio resource management. At a high level of

abstraction, this research identifies commonalities among these systems on the basis of mobility and radio resource management functions that are related to application protocols. Network entities constitute the architectural elements that are analyzed to generate a common network reference model. The layers that are highlighted in gray are out of our scope.

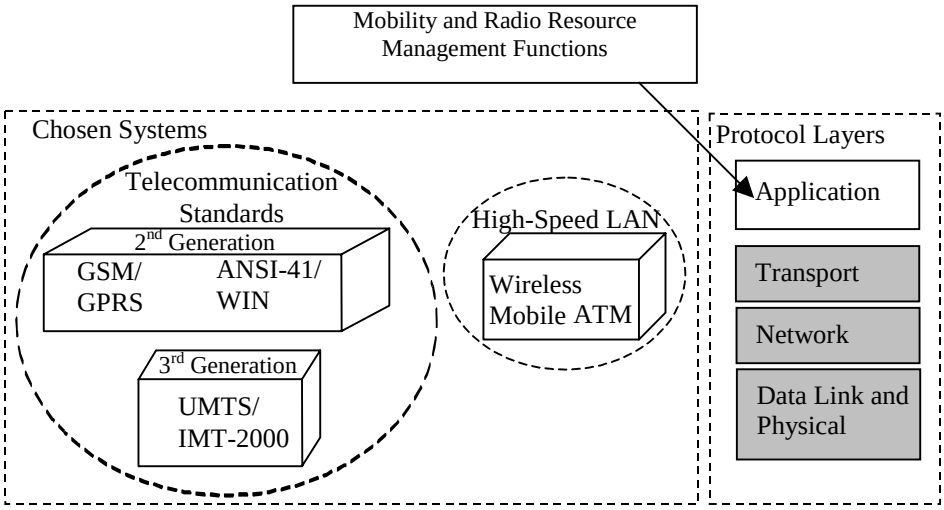


Figure 20 Chosen Systems with Target Functions and Layers

The literature related to software engineering and mobile systems presents several ways to deal with the software development life cycle. As a result, different definitions for the same terminology are introduced in these domains.

For instance, the ITU-T three-stage *methodology* is presented as a combination of text, tables, MSCs, and pseudo-code distributed in different development stages (see details in the next section). On the other hand, a system development *methodology* is defined in [167] as "a collection of the following three elements: a modeling language, modeling heuristics, and work organization." In addition, a *methodology* combines a functional model, a dynamic model, and an object model in [164].

Another example is the term framework that is used differently from the definition presented in the last chapter when describing the software development life cycle. A development *framework* is composed of operational and managerial issues such as concepts and rules involved in different aspects of software development. In this context, a framework is also a synonym for the work organization mentioned in the system development methodology definition.

In order to incorporate these different meanings that are applied to describe the software development, the term *approach* is used in this thesis in a broad sense as a way of dealing with development and evolution of large systems.

The next section presents the development approaches currently applied to the chosen systems. Section 4.3 and Section 4.4 give a quick overview of definitions and of the UCM notation that are used to describe these systems and to capture their commonalities, respectively. The reverse and the forward engineering approaches, which are used to extract information from each system in order to generate its description with UCMs, are presented in Section 4.5. Section 4.6 presents the approach for the capture of common functional behaviors and architectural elements. Section 4.7 describes the common functional behaviors with unbound UCMs. The description of the network reference model that is derived from the common network entities is the focus of Section 4.8. Finally, Section 4.9 discusses related work with respect to the identification of commonalities and variabilities in software engineering and Section 4.10 highlights the main contributions of this chapter.

4.2 Development Approaches used in the Chosen Systems

The development process of systems and services in the telecommunication standards often comprises different *phases* that bring additional features. Each phase is split into three major *stages* as depicted in Figure 21. Services are first described from the user's point of view in prose form and with tables (stage 1). After this, they are expressed with information flows, which are also known in the literature as sequence diagrams [116] or **M**essage **S**equence **C**harts (MSCs) [106] (see more details about MSCs in Section 6.3.2 of Chapter 6). These information flows represent the sequences of messages between the different architectural elements involved in the communication (stage 2). Finally, services are expressed with (informal) specifications of protocols and procedures (stage 3).

This three-stage methodology was first developed by ITU-T to describe services and protocols for ISDN [70]. Subsequently, it has become of general use in the telecommunication area. In short, a mixture of text, tables, and information flows constitutes the telecommunication standard documents.

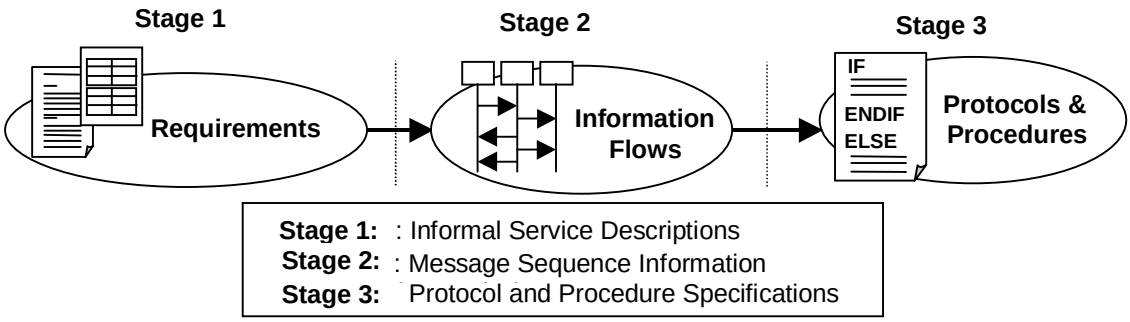


Figure 21 Three-Stage Methodology

Like many mobile communication standards, WIN has used the three-stage methodology shown in Figure 21 in each phase of its service development. For instance, the WIN standard represents services with informal descriptions and tables in stage 1. In stage 2,

the WIN distributed functional model and the network reference model are presented as well as message sequence charts, which illustrate the main service scenarios. The MSC components (the vertical lines) are network entities instead of functional entities.

As mentioned in [12], which describes the WIN standard with UCMs, there is a gap between stage 1 and stage 2 system descriptions of the ITU-T methodology. The WIN standard documents present the informal requirements (stage 1) followed directly by a level of detail that includes exchange of messages and parameters between network entities (stage 2). Neither functional entities (FEs) nor network entities (NEs) that should be involved in each service are mentioned in stage 1. Every decision taken along the way (from stage 1 to stage 2) remains in the heads of the designers who have edited the documents and this situation obviously does not promote the reuse and the openness of the standards. By focusing on the NEs in stage 2, the WIN standards are almost imposing a specific network reference model to the developers.

In addition to the ITU-T methodology, the creation process of telecommunication standards comprises several steps before their approval for publication. Without getting into detail about the whole process, it is important to mention a step called *verification & validation* (V&V). According to [79], each baseline text is analyzed manually, line by line, for correctness and accuracy. The specification can only be submitted after the V&V process is complete. The standards claim to achieve correctness and accuracy with a meticulous reading process. However, we believe that formal validation and verification methods, which have been used in the literature to describe large and complex systems, are the only means to help designers to prove the specification is in conformance with the requirements and free of deadlocks and redundancies [96].

In case of wireless mobile ATM networks, signaling protocol alternatives have been presented in the literature as discussed in Section 2.9 of Chapter 2. Figure 22 shows a summary of the main development approaches currently applied to WmATM network development. They involve *informal descriptions* (e.g., text) at the early stages (for example, ITU-T stage 1) followed by *information flows* [4][5][158], *flow charts* [29][187], or *state models* [54] at the later stages. These later stages correspond to ITU-T stage 2 (information flows) and stage 3 (flow charts and state models).

The previously mentioned notations, which are currently applied to telecommunication standards and WmATM networks, are not adequate to this research that aims to capture commonalities among these systems in order to generate patterns. The description of architecture and protocols of such large systems involve an increasing level of complexity that cannot be expressed properly with informal and semi-formal models. For instance, when signaling protocol requirements are described only with text, they contain redundancies and become cumbersome to read, understand, and manage. Information flows and flow charts are an attempt to solve these problems. However, they are disjoint scenarios with details about messages, parameters, data, and system components that are difficult to manage and analyze when the goal is to capture commonalities at the early development stages. Although state models enable the connectivity between scenarios, they also contain many design details since they demand

full precision for the definition of the underlying architecture and of each state. Information flows, flow charts, and state models are only necessary where detailed designs need to be taken (i.e., at the design stage).

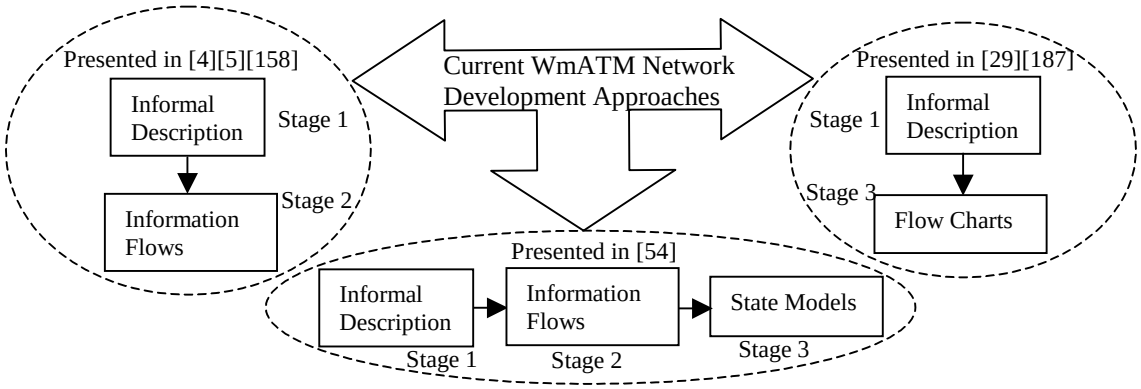


Figure 22 Different Development Approaches for WmATM Networks

A notation called **U**se **C**ase **M**aps (UCMs) appears to overcome, at least in part, the inadequacy of the notations mentioned above to describe the chosen systems uniformly and to capture common behaviors and architectural elements among them. The UCM notation is suitable for this work because of its simplicity, its modularity and its flexibility characteristics. In other words, UCMs are easy to learn and to understand; they help the decomposition of large systems like the mobile systems into small units; finally, they are easy to map the system architecture to the functional behaviour that may be described independently.

We also choose UCMs for their ability to express requirements and analysis models in such a way that the developer can have a bird-eye view of the whole mobile system behavior and structure. This helps to identify the design problems to be solved starting from the beginning. Our purpose is to achieve better and more complete descriptions, and to improve both the human understanding and the technical quality of the telecommunication standards.

UCMs are also used in this work as the technique to represent potential reusable procedures. Possible scenarios for the pattern solutions are graphically specified with bound UCMs by showing general control flows between responsibilities in Chapter 5. When reusing the patterns at the early stages of system development process and evolution, this notation also allows expressing the scenarios regarding the pattern relationships as presented in Chapter 7.

Section 4.3 includes definitions and Section 4.4 presents examples, which refer to the UCM concepts and notation, that are used in the description of the mobile systems discussed in Chapter 2. More details about the current approaches for the development of large systems and their respective notations are also discussed in Chapter 6.

4.3 Definitions

As mentioned earlier, *mobile systems* are composed of a set of distributed architectural elements with different functionalities. In this thesis, a mobile system is described as a set of functional behaviors at the *requirements capture* stage. At the *analysis* stage, these behaviors are mapped to their respective architectural elements (also known as structural elements). At the *design* stage, details related to messages, operations, algorithms, and parameters are included in the system description.

The following definitions help the reader understand the UCM notation presented in this chapter. In addition, they are also used in the approaches for the capture of requirements, analysis, and design information from each chosen system and for the capture of commonalities among them.

- A *condition* is a logical predicate that must be satisfied in order to start the execution of a functional behavior or to enable the execution of a responsibility. It is also satisfied after the execution of a responsibility or a functional behavior. There are two types of conditions as follows. When a condition precedes the functional behavior or the responsibility, it is called *pre-condition* (P). When the condition is a result of the execution of a functional behavior or a responsibility, it is called *post-condition* (Q).

- A *responsibility* (r) describes an action (e.g., operations on data items) or an event that occurs in a functional behavior. A *pre-condition* (cause or context) must be satisfied before enabling each responsibility and a *post-condition* (consequence or resulting context) must be achieved after performing each responsibility.

- A *functional behavior* (FB) of a mobile system describes start points, responsibilities, and end points associated with pre-conditions and post-conditions that are related to mobility, communication, or radio resource management functions.

- A *triggering event* (TE) is an event associated with a *pre-condition* that should be satisfied before a *functional behavior* starts its execution [34]. For example, when a user presses the button to power on a mobile station (i.e., the power on event occurs), a condition such as whether the mobile station responds to the power on event or not is analyzed. If this condition is true (e.g., the mobile station responds), the functional behavior is executed.

- A *resulting event* (RE) is an event associated with a *post-condition* that should be satisfied when a *functional behavior* ends. For example, after a successful power on event, different actions and events can be performed. A post-condition is set true (e.g., call establishment is enabled) and a *resulting event* can be generated (e.g., a confirmation message is sent) when the functional behavior finishes.

4.4 Use Case Map Notation

A *Use Case Map* (UCM) is a visual notation that helps to express the system behavior in terms of causal relationships between responsibilities and the system structure in terms of paths cutting across components [48][51]. These maps describe the system functionalities at the early stages of the development process. UCMs can graphically describe *use cases*, which have been defined [51][116] as structured prose descriptions of a set of *scenarios* that shows the sequence of steps involved in the interaction between a system to be designed and the system users. These scenarios are joined in a use case by a common user goal.

Figure 23 depicts different UCMs. UCM is called a *root map* when it represents the highest level of a system. The root map illustrated in Figure 23a starts with the filled circle labeled StartPoint and ends with one or more bars labeled $EndPoint_1$ and $EndPoint_2$. UCM is called a *plug-in* when it describes a sub-map that is associated with a *stub*.

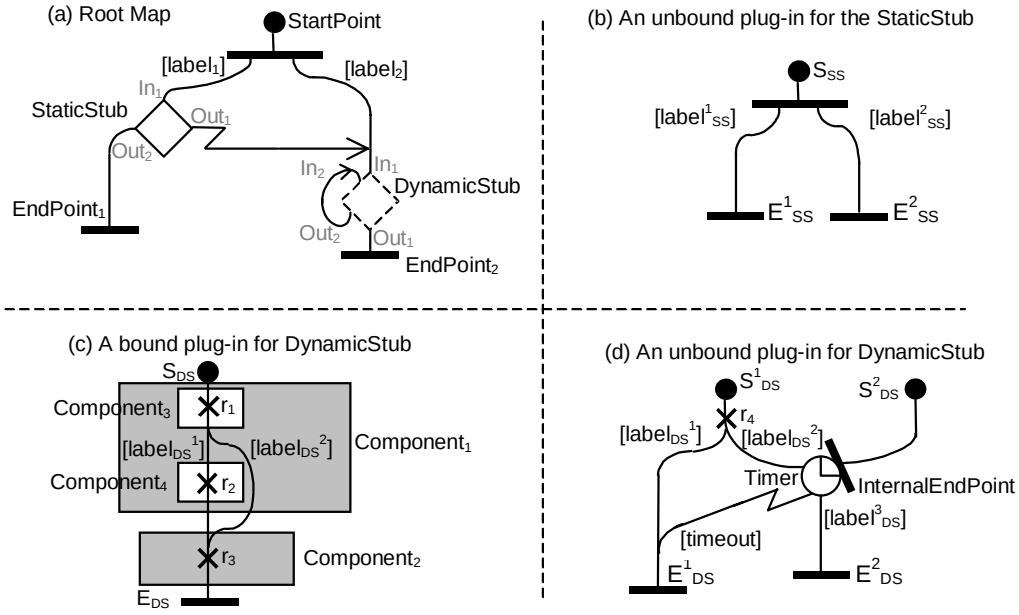


Figure 23 A Root Map and Plug-ins for Static and Dynamic Stubs

A *stub*, such as StaticStub and DynamicStub in Figure 23a, identifies places where details are delayed to sub-maps (the *plug-ins*). The *stub* notation not only hides details, but also decomposes the system into small manageable units.

A *static stub* is represented by a plain diamond and is bound to only one plug-in. Each outgoing path in a static stub should be bound to the respective end point of the plug-in (e.g., Out_1 and Out_2 in Figure 23a are bound to E^1_{SS} and E^2_{SS} in Figure 23b).

On the other hand, a *dynamic stub* is represented by a dashed diamond and is bound to one or more *plug-ins* (as illustrated in Figure 23c and Figure 23d, respectively) according

to a selection policy described in the stub pre-conditions. In case of a dynamic stub, outgoing paths are bound according to the binding between the stub post-conditions and the end points of the selected plug-ins (e.g., Out_2 in Figure 23a is either bound to E_{DS} in Figure 23c or to E^2_{DS} in Figure 23d).

In Figure 23c, *responsibilities* are represented by crosses (labeled r_1, r_2, and r_3) and are enabled along the path to represent the system behavior. These responsibility points can identify events or actions to be performed.

When UCMs represent scenarios as causal paths cutting across structures of components, they are called *bound maps.* The *component* notation describes runtime entities of systems such as objects, functional entities, network entities, processes, or groups of these entities. A component that represents a generic container is called a *team*. $Component_1$, $Component_2$, $Component_3$, and $Component_4$ in Figure 23c are examples of teams. On the contrary, when UCMs combine paths and responsibilities without components, they are called unbound maps (seeFigure 23a, Figure 23b, and Figure 23d).

Direction arrows help designers visualize the UCM flow as in the DynamicStub where an outgoing path (Out_2 label) returns to the same stub (In_2 label). The In_2 ongoing path leads to the S_{DS}^2 start point depicted in the 2nd *plug-in* (Figure 23c).

Figure 24 depicts a detailed unbound UCM to describe a *functional behavior* (FB) that is part of a system A. The representation of the start point (S_A), responsibilities (r_0^A, r_1^A , …, r_n^A), and the end point (E_A) contains *pre-conditions* (starting with P), *triggering events* (starting with TE), *post-conditions* (starting with Q), and *resulting events* (starting with RE) as introduced in Section 4.3.

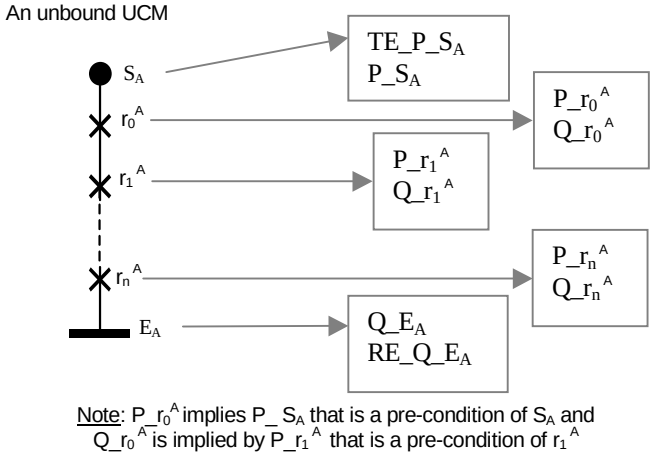


Figure 24 Detailed UCM Representation of a Functional Behavior

The following relations are associated with the definitions presented in the last section and the UCM notation discussed earlier. These relations are introduced in this thesis to give more details about the UCM notation and to show how a system description with UCMs works:

- An *enabling relation* describes either the relation between a *triggering event* and a *pre-condition* defined in a start point or the relation between a pre-condition and its *responsibility* (see Figure 24). A start point enables a path when its triggering event, which satisfies its pre-condition, occurs. A responsibility *r* in a path is activated once its pre-condition is satisfied.

- In a sequence of two or more *responsibilities*, a *causal relation* between responsibilities indicates that the first responsibility causes the next responsibility, which causes its successor if any, and so on. A *post-condition* of a responsibility corresponds to a *pre-condition* of the next responsibility in a sequence of responsibilities (see note in Figure 24).

- A *resulting relation* describes either the relation between a *post-condition* and a *resulting event* defined in an end point or the relation between a *responsibility* and its *post-condition* (see Figure 24). When a path ends, a post-condition associated with an end point of the path is satisfied and its resulting event occurs.

- A *choice relation* denotes an alternative between two or more or-exclusive paths in any single system run. There is no concurrency associated with the choice relation. *OR-forks* represent composite UCMs that can be split into two different paths such as [label$^1_{DS}$] and [label$^2_{DS}$] in Figure 23c and Figure 23d. Figure 23c depicts two path alternatives: the causal sequence of r_1, r_2 and r_3 responsibilities or r_1 and r_3 responsibilities.

- A *join relation* denotes a merge between two or more or-exclusive paths in any single system run. There is no concurrency associated with the join relation. *OR-joins* represent composite UCMs that can be joined to a single path (such as [label$^1_{DS}$] joining [label$^2_{DS}$] and [timeout] joining [label$^1_{DS}$] in Figure 23c).

- A *parallel relation* indicates concurrency between two or more paths in any single system run. When these paths are enabled, they can occur simultaneously. *And-forks* represent composite UCMs that split a path into parts (sub-paths) that proceed concurrently (e.g., [label$_1$] and [label$_2$] sub-paths in Figure 23a).

- A *disabling relation* describes an interaction between an abort or exception path (the so-called watchdog path) and one or more paths, which are disabled once a certain action or event represented by the abort path occurs. The zig-zag UCM notation represents the exception or the abort path that can terminate the execution of another path.

A timeout is a special case of the zig-zag notation as depicted in Figure 23c ([timeout] path). Timers are special waiting places triggered by the arrival of a specific event. In cases where the event does not arrive in time, the timeout path is triggered. The timer notation can be also explained using the *choice relation*. For instance, either the timeout path or the [label$^3_{DS}$] path is triggered in Figure 23d.

This research uses the disabling relation to describe an interaction between an external event, which triggers a path, and one or more paths. The UCM zig-zag notation is used from the path that discovered a failure (e.g., a database failure) or from the path triggered by an external event (e.g., power-off) to the path to be aborted, which comes after an arrow direction, as shown in Figure 23a (Out_1 path). The zig-zag notation in this particular case describes an interaction between a stub, which has a plug-in that is bound to it (see Figure 23b), and another path (as shown in the figure) or between two stubs, which have plug-ins that are bound to them (not shown in the figure).

The plug-in that describes this specific use of the disabling relation is represented as in Figure 23b. For instance, in Figure 23a, the stub has two exits to represent the abort path and the end of the actions that are taken to recover the system (e.g., Out_1 and Out_2 paths in the figure). The StaticStub and the [$label_2$] path interact when a certain pre-condition is satisfied and the start point of the plug-in, which is statically bound to the StaticStub, is triggered as depicted in Figure 23b. This plug-in indicates a failure or another event to the [$label_2$] path through the [$label^1_{SS}$] path that becomes the zig-zag path and replaces the [$label_2$] path with the [$label^2_{SS}$] path.

Besides the previous relations, it is important to define a *route* as a single UCM path that links an initial cause (S_{DS} start point in Figure 23c) to a final effect (E_{DS} end point). Routes are textually expressed by < >. For example, <S_{DS}, r_1, r_2, r_3, E_{DS}> is a route in Figure 23c. In this research, [] and ||| symbols are used to denote, respectively, *choice* and *parallel relations* between two or more routes. One of the alternative routes <S_{DS}, r_1, $label^1_{DS}$, r_2, r_3, E_{DS}> [] <S_{DS}, r_1, $label^2_{DS}$, r_3, E_{DS}> in Figure 23d is chosen depending on the result of a specific condition described in r_1. In this thesis, a *scenario* and a *trace* are used indistinctly to represent a single UCM *route*.

A UCM drawing tool called a UCM Navigator is available and was used in this thesis to edit the maps as well as to check for duplication among responsibility names. The UCMNav tool allows the designer to describe the detailed UCM presented in Figure 24 with pre- and post-conditions, triggering and resulting events. However, the tool presents two restrictions when describing mobile systems in this research. First, the disabling relation is not implemented in the tool. Second, the impossibility of printing the pre-conditions and the post-conditions of each responsibility that are needed for the comparison between functional behaviors. For a detailed description of the UCM notation, the interested reader may refer to [48]. For information about the current status of the tool and references about UCMs refer to [181].

4.5 Approach for Describing the Chosen Systems with UCMs

Capturing requirements with UCMs has been done in several case studies including features for mobile systems such as **I**ncoming **C**all **S**creening (ICS) [12][23], **C**all **N**ame **P**resentation (CNAP) [200], and a mobile group call system [14][15]. These case studies have shown that the UCM notation is suitable to provide a visual description of telecommunication features at a high level of abstraction. UCMs can also describe the

integration of different features. As mentioned earlier, this research has chosen UCMs for its synthesis capability together with the ability to describe the functional behaviors and the architectural elements independently. The UCM relations introduced in Section 4.4 are used to describe the functional behaviors.

The next subsection presents the reverse and forward engineering approaches that are used to extract functional behaviors and architectural elements from the chosen systems for the generation of their UCM descriptions. Section 4.5.2 outlines the UCM documentation conventions adopted to describe these systems.

4.5.1 Reverse and Forward Engineering Approaches

In order to describe the chosen systems (see Chapter 2) with UCMs, a mixture of forward and reverse engineering approaches is used [65][117]. The former is used to describe UCMs from the stage 1 requirements (Figure 25a). The latter is used to build UCMs from stage 2 MSCs (Figure 25b).

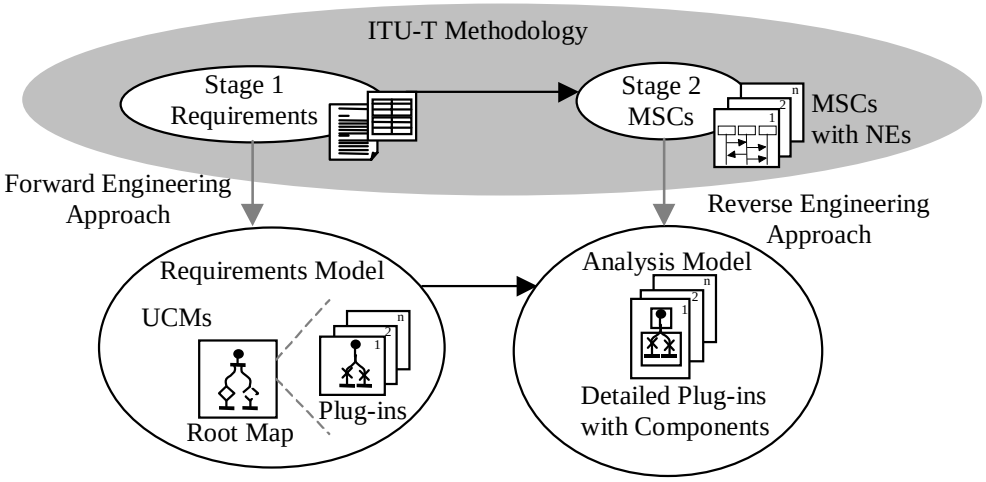
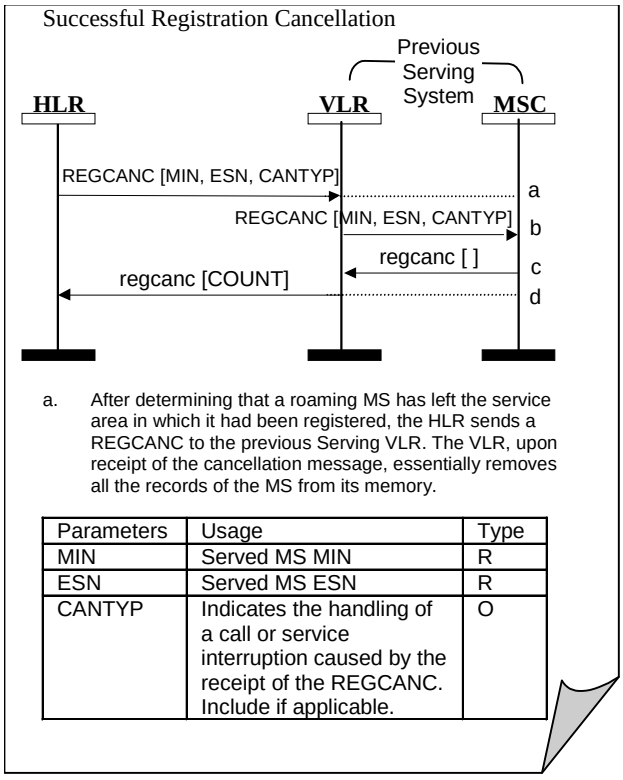


Figure 25 Forward and Reverse Engineering Approaches

Stage 3 documents are only considered when the details provided by the stage 2 disjoint MCSs are not sufficient to integrate different paths (e.g., with the *parallel relation*) or to represent a sequence of responsibilities (i.e., to use the *causal relation*).

Requirements and analysis models are then generated for each system on the basis of the results of these approaches (see more details about these models in Chapter 6). At the requirements stage, unbound UCMs describe a system. A UCM path starts when the pre-condition of the start point is satisfied and it finishes when the post-condition of the end point is satisfied (see *enabling* and *resulting relations* in Section 4.4). At the analysis stage, the architectural elements that are involved in each responsibility are included in the responsibility description and bound to the maps as UCM components.

Figure 26 illustrates a stage 2 document used in the reverse engineering approach to generate a bound UCM. This document describes the ANSI-41 location cancellation function and Figure 28b shows the bound UCM that is generated from it. The information contained in the tables (parameters and their usage) are not translated to the UCM system description but used to understand the messages in each information flow scenario.



a. After determining that a roaming MS has left the service area in which it had been registered, the HLR sends a REGCANC to the previous Serving VLR. The VLR, upon receipt of the cancellation message, essentially removes all the records of the MS from its memory.

Parameters	Usage	Type
MIN	Served MS MIN	R
ESN	Served MS ESN	R
CANTYP	Indicates the handling of a call or service interruption caused by the receipt of the REGCANC. Include if applicable.	O

Figure 26 ANSI-41 standard documents (stage 2)

The reverse and forward engineering approaches were first used in this research to generate UCMs from the WIN standards [176] (see Section 2.6 of Chapter 2 for details on the WIN standards) and the results were presented in [12]. The identification of the functional behaviors and architectural elements, which are necessary to build UCMs, is not an easy task since the information available is spread over different stages and it does not always lead to the information needed (see a discussion about the drawbacks of the current approaches in Section 4.2).

Table 7 and Table 8 show steps used in the description of the chosen systems with UCMs. This experience of how to describe a system with UCMs is not only applied to the standard documents of ANSI-41/WIN, GSM/GPRS, IMT-2000, and UMTS, which are described using the ITU-T three-stage methodology, but also to other systems such as WmATM networks that are presented in the same format (tables, texts, and MSCs). The steps presented in Table 7 and Table 8 are generalized, as follows:

- first, a general description of the system in terms of a UCM root map is given on the basis of the informal descriptions and paths with details hidden in the stubs (step 1 in Table 7);

- then, plug-ins with responsibilities that represent actions or events described in the informal documents are added to the stubs (step 2 in Table 7);

- more paths and new sub-maps with detailed start points, responsibilities, and end points are added to the plug-ins based on the information flows (step 3 in Table 7);

- last, UCM components, which correspond to functional entities, network entities, or the mapping of FEs to NEs, are bound to the maps that describe functional behaviors (respectively, steps 4, 5, or 6 in Table 8).

At the beginning of the system description as shown in Table 7, there is no worry about UCM components. **Step 1** is concerned with the description of the system's global scenario (e.g., the *root map* shown in Figure 23a). The first UCM obtained from the informal description provides a context where specific *functional behavior* related to mobility, communication, and radio resource management can be inserted. Further details such as actions or events to be performed are delayed with the stub notation.

Steps	From Standard Document Notations	To the UCM Notation
1	Informal description of the system (Stage 1: text)	A map to show the global picture of the system (root map)
2	Informal description of individual features (Stage 1: text and tables)	Sub-maps with responsibilities for each scenario (unbound plug-ins for stubs)
3	Information flows (Stage 2: MSCs with exchange of messages between NEs)	Paths and new sub-maps with detailed start points, responsibilities, and end points (unbound plug-ins for stubs)

Table 7 Functional Behavior: From the Standard Document Notation to UCMs

Step 2 focuses on the *enabling relation*, *causal relation*, and *resulting relation*. In other words, start points, causal paths between *responsibilities*, and end points are extracted from stage 1 text and tables to represent each functional behavior.

More details regarding the *start points* (triggering events and pre-conditions), *responsibilities* (pre- and post-conditions), and *end points* (resulting events and post-conditions) are added in **step 3**. Detailed unbound plug-ins such as the one in Figure 24 are the result of this step. The *choice relation*, the *join relation*, the *parallel relation*, and

the *disabling relation* come into play at this step. As a result, new *paths* (or UCM routes) and *sub-maps* (new decisions) are generated

Each responsibility corresponds to one or more exchanged messages in the MSCs and the actions or the events that follow them. In addition, the sequence of responsibilities corresponds to the sequence of actions or events performed as a result of the exchanged messages. However, control messages such as *"forward or relay a message to the next component"* (e.g., *regcan []* and *regcan[COUNT]* in steps c and d of Figure 26) without embedded processing should not be expressed in UCMs (see also m_1 and m_2 in Figure 27a). For this reason, a path that crosses two UCM components does not necessarily correspond to an interface between two functional entities or two network entities (see NE1 and NE2 in Figure 27c).

After these three sequential steps, components can be added to the plug-ins following the description of the architectural elements available in the standard documents (functional entities, network entities, or both). Table 8 presents the components that are extracted from three different standard reference models that are included in the WIN documents. The WIN standard describes **F**unctional **E**ntities (FEs) in the **D**istributed **F**unctional **M**odel (DFM), **N**etwork **E**ntities (NEs) in the **N**etwork **R**eference **M**odel (NFM), and the mapping of FEs to NEs as presented in Section 2.6 of Chapter 2. These steps are not sequential and depend on the availability of each model in the standards.

Steps	From Standard Reference Models	To the UCM Notation
4	Distributed Functional Model (DFM): Stage 2 - Functional Entities (FEs)	Bound plug-ins with components that represent FEs
5	Network Reference Model (NRM): Stage 2 - Network Entities (NEs)	Bound plug-ins with components that represent NEs
6	Stage 2 - Mapping of FEs to NEs	Bound plug-ins with components that represent the mapping of FEs to NEs

Table 8 Architectural Elements and the Mapping of Unbound to Bound UCMs

Step 4 binds the unbound maps to the DFM. This step is difficult and error-prone, because there is not enough information in the current standard documents about which responsibilities are assigned to which FEs. Hence, UCM responsibilities are described according to the purpose of each functional entity. This step is optional since it is not every mobile system that presents functional entities. It was considered during the description of features for WIN [12][23] but is not included in this thesis.

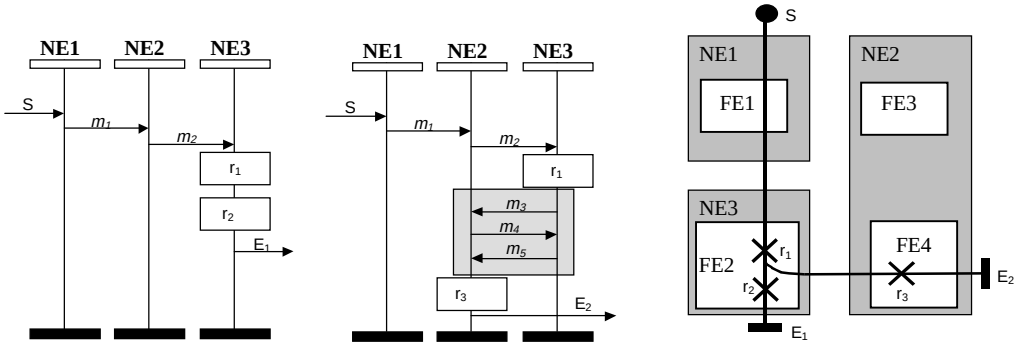
Step 5 describes the network entities as UCM components. As shown in Table 8, the network reference model and information flows with NEs are presented in stage 2. This step is used in the description of every system in this research. The network reference

model of each system is generated in terms of UCM components and is bound to the maps generated in step 3 of Table 7.

The mapping of FEs to NEs is easily translated into UCM components and bound to UCM maps in **step 6** as shown in Figure 27. This mapping is only presented in the WIN and in the IMT-2000 standard documents; hence, it is also an optional step during this process. In [12], the unbound UCMs are mapped to FEs and NEs according to the mapping presented in the annex of the WIN standards and to the information flows between NEs presented in stage 2 (see a bound plug-in with a group of components in Figure 23).

It is important to emphasize that the use of functional entities at the analysis stage is more appropriate than the use of network entities as stated in [12]. However, this thesis expresses UCM components in terms of network entities instead of functional entities due to the diverse representation of architectural elements among the chosen systems, which do not always provide FE descriptions. When functional entities are used in the analysis stage, they can be mapped to the network entities at the design stage according to step 6. Network entities create one more level of abstraction from the physical entities that represent the implementation of the architectural elements.

Figure 27 depicts an example of the reverse engineering approach with two MSCs described by a bound plug-in (steps 3 to 6 are used to generate this UCM). Since our focus is on the early stages, the lack of formality of UCMs constitutes an advantage when describing them from the MSCs.



(a) A MSC to represent Scenario X

(b) A MSC to represent Scenario Y

(c) A bound UCM to represent Scenario X (<S, r_1, r_2, E_1>) and Scenario Y (<S, r_1, r_3, E_2>)

Figure 27 From MSCs to bound UCMs

The next sections present the capture of common functional behaviors and architectural elements among the chosen systems. These commonalities are identified from the UCM descriptions that are generated from the reverse and forward engineering approaches mentioned previously. The generation of the system descriptions with UCMs and the comparison among these descriptions are done manually. The investigation of an

automatic process of system generation and comparison is left as an open area for research.

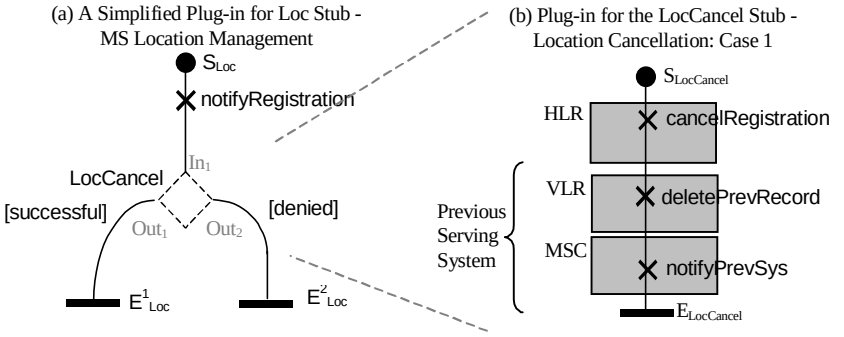
4.5.2 UCM Conventions

In this research, the following documentation conventions are adopted when describing a system with UCMs (see Figure 23 and Figure 24 for examples of the UCM notation):

- A *UCM* name (e.g., a root map name or a plug-in map name) is representative of a functional behavior;
- A *component* name is representative of an architectural element;
- A *stub* name is an acronym for a plug-in name;
- *A start point* name is represented either by S in case of a root map or by $S_{StubName}$ or $S^{StubName}$ in case of a plug-in bound to a stub. Numbers are added when there are two or more start points in a map (e.g., $S^{1}_{StubName}$ or $S_{1}^{StubName}$);
- A *responsibility* name is representative of events or actions to be performed. Each responsibility is composed of a verb followed by a noun or a noun phrase that represents actions or events;
- A *pre-condition* name is represented by P_*start point name* or P_*responsibility name*;
- A *triggering event* name is represented by TE_ P_*start point name*;
- A *post-condition* name is represented by Q_*end point name* or Q_*responsibility name*;
- A *resulting event* name is represented by RE_Q_*end point name*;
- An *End Point* name is represented either by E in case of a root map or by $E_{StubName}$ or $E^{StubName}$ in case of a plug-in bound to a stub. Numbers are added when there are two or more end points in a map (e.g., $E^{1}_{StubName}$ or $E_{1}^{StubName}$);
- A *path label* is represented by [*pre- or post-condition name*] if it is associated with the *choice* relation (post-conditions of a responsibility), with a stub post-condition or with the *parallel* relation (pre-conditions of a start point or post-conditions of a responsibility).

Figure 28 depicts two examples of the use of these conventions in the description of the ANSI-41 location management function. Figure 28a shows a simplified version of the MS Location Management plug-in, which is bound to the Loc stub (not shown here). Figure 28b illustrates one of the plug-ins bound to the LocCancel dynamic stub. The architectural elements involved in the Location Cancellation functional behavior are also

shown in the figure (see the ANSI-41 successful location cancellation scenario in Figure 26).



S_{Loc}	TE_P_S_{Loc} : a serving system detects a new mobile station within its area.
	P_S_{Loc} : a new mobile station has entered in a system.
notifyRegistration	P_ notifyRegistration: a new mobile station has entered in a system.
	Q_ notifyRegistration: a registration request has been sent.
LocCancel	P_ LocCancel: a location cancellation has been requested.
	Q1_ LocCancel: a location cancellation has been denied.
	Q2_ LocCancel: a location cancellation has been completed successfully.
E^1_{Loc}	Q_ E^1_{Loc}: a location cancellation has been completed successfully.
	RE_ Q_ E^1_{Loc}: a successful notification is sent.
E^2_{Loc}	Q_ E^2_{Loc}: a location cancellation has been denied.
	RE_ Q_ E^2_{Loc}: an unsuccessful notification is sent.

Figure 28 UCM Documentation Convention: ANSI-41 Location Management

4.6 Approach for Capturing Commonalities with UCMs

As discussed in Section 4.4 and Section 4.5, the uniform description of the chosen systems with UCMs includes their functional behaviors, which are represented by unbound UCMs, and their architectural elements, which are represented by UCM components. These system descriptions are considered requirements and analysis models (respectively, unbound and bound UCMs) in this research.

Commonalities among the system descriptions (i.e., functional behaviors and architectural elements) are discovered by inspection. Our inspection focuses on mobility and radio resource management functions.

The system functional behaviors are analyzed to determine start points, responsibilities, and end points that are common among them [130][118][136]. An unbound UCM description is then generated to represent each commonality that is identified according to the approach presented in Section 4.6.1. The approach also shows that this UCM is a sub-map of the system UCMs.

Once a common functional behavior is found, the system analysis models are used to identify common architectural elements. We concentrate on the bound UCMs that include common responsibilities, start points or end points. We generate bound UCMs for the

common functional behaviors on the basis of these common architectural elements. Section Figure 30 presents the approach for the capture of common architectural elements.

These general steps are summarized as follows:

Step 1. Determine common functional behaviors among the system descriptions on the basis of pre- and post- conditions associated with start points, responsibilities, and end points.

Step 2. Capture responsibilities with their pre-conditions and post-conditions, start points with their pre-conditions and triggering events (if any), and end points with their post-conditions and resulting events (if any) that describe commonalities among functional behaviors and generate unbound UCMs.

Step 3. Determine common architectural elements among the system descriptions on the basis of the architectural elements that perform the common functional behaviors in the system descriptions.

Step 4. Capture common architectural elements and generate bound UCMs to describe them and their corresponding common functional behavior.

The next subsections describe these steps in detail within the approaches for capturing common functional behaviors (step 1 and step 2) and common architectural elements (step 3 and step 4).

4.6.1 Approach for Capturing Common Functional Behaviors

At the requirements stage, a system is described as a set of functional behaviors (see definitions in Section 4.3) that are described with UCMs. As mentioned in the *condition* definition, a logical assertion describes a condition, which is true at a given point in the execution of a functional behavior. With respect to each responsibility, we call the assertion that precedes it a *pre-condition* (e.g., $P\_r_0^A$ in Figure 24) and the assertion that follows it a *post-condition* (e.g., $Q\_r_0^A$ in Figure 24). In case of start points, *pre-conditions* can be associated with *triggering events* (e.g., P_S_A and $TE\_P\_S_A$ in Figure 24) and in case of end points, *post-conditions* can be associated with *resulting events* (e.g., Q_E_A and $RE\_Q\_E_A$ in Figure 24).

A *common functional behavior* is identified in terms of common start points (triggering events and pre-conditions), common responsibilities (pre-conditions and post-conditions), or common end points (post-conditions and resulting events). For instance, in FB contexts that occur in two or more systems:

- two or more start points are *common* if they are enabled (see the *enabling relation* in Section 4.4) in a common context (pre-condition) associated with the respective triggering events if any;

- two or more responsibilities are ***common*** if they are enabled (see the *enabling relation* in Section 4.4) in a common context (pre-condition) and if they achieve a common resulting context (post-condition) as illustrated in Figure 29.

- two or more end points are ***common*** (see the *resulting relation* in Section 4.4) if they achieve a common resulting context (post-condition) associated with the respective resulting events if any.

The following definitions will clarify these ideas. These definitions constitute an attempt to formalize our intuition behind the concept of commonality. The reader will see that these definitions are not used explicitly in the rest of the thesis. This is because we are using realistic examples, and it would have been excessively complicated to formulate pre- and post-conditions for every part of the examples. However, Appendix A presents detailed responsibilities, start points and end points of the identified commonalities among the chosen systems. We leave to further research the task of developing these detailed models.

It is also important to notice common responsibilities, pre- and post conditions, start points, and end points are not called in the same way in different systems (e.g., "handoff" and "handover" are equivalent terms used respectively in North America and in Europe). We assume for simplicity that, at the time a commonality is found by inspection, a common naming scheme is created, so that common entities in the various systems get the same names.

For the purpose of the following definitions, we assume that all responsibilities, start points, and end points in a given UCM have different names. If this is not true in practical cases then we can always assume that identical names can be made different by renaming them.

Let FB be a functional behavior, S be a start point, E be an end point, and r_0, r_1, …, r_n be a set of responsibilities that can be performed along a UCM path (see Figure 29). We use the *functional behavior*, *condition*, *triggering event*, *resulting event*, and *responsibility* definitions presented in Section 4.3 as follows:

(A1) FB_A is a map that represents a functional behavior of a system A (e.g., the UCM route expressed by $<S_A, r_0^A, r_1^A, \ldots, r_n^A, E_A>$), FB_B is a map that represents a functional behavior of a system B (e.g., a UCM route expressed by $<S_B, r_0^B, r_1^B, \ldots, r_n^B, E_B>$), and FB is a map that represents a set of commonalities discovered by inspection (e.g., a UCM route expressed by $<S, r_0, r_1, \ldots, r_n, E>$);

(A2) P_S is a pre-condition, defined in a start point S, Q_E is a post-condition defined in an end point E, and P_r and Q_r are pre- and post-conditions defined before and after a responsibility r;

(A3) A triggering event TE_P_S defined in a start point S satisfies a pre-condition P_S;

(A4) When Q_E is satisfied, a resulting event RE_Q_E defined in an end point E occurs.

So we use (A1) to (A4) to precisely define a *common* S, a *common* E, and a *common* r:

(D1) S is *common* for S_1^A and S_1^B if $P\_S_1^A$ implies P_S and $P\_S_1^B$ implies P_S;

(D2) E is *common* for E_1^A and E_1^B if $Q\_E_1^A$ implies Q_E and $Q\_E_1^B$ implies Q_E;

(D3) r is *common* for r_1^A and r_1^B if $P\_r_1^A$ implies P_r and $P\_r_1^B$ implies P_r or $Q\_r_1^A$ implies Q_r and $Q\_r_1^B$ implies Q_r.

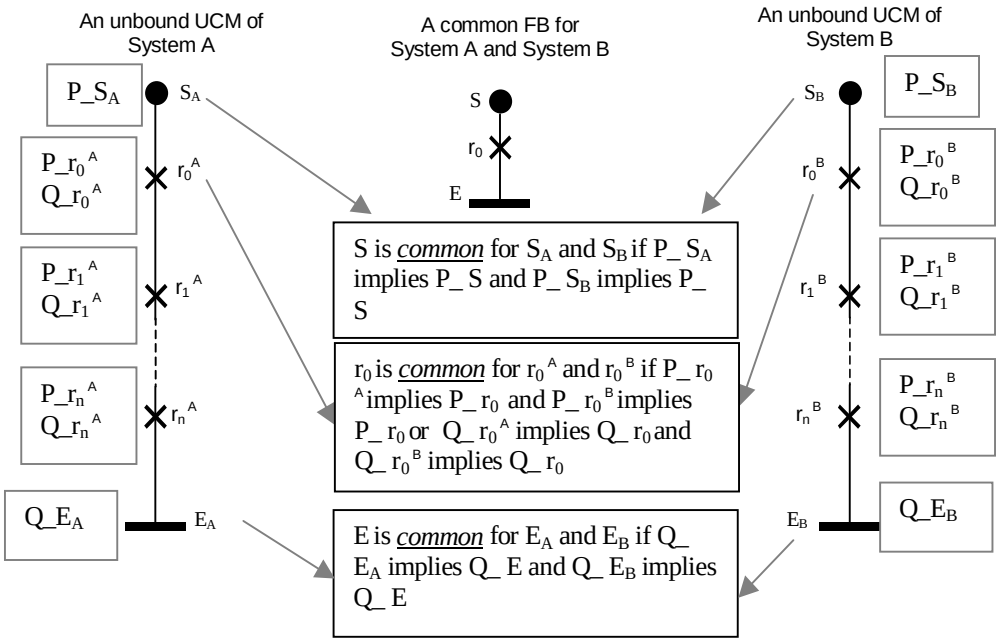


Figure 29 Capture of Common Responsibilities, Start Points, and End Points

A UCM A' is said to be a sub-map of a UCM A if the following conditions hold:

(C1) The set of responsibilities of A' is a subset of the set of responsibilities of A;

(C2) Whenever there is a path from a responsibility to another in A, there is a path between the same responsibilities in A', possibly including all responsibilities.

A commonality between UCM A and B is a UCM A'B' that is a sub-map of both A and B. Then, we use (D1) to (D3), (C1) and (C2) to precisely define a *common* FB:

(D4) a *common* FB for system A and system B is a sub-map of both A and B that describes one or more *common* r, *common* S, or *common* E.

Note that a *common* functional behavior does not need to be a complete sub-map in the sense that they may lack start and end points as depicted in Figure 30. If there is no common start points or end points, the pre-condition of the first common responsibility

becomes the pre-condition of the start point and the post-condition of the last responsibility becomes the post-condition of the end point.

In addition, when a *common* functional behavior is identified and extracted from system A and system B, the commonalities are renamed.

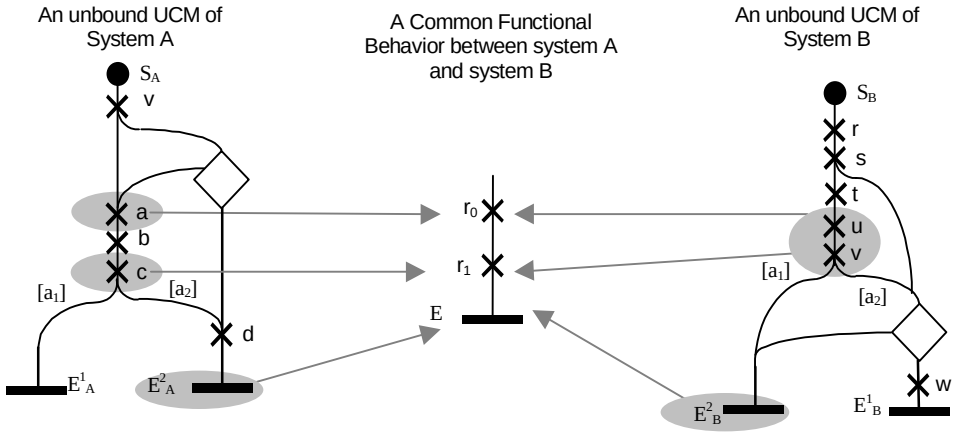


Figure 30 Capture of Common Functional Behaviors

4.6.2 Approach for Capturing Common Architectural Elements

At the analysis stage, a system is described as a set of architectural elements and their respective FBs. An architectural element is determined *common* for two or more systems on the basis of *common* start points, *common* end points, and *common* responsibilities.

Let AE_A be an architectural element of system A, AE_B be an architectural element of system B, AE be a common architectural element discovered by inspection, S be a *common* start point, E be a *common* end point, and r be a *common* responsibility, we use (D1) to (D3) introduced in the last section to precisely define a *common* AE:

(D5) AE is *common* for AE_A and AE_B if AE performs a *common* r that is also performed by AE_A and AE_B;

(D6) AE is *common* for AE_A and AE_B if AE includes a *common* S that is also included in AE_A and AE_B;

(D7) AE is *common* for AE_A and AE_B if AE includes a *common* E that is also included in AE_A and AE_B;

(D8) a *common* AE performs one or more *common* responsibilities (r), includes one or more *common* start points (S) or *common* end points (E).

Figure 31 illustrates the capture of common architectural elements between a bound UCM of system A and a bound UCM of system B.

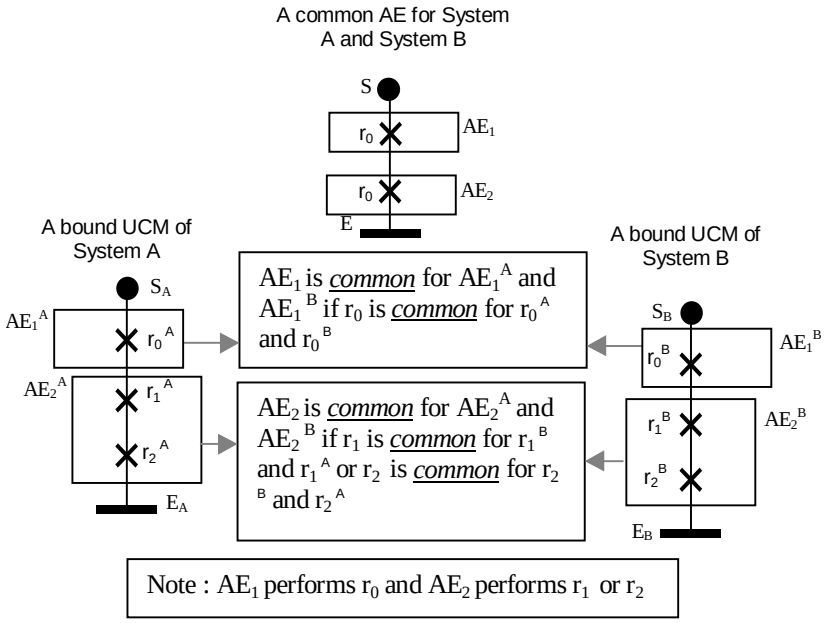


Figure 31 Capture of Common Architectural Elements

4.7 Common Functional Behaviors

As mentioned in Section 4.5, in order to capture commonalities among the chosen systems, we start by expressing them with UCMs (see Appendix A). After this, *common* functional behaviors (FBs) are identified in terms of start points, responsibilities or end points as defined in (D4) of Section 4.6.1. The visual representation of common functional behaviors without considering which architectural element performs them is the main advantage of using unbound UCMs at this stage.

When a *common* FB is identified among at least three of the chosen systems (according to the pattern definition in Chapter 3), they are extracted from the system descriptions and represented by an unbound UCM (see Figure 32). Each map is built following the *enabling relation*, the *resulting relation*, the *causal relation*, the *choice relation*, and the *join relation* on the basis of the identified common FBs. Each common responsibility, common start point or common end point is re-named in this unbound map. However, the new names still follow the UCM conventions presented in Section 4.5.2.

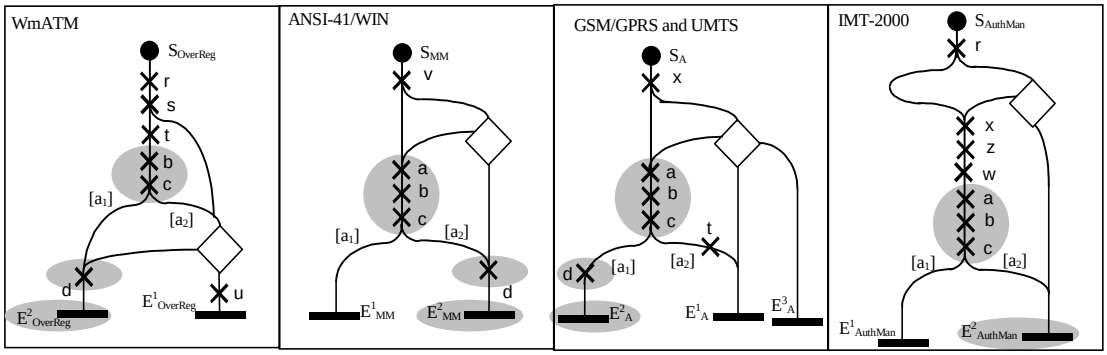
The next sub-sections present unbound UCMs that describe the identified common functional behaviors related to mobility and radio resource management functions. At this point, the focus of this research is on the UCM description of every common functionality as an isolated map, which represents a pattern solution in Chapter 5. However, these maps become plug-ins bound to stubs in Chapter 6, which introduces an approach for reuse and validation of patterns in the development and evolution of mobile systems.

4.7.1 Unbound UCMs related to Common Mobility Management Funtions

Every system considered in this study performs mobility management functions to update information about a mobile user location, to locate a user, and to guarantee location access security. Every system also invokes a location cancellation function to delete information from the previous location area of the mobile station.

Figure 32 shows the capture of functional behaviors with a simplified example related to the mobility management functions of WmATM, ANSI-41/WIN, GSM/GPRS, UMTS and IMT-2000 systems. These maps describe how each system handles the authentication of a mobile user (see more details in Section 2.3.2 of Chapter 2 and in Chapter 5).

(a) Simplified Functional Behavior Descriptions with UCMs



(b) List of Commonalities

Responsibility and End Point	Pre-Condition (P), Post-condition (Q), and Resulting Event (RE_Q_E)
a: send authentication information	P_a : an authentication request has been accepted. Q_a : the MS authentication information has been received.
b: apply authentication algorithm	Q_b : an authentication result has been generated as a result of specific calculations.
c: check authentication result	Q1_c : MS authentication result is different from the network authentication result; Q2_c : otherwise.
d: notify a denied authentication	Q_E^2_{Auth} implies Q_d : a notification of the MS unsuccessful authentication has been sent.
E^2_{Auth}	RE_Q_ E^2_{Auth}: unsuccessful authentication is complete.

(c) Common Functional Behavior: Authentication

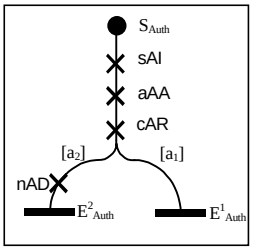


Figure 32 An Example of the Capture of Commonalities: Authentication Functional Behavior

Figure 32b lists a set of *common responsibilities* that are performed by all systems during the authentication function. In addition, a *common end point* (*post-condition* and *resulting event*) is also identified.

In Figure 32c, an unbound UCM map describes the identified sequence of *common* responsibilities and end point (the *causal relation*, the *choice relation*, and the *resulting relation* are used) for extracting the authentication FB, which includes two alternative routes as follows: <S_{Auth}, sAI, aAA, cAR, nAD, E^1_{Auth}> [] <S_{Auth}, sAI, aAA, cAR, E^2_{Auth}>. The commonalities are renamed after being identified among the systems. In addition, S_{Auth} is a start point generated from the P_a pre-condition and E^1_{Auth} is an end

point generated from the Q_c post-condition. Responsibilities, such as r, s, t, u, v, x, z, and w are not common among at least three of the chosen systems, and so they are disregarded as common functional behavior and left out of the responsibility list.

The Authentication map depicted in Figure 32c is triggered when the network side or the mobile user side (e.g., when a power-on event or a change of location area occurs) requests the authentication (triggering event of the S_{Auth} start point). After this, the result of an authentication operation performed by the mobile station is sent to the network (the send Authentication Information responsibility, called sAI). Then, the aAA (apply Authentication algorithm) responsibility performs the same authentication operation at the network side. The cAR (check Authentication result) responsibility generates successful or unsuccessful outcomes (respectively, E^2_{Auth} or E^1_{Auth} end points) depending on the outcomes of the comparison between the respective results. In the case of denied authentication, the mobile user is notified ([a2] path with nAD responsibility). Otherwise, a successful authentication ([a1] path with) occurs and a resulting event is generated.

Figure 33, Figure 34, and Figure 35 illustrate the unbound UCMs related to the other common mobility management functions.

The Temporary identification assignment map depicted in Figure 33a is triggered when a mobile user powers on a mobile station or a mobile station changes location area (two different triggering events of the S_{TempID} start point). First, the network assigns the mobile station's temporary identification (the assign Temporary Identification responsibility, called aTID) and sends it to the mobile station. Then, the cATID (check the assigned Temporary Identification) responsibility checks whether the mobile station gets this temporary identification or not. This responsibility generates successful or unsuccessful outcomes (respectively, E^1_{TempID} or E^2_{TempID} end points) depending on the different outcomes. If the confirmation is not received by the network, the operation is not successful ([a2] path). Otherwise, the network completes this assignment successfully ([a1] path).

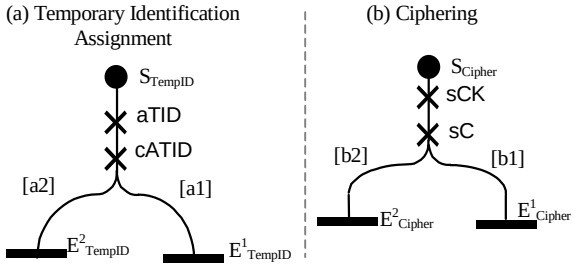


Figure 33 Unbound UCMs for Temporary Identification and Ciphering

The ciphering map is illustrated in Figure 33b. This function is responsible for starting a ciphered communication over the air interface (ciphering and deciphering the data information). The sC responsibility ciphers the data that is sent by the mobile station and the network deciphers it. The ciphering key, which is generated by the sCK (send

ciphering key) responsibility, and the respective ciphering algorithm are used in the ciphering/deciphering procedure. This ciphering FB starts when the network sends a ciphering mode request to a mobile station (S_{Cipher} start point). The change to the ciphering mode ends successfully ([b1] path) after the network and the mobile station agree upon the ciphering/deciphering procedures (i.e., mobile station acknowledge the use of ciphering/deciphering). Otherwise, the map exits at the [b2] path.

The Paging, which map is illustrated in Figure 34, is responsible for finding a mobile station within its current location area. The pU responsibility issues the paging that starts when an incoming call arrives to an idle terminating party (S_{Pag} start point). The paging ends after either the mobile station is reached (i.e., it responds to the paging). A channel is then assigned for the communication (successful outcome). Otherwise, the mobile station is unreachable (unsuccessful outcome). These outcomes are represented by, respectively, <cAMS, E^1_{Pag}> [] <nUU, E^2_{Pag}> sub-routes.

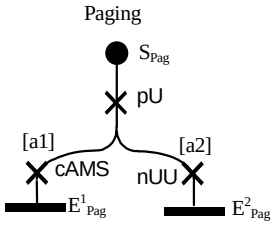


Figure 34 Unbound UCMs for Paging

Figure 35 describes the Location Registration map that is triggered when the mobile user roams and needs to be registered in the current location area (S_{Updt} start point). After getting the location information (sLI responsibility), the cL responsibility (check Location) generates different outcomes according to the following post-conditions: either the mobile user is visiting a new location area (both visitor and home databases are updated) or not (just the home database is updated). The uPL (update home user Profile when not roaming), uPR (update home user Profile when roaming) and uTP (update Temporary user Profile) responsibilities are operations on the home and visitor database records. Sub-paths labeled [a1] (visiting location area) and [a2] (home location area) are joined to perform the location cancellation.

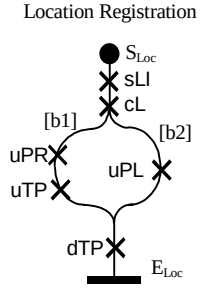


Figure 35 Unbound UCMs for Location Registration

The purpose of the location cancellation is to delete the user profile in the location area previously visited by the mobile station. The temporary profile is deleted from the previous visiting database (dTP responsibility in the figure). Unsuccessful outcomes are not shown in the figure but they can occur due to network or database failures as presented in Chapter 7.

4.7.2 Unbound UCMs related to Common Radio Resource Management Functions

The identified common functional behaviors among radio resource management functions are represented by the handoff functions performed to maintain the quality of the link when the user moves from one location to another as well as the functions performed to release resources (see Figure 36 and Figure 37). We concentrate on the handoff involving two mobile switching centers since this is the only case that affects the signaling in the application layer. This function also handles handoff failures, which occur due to the lack of good links surrounding a mobile station that is roaming.

The *handoff decision* is taken according to measurements represented by take measurements (tM) responsibility in Figure 36a. After this, a comparison is done (compare measurements (cM) responsibility). The $E^1_{Hdecision}$ and $E^2_{Hdecision}$ end points represent the need of having a handoff or not.

Figure 36b depicts a map with the rR responsibility that is performed after a successful inter-system handoff ([a1] path with the E^1_{Hand} end point in Figure 37a) to release the resources allocated in the previous area. In addition, this map is performed after a handoff failure ([a2] path with the E^2_{Hand} end point in Figure 37a that leads to Figure 37b) to release the resources allocated in the new area before the handoff failure.

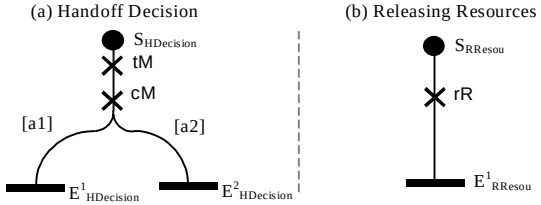


Figure 36 Unbound UCMs for Handoff Decision and Releasing Resources

Figure 37a starts with a handoff request triggering event (S_{Hand} start point). After this, a new channel is allocated (aC responsibility). The mobile station tunes to the new channel (tNC responsibility). The new channel is verified to guarantee that the new link has better quality of transmission than the previous one (vC responsibility). Alternative sub-paths labeled [a1] and [a2] are generated as a result of this action. In case of negative result, the (the resulting event of the E^2_{Hand} end point triggers the S_{HFail} start point in Figure 37b).

The map depicted in Figure 37b describes the actions to be performed when a handoff failure occurs. First, the mobile station tunes to the previous channel. Second, all network resources, which are allocated for the new channel and are no longer needed, are released (rRA responsibility). This last action is also described by the Releasing Resources FB.

The next section describes the *common* architectural elements (AEs) among mobile wireless communication systems. A network reference model incorporates these elements that are specified in terms of UCMs components. The mapping of the *common* FBs previously illustrated to the *common* AEs is done in Chapter 5.

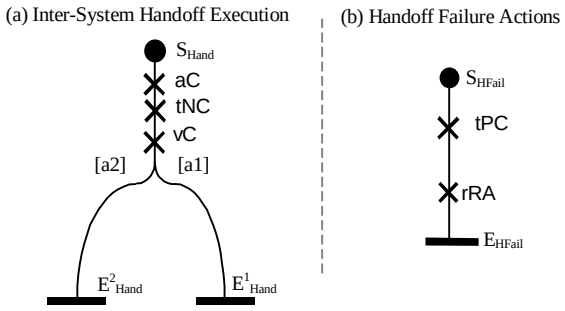


Figure 37 Unbound UCMs for Inter-system Handoff and Handoff Failure Actions

4.8 Common Architectural Elements

The last step of the capture process involves the extraction of *common* architectural elements (AEs) from the system descriptions with bound UCMs. First, the functional behavior (FB) of each architectural element is investigated on the basis of the process introduced in Section Figure 30. Then, *common* AEs are identified and described in a common network reference model with the UCM component notation. These *common* architectural elements are then bound to the previously identified *common* FBs, which are mobility and radio resource management functions captured from the chosen systems (see bound UCMs in the pattern solutions presented in Chapter 5).

Figure 38 illustrates the UCM component models of ANSI-41/WIN systems and WmATM networks. The WIN network reference model and the WmATM environment (see, respectively, Section 2.6 and Section 2.9 in Chapter 2 for a detailed description of these elements) are described with UCM components and their commonalities are identified on the basis of the identified *common* FBs (not shown in the figure).

As shown in Section 2.3.1 of Chapter 2, each of the chosen systems shares common architectural elements and concepts. However, they are presented in different formats. For instance, in WIN, which is supported by ANSI-41-D, these architectural elements are expressed in terms of network entities in the WIN network reference model. These network entities, which are introduced in the WIN standard [169] as discussed in Section 2.6.4 of Chapter 2, are an abstraction of the physical components often called physical entities in the literature [25][111]. Even though they determine a specific structure for a mobile system, these entities are still one step away from the physical entities. Implementations ultimately require that network entities be allocated to specific physical entities. In other words, one or more network entities are grouped into a physical entity at the implementation stage.

Furthermore, the WmATM network descriptions in the literature consider home and visitor databases indistinctly and they do not mention the security database explicitly. However, we include visitor, home, and security databases in the common network reference model illustrated in Figure 39 since they represent commonalities for security and location problems of three other systems as follows (see pattern concept in Chapter 3): ANSI-41/WIN, GSM/GPRS and IMT-2000 systems.

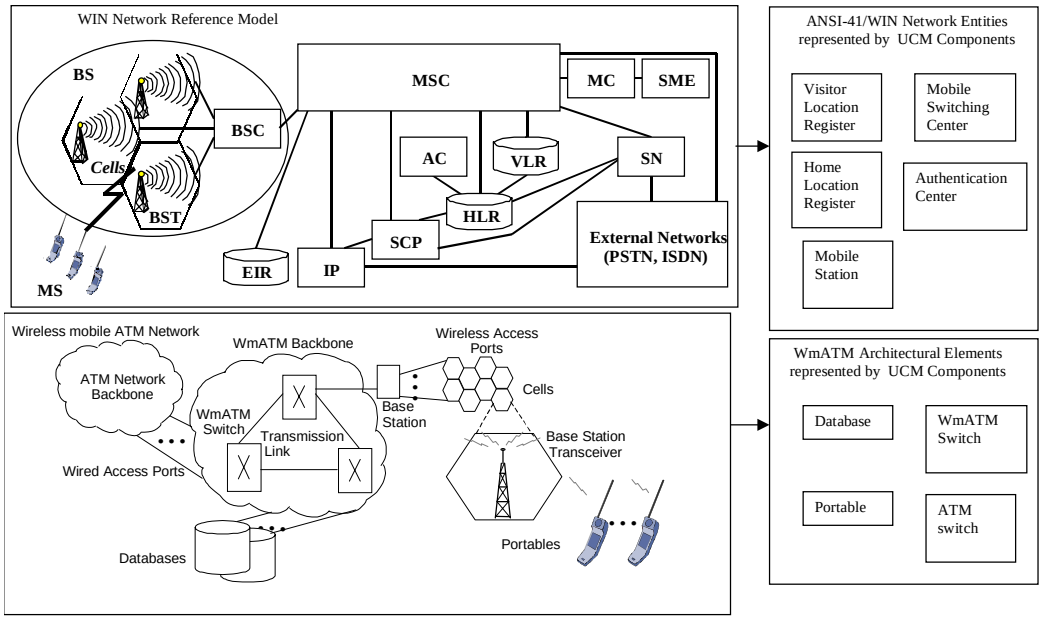


Figure 38 The UCM Component Model of ANSI-41/WIN Systems and WmATM Networks

As mentioned earlier in this chapter (see Figure 20), this research focuses on the architectural elements that support mobility and radio resource management operations in the application layer. For instance, mobile switching center, security database (e.g., the GSM authentication center), home database (e.g., the ANSI-41 home location register), and visitor database (e.g., the IMT-2000 visitor location register) are examples of these architectural elements as depicted in Figure 38. Although base station transceivers and base station controllers are also involved in these operations, they are related to the lower layer protocols and are not represented as UCM components in the system descriptions. During the system descriptions with bound UCMs, we assume that mobile stations communicate directly with the mobile switching centers and the users' movements are represented by the change of location area.

Figure 39 illustrates a typical mobile wireless environment which is an example of the *common* architectural elements (see also Figure 66 in Chapter 7) that are captured among the chosen systems (see also Section 2.3.1 of Chapter 2) as follows: *mobile station*, *home database*, *visitor database*, *security database*, and *mobile switching center*. This scenario depicts a mobile station (represented by a car) that is roaming from its home location area to another location area (called visitor location area). The mobile station movement is represented by gray (previous locations) and black (current location) colors.

The mobile switching center is the interface between the base station controller (not shown in the figure) and the home database as well as the visitor database. Each mobile switching center is responsible for one or more location areas and for the exchange of messages between the network side and the mobile stations through common or dedicated radio channels.

The home database permanently keeps information about the mobile user profile while the visitor database temporarily keeps part of the mobile user profile. The security database is responsible for the sensitive data related to authentication and ciphering functions. In short, these databases are responsible for keeping information about mobile users' location, services, and equipment. Chapter 5 presents more details about these databases.

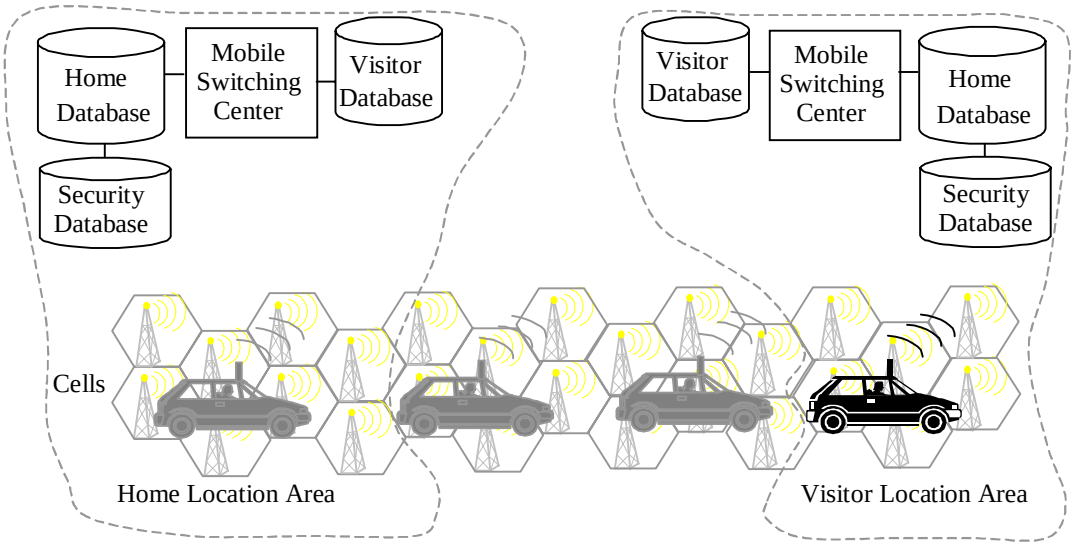


Figure 39 Typical Components of a Mobile Wireless Communication System

4.9 Related Work

The software engineering community has been focusing on commonalities and variabilities as a solution for the rapid development of software [27][56]. We believe that mobile communication software can also benefit from the advantages presented in these papers and we focus on capture, reuse, and validation of commonalities. Our target is to provide a seamless network environment for the development and evolution of mobile systems at the early stages.

This section discusses the results introduced in this thesis with respect to Scope, Commonality, and Variability (SCV) analysis and Family-Oriented Abstraction, Specification, and Translation (FAST) approaches that are presented in [56] and [27].

SCV analysis is proposed in [56] for the identification of commonalities and variabilities in software engineering. According to the authors, the SCV analysis is suitable for different stages of the software development life cycle and it decreases the software development cost as well as the time for creating new members of a software family, which is composed of "similar systems with many variations". These benefits are perceived when large numbers of family members are developed using the SCV analysis.

The SCV analysis consists of five steps as follows: (1) the establishment of the scope; (2) the identification of commonalities and variabilities, (3) the definition of boundaries for the variabilities; (4) the exploitation of commonalities; and (5) the accommodation of variabilities. Commonality and variability assumptions are chosen according to the scope.

With respect to the identification and capture of commonalities among mobile systems, which are introduced in this chapter, this research shows an application of SCV analysis for the mobile system domain. Our focus is on the first and second steps of the SCV analysis and variabilities are disregarded in this chapter. However, Chapter 6 introduces an approach for reuse of the commonalities that incorporates steps 4 and 7. We do not tackle step 3 since we concentrate on the reuse of commonalities that are documented as patterns and we leave variabilities and, in consequence, the boundaries applied to them, to the designer's decisions. Our results can be represented in terms of the systematic thinking process proposed by SCV analysis, as follows:

S: the chosen mobile systems;

C: *common* functional behavior and architectural elements;

V: functional behaviors and architectural elements that are not common among the chosen systems.

The results presented in this thesis are complementary to SCV analysis. While SCV analysis presents a systematic way of dealing with commonalities and variabilities without getting into details on how to capture commonalities from existing systems, this chapter proposes an approach for capturing commonalities among family members, which are mobile systems. Furthermore, we document the commonalities as patterns (see Chapter 5) to improve the reuse at the early stages and we add rigor to it with a validation part (see Chapter 6).

The FAST approach discussed in [27] and [56] applies the results of SCV analysis to program families. Then, it produces "a language for specifying family members" and generates "members from these specifications." With the focus on the design and implementation stages, the FAST approach is a step beyond our thesis contributions. We believe that the FAST approach can be combined with our results to generate a seamless mobile system environment at the design and implementation stages. As mentioned earlier, this thesis identifies commonalities, documents them as patterns, and formalizes the pattern solutions for the validation part. With respect to FAST, the MoRaR pattern language presented in Chapter 5 and the approach for reuse and validation proposed in Chapter 6 are considered, respectively, as the language and the systematic way for

generating family members for mobile systems at the requirements and analysis stages. Since our focus is not on variabilities, SCV analysis and FAST approach can be used to select and specify variabilities for these systems.

4.10 Conclusion

The diversity of systems that provide telecommunication services to mobile users have substantial differences related to their architecture, protocols, and services. However, these systems also adopt common solutions for dealing with design problems associated with their mobility, communication and radio resource management functions. While the differences reside in design details such as parameters, interfaces, and messages, the commonalities are perceived at the early stages when decisions regarding the choice of functional behaviors and architectural elements are taken.

This research focuses on mobility and radio resources management functions and presents a systematic way of investigating and capturing commonalities among mobile systems. We abstract design details with our target on requirements and analysis models in order to capture commonalities. ANSI-41/WIN, GSM/GPRS, WmATM networks and IMT-2000/UMTS systems are investigated for this purpose. As discussed in Section 4.2, the UCM notation is chosen to describe these systems and to capture the system commonalities.

First, the chosen systems are uniformly described with UCMs using the reverse and forward engineering approaches shown in Section 4.5.1. Then, these descriptions are analyzed in order to identify common functional behaviors and architectural elements, which are involved with each common responsibility. The graphical and abstract nature of UCMs facilitates the comparison among these systems. Their commonalities are identified in terms of responsibilities, start points and end points and represented as unbound UCMs. Finally, common architectural elements are identified as the basis for their common functional behaviors.

The next chapter presents patterns that are extracted from these common functional behaviors and architectural elements related to mobility and radio resource management functions. A pattern language is also introduced to group these patterns.

Chapter 5 MoRaR: A Pattern Language for Mobility and Radio Resource Management

As discussed in Chapter 4, the different mobile wireless communication systems that are currently available adopt common solutions for dealing with recurring mobility and radio resource management problems associated with similar architectural elements. This chapter extracts and documents patterns that identify such common problems and solutions among second and third generation systems. These patterns are grouped into a pattern language that shows how they interact. At a high level of abstraction, the pattern language makes it possible to generate different scenarios of the mobile system behavior.

5.1 Introduction

Mobile users are able to roam with a large variety of mobile wireless communication systems (see details in Chapter 2). For instance, the Global System for Mobile communications 900 (GSM-900) is a European-based technology that is the foundation for the digital cellular system 1800 (GSM-1800) [139] and the Personal Communication System 1900 (PCS-1900) [35]. Furthermore, the Digital Advanced Mobile Phone System (D-AMPS) (also known as Interim Standard 54-B) [35], which is a North American technology, defines a hybrid air interface that allows mobile stations to operate in a dual mode fashion (analog and digital). These cellular systems provide basic and supplementary telecommunication services [35][139]. The D-AMPS air interface is supported by the **A**merican **N**ational **S**tandards **I**nstitute – 41 (ANSI-41) [26][79]. ANSI-41 provides registration, roaming, call features, and other mobile application protocol features at the network side.

These systems have substantial differences related to their architecture, protocols, and services. More specifically, interfaces among components, cryptography algorithms, and types of handoff are proprietary solutions. However, they adopt common solutions for dealing with recurring mobility and radio resource management problems [86] as discussed in Chapter 4.

Third generation systems such as the International Mobile Telecommunications 2000 (IMT-2000) Systems [112][113][114] are under development to overcome the incompatibilities among the second generation systems mentioned earlier and to integrate the current standardization activities for the air interface and the cellular networks. Meanwhile, other solutions such as signaling protocols for Wireless mobile

Asynchronous Transfer Mode (WmATM) systems have been presented in the literature to provide mobility and radio resource management functions for high-speed Local Area Networks (LANs) and Wide-Area Networks (WANs) [4][54][191]. Both IMT-2000 systems and WmATM networks present commonalities with second generation systems regarding the architectural elements and the functional behaviors involved in mobility and radio resource management.

This research investigates the commonalities among mobile wireless communication systems as presented in Chapter 4 in order to identify, to capture, and to document patterns [57][80][160]. These patterns are grouped into a pattern language [6] for mobility and radio resource management (MoRaR) that shows possible relationships among them. Alternative scenarios can be derived from these relationships. In addition, this set of patterns allows designers to recognize similarities among legacy systems at the early stages of the system development process and evolution [116] and to re-use good solutions independent of implementation.

The next section summarizes the pattern writing steps used in this research. Section 0 introduces the MoRaR pattern language with pattern names indicated in italic. Section 5.4 describes patterns related to mobility management functions. The radio resource management patterns that include the handoff functions are presented in Section 5.5. Finally, Sections 5.6 and Figure 52 address our main contributions and potential future work. Conclusions are summarized in Section 5.8. A table with a summary and an index of all patterns presented in this chapter is provided in the Appendix C.

5.2 Pattern Decision, Capture, Search, and Writing

The common functional behaviors that are presented in Chapter 4 with UCMs are first analyzed and when it is appropriate, a pattern is captured and documented. These patterns are called *behavioral patterns* (as illustrated in Figure 2a of Chapter 1). As presented in Section 4.8 of Chapter 4, each architectural element (also called network entities in [174]) is then described with UCM components and the common elements are extracted. A common network reference model is proposed in Figure 38 of Chapter 4. Accordingly, when the pattern concept can be applied, these common network entities are translated to the pattern template and they constitute the *structural patterns*. As mentioned in Chapter 1 and Chapter 3, these patterns are suitable for requirements and analysis stages.

As shown in Figure 1c of Chapter 1 and discussed in Chapter 3, each pattern is documented using the following pattern template: *name*, *context*, *problem*, *forces*, *solution*, *rationale*, *resulting context*, and *known uses* [59][133]. Related patterns are mentioned when is necessary. A pattern is captured and documented every time a common recurring design problem and its respective solution can be represented in this pattern template.

The UCM title represents the pattern *name* and the map description as well as the pre-conditions associated to the start points express the *context* and the *problem*. The

sequence of responsibilities graphically represents a possible scenario that describes common causality relationships between the functionalities documented in the pattern *solution*. *Resulting context* is represented by the post-conditions associated to the end points. *Forces* describe the tension that makes the solution good for the design problem. Although forces are not represented with the UCM notation, they are essential to determine whether a common solution to a recurrent problem, which is captured when investigating the chosen systems, corresponds to a pattern or not. *Related patterns* within the MoRaR pattern language are shown in the UCMs that describe the relationships between the patterns (see Figure 2). The former helps to understand why the pattern solution works and should be applied in a particular system. The latter shows the relation with other patterns that have been presented in the literature. UCMs do not describe *rationale* that comes from the motivation for the use of each pattern. *Known uses* are the chosen systems mentioned in Chapter 4, for instance, second generation systems such as GSM, ANSI-41, and Wireless mobile ATM networks, which are outlined in the next section.

Chapter 4 has shown that most requirement documents used in this research to capture common behaviors are expressed with text (informal description), tables, and information flows and there is a need for finding a better way to express them. **U**se **C**ase **M**ap**s** (UCMs) are chosen in this research to express system requirements and analysis models as well as the capture approaches. However, as pointed out in [153], pattern descriptions are more efficient when complemented with a text version since they should be understood for both end users and developers. In order to overcome the problems caused by the informality of each pattern textual description, we adopt UCMs and basic **M**essage **S**equence **C**harts (MSCs) to visually describe the behavioral pattern solutions introduced in the next sections.

The UCM graphical specification takes a somewhat simpler view of the pattern solutions, in the sense that they are used to capture common causality relationships between the behavioral pattern solutions. Such relationships are apparent from the design of the chosen systems mentioned in Chapter 2. These causality relationships constitute basic documentation for existing systems and are the source of initial design decisions for new systems as presented in Chapter 6.

Furthermore, even though the MoRaR patterns are not presented in the object-oriented paradigm, designers should be able to apply object-oriented approaches and languages to implement them. Chapter 6 presents an approach for pattern reuse and validation that adds rigor to the pattern reuse and validates requirements, analysis, and design models against validation test cases that are derived from the pattern solutions.

Reusable Software	**Development Stage**
Behavioral Patterns	Requirements
Structural Patterns	Analysis

Table 9 Reusable Units of the MoRaR Patterns

Table 9 presents an overview of the MoRaR patterns in terms of its reusable software units and their correlation to the software development stages. These units can be reused at the requirements and analysis stages.

5.3 The MoRaR Pattern Language

After identifying behavioral and structural patterns from the commonalities presented in Chapter 4, the motivation for designing a pattern language arises from the need of showing the interactions among them. The pattern language for mobility and radio resource management gives designers the possibility of generating different scenarios related to mobile systems.

The pattern language facilitates a concise description of mobility and radio resource management functions suitable to be adapted to different mobile systems at the requirements and analysis stages. The pattern language gives an overview of the pattern relationships. Designers are then able to find the patterns that are relevant for what they intend to do. It should be mentioned that the pattern language shows alternative ways to combine patterns (see case studies in Chapter 7).

Behavioral and structural patterns are gathered in the MoRaR pattern language are general and abstract enough to allow freedom with respect to future implementation decisions and to be re-used at the early stages of the system development process and evolution of mobile systems.

Figure 40 depicts the pattern language with the behavioral (ovals) and structural (plain rectangles) patterns classified into two categories (dashed rectangles) as follows: mobility management and radio resource management. In short, these two categories describe the functional layers discussed in Chapter 2 that are often used in the literature when discussing protocols for mobile communication systems. In addition to a name, each pattern has a number that identifies the sub-section in which it is discussed. The dashed and plain black arrows illustrate the relationship among the patterns. The gray arrows with outgoing call, incoming call, and roaming labels depict three possible start points for scenarios derived from the pattern language.

Dashed arrows represent the exchange of the following information requests between the behavioral and the structural patterns: a request for data items, a request for operations on data items, or a request for control or release of resources. *Location registration* requests the *home and visitor databases* to store, update, and delete data items related to the mobile users' location. On the other hand, *paging*, *temporary identification* assignment, *authentication*, and *inter-system handoff execution* request data items that are stored in the databases. In addition, the *home database* requests data items that are stored in the *security database* and vice-versa. The *anchor mobile switching center*, which controls the resources (e.g., *releasing resources*) for the *inter-system handoff execution*, is triggered by a request from the *handoff decision* or from the *location registration*.

Plain arrows represent the order in which the patterns occur. A designer chooses either a set of patterns or an individual pattern that best suits the system needs. In other words, the first pattern in a sequence or a single pattern is chosen according to the specific scenario that the designer wants to generate (e.g., outgoing call, incoming call, or roaming in the figure). The context of each pattern in a specific sequence is related to the resulting context of the previous pattern. As mentioned earlier, the following systems use these patterns: GSM/GPRS, ANSI-41/WIN, and IMT-2000 based systems as well as WmATM networks.

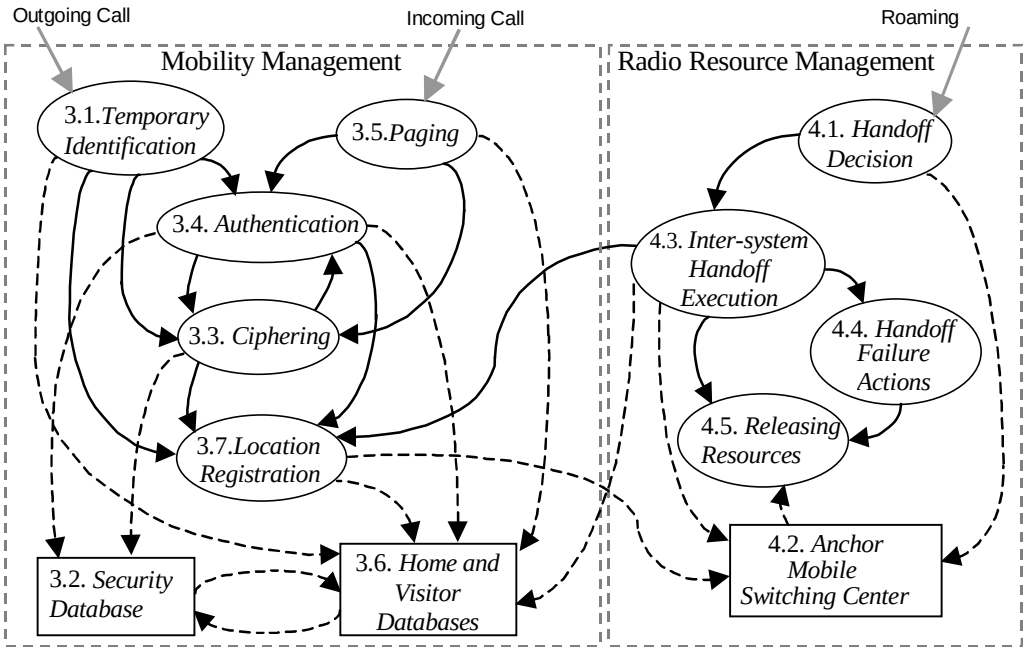


Figure 40 Relationship among the Patterns within the MoRaR Pattern Language

For instance, a scenario with the sequence of patterns that describe what happens when a user powers on the mobile station and tries to make a call starts with an outgoing call request shown in Figure 40. First, the network requires the *temporary identification* of the mobile station, then, *authentication* and *ciphering* provide a secure environment for the communication. The *security database* separates the user's authentication information from the user's profile and it is accessed during the previous two steps. After this, a new *temporary identification* is assigned to the mobile station and the *location registration* updates the *home and visitor databases* that keep track of the mobile user's current location information whenever the user roams.

When an incoming call (see gray arrow in Figure 40) arrives to a mobile station that is powered on, *paging* is performed to reach the mobile station before *authentication* and *ciphering* that guarantee security and privacy for the establishment of the connection between the users. A *temporary identification* can be assigned to the mobile user to avoid sending the real user's identity through the air interface.

Meanwhile, a *handoff decision* monitors the quality of the link between the mobile station and the network whenever a user moves from one location to another (see the roaming gray arrow in Figure 40). The *inter-system handoff execution* guarantees the communication when the roaming occurs. However, unsuccessful handoff outcomes also occur in this case and *handoff failure actions* handle them. The ability to release resources (*releasing resources*) after a handoff is also supported by the network. As presented earlier, an *anchor mobile switching center* maintains the control of the resources (e.g., allocation and release of transmission links) during the call processing.

Section 7.2 of Chapter 7 presents a framework that graphically describes the MoRaR pattern language with UCMs. This framework includes not only mobility and radio resource management functions (commonalities) but also variabilities such as communication management functions and other network entities presented in Chapter 4 and not described as patterns.

The next sections introduce mobility and radio resource management patterns, which are used in different second and third generation systems as stated earlier and are suitable for requirements and analysis models. It is important to mention that pattern names are given in sub-section headings. When we reference patterns that are included in the MoRaR pattern language, we identify them within the pattern context sub-section or the pattern resulting context sub-section. However, a separate sub-section describes other related patterns to *ciphering*, *authentication* and *location registration*.

5.4 Patterns Related to Mobility Management Functions

Mobility management functions solve problems related to the user's security and location. For instance, mobile communication systems use wireless technologies, such as the Time Division Multiple Access (TDMA) technique, that are by nature more prone to eavesdropping and fraud on radio interfaces than fixed networks. Furthermore, in such systems, other facilities are required to manage location aspects related to users that roam from cell to cell and to reach a user that is being called. This contrasts with the situation in fixed systems where the location of the user (or the user's terminal) is always known since it is associated with the subscriber's number.

The next sub-sections present patterns that solve mobility management problems such as: to guarantee security and privacy (*temporary identification*, *security database*, *ciphering*, and *authentication*); to reach a mobile user that receives an incoming call (*paging*); to keep a record of the subscriber's location information that enables the establishment of calls efficiently (*home and visitor databases*); and to keep up to date the location information of the mobile station (*location registration*).

5.4.1 Temporary Identification

Context

A user has just powered on a mobile station, traveled to a new location area, or an incoming call has just arrived to a mobile user. In these cases, common control channels are used for network management messages before the establishment of a dedicated control channel between the mobile station and the network [160]. At this time, privacy and security operations are the main concerns of the network in order to protect the communication over the air interface against illegal scanning of these control channels [79].

Problem

How does one ensure privacy of the subscriber's identity when sending it on the radio path?

Forces

- All the information exchanges in clear on the radio path are vulnerable to a third party listening to the control channels. As a result, the subscriber's identity can be easily captured on the air interface. Then, a fraudulent mobile station can be programmed with this valid identification and can make calls at the original mobile station's expense (known as mobile cloning fraud);

- Although encryption is effective at providing confidentiality, it is not possible to protect every single information exchange on the radio path, for example:

  – when common channels are used simultaneously by all mobile stations in the cell and in the neighboring cells, *ciphering* (Section 5.4.3) is not applied since a key known to all mobile stations has a low level of security;

  – when a mobile user moves to a dedicated channel, there is a period during which the subscriber's real identity is unknown to the network and ciphering methods are not applied.

Solution

Assign a temporary identification to the mobile station in order to avoid exchanging the subscriber's real identity and the electronic serial number of the mobile station over a non-*ciphering* (Section 5.4.3) radio path.

Each mobile station has a unique temporary identification that is composed of a location area identity and a digit string. The temporary identification, which is dynamically allocated by the network when the mobile user registers in the location area, is stored in the mobile station and in the *visitor database* (Section 5.4.6). When a mobile user powers off the mobile station or changes to another location area, this identification

is released in the old *visitor database*. The network reduces signaling messages and resources by storing the temporary identification in the *visitor database*.

Figure 41 illustrates two scenarios that represent, respectively, a *temporary identification* inquiry and a *temporary identification* assignment. The former can be performed by these systems when a mobile station receives a call or changes location area. The latter is used by GSM, ANSI-41-C, and IMT-2000 based systems when a mobile station powers on or tries to make a call. When the mobile station changes location, in addition to the temporary identification assignment, the old temporary identification is released from the old *visitor database*. The figure also illustrates a bound UCM that corresponds to the temporary identification assignment scenario shown in Figure 33a of Chapter 4 mapped to the common architectural elements.

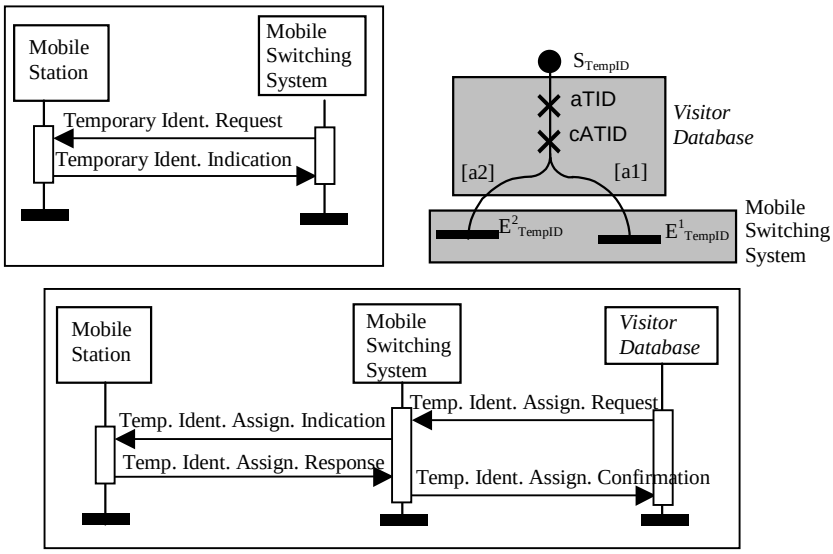


Figure 41 Temporary identification Inquiry and Assignment

Rationale

Temporary identification adds an extra level of protection in mobile wireless environment. It is used instead of the real subscriber's identity when the user powers on a mobile station or tries to make a call and it has been previously assigned by the network. The advantage of the use of this identity is observed when *ciphering* (Section 5.4.3) is not applied to the traffic. In this case, even if someone is listening to the radio path, this identity does not have any meaning outside the serving network (e.g., an illegal mobile station cannot be programmed with this number).

Resulting Context

The *temporary identification* protects the mobile user and the network from third parties that could otherwise get information about the subscriber's real identity by listening on the radio path. Once the temporary identification has been assigned to the mobile station, the user identity can be validated through *authentication* (Section 5.4.4) and the

exchanged information over the dedicated channel can be secured through *ciphering* (Section 5.4.3).

Known Uses

Temporary identification is called Temporary Mobile Subscriber Identity (TMSI) in the location management procedures of GSM/GPRS and ANSI-41 Specifications and Temporary Mobile User Identity (TMUI) in the TMUI assignment information flows of IMT-2000 Systems and UMTS.

5.4.2 Security Database

Context

Security and privacy management functions handle information such as authentication keys and security-related parameters. Furthermore, they check the validity of received authenticated data, and perform confidentiality controls.

On the other hand, location management functions manage mobile users' location information as well as their identification and profile that are relevant to the provision of telecommunication services. This information is stored in the *home and visitor databases* (Section 5.4.6).

Problem

How does one handle the mobile user's sensitive information while assuring its protection on the network side?

Forces

- All the security mechanisms including keys and algorithms should be a concern for operators and manufacturers of mobile systems. For example, *ciphering* (Section 5.4.3) and *authentication* (Section 5.4.4) not only rely on the secrecy of the information that are provided by them, but they also rely on the secrecy of their keys and algorithms;

- Even though security management functions involve the same protocols and architectural elements as location management functions, the location information and user profile are often accessed by the network while performing *paging* (Section 5.4.4) and *location registration* (Section 5.4.6). Consequently, the information in the *home and visitor databases* is vulnerable to attacks and failures due to this frequent access;

- It is also not possible to store the security-related information only in the *visitor database* since this database is in charge of storing temporarily information related to subscribers who are currently in its location area.

Solution

Create a repository of the user's sensitive information that is only accessed by functions involved in the security management process. This database does not transmit any sensitive information (e.g., secret keys and algorithms) but performs the *ciphering* and the *authentication* computations itself.

Figure 39 of Chapter 4 illustrates the security database in a mobile environment. The authentication center is the security database for ANSI-41 and GSM based systems (see Chapter 2). The same secret keys and algorithms are permanently stored in the internal memory of the mobile station when its activation occurs. In GSM, the Subscriber Interface Module (SIM) stores this information.

The *home and visitor databases* (Section 5.4.6) have small roles during the security management process. For example, they store complementary information that is necessary to perform the authentication and ciphering computations.

Rationale

The amount of security information in a wireless environment such as keys and algorithms that are used to perform ciphering and authentication calculations justifies their separation from the user profile information stored permanently in the home database. Any problem that home databases face will not affect the *authentication* or *ciphering* procedures. This decision gives an extra level of security to any mobile system.

Resulting Context

Sensitive data has been stored in the *security database*, which provides an additional layer of protection around this information. Consequently, the *ciphering* (Section 5.4.3) and the *authentication* (Section 5.4.4) functions use this information in order to guarantee privacy and security to the mobile user.

Known Uses

Security Database is called Authentication Center (AC) in ANSI-41 Specifications, IMT-2000 Systems, and GSM/GPRS/UMTS.

5.4.3 Ciphering

Context

In the idle mode, common control channels are used for all mobile stations that are in a particular cell as well as for the ones in the neighboring cells. Once a mobile station sends its *temporary identification* (Section 5.4.1) to the network, a dedicated control channel and a traffic channel, which is reserved for user information, are allocated. As a result, the mobile user changes from the idle mode to the dedicated mode.

After this, the type of information transmitted through the radio path belongs to different categories, such as: user information (voice or data), user-related signaling (messages carrying user's identity numbers), and system-related signaling (messages carrying radio results measurement).

Problem

How does one protect the privacy of the communication over an insecure wireless communication channel?

Forces

- When the information is sent in clear text on the radio path, third parties are able to eavesdrop the communication;

- A good protection against unauthorized listening is not easy with analog transmission, which is used by first generation systems such as AMPS. For instance, analog mobile systems have to apply more than one mechanism to encrypt selected parameters in the signaling messages or to encrypt the user traffic. However, digital transmission, which is used by the second generation systems such as GSM, provides privacy and security to mobile subscribers by protecting all the transmitted information (e.g., voice, data, and signaling);

- Encryption of the same data must not generate the same encrypted sequence on the network each time to prevent replication fraud (e.g., play back of the encrypted sequence from a previously intercepted sequence).

Solution

Apply encryption mechanisms to the digital information that is transmitted through control and traffic channels when the mobile station is in the dedicated mode. These mechanisms are independent of the exchanged information type.

Figure 42 depicts two scenarios that involve the setup of the ciphering mode and the transmission of the ciphering information in IMT-2000 systems. In Figure 42a, the network is instructing the mobile station that the *ciphering* mode should be employed during the transmission process. Figure 42c illustrates the mobile station sending an encrypted data stream on the radio interface. The bound UCM that corresponds to the ciphering data exchange scenario is also shown in Figure 42b and corresponds to the unbound UCM depicted in Figure 33b of Chapter 4.

Before setting up the encryption mechanisms as shown in Figure 42a, the mobile station and the network have already agreed on the inputs that allow the ciphering and deciphering methods, which are the following ones: a frame number, an encryption algorithm, a ciphering key. The frame number, which is provided by the network for each ciphering sequence, prevents the replication fraud. The encryption algorithm, which is specified for use in several countries, generates the ciphering sequence. The ciphering key

is calculated for each communication according to a random number, a computation algorithm, and a secret key. The mobile station, the *home and visitor databases* (Section 5.4.6), and the *security database* (Section 5.4.2) are responsible for storing the inputs and calculating the operations mentioned previously.

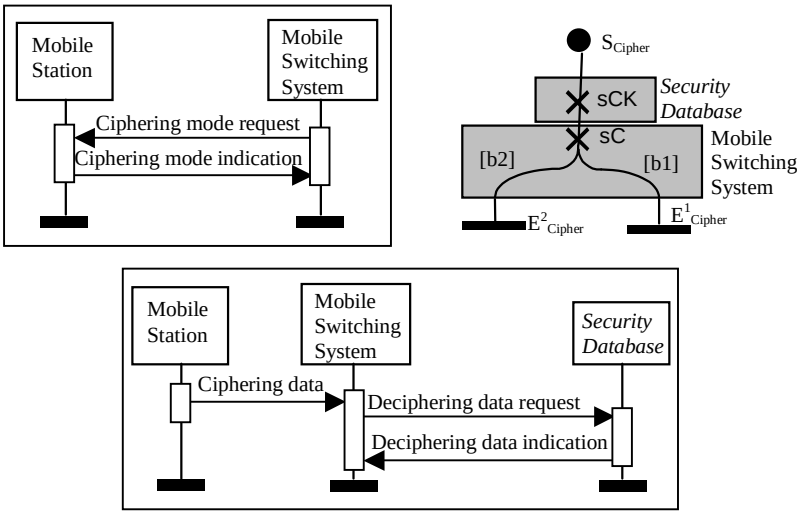


Figure 42 Ciphering Mode Setup and Ciphering Data Exchange

In order to ensure that the ciphered data from the mobile user's side can be deciphered on the network side, the encryption algorithm should be reversible. For instance, if the encryption algorithm is used to cipher a data stream, the same algorithm is used twice to decipher this data and get back to the original stream.

Rationale

Wireless environment does not bring any protection against eavesdropping. This is the reason for the need of ciphering the traffic in the air interface. Ciphering provides algorithms to encrypt signaling messages and data that are exchanged in the air interface.

Resulting Context

In the dedicated mode, encryption mechanisms provide an enhanced degree of privacy over the radio channel by preventing unauthorized access to the information exchanges. For instance, when an incoming call has been requested through *paging* (Section 5.4.3), *ciphering* is applied to assure privacy of the information exchanges between the network and the user who is being called.

In addition to ciphering, the mobile user's authentication (Section 5.4.4) should be performed to prevent fraudulent access.

Known Uses

Ciphering is used in the authentication and signaling message encryption procedures of the ANSI-41 Specifications, in the authentication management of the IMT-2000 Systems, in the ciphering procedures of GSM/GPRS/UMTS, and in the overlay registration of the WmATM networks.

Related Patterns

Secure-channel communication, *information secrecy*, *secrecy with sender authentication*, and *secrecy with signature* are presented in [44] within a pattern language for cryptographic software.

Information secrecy has the same purpose as *ciphering*. In other words, both patterns support encryption and decryption of data. However, *ciphering* documents specific characteristics of mobile systems and concentrates on the requirements and analysis stages while *information secrecy* focuses on the design stage of cryptographic software, which are used in different domains. *Secrecy with sender authentication* and *secrecy with signature* are modifications of the *information secrecy* with the addition of *sender authentication* (see related patterns of Section 5.4.4) and *signature*, which guarantees the authorship of the message (non-repudiation objective of cryptographic process).

5.4.4 Authentication

Context

A *temporary identification* (Section 5.4.1) has been requested from a mobile user that has powered on a mobile station, or entered a new location area. Once the network has provided a dedicated channel, *ciphering* (Section 5.4.3) prevents the eavesdropping of communications through the radio path. Nevertheless, transmissions could have been intercepted before using the temporary identification or enabling the ciphering methods whenever a *location registration* (Section 5.4.6) has occurred or when a mobile user has received an incoming call request through *paging* (Section Figure 44). As a result, the stolen identification codes could have been used to obtain airtime fraudulently.

Problem

How does one prevent unauthorized or fraudulent access to cellular networks by mobile stations illegally programmed with a counterfeit identification and electronic serial number?

Forces

- When the subscriber's identity is transmitted without any form of encryption over the air interface and special radio scanners capture this information, the valid user's identity (and the user's account) can be illegally used by someone else;

- Without verifying with a strong degree of confidence the identity of the mobile station, any fraudulent mobile station programmed with a valid subscriber's

identification can make calls, which lead to incorrect billing to the legitimate user, or receive calls as if it were the original legitimate mobile station;

- Passwords only limit physical access to the mobile station, but are of little value when sent over an open channel on the air interface;

- A robust method of validating the true identity of a subscriber in a wireless environment requires no subscriber intervention and no exchange of keys and algorithms through the air interface.

Solution

Perform an authentication operation in both the mobile station and the network sides based on an encryption algorithm and a secret key number. These values are stored in the legitimate mobile station and in the *security database* (Section 5.4.2) at the initiation of the service (e.g., at the mobile station activation). Furthermore, they are neither displayable nor retrievable and never transmitted over the air or passed between systems.

The operation consists of applying the encryption algorithm with the following inputs: a random value dynamically provided by the network, the secret key number, an electronic serial number, which identifies the mobile station, and the user's identification number. According to the comparison of both authentication results, the mobile station is either authorized or denied access to the network. The user's identification number is sent over a *ciphering* (Section 5.4.3) radio path and the electronic serial number has been previously stored in the *home database* (Section 5.4.6).

Figure 43 illustrates a typical mobile wireless communication environment with a scenario that shows the authentication operations performed by a mobile station and its home network provider. The architectural elements shown in the figure are related to the following scenario: a mobile user powers on a mobile station inside the home location area and an *authentication* is requested.

First, the network sends to the mobile station a random value from a pool of free numbers. The mobile station performs the authentication operation based on the encryption algorithm with this random value and the inputs assigned previously. The operation result is sent to the network. The comparison between these results (respectively, UsrAuth and NetAuth) is made at the network side with the following network entities involved: mobile switching center, *home database* (Section 5.4.6), and *security database* (Section 5.4.2). The mobile user is successfully authenticated and able to access communication services in case of identical results. Otherwise, the access is denied.

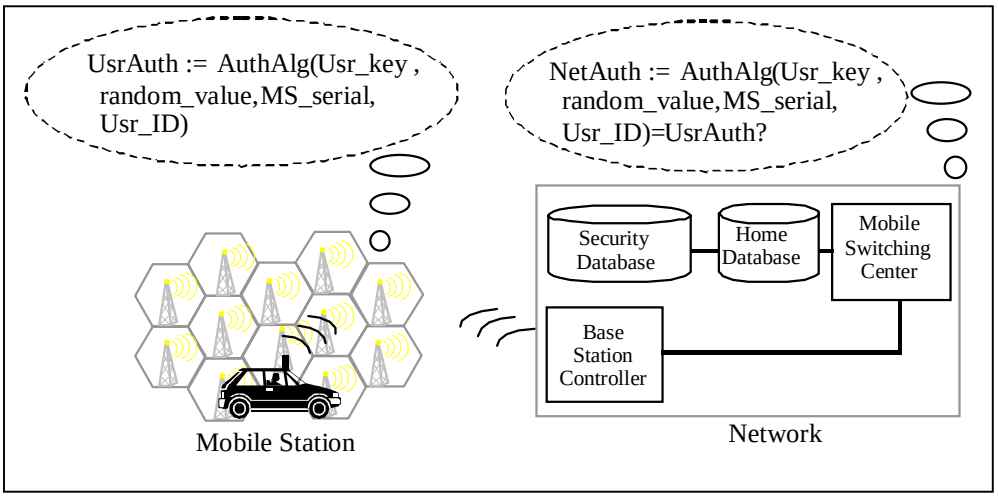


Figure 43 Authentication Operations

The message sequence chart shown in Figure 44 illustrates the architectural elements of an ANSI-41-C based system that are involved in the authentication of the mobile station shown in Figure 43.

Rationale

There is a need for verifying the authenticity of every mobile station that tries to access a network to avoid problems such as mobile cloning. Different from the fixed networks, in a wireless environment, extra protection is also necessary to avoid exchanging security information over the air interface when ciphering is not available. *Authentication* combines the verification of the mobile station's identity in the network side with the exchange of random numbers through the radio ports instead of the mobile user's identity.

Resulting Context

The *authentication* operation has protected the network against unauthorized access and the mobile user from fraudulent impersonations by certifying her identity. As a result, a secure mobile communication environment is offered to the subscribers before the user's *location registration* (Section 5.4.6), before an attempt to make a call, or when a mobile user has received an incoming call request through *paging* (Section Figure 44).

Known Uses

On the basis of the air interface protocol (e.g., IS-95 or IS-136), which is used to access mobile systems, or of a roaming agreement between the home and the serving systems, it is possible to determine whether the MS is authentication-capable or not. The ANSI-41-C mobile application part (MAP) contains operations and algorithms that are responsible for the authentication tasks [79]. The authentication operations are done by a set of algorithms called Cellular Authentication and Voice Encryption (CAVE) and two secret keys: A-key and SSD (see also Section 2.5).

Related Patterns

Secure-channel communication, *sender authentication*, and *secrecy with sender authentication* are presented in [44] within a pattern language for cryptographic software. *Sender authentication* has the same purpose as *authentication*. In other words, both patterns guarantee that the sender is genuine and authentic. However, *authentication* documents specific characteristics of mobile systems and concentrates on the requirements and analysis stages while *sender authentication* focuses on the design stage of cryptographic software, which are used in different domains. *Secrecy with sender authentication* is a design pattern that combines *information secrecy* (*ciphering*) and *sender authentication* (*authentication*).

In addition, *authenticator* is presented in [46] to describe identification and authentication mechanisms for distributed object systems. This design pattern uses a login-and-authenticate protocol to grant or deny access to individual requestors. Although the authenticator and the authentication intents are related, these patterns have differences regarding their problems, solutions, resulting context, and applicability.

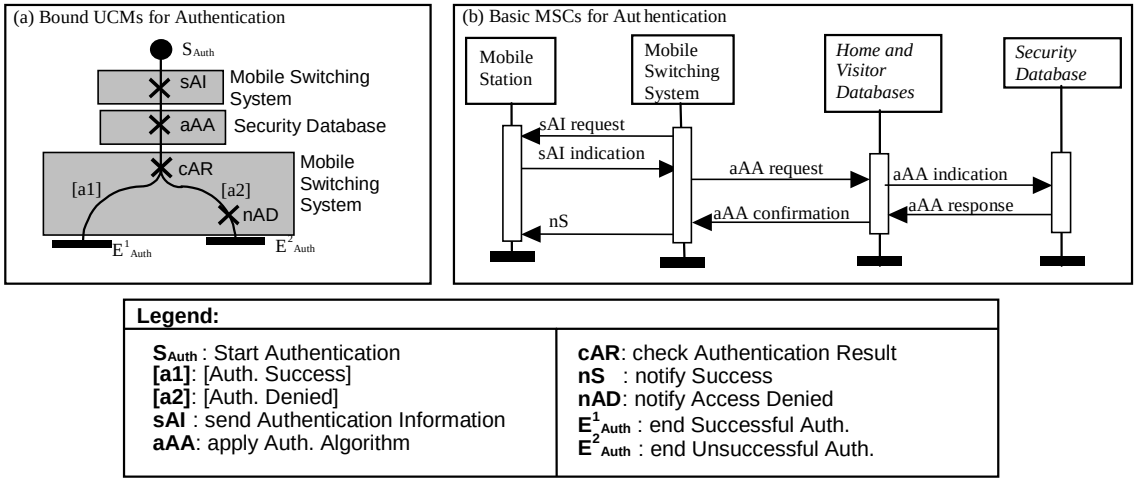


Figure 44 Authentication: Bound UCMs and ANSI-41 Message Sequence Charts

5.4.5 Paging

Context

A network has received an incoming call request that is addressed to a mobile user whose current location area is kept by the *home and visitor databases* (Section 5.4.6). This request contains the mobile user's identity (dialed number). The network refers to the user as the terminating or called party.

Problem

How does one reach the terminating mobile user and route the call to the actual location of the mobile station?

Forces

- The precise location of a terminating mobile station needs to be known in order to establish the communication;
- In a fixed telecommunication environment, the user's location is always known since it is associated with the subscriber's number. On the contrary, in a mobile network, the dialed number has no information about the terminating user's current location that changes with the user's wanderings;
- The mobile environment is split into cells and each location area includes several cells managed by a single mobile switching center. When a user roams within a location area, this user eventually changes cells;
- When an incoming call request is sent to the network, *home and visitor databases* (Section 5.4.6) are queried about the terminating user's current location area. However, each location area is composed of several cells and the current cell where the mobile station is camped on needs to be found by the network;
- The amount of information transmitted and processed by the network increases considerably if smaller cells are used. For instance, when a large number of mobile users are transmitting information about their location through every cell, both base stations and the spectrum in use by them are overloaded.

Solution

Send a paging message to reach the terminating mobile user only in the cells within the user's current location area (for example, several dozen cells). The location area information is retrieved from the *home and visitor databases* (Section 5.4.6), which are kept updated by the *location registration* (Section 5.4.7). Based on the mobile station reply, the network precisely knows the cell where the terminating user is currently located.

Figure 45 shows a successful *paging* scenario after the network receives an incoming call request.

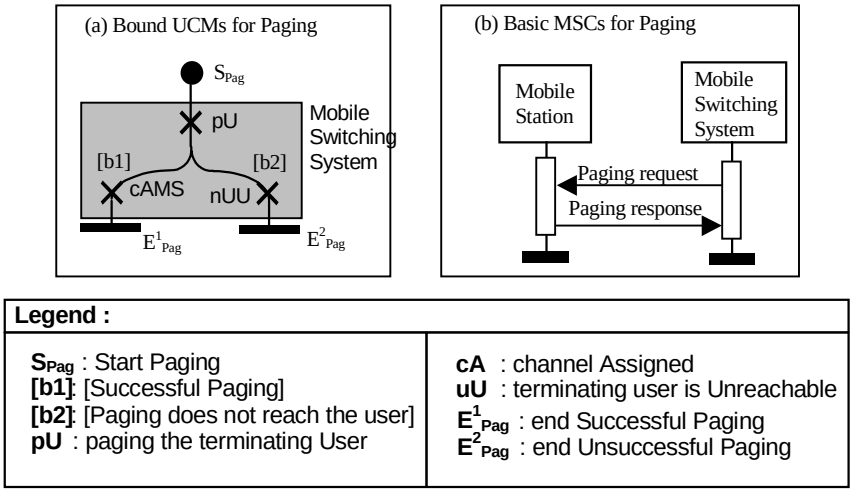


Figure 45 Paging: Bound UCMs and Message Sequence Charts

Rationale

When there is an incoming call for a mobile user, it is not possible to locate the user on the basis of the dialed number. The location of the terminating user is not included in the number since it is not fixed. *Paging* solves the problem of locating the terminating user with the information provided in the user profile of the home database.

Resulting Context

Once the network has found the terminating mobile user and allocated a dedicated channel, the call establishment proceeds. The terminating user's *authentication* (Section 5.4.4) and *ciphering* (Section 5.4.3), which is used to protect the information on the radio path, can also follow *paging* depending on implementation decisions.

In large networks, this solution provides a balance between the amount of location information to be exchanged among architectural elements and the number of necessary *paging* messages to be sent on the radio path. Besides locating the user, the paging procedure minimizes resource consumption regarding both the signaling load on the radio path and the processing load on the network.

Known Uses

ANSI-41 Specifications, IMT-2000 Systems, and GSM/GPRS/UMTS, have *paging* in their specifications. *Paging* is also part of connection establishment in the overlay originating party procedures of the WmATM networks.

5.4.6 Home and Visitor Databases

Context

In the mobile wireless network world, the capability of users moving from one location area, which groups a set of cells, to another is called “roaming”. As a result of roaming, users change to location areas other than their own home area.

Problem

How does one enable a user’s mobility between location areas of the same provider or between location areas of different providers?

Forces

- Different from fixed users’ identification, a mobile user’s identification includes no information about the subscriber’s current location;
- Mobile communication providers are responsible for keeping information about the users permanently registered in their networks as well as about the ones currently visiting them;
- Information about mobile users’ home, previous, and current locations are essential to enable telecommunication services when one or several providers are involved in the roaming;
- Telecommunication services are only obtained from a current visiting location area if a mobile station is registered in this particular area. This registration relies on the home and previous location area in order to get information about the mobile user.

Solution

Create two types of repositories to handle the mobile user’s information. One is the home database, which is a primary repository for information (such as current location) about a set of users permanently registered in a location area. The other repository is the visitor database, which temporarily stores part of the information (e.g., home location) about the users that are currently visiting a particular location area. The visitor database reduces the signaling messages to the home database when the user is roaming.

Every time a mobile user registers in a new location area, the *location registration* (Section 5.4.6) procedure updates the home, previous, and current location information.

Figure 39 of Chapter 4 illustrates the home and visitor databases in a typical mobile environment. Chapter 2 shows GSM, ANSI-41, and IMT-2000 based systems, which maintain a home database that is called Home Location Register (HLR) and a visitor database that is called Visitor Location Register (VLR). The GSM-900 VLR is physically integrated with the Mobile Switching Center (MSC).

Rationale

Databases are used in fixed networks to keep information about their subscribers such as billing information and features. In mobile systems, these databases are still necessary and they also have to keep information about user's current location area, *temporary identification,* and security information, among others. There is information about the user to keep permanently in a database and information to keep temporarily. The permanent information is stored in the user's home network that can be far from the current user's location. The temporary information is often accessed by the network and should be close to the current user's location. *Home and visitor databases* solve these problems.

Known Uses

ANSI-41's specifications describe the *visitor database* as the VLR functional entity. The ANSI-41serving system is described as a single entity composed of the MSC and VLR functional entities. Most of the ANSI-41 implementations currently available in service also describe a single MSC/VLR; however, there is potential for their separation at the implementation level.

Resulting Context

Home and visitor databases keep track of users' information through *location registration* (Section 5.4.6). As a result of having these databases, the roaming capability is guaranteed and the restriction of offering services only in a specific area within a particular network is removed. In addition, the *visitor database* stores the *temporary* identification (Section 5.4.1) as well as part of the user's profile information.

The mobile user's location information is needed for *authentication* (Section 5.4.4) purposes and for *paging* (Section Figure 44) the terminating mobile user. Furthermore, the *security database* (Section 5.4.2) requests information from the *home and visitor databases* in order to perform the *authentication* and *ciphering* calculations, whose results are requested by the *home and visitor databases*.

5.4.7 Location Registration

Context

A user has changed location area in a mobile wireless environment that contains *home and visitor databases* (Section 5.4.6). The user's location has changed as a consequence of a power-on event[1], an outgoing call, or an incoming call. Optionally, an *authentication* operation (Section 5.4.4) has been successfully performed.

[1] A mobile user changes location area before powering on the mobile station.

Problem

How does one keep up to date information about a mobile user's location every time the user changes location area?

Forces

- A visitor database has limited storage capacity that is easily overloaded with a large number of mobile users roaming in its location area;
- The accuracy of the location information is necessary in order to offer telecommunication services such as information exchange between users;
- The location information in the previous *visitor database* is no longer up-to-date or useful when the mobile user moves to a new location area.

Solution

Perform a location registration procedure that consists of updating and inserting the mobile user's location, respectively, in the *home and* current *visitor databases* (Section 5.4.6) every time the mobile user changes location area. This registration operation also includes a request message to the previous *visitor database* in order to delete the mobile user's temporary location record.

Figure 47 illustrates a location registration in ANSI-41-C networks. This scenario considers that the mobile user's home network is different from the previous location area (old network). The user is registered in a new network (new location area).

A location registration failure occurs in two cases: the mobile user's previous location information is not reachable or the mobile station does not support the new area for technical reasons, such as network failure. As a result, the location information is not updated and the network notifies the mobile user.

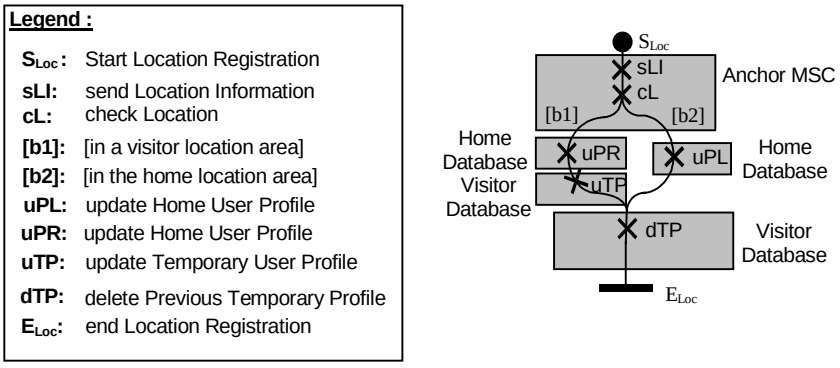


Figure 46 Location Registration: bound UCMs

Rationale

In order to handle the user's mobility, there is a need for updating the *home and visitor databases* every time a user enters in a new location area. Without this constant operation, it not possible to complete calls or offer services to mobile users. *Location registration* not only takes care of this problem but also maintains the *visitor database* with a temporary information about the users that are currently roaming its respective location area.

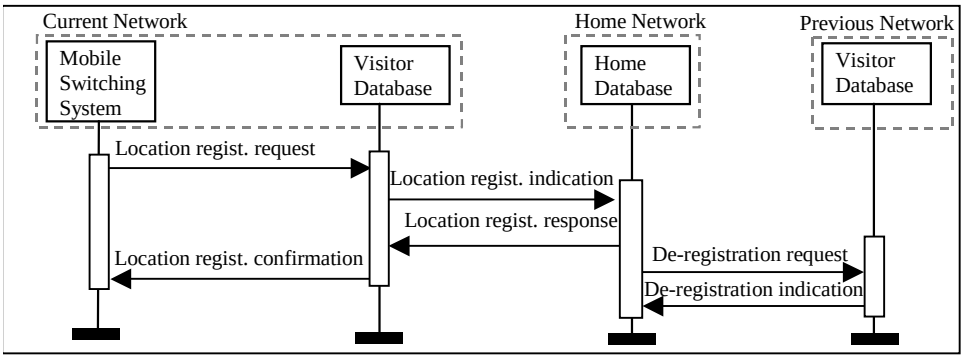


Figure 47 Location Registration: Message Sequence Chart

Resulting Context

When the current mobile switching center has successfully registered a mobile station in the new location area, the *home database* contains the current location of the mobile station and the current *visitor database* contains a temporary record of the mobile station. The mobile station is allowed to camp in this area, as well as to request services, to establish, and to receive calls. In addition, the temporary record containing out-of-date location information has been deleted from the previous *visitor database*. As a result, storage resources have been released.

On the other hand, information about user location does not change in case of location registration failure.

Known Uses

The triggering events for location registrations are dependent on the protocol used for the air interface and on the internal algorithms implemented in the serving systems [79]. However, the air interfaces standards for AMPS, CDMA, and TDMA registrations support the following common registration events: mobile station power-on, timer-based (i.e., registration occurs at periodic intervals while the mobile station is powered on), transition to another system, and call origination. At a high level of abstraction, the actions involved in the location registration are common to ANSI-41 (registration notification and location cancellation procedures) and GSM based systems (location management), and WmATM networks (overlay registration procedure).

Related Patterns

The *Parameter Database* [184] that is described within a *Pattern Language of Feature Interaction* solves the problem of two or more features accessing and modifying the same database parameter. This pattern language handles similar feature interactions occurring in telecommunication services (for fixed and mobile users).

5.5 Patterns Related to Radio Resource Management Functions

In a mobile wireless communication environment, the network controls the provision of a dedicated channel to the mobile station over the radio interface (see Section 2.3.2.3 of Chapter 2). The main concern of the network is how to maintain this dedicated channel despite the wanderings of the users. A handoff function is responsible for this maintenance [78][86][113][139].

As mentioned in Figure 38 of Chapter 4, base station transceivers and base station controllers are important components of the handoff process; however, this research considers only the handoff that generates network traffic and involves different mobile switching centers (called inter-system handoff). At the upper layers, the inter-system handoff is managed by mobile stations (MSs) and mobile switching centers (MSCs). Base stations act as complex transmission systems. This handoff requires specialized signaling protocols between the current and the candidate mobile switching centers involved.

The next sub-sections present patterns that solve radio resource management problems as follows. A *handoff decision* is taken every time it is appropriate to change a radio communication link. The execution of an inter-system handoff maintains the stability of the dedicated channel despite the user's movement. An *anchor mobile switching center* handles the resources for the exchanges of information during the inter-system handoff process. The *handoff failure actions* are responsible for handling unsuccessful outcomes during the *inter-system handoff execution*. Radio resources are minimized by *releasing resources*.

5.5.1 Handoff Decision

Context

A dedicated radio communication channel has been assigned between a mobile station and a mobile switching center. The mobile user is roaming from one place to another. This possibility of changing cells (and possibly location area) is the major source of complexity for mobile networks [79][139].

Problem

How does one control the quality of the radio communication link between the mobile station and the network?

Forces

- The radio communication is interrupted as soon as the user leaves the radio coverage area of the current cell whether a call is in progress or not;
- When a call is in progress, the radio communication cut-off has an important weight in the overall perception of quality from the user's point of view;
- When comparing the capability of two cells, the load of each base station transceiver and the overall interference level in each cell affect the radio link quality;
- Local geographic peaks can occur in events such as sport competitions, concerts, and festivals. It is possible in these situations that a cell is congested in the peak area while its neighbor cells are not.

Solution

Take measurements on the quality of the transmission for ongoing dedicated radio connections, check the transmission quality on the neighbor cells, and verify the load of the base station transceiver. Then, decide whether a handoff should be triggered or not. This decision is based on the received signal measurements concerning the current cell and when it is necessary, the neighbor cells.

These measurements are best taken by the current base station or both the current base station and the mobile station with parameters, such as: transmission error rate, the propagation path loss, the propagation delay, traffic considerations, as well as the cell capacity and load. Mobile stations provide measurements of the received base station signal strength to the current mobile switching center. Adjacent base stations also provide measurements to the current mobile switching center while the mobile station moves towards their coverage areas.

The handoff decision can be taken by the mobile station (MS-controlled handoff), the serving MSC (network-controlled handoff) or both (MS-assisted handoff - MAHO) depending on implementation issues.

Figure 48 depicts a possible solution for applying a *handoff decision* (see unbound UCMs in Section 4.7.2 of Chapter 4). In this scenario, the MSC decides whether the handoff is necessary or not. As stated earlier, two or more mobile switching centers (MSCs) are involved in an inter-system handoff. On the other hand, when the handoff involves two radio channels that are controlled by the same MSC (intra-system handoff), the base station controller takes the handoff decision. The map starts ($S_{HDecision}$) when the user moves to another location area and the *handoff decision* is taken according to measurements (tM and cM responsibilities). The alternative paths represent different outcomes. For instance, the endpoints represent the need of having an inter-system handoff or not (respectively, $E_2^{Hdecision}$ or $E_1^{Hdecision}$).

Rationale

While a mobile user roams, there should be a process to monitor the quality of the current link and the surrounding links in order to guarantee that a user can make calls, receive calls, and access other network services.

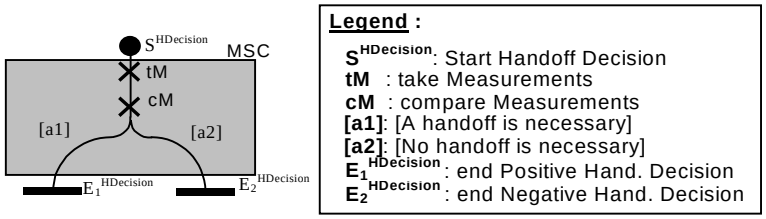


Figure 48 A Solution for *Handoff Decision* with bound UCMs

Known Uses

The handoff decisions taken by the signaling alternatives for WmATM networks presented in [4], the ANSI-41 Handoff Measurement Request scenario presented in [26], and GSM based systems such as GSM-900 and GSM-1800 [34] follow this pattern. ANSI-41 supports network-controlled handoff and MAHO only [79].

Resulting Context

After the decision about the need for an inter-system handoff, an *anchor mobile switching center* should handle the resources during the *inter-system handoff execution* (Section 5.5.3) without being perceived by the mobile user. Otherwise, either no handoff or other type of handoff is necessary (not shown in this pattern).

5.5.2 Anchor Mobile Switching Center

Context

A *handoff decision* (Section 5.5.1) has been taken and a handoff, which involves the modification of the dedicated transmission path between the mobile station and the network, should be performed [78][139]. The dedicated channel has been allocated for transmitting signaling and data. As a result, the mobile station has changed from the idle to the dedicated mode. When the mobile station goes back to the idle mode, this path is released.

Problem

How does one handle the resources for the exchanges of information during an inter-system handoff?

Forces

- The physical transmission path, which is created when a mobile station enters the dedicated mode, is modified by handoffs;

- At the lower layers, when a handoff occurs, the previous channel connection is released and the information completely destroyed. This happens regardless whether a new channel is allocated to the mobile station in the same mobile switching center or not;

- Charging is complicated with more than one mobile switching center responsible for a call from the beginning to the end.

Solution

Choose the first mobile switching center serving the dedicated channel as an anchor that will be in charge of the communication (e.g., the call processing information) and will keep the control of the resources during the handoff.

When an inter-system handoff occurs, the network configuration changes and other mobile switching centers become involved. However, the anchor remains the same and these centers become just relays. For example, if the new channel is worse than the previous one, the anchor requests the previous base station controller to re-direct the transmission to the previous mobile switching center. In this case, the anchor mobile switching center is different from the previous mobile switching center.

Rationale

When an inter-system handoff occurs, there is a possibility that more than one mobile switching centers is involved in the exchanges of information between the mobile station's current network and its home network. An *anchor mobile switching center* solves the problem of how to handle the allocation and de-allocation of resources during the handoff as well as how to control the information exchanges.

Resulting Context

The *anchor mobile switching center* is responsible for providing the resources that guarantees the signaling and data information exchanges during inter-system handoffs. The handoff process includes *inter-system handoff execution* (Section 5.5.3) and eventually *handoff failure actions* (Section 5.5.4)

When a *location registration* (Section 5.4.6) occurs in consequence of a handoff, the *anchor mobile switching center* is also in charge of the resources for the information exchanges.

Known Uses

Anchor Mobile Switching Center is used in ANSI-41 Specifications (called anchor MSC), IMT-2000 Systems (called MSC), and GSM/GPRS/UMTS (called anchor MSC).

5.5.3 Inter-system Handoff Execution

Context

A *handoff decision* (Section 5.5.1) has been taken and it is necessary to perform a handoff that involves different mobile switching centers. An *anchor mobile switching center* (Section 5.5.2) has been chosen to control the resources for the information exchanges.

Problem

How does one continuously guarantee communication service assessment for mobile users?

Forces

- The current service, which the user has been accessing, can be cut off as a result of:
  - the candidate mobile switching center (MSC) does not support the requested radio channel characteristics. For example, a TDMA digital channel is required but not available;
  - the signal quality of the candidate MSC is below an acceptable threshold;
  - the current traffic conditions on the candidate MSC does not allow handoff traffic at the given time;
- Users' expectations are not met with reliability and consistency of the signaling and data transmission.

Solution

Identify the candidate MSCs, which are considered for handoff purposes, and evaluate them before executing the handoff. Then, select the most reliable one to handle the communication. After this, the candidate mobile switching center detects and accepts the mobile station in its location area and the mobile station tunes to the new channel.

Figure 49 depicts a possible solution for perfoming an inter-system handoff (see unbound UCMs in Section 4.7.2 of Chapter 4).

Rationale

If the quality of the link is not adequate to provide services to mobile users, a handoff should be triggered to avoid either the end of the communication between two users or the network's incapacity to support services. Without the handoff capability, roaming will not be possible.

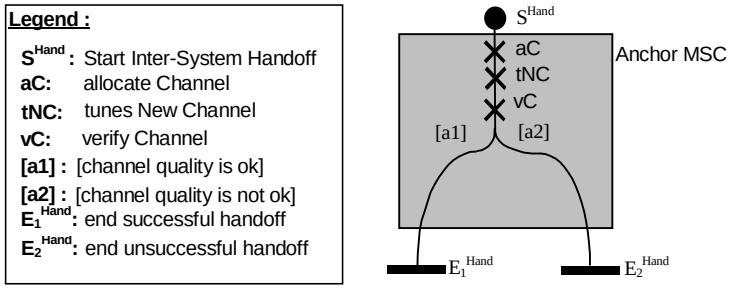


Figure 49 An *Inter-System Handoff Execution* Pattern Solution with bound UCMs

Resulting Context

The *inter-system handoff execution* (Section 5.5.3) has been successfully performed and the mobile station characteristics are stable. The *anchor mobile switching center* (Section 5.5.2) is then able to release the resources that are no longer needed (*releasing resources* in Section 5.5.3). Meanwhile, *home and visitor databases* (Section 5.4.6) update the mobile user's location information (Section 5.4.6). This successful outcome is transparent to the user.

On the other hand, when a handoff failure occurs during this process, the network takes *handoff failure actions* (Section 5.5.4) and notifies the user.

Known Uses

Inter-System Handoff Execution is used in the handoff back, handoff forward, and handoff to third information flows of the ANSI-41 Specifications, in the handover procedures of GSM/GPRS/UMTS, and in the overlay handoff previous of the WmATM networks.

5.5.4 Handoff Failure Actions

Context

A failure has occurred during *inter-system handoff execution* (Section 5.5.3) due to the lack of radio or terrestrial resources or an unexpected propagation loss (for example, obstacles such as bridges or tunnels). As mentioned earlier, an *anchor mobile switching center* (Section 5.5.2) controls the allocated resources during inter-system handoffs when the mobile station is in the dedicated mode.

Problem

How does one handle an inter-system handoff failure?

Forces

- *Handoff decision* (Section 5.5.1) and *inter-system handoff execution* reduce the chances of a handoff failure. However, a failure is not suppressed completely and communication with the current cell can be effectively lost;
- The current communication service, which the user has been accessing, is cut off with possibly loss of information. This failure can be perceived by the user;
- Before the failure has occurred, several communication resources have been already allocated including the transmission path;

Solution

Choose one of the following alternatives: a new handoff attempt towards the same cell, a new handoff attempt towards another cell, or tune to the previous channel (see Figure 50). Then, request the release of all the previously allocated resources (*releasing resources* in Section 5.5.4) along the path in which the failure has occurred. One alternative that should be avoided is to lose the connection between two users or the ability to access network services. The *anchor mobile switching center* (Section 5.5.2) chooses one of the previous alternatives.

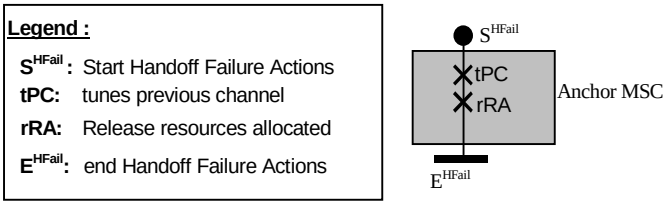


Figure 50 A Solution for *Handoff Failure Actions* with bound UCMs

Figure 50 depicts a possible solution for performing *handoff failure actions* (see unbound UCMs in Section 4.7.2 of Chapter 4).

Rationale

The user should not perceive a handoff failure that occurs when the network verifies that the new channel does not have a satisfactory quality of connection. Actions must be taken to minimize these failures and avoid interrupting the access to the network services.

Resulting Context

Either a handoff has been re-initiated (first and second alternatives) or the mobile station has tuned to the previous channel (the last alternative). Furthermore, all the resources in use during the failed handoff have been de-allocated (see *releasing resources*).

Known Uses

Handoff Failure Actions is used in the call re-establishment of GSM/GPRS/UMTS and in the handoff failure actions of the WmATM networks.

5.5.5 Releasing Resources

Context

An *inter-system handoff execution* (Sections 5.5.3) has successfully occurred or *handoff failure actions* (Section 5.5.4) have been taken. Meanwhile, the *anchor mobile switching center* (Section 5.5.2) is controlling the allocated resources despite handoffs.

Problem

How does one minimize the use of resources that are no longer needed, such as the inter-mobile switching center circuits?

Forces

- Besides the amount of inter-mobile switching center circuits available being limited, these resources are necessary not only for handoff execution, but also for location registration;
- If the user is out of coverage, or has powered off the mobile station in the middle of a call, the network infrastructure has to detect that the resources are no longer needed and make sure that the mobile station is back to the idle mode;
- Before a mobile station is placed in the idle mode (after finishing a call or due to a network failure), a frame loss can occur and the mobile station can still transmit on its dedicated channel while the network is allocating the same channel to another mobile station.

Solution

Release the unnecessary inter-mobile switching center circuits using a request from the *anchor mobile switching center* (Section 5.5.2) to the previous mobile switching centers involved in the handoff. In order to avoid conflict on the allocation of channels, the mobile station goes back to the idle mode and stops using the channels before the network releases the resources.

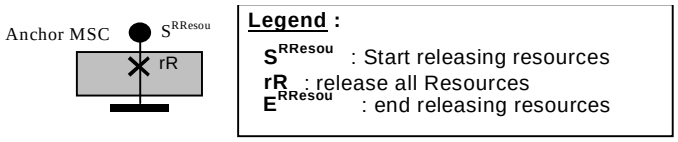


Figure 51 A Solution for Releasing Resources with bound UCMs

Figure 51 depicts a possible solution for *releasing resources* (see unbound UCMs in Section 4.7.2 of Chapter 4).

Rationale

Resources such as radio channels and inter-mobile switching center circuits are limited in mobile systems. In order to avoid failure in the execution of paging, call connection, and handoff procedures, among others, for lack of these resources, there is a need for releasing all resources that are no longer necessary after the end of these procedures.

Resulting Context

Once the inter-mobile switching center circuits have been released, they are available for allocation to other purposes.

Known Uses

Resources are released by the following signaling alternatives for WmATM networks: overlay registration, call setup and handoff. This pattern can be also found in the ANSI-41 handoff scenarios and in the GSM/GPRS handoff scenarios.

5.6 Discussion: Variabilities

Figure 52 depicts the relation between commonalities represented by the MoRaR pattern language and variabilities. The gray part of the figure corresponds to the specific functional behaviors (e.g., communication management functions) and architectural elements that are not described as patterns and can be added to the system.

The patterns documented in the previous sections represent the commonalities among the chosen mobile systems at a high level of abstraction. In order to develop protocols related to the application layer (e.g., MAP), parameters, internal algorithms, operations, and message names are dependent on the implementation issues. This constitutes the variabilities that make the current systems incompatible.

For instance, CDMA systems support *location registration* when the mobile station is powered-off [79]. This registration event is triggered to inform the network that the MS is unavailable for service and constitutes a power-off de-registration (location cancellation). This solution solves the problem of a subscriber who turns off the MS and remains registered in a system for several minutes or even hours until her inactivity is

detected. The inactivity can be detected when the subscriber registers in another system or by periodic maintenance procedure.

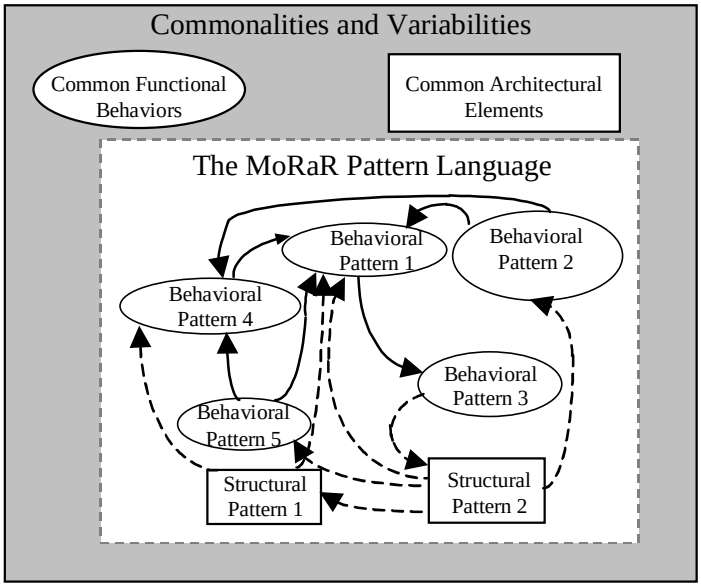


Figure 52 The MoRaR Pattern Language and Variabilities

5.7 Future Work

This research does not intend to cover all common functional behaviors and architectural elements among mobile wireless communication systems. We believe that common functionalities for communication management can be further investigated using the approach for capture proposed in Figure 29 of Chapter 4 and patterns can be extracted from these commonalities. These patterns can be included in the MoRaR pattern language within a new category called communication management.

Chapter 7 presents case studies of the pattern language reuse and validation (e.g., in a WmATM environment). As future case studies, the MoRaR pattern language can be applied to other domains and to design a hybrid network that aims to integrate cellular and IP networks as presented in [97]. Furthermore, the mobility and radio resource management functions used by the IS-95 systems (i.e., CDMA systems) [35] can be investigated in order to find which of the radio resource management patterns are also suitable for them.

In addition, we believe that a more complete identification of the intersection between the software patterns presented in the literature and the requirements and analysis patterns documented in this research should be done. Figure 53 depicts these two groups of patterns and their commonalities, which are called *related patterns* like in the pattern template. An attempt to do this is presented in some of the behavioral patterns. This identification can help to migrate from the requirements and analysis models to object-oriented design and implementation.

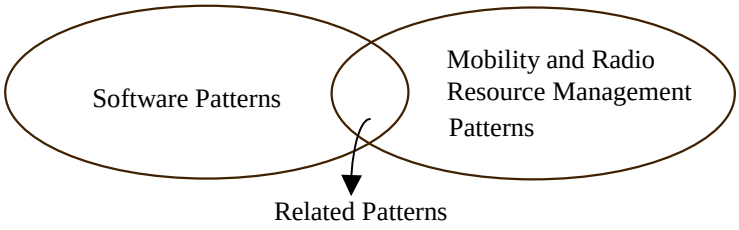


Figure 53 Intersection of Patterns

5.8 Conclusion

This chapter presents a pattern language to describe mobility and radio resource management functions at a high level abstraction. The patterns are described in a general way, capable of different implementations. In practice, the pattern problems and their respective solutions are recognized by investigating different mobile systems and by capturing what they have in common. Chapter 4 shows the capture of common functional behaviors and architectural elements used in various mobile systems while looking for similarities and how they solve specific design problems associated with them. GSM/GPRS/UMTS [139], ANSI-41/WIN [25] (D-AMPS [35] is an ANSI-41 based system), IMT-2000 [112][113][114][118], and WmATM [4][54][191] are the systems investigated in this work. In [19], we have published the patterns related to mobility management functions.

Whether designers are maintaining existing systems or building new ones, they can identify similarities and differences with respect to actual or future systems. In Chapter 7, we present case studies that use the set of patterns that we have documented. The MoRaR pattern language avoids the representation of these patterns as isolated entities. We believe that once one recognizes commonalities among existing systems, it is possible to iron out differences and enable them to interwork. Furthermore, these solutions become more accessible and better understood to novices and experts when documented as patterns.

The next chapter discusses the current approaches for developing large systems such as mobile networks and introduces an approach for reuse and validation of the pattern solutions presented earlier.

Chapter 6 Approach for Reuse and Validation

An approach for reuse and validation of the behavioral and the structural patterns that are grouped in the MoRaR pattern language presented in Chapter 5 is now introduced. At the early stages of the development and evolution of mobile systems, this approach applies a combination of **U**se **C**ase **M**aps (UCMs), **L**anguage **O**f **T**emporal **O**rdering **S**pecifications (LOTOS), and **M**essage **S**equence **C**hart**s** (MSCs) in order to reuse and to validate the pattern solutions. First, the UCM scenarios are chosen according to the system needs. Then, these scenarios are formally specified and validated with LOTOS and its tools, which provide confidence in the correctness of the reuse of the pattern solutions. MSCs are used to represent the results of the LOTOS validation traces and to describe how behavioral and structural patterns interact.

6.1 Introduction

As discussed in Chapter 2, there is a need for recognizing and capturing common functional behaviors and architectural elements among different mobile systems in order to generate a seamless environment. This research presents patterns for mobility and radio resource management functions to fulfill this need (see Chapter 5). These patterns document the identified commonalities among second and third generation systems as shown in Chapter 4.

An approach on how to reuse and how to validate these patterns is now introduced. This approach is a good starting point towards ironing out differences and leading possibly to better ways to interwork diverse mobile systems and to develop new ones. For instance, the behavioral and the structural patterns are documented in a general and abstract way in Chapter 5 to allow telecommunication designers to reuse them at the early development stages (e.g., the ITU-T stage 1 and stage 2 depicted in Figure 21 of Chapter 4). These patterns, which are grouped in the MoRaR pattern language, offer a common foundation for reuse and designers can adapt and make changes according to the system needs.

The proposed approach combines the application of semi-formal and formal methods with pattern reuse and validation in the development and evolution of large and complex systems such as mobile systems [8][13]. **U**se **C**ase **M**aps (UCMs) [51] and the **L**anguage **O**f **T**emporal **O**rdering **S**pecifications (LOTOS) [145] are applied at the requirements and analysis stages. At the design stage, LOTOS and **M**essage **S**equence **C**hart**s** (MSCs) [106] are the chosen methods. LOTOS is able to provide confidence in the correctness of the pattern reuse and MSCs are generated from the LOTOS traces to help the implementation of protocols.

According to [159], software reuse improves development productivity and quality by saving time when reusing parts from other projects that have previously been developed, tested and debugged. Besides development productivity and quality improvements, cost reduction is another factor for the need and popularity of software reuse. In addition, the increasing demand for large and complex software systems, and the difficulty of supplying them on time and within budget make the reusability concept attractive.

Although software reuse has many advantages as stressed in [81][124][159], the investment to make the software reusable by ensuring its quality as well as flexibility and by providing more documentation is higher than the one required for non-reusable systems. Furthermore, reuse has long-term benefits.

As mentioned in [124], when reusability is taken into consideration at the early stages, it saves time and costs by reducing the effort to understand and capture the user requirements. It also reduces the activities associated with the analysis stage such as describing how functional behavior and structure models work together and it reduces the changes necessary to the reusable code at later stages. We believe that the proposed approach brings the reusability benefits to the mobile system domain and overcomes its cost disadvantages with our focus at the requirements, analysis, and design stages.

Methodologies, modeling languages, development frameworks, and conceptual models currently used in the development and maintenance of large systems are outlined in the next section. After this, an overview of LOTOS and MSCs is given in Section 6.3. Section 6.4 introduces the proposed approach followed by a discussion of related work in Section 6.5. Last, Section 6.6 addresses our main conclusions.

6.2 Development Approaches for Large Systems

Figure 54 illustrates several approaches and their respective notation that are applied at different stages of the software development life cycle. Methodologies (e.g., *object-oriented methodologies* such as UML-RT and *structured methodologies* such the ITU-T three-stage methodology), modeling languages (e.g., UML), development frameworks (e.g., Architectural notes and ODP), and conceptual models (e.g., IN) are among the approaches we are considering and they are outlined in the next sub-sections. As shown in the figure, this research focuses on the requirements capture, analysis, and design stages.

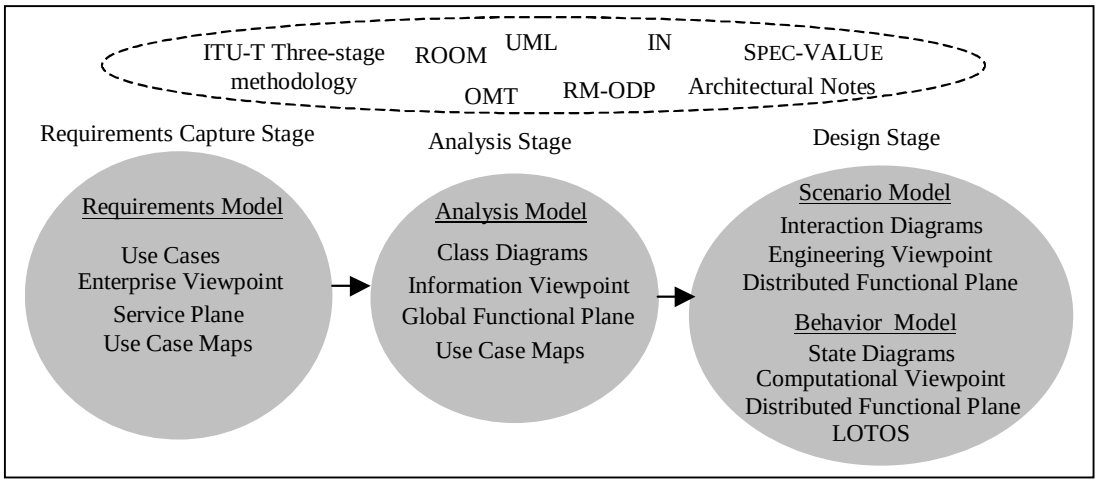


Figure 54 Current Approaches and their respective Notation at Different Development Stages

6.2.1 Object-oriented Approaches

Several object-oriented approaches for software development have been presented [39][55][116][164][167] and discussed [8][153][195] in the literature. These approaches change the focus from functional decomposition, which is used in the structured approaches, to objects as a better way to deal with the complexity of large systems [60].

According to [153], the object-oriented methodologies that appeared between 1988 and 1992 are "conceptually quite similar despite significant differences in their notations." In addition, we consider that they have differences regarding their use in the software development stages. In this context, object-oriented methodologies are discussed in terms of their notation and their use in the different development stages as follows.

Object-**O**riented **A**nalysis and **D**esign (OOAD) models such as the *use case driven approach* [116] use different techniques for each development activity. OOAD approaches target early stages while neglecting implementation and testing issues. For instance, use cases, interaction diagrams, and finite state machines represent requirements, analysis and design models. The *use cases* notation describes the requirements model as a sequence of events an actor (also known as an external agent) uses in a system to complete a process (see also definition in Chapter 4). This notation guides the modeling of a domain as a set of objects, which are detailed in the design stage.

The **O**bject **M**odeling **T**echnique (OMT) [164] and the **R**eal-time **O**bject-**O**riented **M**odeling (ROOM) [167] focus on analysis, design and implementation stages. Both approaches use the following notation: sequence diagrams, class diagrams, object diagrams, and state diagrams.

OMT is a *methodology* with functional, dynamic, and object models. The functional model describes what a system does without concern of how and when this is done. The dynamic model "describes the sequence of operations that occur, without regard for what the operations do, what they operate on, or how they are implemented." The object model offers a *framework* with static, structural, and data aspects of a system within which the dynamic and functional models are placed.

ROOM is the *system development methodology* mentioned earlier. The *modeling language* describes the basic vocabulary (i.e., the mainly graphical notation) that can be used to express high-level system properties such as domain knowledge and design ideas. *Modeling heuristics* are informal guidelines on how to apply the modeling language in specific situations. The *work organization* offers the structure to use the modeling language and the modeling heuristics.

The Unified Modeling Language (UML) [66][75][94] is a modeling language that incorporates the notation of Booch (the object-oriented design methodology) [39], Jacobson (the use case driven approach), and Rumbaugh (OMT). UML is independent of process, which specifies the steps to take when doing the design. Notation such as a use case diagram, class diagram, interaction diagram, package diagram and collaboration, state diagram, activity diagram, and physical diagram is part of UML. These UML diagrams are used to specify requirements, analysis, and design models.

Use case diagrams are a graphical representation of use cases, which are suitable for describing requirements. *Class diagrams* are used in the analysis, design, and implementation models to describe objects and static relationships among them. An *interaction diagram* represents the behavior of a use case and how groups of objects collaborate. *Package diagrams* are high-level units that group classes together and show the dependency among them. The interaction among two or more classes is called *collaboration*. *State diagrams* describe the behavior of a system in terms of all possible states that a particular object can reach. A variant of a state diagram is an *activity diagram* that describes a sequence of activities with support for conditional and parallel behavior. *Physical diagrams* show both the system hardware and software components and the physical relationships (e.g., interfaces) among them.

6.2.2 Development Frameworks, Structured Approaches, and Conceptual Models

Reference **M**odel **O**pen **D**istributed **P**rocessing (RM-ODP) [71][72][105] is considered a *framework* for the development of object-oriented distributed systems. The ODP framework applies five different viewpoints to deal with the complexity of distributed systems, as follows: enterprise, information, computation, engineering, and technology.

Each viewpoint represents the system with emphasis on a specific set of concerns. For instance, purpose, scope, and policies of the system are the *enterprise viewpoint* concerns. The *information viewpoint* deals with the information that is managed by the system as well as the constraints carried out by the use of this information. The *computational viewpoint* decomposes the system into smaller units (distributed objects) and identifies

the object interfaces and their interaction. Finally, the *engineering viewpoint* describes the infrastructure and the technology viewpoint, which are composed of the software and hardware that are necessary to implement the system components. These viewpoints allow a gradual and iterative system development, which is similar to the stages mentioned earlier (see viewpoints in Figure 54).

Another *framework* for distributed systems development called *architectural notes* is presented in [150]. This methodology is based on step-wise refinement and considers a system from an integrated perspective and from a distributed perspective. LOTOS is the notation used to illustrate the concepts that focus on the design stage.

Besides the previously mentioned approaches, *conceptual models* such as the Intelligent Network (IN) [110][111] have been presented in the literature for the development of distributed systems as well as services and protocols related to the telecommunications area. In particular, the IN conceptual model contains four *planes*, each of which focuses on a different system viewpoint or level of abstraction, hence, encouraging an early separation of concerns. These planes are, from the more abstract to the more concrete: service plane, global functional plane, distributed functional plane, and physical plane (see more details about IN in Chapter 2).

The three stages of the ITU-T methodology (see Figure 21 and more details in Chapter 4), which is a *structured approach*, correspond roughly to IN's service plane (stage 1), distributed functional plane (stage 2), and physical plane (stage 3). The IN service plane contains informal descriptions of service operations that correspond to Stage 1 documents such as textual descriptions of normal procedures with successful outcome, exception procedures or unsuccessful outcome, and interactions with other services. The Distributed Functional Model (DFM), related to the distributed functional plane, is described in stage 2. The third stage (and often the second one too) presents services with the help of the Network Reference Model (NRM) which corresponds to the IN physical plane. The IN's global functional plane (GFP) deals with services in the context of the basic activities to support a single call such as the sequence of instructions related to a call instance to be processed (e.g. playing announcements and charging). Since ITU-T methodology focuses on individual services, IN's GFP does not have a clear counterpart in this methodology.

The **Spec**ification-**V**alidation **A**pproach with **L**OTOS and **U**CMs (SPEC-VALUE) [8] is a rigorous scenario-driven approach to the description and validation of complex system functionalities at the early stages of design (see UCMs and LOTOS in Figure 54). UCMs capture functional requirements and the UCM scenarios are translated into LOTOS specifications that are validated with the help of tools. As shown in the figure, UCM is the SPEC-VALUE notation for requirements capture as well as for describing the analysis model. LOTOS is used in different stages even when details regarding the architecture are not available to produce specifications that are executable and that can be validated. The validation testing method introduced in this approach proposes the generation of test cases from the information provided by the requirement model, which is graphically specified in UCMs. These test cases are used in the validation of the

LOTOS specification against the functional requirements expressed with UCMs. There is no formal approach for the generation of these test cases. Another rigorous approach that uses the LOTOS notation for the specification and the validation of mobile telecommunication standards is presented in [25]. This validation methodology proposes a formal prototype with LOTOS using several steps that includes requirements capture, modeling, and testing. Requirements are extracted from the standards documents and expressed with LOTOS with a level of details corresponding to the design stage. The LOTOS specification is then validated and verified.

6.2.3 Discussion: Describing Early Stages with Rigor

The object-oriented approaches, the ODP framework, the IN model, and the SPEC-VALUE approach discussed in the previous sub-sections help the system development by promoting a more uniform description of the stages (e.g., requirements, analysis, and design) and by bridging the gap between them. Furthermore, the concepts of ODP viewpoints and IN planes as well as the concept of layers [173], which is discussed in 2.3.2 of Chapter 2, are useful for the decomposition of large and complex systems into small manageable units. However, approaches such as architectural notes, which focuses only on the design stage, produce specifications that are difficult to maintain due to the early emphasis on the details required at the design stage. In addition, we believe that the ITU-T three-stage methodology, which clearly presents a gap between the stages, is more error-prone.

OO approaches have intuitive definitions rather than formal ones. However, they offer semi-formal notations that are more precise than natural language and more readable and understandable than formal languages. Such semi-formal notations such as UCMs can generate formal specifications as shown in the SPEC-VALUE approach.

In the mobile system domain, the structured approaches have been successfully applied in many large projects and have shown their usefulness for describing systems within specific domain applications [1][23]. The overall mobile system behavior is described as a function (see Chapter 2), which is broken down further into mobility, communication, and radio resource management sub-functions.

This research focuses on mobile systems and addresses the various problems encountered in developing and maintaining these systems due to the lack of formality of the current approaches. As shown in Section 4.2 of Chapter 4, most development approaches of the chosen systems include only text and information flows at the early stages. As a result of the complexity involved in handling mobility, communication, and radio resource management functions, these approaches can lead to ambiguities, gaps, inconsistencies, and undesirable interactions at the later stages.

For example, the current approaches for the development of WmATM signaling protocols, which are depicted in Figure 22 of Chapter 4, use *informal description*s, *information flows*, *flow charts*, and *state models*. As a result, from the informality of the text to formal models, a description gap can be identified that leads to protocol

inconsistencies and undesirable interactions at later stages. Even though *flow charts* after informal descriptions are more adequate in reducing this gap, they quickly become difficult to manage due to the increasing complexity involved in the description of architecture and protocols of large systems such as WmATM networks. In addition, *information flows* and *flow charts* produce disjoint scenarios that cannot be validated. Thus, completeness and consistency can only be checked at the implementation stage.

On the other hand, **F**ormal **D**escription **T**echniques (FDTs) have been successful in specifying and validating protocols in different application domains resulting in clear and concise specifications [15][25][61]. FDTs, such as LOTOS [145] (the Language of Temporal Ordering Specifications) and MSCs [106] (Message Sequence Charts) have not only shown resiliency in their usability, but have also improved in tool support and training over the last 15 years [13][61][69]. The tool support helps the understanding, manipulation, and execution of formal specifications. In addition, validation methods can be also used to guarantee the system specification meets the requirements of the system.

Even though LOTOS and MSCs can be used at different levels of abstraction, they require precision in the description of action sequences and exchanged messages. Thus, these formal techniques are more suitable to be applied at intermediate stages of the development process. In contrast, as mentioned in the previous chapters, **U**se **C**ase **M**aps (UCMs) [50][48] give the designer the capability to work with whatever amount of detail is available. This notation is thus appropriate for the early development stages. For instance, UCMs are applied to describe the chosen systems (functional behaviors and the mapping of functional behaviors to architectural elements) in Chapter 4. In addition, as shown in Chapter 5, UCMs are chosen to better specify scenarios for the common functional behaviors described in the pattern solutions.

The combined use of semi-formal and formal techniques at different stages of the system development process is proposed in [13] and further discussed in [127][93]. This combination overcomes the lack of formality in the development of telecommunications systems. As mentioned earlier, in the SPEC-VALUE approach the informality of UCMs is combined with the power of LOTOS. This approach is used in the specification and validation of GPRS and WIN standards, respectively, in [8] and [200].

Our approach for reuse and validation follows the SPEC-VALUE approach with the combination of UCMs and LOTOS notations to better express the common functional behaviors and architectural elements described by the MoRaR pattern language in Chapter 5. However, the proposed approach also aims to provide a way to reuse and to validate these commonalities when developing or maintaining a mobile system. In addition, MSCs are included to represent LOTOS validation traces and to express the scenario model in the design stage. More details about the SPEC-VALUE approach will be outlined together with the various steps of our approach, which is intertwined with it, in Section 6.4.

The validation part of the proposed approach shows how to add rigor to the pattern reuse. LOTOS tools helps to achieve correctness and accuracy automatically. As an open

area of research, properties can be formulated in temporal logic and checked against the behavior trees generated from the LOTOS specification as proposed in [25]. We believe that this step augments the precision of the validation part by addressing specific properties instead of scenarios.

In the next sub-section, the main LOTOS and MSCs notations applied to this thesis are presented.

6.3 The Chosen Notations

The approach for reuse and validation, which is detailed in Section 6.4, applies UCMs, LOTOS, and MSCs at the requirements capture, analysis and design stages. UCMs are discussed in Section 4.4 of Chapter 4. Basic concepts of LOTOS and MSCs are briefly outlined in the next sub-sections. These notations are introduced in this thesis in order to explain the validation process and to enable the reader to follow the LOTOS specifications and MSCs scenarios presented in Chapter 7.

[8] and [200] have presented formal specification and validation of, respectively, GPRS and WIN services that were developed from UCM descriptions and were verified against the requirements. The University of Ottawa LOTOS Group has successfully applied LOTOS to the specification and validation of mobile network standards, such as Global System for Mobile Communication (GSM) [139], and UCMs to the description of Wireless Intelligent Network standards [174][175][176] as presented in [12].

6.3.1 LOTOS Notation

LOTOS specifications represent a formal system prototype that describes temporal relations that correspond to the externally observable behaviors of the system. The LOTOS notation, which has formally defined syntax, static semantics, and dynamic semantics, is an ISO standard [145].

A LOTOS specification is composed of a hierarchy of *processes* that perform internal unobservable actions and interact with the environment through *gates*. Behavior expressions and abstract data types (ADTs) are part of this formal technique. LOTOS operators such as action prefix, choice, disable, enable, guard, and parallel composition are used to combine processes, actions, and behavior expressions to form other behavior expressions as follows:

- the *action prefix* operator ";" as shown in "a;B" means that a gate or an action "a" precedes a behavior "B";

- the *choice* operator "[]" as shown in "B_1[ ]B_2" means that the process will behave as either "B_1" or "B_2";

- the *disable* operator as shown in "B_1[>B_2" means that at any time during the execution of "B_1", "B_2" can be triggered and terminate "B_1";

- the *enable* operator as shown in "B_1>>B_2" means that "B_2" can only be activated after the successful completion of "B_1";

- the *guard* operator as shown in "[P]->B" means that "B" can only be performed if the predicate "P" is true;

- the *full synchronization parallelism* operator as shown in "B_1||B_2" means that "B_1" and "B_2" must synchronize in every action that they perform;

- the *interleaving* operator as shown in "B_1|||B_2" means that "B_1" and "B_2" are performed in parallel without any synchronization between them;

- the *interleaving* operator as shown in "B_1|[g_1,g_2, …, g_n]|B_2" means that "B_1" and "B_2" are performed in parallel with synchronization required on gates g_1,g_2, …, g_n. This operator is also called the *selective interleaving* parallel operator.

The LOTOS notation combines concepts presented in pre-existing notations such as CSP [92] and CCS [136] for the control part and ACT ONE for the data part of a specification. For instance, CSP inspires the notation for the offer and the acceptance of values between processes that are denoted by, respectively, ! and ? as shown in g !dialTone ?user_ID: integer where the network offers the dialTone message and receives the dialed number at gate g. The LOTOS formal semantics are based mainly on CCS. More details about LOTOS can be found in [36][127][145].

LOTOS has many advantages for specifying and validating complex and large systems. For example, the ability of LOTOS to express functional behaviors at several levels of abstractions is used in the SPEC-VALUE approach at different development stages. This ability allows the translation of UCMs into LOTOS [8][21] at the requirements and analysis stages and the addition of detailed messages and parameters to the LOTOS specification at the design stage.

Furthermore, the LOTOS ability of process instantiation and parallel composition is useful to map common functional behaviors, which are described as sequences of responsibilities in each behavioral pattern solution, to the system reference architecture, which is documented in the form of structural patterns.

LOTOS is executable, modular, and capable of synchronization between processes and these characteristics are also considered LOTOS advantages in the specification of mobile systems. LOTOS specifications integrate behavior and architecture in a single executable prototype that can be easily validated against the requirements represented by UCMs. As mentioned in Chapter 4, the UCM stub notation expresses modularity that is translated into LOTOS as a result of the stepwise decomposition of processes. These processes become reusable by applying parameterization. Synchronization between the

processes, which correspond to the architectural elements and their functional behaviors, is essential to describe large and distributed systems such as mobile systems.

Tools that support the language are available to offer ways of checking completeness and consistency [37][148]. LOTOS tools provide validation and verification methods that allow the detection of design errors, inconsistencies and incompleteness at early stages.

Among others, **LO**tos **LA**boratory (LOLA) is a set of tools developed by the Department of Telecommunication Engineering (ETSIT) of the University of Madrid [148]. LOLA is a transformational and state exploration environment that supports execution and testing of LOTOS specifications. LOLA includes a set of tools that help designers analyze the behavior of a system before the implementation stage: a step-by-step executor, a tool for obtaining the labeled transition system, and a tool for testing. The step-by-step executor (also called *simulation* or *debugging* tools) simulates the behavior step by step and evaluates data value expressions. The *testing* tools calculate the response of a system specification to a test according to testing equivalence. These expansion transformation tools are also used to generate a trace (one possible scenario of the specification). LOLA is applied to the case studies shown in Chapter 7.

6.3.2 MSC Notation

MSCs [106][163], standardized by ITU-T, describe interactions between system components. In the simplest form of the notation, each MSC represents exactly one scenario by focusing on the communication behavior of system components and their environment through message exchanges. These basic MSCs are simple and intuitive and are the most used notation by telecommunications companies and standards. This language has a graphical (MSC/GR) and a textual (MSC/PR) syntax form. This thesis uses the graphical form to visualize the selected system runs (traces) that are the results of the LOTOS validation process.

Figure 55 depicts an example of the basic MSC elements applied to this research. These elements are the only ones necessary in order to represent the LOTOS validation results. These constructs are instance, message, and environment. They are explained as follows:

- the *instances* represent LOTOS processes, SDL entities, blocks, or services. They are represented by a rectangle with vertical lines in the graphical representation. Within the rectangle, the instance heading such as LOTOS process type is described in addition to the instance name;

- the *messages* describe the communication events. Horizontal direction arrows represent the message flow. The arrow describes the event message consumption and the opposite side describes the event message production. Besides the message name, message parameters, which are depicted in parentheses, can be assigned;

- the system *environment* is graphically represented by the frame symbol that forms the boundary of an MSC diagram.

Each instance axis illustrates a complete ordering of the described communication events. Different instance events are ordered through the messages, since a message must be sent before it is consumed.

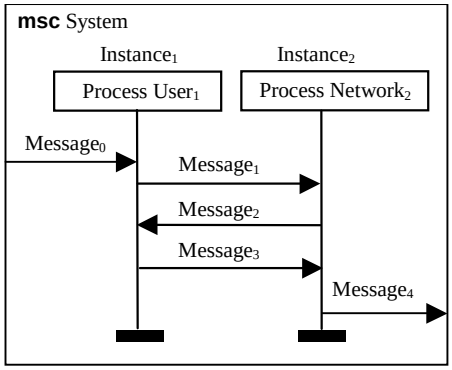


Figure 55 Basic Message Sequence Chart Elements

Conditional (e.g., alternatives) and looping behaviors are not allowed in conventional MSCs. However, the MSC'2000 version allows multiple traces and the High-Level MSCs (HMSCs) [130] allows multiple structured scenarios and causality instead of time for the sequence of messages. As mentioned earlier, information flows, UML sequence diagrams, Jacobson's interaction diagrams are similar notations to MSCs.

6.4 The Approach for Reuse and Validation

The approach for reuse and validation combines UCMs, LOTOS, and MSCs notations with pattern reuse and validation. UCMs are applied at the *requirements capture* (unbound maps) and *analysis* (bound maps) stages. These UCMs are translated into the LOTOS notation for the validation part. At the *design* stage, MSCs and LOTOS describe, respectively, the scenario and the behavior models. As mentioned in Section 6.2.3, the use of UCMs and LOTOS in the development of mobile systems follows the guidelines proposed in the SPEC-VALUE approach [10].

Figure 56 and Figure 57 illustrate the proposed approach. Stage 1 and stage 2, which are part of the ITU-T three-stage methodology (compare with Figure 21) are also shown in these figures. As mentioned in Chapter 1, the implementation stage is left out of this research mainly due to the confidentiality problems among telecommunications industries that makes it difficult to capture, to reuse, and to validate commonalities of services and protocols at lower levels. Since implementation is out of the scope of this thesis, the implementation stage as well as the testing stage is omitted in the figures.

In the proposed approach, different cycles are used to allow the system behavior increases with designer and user needs (similar to the *phases* mentioned in Section 4.2 of

Chapter 4). Each development cycle brings additional details regarding new functional requirements as well as new system components (called variabilities in [56]). This incremental characteristic is useful when developing a large system. For instance, functional layers, which are discussed in Section 2.3.2 of Chapter 2, can be described at different development cycles as follows: mobility management functions are described in the development cycle 1, followed by communication and radio resource management functions in the development cycle 2 and development cycle 3, respectively (see case studies in Chapter 7). In addition, if any modifications are required after validating the prototype generated with LOTOS, it is possible to revisit the respective model for several iterations.

The next sub-sections give more details about the reuse with UCMs and the specification and validation with LOTOS. MSCs come into the approach at the design stage.

6.4.1 Reuse with UCMs at the Requirements and Analysis Stages

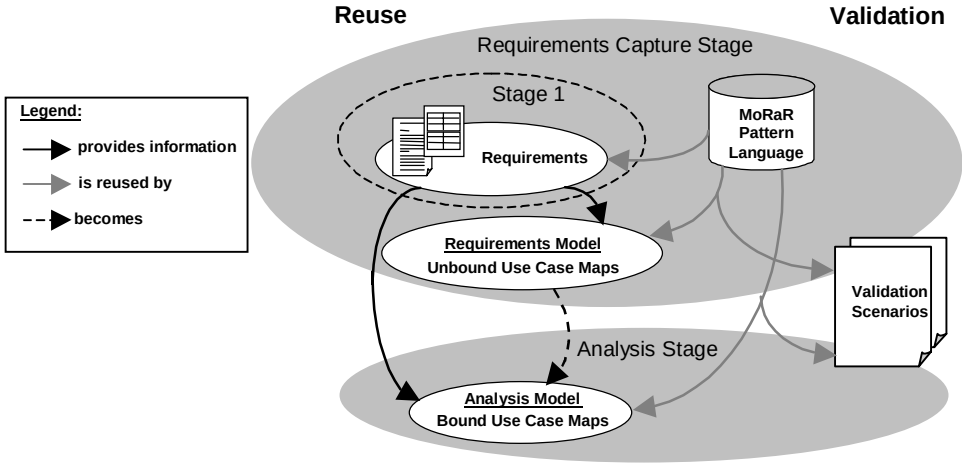
The first design decisions regarding functional behaviors and architectural elements to be added to the system are taken at the requirements and analysis stages. The reuse is done in the stages where the required functionalities (*behavioral patterns*) and components (*structural patterns*) are extracted from the *MoRaR pattern language*.

Figure 56 illustrates the requirements and analysis stages with the pattern reuse and the generation of validation scenarios from the chosen pattern solutions. The resulting models (requirements and analysis models) generated at these stages are also depicted in the figure. Plain gray arcs represent the *reusability process*. Behavioral and structural patterns are reused at the requirements and analysis stages, respectively. The derivation of the validation scenarios from the pattern solutions described by unbound and bound UCMs is also part of the reusability process. Black arcs express the capture of information from the requirements to graphically specify unbound and bound UCMs. Dashed black arcs represent a more detailed view of the system with the derivation of bound UCMs from the unbound UCMs that are mapped to the chosen structural patterns.

As mentioned in Section 6.2, meaningful requirements identify and document what the system is supposed to do and what are the main functions to be described. In order to avoid ambiguities caused by narrative documents such as simple text or even textual use cases, the informal descriptions, which can be stage 1 standard documents, are expressed in a *requirements model* with UCMs (see Section 4.4 of Chapter 4 for more details about this notation). At the requirements stage, when organizational structure details (e.g., the network reference model shown in Figure 39 of Chapter 4) are not available, the *unbound UCMs* are specified reusing the *behavioral patterns* and adding new functional behavior if necessary. The stub notation is used to hide information that is detailed at the analysis and design stages.

Decisions regarding which system component is responsible for a specific action or event are taken during the *analysis* stage. The *structural patterns* come into play at this

point and a network reference model is described with UCM components. The functional behavior (represented by the requirements model) is then mapped to the network reference model. *Bound UCMs* that constitute the *analysis model* are the result of this mapping. Detailed descriptions about what the system does are represented in terms of new *plug-ins*, *responsibilities* are refined with *pre-* and *post-conditions*, *start points* are



refined with *pre-conditions* and *triggering events*, and *end points* are refined with *post-conditions* and *resulting events*.

Figure 56 Approach for Reuse and Validation: UCMs

As mentioned in [81], the best place to select components is at the early stages of the software development life cycle. Inconsistencies between decisions without components at the early stages and decisions with components at the implementation stage can be avoided. We believe that some of these drawbacks of choosing components as an implementation decision are solved with the proposed approach. In addition, we believe that component deficiencies and new functionalities that require extra coding are also reduced at later stages. Chapter 7 applies the proposed approach to three case studies to show the development of requirements, analysis, and design models on the basis of the reuse of structural patterns (i.e., components) and behavioral patterns (i.e., functional behaviors).

The validation scenarios are unbound and bound UCMs derived from the chosen pattern solutions. For instance, Figure 61 illustrates the unbound UCM that represents the *inter-system handoff execution* pattern solution. Two validation scenarios can be derived from this map and described as LOTOS validation test cases as presented in the next sub-section.

6.4.2 Specification with LOTOS at the Requirements, Analysis, and Design Stages

LOTOS is the formal underlying model that supports UCMs. In this research, LOTOS specifications are generated from unbound UCMs at the requirements stage and from bound UCMs at the design stage. In order to specify a system with LOTOS from bound UCMs, construction guidelines (CG) are taken from the SPEC-VALUE approach [10][8]

that split them into the following categories: *interaction points and responsibilities*, *causal paths*, *stubs and plug-ins*, *other path elements*, *structure*, *unrelated path segments*, *inter-component causality*, and *data*.

The CGs are mentioned throughout this section to explain the translation of the UCM descriptions to the LOTOS specifications at the *requirements* and *analysis* stages as shown in Figure 57 (see dotted gray arrows). The behavior model (i.e., the LOTOS design specification) is derived from the previous LOTOS specifications with the addition of design details.

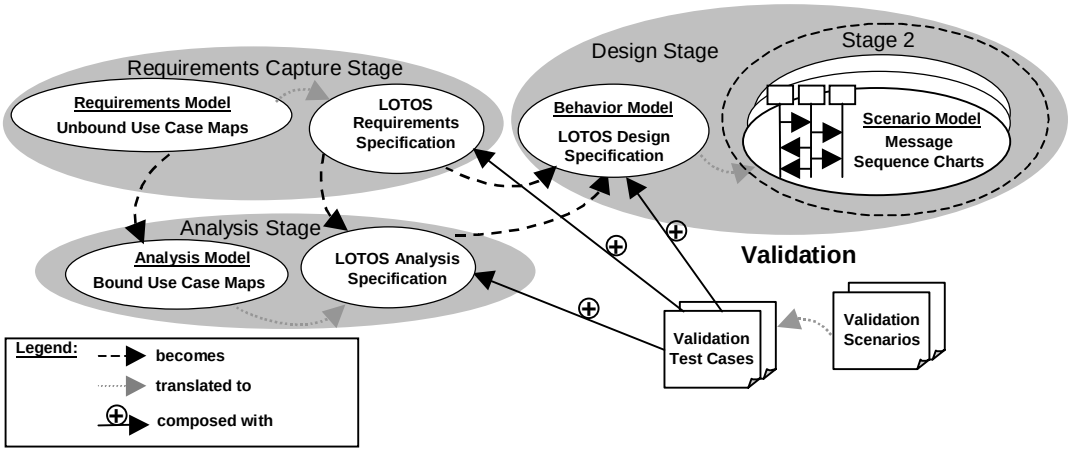


Figure 57 Approach for Reuse and Validation: LOTOS

The following guidelines help the generation of the LOTOS requirements and analysis specifications shown in Figure 57 (see details about LOTOS operators in Section 6.3.1):

- At the requirement stage, stubs and their plug-ins are mapped to LOTOS processes as illustrated in Figure 58. A plug-in is represented as an independent LOTOS process ready to be bound to different stubs. A stub process describes the binding relation with its plug-in(s). These guidelines are explained in the CGs for *stubs and plug-ins* of the SPEC-VALUE approach [8];

- According to the CG for *interaction points and responsibilities*, start points and endpoints, which often describe interactions with the environment through their triggering and resulting events, are specified as LOTOS gates (e.g., start and end gates in Figure 58). We also add a gate to represent the exchange of responsibilities with the environment (e.g., resp gate in Figure 58);

- At the design stage, UCM components (see Chapter 7) are specified as LOTOS processes that synchronize using the LOTOS parallel operators (see *structure* CGs in the SPEC-VALUE approach). LOTOS parallelism allows more than one instance to execute concurrently and for execution purposes the designer can limit the maximum number of instances using ADTs;

- When a UCM path crosses different components, gates represent the interfaces between these components (see *structure* CGs). Figure 72 shows the MSC and HLR synchronization through the gates hlr_to_msc and msc_to_hlr;

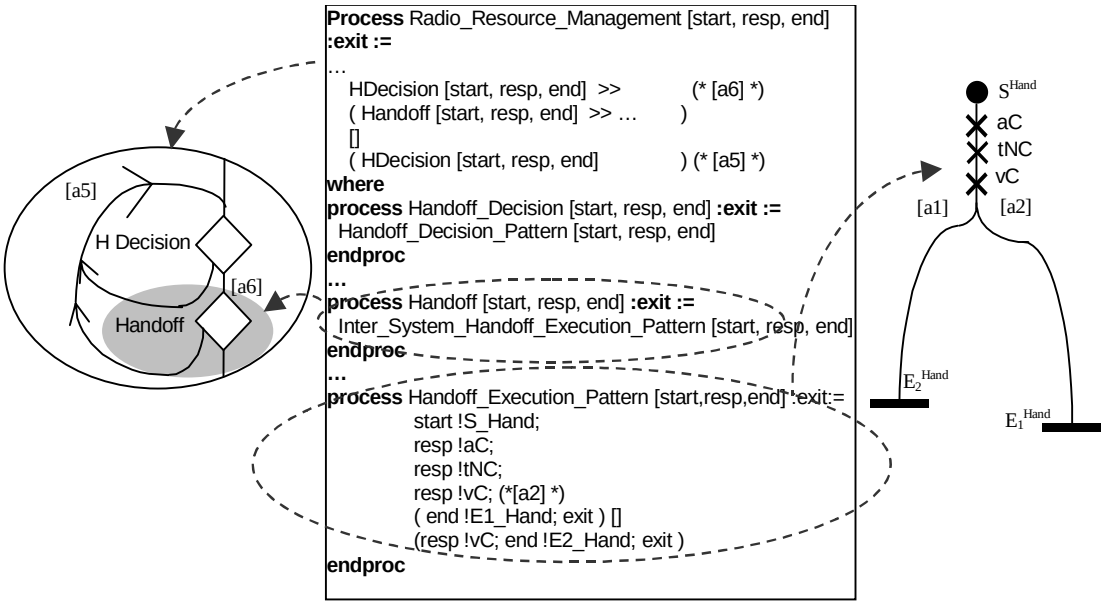


Figure 58 From UCMs stubs and plug-ins to LOTOS processes

- Each LOTOS process behavior corresponds to the causality sequence of responsibilities at the requirement stage. This translation is straightforward at the requirement stage; however, the responsibilities for each UCM component are not always included in one single map when generating the design model (see CGs for *causal paths* for more details and Chapter 7 for an example);

- The sequences of responsibilities are represented by the enable operator ">>" or by the *action* prefix operator ";";

- Choices among paths in UCMs, which are represented by OR-Forks as shown in Figure 58, are translated into the *choice* operator "[]" (see *causal path* CGs in the SPEC-VALUE). These choices can be guarded in LOTOS with the guard operator that represents the post-conditions attached to the responsibilities. This solves most of the non-determinism problems associated with choices;

- The concatenation of UCM paths described with AND-joins and OR-joins are translated into the *enable* operator. The enable operator also captures the exit in the following scenario and continues with the rest of the UCM path: when an OR-join causes a path to loop as shown in Figure 58, a sub-process describes the loop path, which is instantiated and it re-instantiates itself recursively for each iteration until the loop ends with an exit (see *causal path* CGs in the SPEC-VALUE);

- The zig-zag outgoing path is translated into the *disable* operator (see *other path element* CGs).

- In situations where there are concurrent paths in UCMs (AND-Forks), the *interleaving* operator is used (see CGs for *causal paths*).

- UCM timers and the timeout path are translated into the choice operator in this thesis. Another option is to specify timers using a reset event and a timeout event as proposed in the *other path element* CGs construction rule of the SPEC-VALUE.

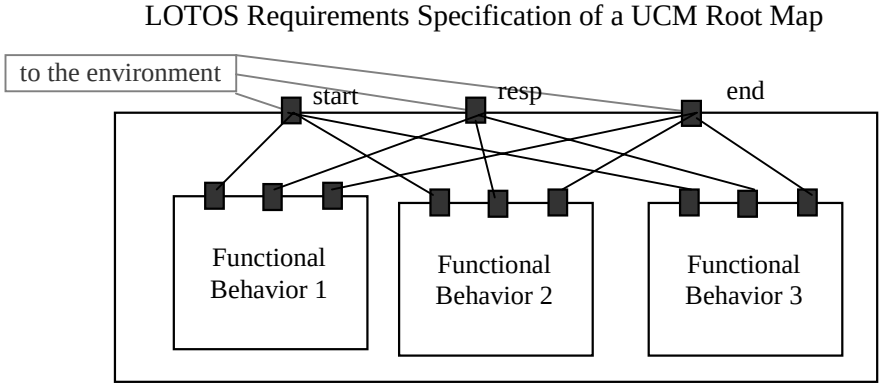


Figure 59 Graphical Representation of LOTOS Requirements Processes

Figure 59 illustrates a graphical representation of processes that constitutes a LOTOS requirements specification. This LOTOS prototype is built on the basis of the unbound UCM. The translation from UCMs to LOTOS follows the construction guidelines mentioned earlier. In addition, we translate UCM stubs and plug-ins, which describe the pattern solutions, into LOTOS processes that can communicate through *resp*, *start*, and *end* gates. These gates receive and send responsibilities, triggering events, and resulting events and they describe the communication between the environment and the system.

At the analysis stage, the functional behaviors illustrated in Figure 59 are associated with architectural elements (e.g., structural patterns) that are described as LOTOS processes (see Figure 60). The gate splitting technique [37] is applied when producing the analyis model with the addition of details to the LOTOS requirements specification. This technique allows the identification of which responsibility comes from which architectural element or which responsibility is sent to which architectural element (see Figure 72 of Chapter 7). Interfaces between architectural elements are not taken in consideration at the analysis stage. *Resp*, *start*, and *end* gates provide the communication between these elements through the environment. The purpose of the LOTOS analysis specification is to provide a prototype of the bound UCMs without adding details that are not specified in the maps such as interfaces and detailed messages.

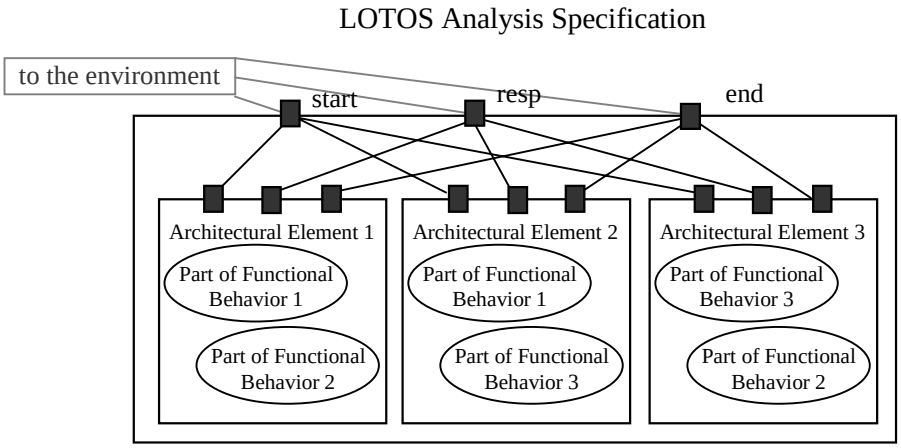


Figure 60 Graphical Representation of the LOTOS Analysis Processes

As mentioned in [10] and [13], the gap between the early stages is reduced by moving from text to unbound UCMs, and then from bound UCMs to LOTOS. LOTOS processes at the analysis stage are derived from the UCM system components and their functional behaviors. However, decisions regarding how these processes directly communicate through gates are not always straightforward and depend on actual interfaces and synchronization needs of the system behavior. These decisions are taken at the design stage as discussed in Section 6.4.4.

6.4.3 Validation with LOTOS at the Requirements and Analysis Stages

Validation and verification techniques are not possible with the UCM notation due to its informality. On the other hand, LOTOS handles different validation and verification techniques and has tools to apply them. When LOTOS is combined with UCMs in [10] and [18], the goal is not only to add rigor to the UCM descriptions (see Section 6.2.3) but also to validate the LOTOS specification that is a prototype of UCMs.

In this research, the use of LOTOS, which starts in the requirements stage, reduces the semantic gap between the translation of UCM requirements and analysis models to LOTOS at the design stage. As a result, when the LOTOS specification is validated, several aspects of the UCM notation are also validated including: causal sequences of responsibilities, choice relations, enabling relations, disabling relations, join relations, and parallel relations (see relations in Section 4.4 of Chapter 4). Our validation part focuses on the reuse of pattern solutions in the specifications.

In short, UCMs and their corresponding LOTOS specification can be validated using simulation, testing, and verification techniques. For example, simulation provides the execution of the specification that can help designers to check the ordering of events and the interactions between processes (e.g., the binding between plug-ins and stubs). In addition, verification techniques help to demonstrate consistency between successive models such as requirements, analysis, and design models as shown in Figure 57. Properties such as absence of unwanted deadlocks can be also verified in a specification.

Testing is the target validation technique in this thesis. The different executions of a LOTOS specification or of an implementation are called *test cases* in [18]. The term *validation test cases* is used in [10] and [8] to differentiate the testing strategy that aims at testing validity of the specification against UCMs and the requirements from the one that aims at testing conformance of the implementation under test against the specification (i.e., conformance testing). In other words, unlike conventional testing which tests implementation, the validation testing is entirely at the specification level.

Our goal is to execute the specification that contains commonalities and variabilities against acceptance test cases that describe the pattern solutions (i.e., commonalities). Since a pattern is a good solution for a design problem, our intention is to assure that the pattern solution is well captured in the requirements and analysis models. In short, we validate the specification against the pattern solutions to guarantee that these solutions are preserved in the specification with the addition of new behavior and structure (variabilities). However, this kind of validation does not guarantee that the pattern is reused properly in the specification, which can be validated with test cases that cover complete scenarios on the basis of the requirements and analysis models.

Testing in LOTOS is the composition of *acceptance* and *rejection test cases*, which represent valid and invalid scenarios, respectively, with the specification. The specification should be able to execute the acceptance test cases without deadlocks and should be able to refuse the rejection test cases. We consider acceptance test cases that are derived from the pattern solutions. Figure 57, which depicts plain black arrows connected to a circle filled with a plus symbol, shows how the LOTOS specification is composed with the validation test cases that are derived from the validation scenarios. In order to facilitate the validation part, the UCM documentation conventions presented in Section 4.5.2 of Chapter 4 are also used in the LOTOS specifications. When a problem is found using LOTOS validation tools, it can be easily related to its UCM corresponding problem.

At the requirements stages, LOTOS validation test cases are generated on the basis of the UCM validation scenarios that are derived from the chosen *behavioral pattern* solutions described with unbound UCMs (see Figure 56). The *LOTOS-requirement specification*, which is the first system prototype, is validated against these test cases using the LOLA tool. The same rules to transform systematically bound UCMs into a LOTOS specification, which are introduced in [18] and are summarized in the last sub-section, are used in the generation of these test cases.

At the analysis stage, new test cases are generated in LOTOS from the UCM validation scenarios bound to the chosen *structural pattern* solutions, if any. The *LOTOS-analysis specification*, which is derived from the requirements and from the *LOTOS-requirement specification*, is then validated.

Figure 61 depicts the validation test cases derived from the *inter-system handoff execution* pattern solution (unbound UCMs). As mentioned in [25], these validation test cases are specified as LOTOS test processes and synchronized with the LOTOS specification under test (see also the testing equivalence definition in [144]). These test cases are derived from the unbound UCMs and they are composed with the prototype to detect possible errors. LOLA performs this composition automatically.

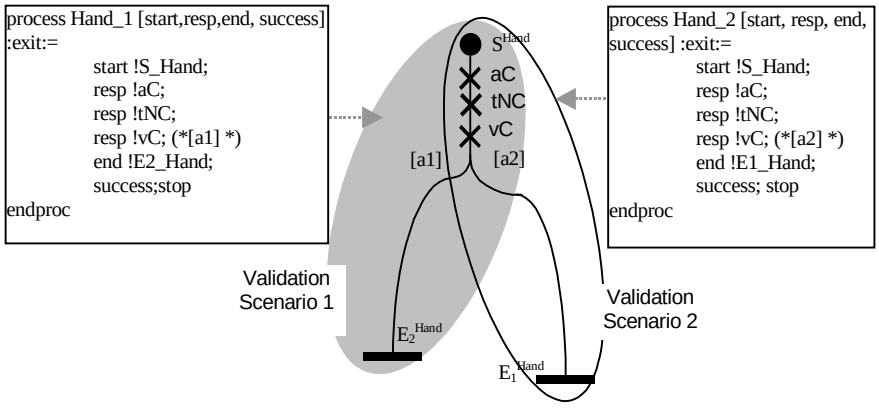


Figure 61 *Inter-system Handoff Execution* Pattern and Validation Test Cases

The special terminating event *success* depicted in the figure is included in every LOTOS test process to indicate the end of every possible execution sequence. The successful termination of a test case means that the termination event has been reached. Unsuccessful test cases represent the case of not reaching the terminating event due to a deadlock situation or internal livelocks. As stated in [144], one of the following test results are obtained when composing a formal specification with a test:

- *must pass*, which indicates that all possible executions terminate successfully;
- *may pass*, which indicates that at least one execution terminates successfully;
- or *reject*, which indicates that none of the executions terminate successfully.

Must pass and *may pass* are considered successful results that guarantee a good level of confidence in the reuse of the pattern solutions with the new behaviors, which are called variabilities in [56], introduced in the specification. However, a *may pass* result can indicate abnormal sequences that can be further investigated to find the unsuccessful paths (see the test exploration step in [25] and the CADP tool in [73]). When a test case fails, the functionality, which has been tested, contains a logical error (*reject* result). As mentioned in [10], this functional behavior was incorrectly specified according to UCMs or was incorrectly integrated with the others.

We believe that other validation techniques such as verification methods and rejection test cases could be used in the validation part of the proposed approach. However, at this

point, the acceptance test cases provide a good level of confidence in the reuse of the pattern solutions.

6.4.4 LOTOS and MSCs at the Design Stage

At the *design* stage, details regarding data types, parameters, interfaces, and exchange of messages are added to the *LOTOS-analysis specification*, which describes the system behavior and its respective network reference model as shown in Figure 57. Since the requirements and analysis models are iteratively and incrementally specified as well as validated, the LOTOS specification becomes easier to develop at the design stage. A *behavior model* is then generated at this stage and validated using the validation testing technique explained in the last sub-section.

The results of the LOTOS validation (e.g., successful and unsuccessful outcomes of the system) are translated into several MSCs (*scenario* models). In Chapter 7, a case study shows how these MSCs are useful for comparing the new scenarios that are generated after the reuse of individual patterns with the ones that originally represent the system behavior. According to the validation approach proposed in [25] (see the test exploration step), these scenarios are also used as input to the implementation and conformance testing stages (out of the scope of this thesis).

As mentioned earlier, the developers of mobile systems often use variations of the MSC notation to describe scenarios at the early stages. For instance, mobile wireless standards documents describe protocols using information flows. As well, MSCs are used to represent early behavior models in object-oriented approaches (e.g., UML sequence diagrams and interaction diagrams in the Use Case driven approach). However, these diagrams are static and disjoint, and only one sequence of events can be observed in each of them. As a result of these characteristics, validation and verification techniques are not possible.

Due the popularity of message sequence charts, most tools for formal methods provide the capability of generating MSCs from the validation results. For instance, the **SD**L Development **T**ool set (SDT) [109] and the SPIN tool [96] support the process of going from the formal design to MSCs. Work on providing the requirements first in terms of sequence diagrams and then applying formal verification techniques such as the ones supported by the SPIN model checker to these diagrams is presented in [95]. This process is also of interest for the SDL and LOTOS communities. In this context, [13] proposes the use of MSCs as a complement of formal methods. We believe that these tools as well as our approach are valuable and lead to a more effective and attractive way to design a system and present the validation and verification results to users and developers.

In this thesis, MSCs enable us to clearly represent the results of the LOLA validation. MSC scenarios are more readable than LOTOS traces and they can be used for implementers to generate the protocols. For instance, these MSCs can replace the information flows of stage 2 standard documents, which are currently used by implementers to generate protocols.

A Lotos2MSC Converter tool is under development to generate MSCs automatically from LOTOS traces [170]. A beta version of this tool is applied to the WmATM case study presented in Chapter 7.

This tool uses a configuration file that interprets the LOTOS traces and generates proper MSC scenarios. To make this possible, the converter uses conventions and additional configuration information to decode a LOTOS action and its elements (the sequence of values) to derive MSC components, messages and parameters. Since the LOTOS generic concept of action defines messages and parameters implicitly in terms of abstract data types, the tool can only recognize messages and parameters when they are described in a LOTOS action.

The converter also allows filtering specific LOTOS actions that the designer wants to be displayed on the MSC graph using gate names as filtering criteria. More details about this converter can be found in [170].

However, this tool restricts the LOTOS capability of full-duplex communication through gates (i.e., no direction is associated with the execution of LOTOS actions among processes) by demanding that gate names represent directions and components (e.g., vlr_to_wsh becomes the gate name in Figure 78 of Chapter 7). According to [170], a direct mapping of LOTOS to MSC concepts is not possible due to the LOTOS synchronization of many simultaneous actions between processes in contrast to the MSC exchange of asynchronous messages between components.

6.5 Discussion

The proposed approach does not depend strictly on the notation used. If the designer chooses another visual technique or formal method that is more appropriate to a particular application, the informal description of the patterns and the pattern language presented in Chapter 5 can still be used. However, validation scenarios and validation test cases should be derived from these techniques to fulfill the validation part.

With the increasing popularity of the Unified Modeling Language (UML) for modeling systems using object-oriented concepts, the UML activity diagrams have potential to replace the UCM notation in the proposed approach. As discussed in [8], like UCMs, activity diagrams focus on functional behaviors rather than on the communication between architectural elements (like MSCs). They also express sequences of actions or events (the *causal relation between responsibilities* defined in Chapter 4), alternatives (the *choice* and *join relations*), and concurrency (the *parallel relation*). In addition, refinement of complex activities and mapping of functional behaviors to architectural elements (e.g., by using *swimlanes*) are possible. However, UCMs enable the description of the *disabling relation* and the use of *dynamic stubs* that are necessary for the description of mobile systems. These features are not allowed in activity diagrams.

UCMs are currently being investigated for inclusion in UML to help the system development process. According to [11], UCMs can help bridge the gap between the requirements model represented by use case diagrams and the analysis and design models represented by the UML behavioral diagrams (e.g., sequence diagrams, state diagrams, and activity diagrams).

UCMs resemble Petri Nets at first sight. Their graphical notations look similar and both are based on causality events. However, many differences can be quickly perceived. First, semantics are not defined for UCMs. This notation is informal and intuitive and it is used in the early stages of the development process to give a global picture of the system. In addition, UCMs are easy to learn and understand (i.e., they are a light-weight notation) and they can also express the underlying architecture. On the other hand, Petri Nets have strict semantics, are rich in analysis methods, and have automated tools that are rigorous and sound. They require precision like SDL and are more appropriate to the design stage. Thus, a mapping of UCMs to Petri Nets, which is left as an open area of research, is more feasible than a replacement.

In the context of this research, a disadvantage of mapping UCMs to Petri Nets is that common varieties of Petri Nets do not allow the mapping of responsibilities to components (bound UCMs). As mentioned earlier, this mapping instead is one of the advantages of the UCM notation and LOTOS allows us to represent both unbound and bound UCMs at different stages and at comparable levels of abstraction.

6.6 Conclusion

This chapter presents an approach for pattern reuse and for validation of pattern solutions. This approach enables designers to re-use common network entities and their respective functional behaviors starting from the early stages of the development process and evolution of mobile wireless communication systems. These solutions become more accessible and better understood to novices and experts alike by being specified with UCMs. By using translation from UCMs to LOTOS, validation test cases can be generated.

On one hand, the pattern concept adds reusability to the SPEC-VALUE approach, which brings more powerful tools to tackle specification and validation problems in telecommunication systems. On the other hand, the formal specification and validation with LOTOS in the SPEC-VALUE approach provides confidence in the correctness of the reuse of these patterns. The reusability-driven characteristic of our approach attends the designer needs in the development of new systems or in the evolution of existing ones.

Whether designers are maintaining existing systems or building new ones, they can identify what makes their actual or future systems similar by taking into consideration the set of patterns that capture the common functional behaviors and architectural elements of legacy systems. Once the commonalities among existing mobile communication systems

are recognized, it may be possible to iron out differences among them and enable them to interwork.

The graphical specification of the requirement and analysis models with unbound and bound UCMs, respectively, allows the designer to have an overview of the basic behavioral properties of a system from the early stages of the system development process or evolution. The UCM notation provides a better human understanding of the system and helps network designers to produce descriptions of the requirements more legibly. As well, it facilitates system evolution. Furthermore, UCMs are a complementary notation for use in the early stages of the ITU-T three-stage methodology, which correspond to the requirements and the design stages of our approach.

We believe that a new system developed on the basis of this common foundation is also more suitable to be integrated with existing systems. Therefore, the pattern reuse concept is a promising way to overcome potential incompatibilities between existing and new systems. This research can also provide means to the development of the global roaming capability and seamless mobile wireless services as discussed in [86].

The next chapter presents the use of the proposed approach in three case studies as follows: A UCM framework for mobile system requirements and analysis models, a third generation system, and a prototype for a typical WmATM networks.

Chapter 7 Case Studies

This chapter presents case studies that apply the approach for reuse and validation presented in Chapter 6. The first case study reuses all the patterns included in the MoRaR pattern language and some of their relationships for the development of requirements and analysis models of a third generation system. The WIN Incoming Call Screening feature is then added to these models to show their integration with second generation systems. The third case study presents the design model of a typical wireless mobile ATM network with the addition of individual patterns.

7.1 Introduction

The approach proposed in Chapter 6 combines UCMs, LOTOS, and MSCs with reuse and validation of the behavioral and structural patterns presented in this thesis. This is done at different stages of the system development life cycle, which are summarized as follows (see Figure 57 in Chapter 6).

At the *requirements capture* stage, the focus is on reusing *behavioral pattern* solutions described as UCMs or textually without considering architectural element. When it is necessary, the designer adds new functional behaviors to build specific characteristics of the mobile system (i.e., variabilities). Unbound UCMs and a LOTOS-requirements specification, which is generated from UCMs, describe the requirements. LOTOS validation scenarios are generated from the chosen pattern solutions to validate the requirements.

At the *analysis stage*, the goal is to re-use the *structural patterns*. These patterns are added to the other architectural elements of the system, which are represented by UCM components. After this, the unbound UCMs are mapped to the architectural elements that constitute the network reference model and bound UCMs are generated. New functionalities can also be included at this stage. A LOTOS-analysis specification is generated from the bound UCMs and the LOTOS-requirements specification. This specification is validated with validation test cases that are generated from the bound pattern solutions.

At the *design stage*, details regarding interfaces, parameters, and data types are included in the LOTOS analysis specification. This detailed system behavior model is again validated against validation test cases that are derived from the previous test cases. MSCs are automatically generated from the *LOTOS-design specification* in order to represent the results of the validation and to facilitate the implementation of protocols (out of this thesis' scope).

This chapter presents the improvements that can be obtained with the reuse and validation of patterns in the development and evolution of mobile systems (see existing approaches in Section 4.2 of Chapter 4). Our case studies exemplify how the proposed approach can improve the overall quality of mobile system development by increasing the following software quality attributes presented in [153]: reusability, portability, correctness, ease of use, and readability. We add software reusability and portability to the previous approaches with the pattern reuse. Rigor is introduced with LOTOS and its validation techniques as proposed in the SPEC-VALUE approach (see Section 6.2 of Chapter 6). UCMs are easy to handle and readable as well as reduces the gap between requirements and design stages. Inconsistencies of messages and parameters as well as incompleteness of the requirements and analysis semi-formal descriptions with UCMs are detected and corrected at the early development stages. As a result, correctness, ease of use, and readability are achieved with the mixture of semi-formal and formal techniques.

The next section presents the first case study that is *a framework for mobile system requirements and analysis models*. The requirements and analysis models are described with UCMs on the basis of the MoRaR pattern language. These models show the relationships among commonalities, which are represented by mobility and radio resource management functions, and variabilities, which are represented by communication management functions. A new feature and the IMT-2000 generic reference model are introduced in these models in Section 7.3. Section 7.4 presents the evolution of a typical wireless mobile ATM (WmATM) system with the proposed approach. We add two patterns to the existing UCM requirements and analysis models (see Chapter 4) and produce a behavior model in LOTOS and a scenario model with MSCs at the design stage. A discussion about other case studies for future work are addressed in Section Figure 82. Finally, conclusions are summarized in Section 7.6.

7.2 A UCM Framework for Mobile System Requirements and Analysis Models: First Case Study

This section presents a UCM framework for mobile system requirements and analysis models that includes the graphical specification of the pattern language introduced in Chapter 5. We use the term framework to describe collections of functional behaviors that are mapped to a network reference model at the requirements and analysis stages. Our framework includes reusable requirements and analysis models in contrast with the OO framework in which developers often reuse either design elements or code. In other words, their focus is on design and implementation stages (see Chapter 3 and Chapter 4 for other uses of the term framework).

Additional requirements, which are necessary to describe the mobile system behavior, are also specified in our framework. This framework helps mobile system designers to reuse the pattern solutions presented in this thesis by providing requirements and analysis models that are ready to be reused at the early stages of the system development process and evolution. The analysis model includes the common architectural elements shown in

Chapter 4 that are mapped to the unbound UCMs. The development of *requirements* and *analysis* models with UCMs applies the approach depicted in Figure 56 of Chapter 6.

In order to develop this framework, we first select all *behavioral* patterns and choose part of the relationships presented in the MoRaR pattern language (see Figure 40 of Chapter 5). Then, communication management functions are added to the requirements model to complete the behavior of a mobile system. This requirements model is represented by unbound UCMs. After this, all *structural patterns* are selected and mapped to their respective functional behavior. Other network entities are chosen from the common network reference model shown in Chapter 4 (see Figure 39). The UCM notation is appropriate to show the mapping of the functional behaviors to the architectural elements as illustrated in Chapter 5, which applies bound UCMs to graphically specify some of the pattern solutions. This mapping constitutes the analysis model of the framework.

At a high level of abstraction, the UCM framework describes the whole functional behavior of a mobile system (e.g., which functions are enabled in the system when mobile users power on their mobile stations). The next section presents UCMs with the description of commonalities and variabilities, which are encapsulated into stubs. Section 7.2.2 presents the mapping of the unbound UCM plug-ins to the architectural elements that are described by UCM components.

7.2.1 Functional Behaviors: Pattern Solutions and Variabilities Encapsulated into Stubs

The description of the requirements model starts with the graphical specification of the MoRaR pattern language (mobility and radio resource management functions) working together with variabilities such as communication management functions. In this first step (see the proposed approach for reuse and validation in Chapter 6), there is no description of which component performs each functionality.

Figure 62 depicts the top level of the framework. This UCM root map illustrates the “big picture” of a simplified mobile system. As mentioned in Chapter 4, the UCM stub notation enables the designer to have a global view of the system at a high level of abstraction.

The reuse of patterns is done in the MM and RRM stubs illustrated in the figure that encapsulate the following pattern solutions for mobility and radio resource management functions presented in Chapter 5: *temporary identification*, *ciphering*, *authentication*, *paging*, *location registration*, *inter-system handoff execution*, *handoff failure actions*, and *releasing resources*. These pattern solutions are graphically specified with UCMs in Figure 32, Figure 33, Figure 34, Figure 35, and Figure 37 of Chapter 4, which presents more details about these plug-ins. Plug-ins related to communication management functions (CM stub) represent the following functional behaviors (see Figure 65): routing, status, and disconnection.

The relationships among these functions are better understood by following the root map flow shown in Figure 62. Each stub has incoming and outgoing paths that graphically specify the interaction among mobility and radio resource management functions as shown in the MoRaR pattern language. Communication management functions and new functional behaviors that describe external events such as power off the mobile station and network failure, which also includes database failure, are also introduced in this root map in order to represent the overall system behavior.

The root map describes the mobile system behavior that starts with the following event: a user powers on a mobile station. This triggering event satisfies the pre-condition associated with the UCM start point labeled S_2. The plug-in, which is bound to the MM static stub (see Figure 63a), also starts after this event. If the mobile station's authentication fails, the plug-in can end with the bar labeled E_2. Otherwise, in case the authentication is successful, the [a2] path is split into two outgoing paths ([a3] and [a4] paths) that are performed concurrently and independently. As a result, communication management functions (described by the CM stub) and the radio resource management functions (described by the RRM stub) can be triggered at anytime ([a3] and [a4] paths followed by [a5] and [a7] iteration paths in the figure) while a mobile station is powered on.

The S_3 start point allows an incoming call to arrive to a user who is powered on, an outgoing call to be generated, and a disconnection by one of the users. S_1 and S_3 can happen only after the map has started with S_2. The S_1 start point corresponds to external behaviors such as power off and network failure. These sub-routes end in E_1 when the user powers off the mobile station. In addition to the Power Off plug-in, the ExternalBs dynamic stub in Figure 62 can be bound to Network Failure and Database Failure plug-ins (not shown in this thesis).

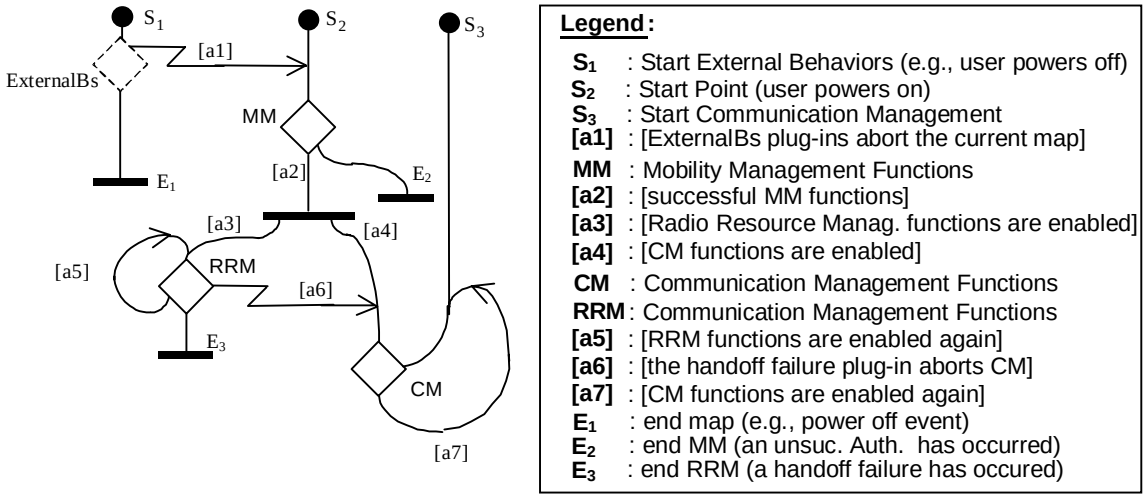


Figure 62 The Root map of the UCM Framework Requirements Model

Communication management functions deal with a mobile user who tries to make a call (*originating mobile user*'s behavior starts with an outgoing call request) and a mobile user who receives a call (*terminating mobile user*'s behavior starts with an incoming call request). According to the satisfied pre-condition, either the plug-in related to the originating party or the plug-in related to the terminating party is bound to the Party stub (see Figure 64a) within the CM stub. After an establishment of communication between two users or an unsuccessful communication attempt, the [a7] outgoing path enables the CM stub execution while the mobile station is powered on. The S_3 start point can be triggered when one of the following two events satisfies the start point pre-condition: either the terminating user answers the call or one of the users disconnects.

The RRM stub includes the handoff decision (Hdecision stub), the inter-system handoff execution (Handoff stub), the handoff failure actions (HFailure stub), and the releasing resource (RelRes stub) functional behaviors as illustrated in Figure 63b. These scenarios can end when the handoff failure actions plug-in ends unsuccessfully. In this case, the [a6] zig-zag path finishes the right hand-side map and the handoff failure actions plug-in takes over the communication (eventually, this path will finish at the bar labeled E3). After every handoff attempt, the [a5] outgoing path enables the RRM stub execution while the mobile station is powered on.

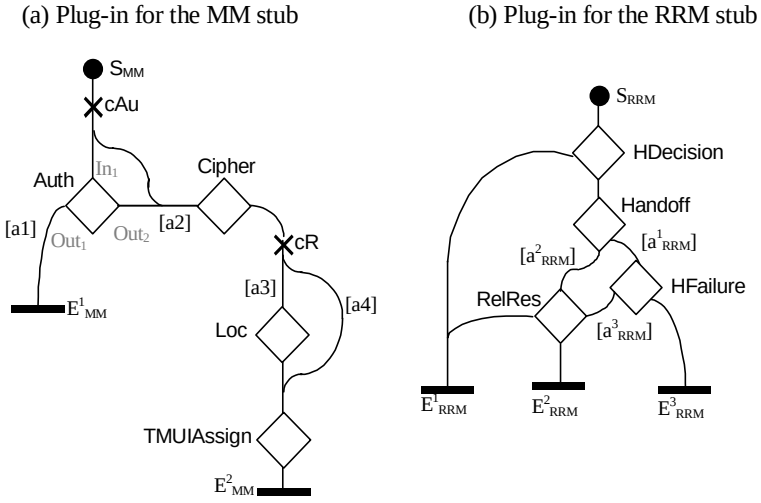


Figure 63 Plug-ins bound to MM and RRM Stubs

As mentioned earlier, a user is able to power off the mobile station or a network failure can occur at any time. These two different behaviors depend on external events and they are represented by different plug-ins bound to the dynamic stub labeled ExternalBs. These plug-ins are not shown in this thesis; however, they take over ([a1] zig-zag path label) the current map, which have started at the S_2 start point and has been performed, and release all the resources that are no longer needed. As a result of these external events (a power-off or a network failure), the map ends with the bar labeled E_1.

Figure 63a illustrates the plug-in for the MM static stub. This plug-in describes the following mobility management functions that are performed when a mobile user powers on: authentication (Auth stub), ciphering (Cipher stub), location registration (Loc stub), and TMUI assignment (TMUIAssign stub). First, the cAU responsibility checks whether the originating party needs authentication or not. After this, one of the following paths can be performed: <Auth, [a2], Cipher, cR, [a3], Loc, TMUIAssign, E^2_{MM}> [] <[a2], Cipher, cR, [a3], Loc, TMUIAssign, E^2_{MM}> [] <Auth, [a2], cR, Cipher, [a4], TMUIAssign, E^2_{MM}> [] <[a2], Cipher, cR, [a4], TMUIAssign, E^2_{MM}> [] <Auth, [a1], E^1_{MM}>.

In the figure, the Authentication plug-in, which is bound to the Auth stub, represents the *authentication* pattern (see Section 5.4.4 of Chapter 5); the Cipher stub encapsulates the *ciphering* pattern solution (see Section 5.4.3 of Chapter 5); the Location Registration plug-in, which is bound the Loc stub, graphically specifies the *location registration* pattern (see Section 5.4.7 of Chapter 5); and the TMUIAssign stub encapsulates the *temporary identification* pattern solution (see Section 5.4.1 of Chapter 5).

The Auth stub has two outgoing paths labeled [a1] and [a2] that correspond to the end points of the Authentication plug-in shown in Figure 34a (Out1 corresponds to E^1_{Auth}, and Out_2 corresponds to E^2_{Auth}). The ciphering plug-in is triggered after authentication (see Figure 33b) and then the cR (check registration) responsibility is activated along the [a2] sub-path to decide whether the mobile station is registered at the current location area or not. One of the alternatives sub-paths ([a3] or [a4] labels) is followed after this decision. This path ends (E^2_{MM} end point) after performing the location registration and the temporary identification assignment plug-ins (see Figure 35 and Figure 33a).

Figure 63b depicts the plug-in for the RRM static stub. The handoff decision is taken in the HDecision stub. The HFailure stub manages unsuccessful outcomes from the handoff functions (Handoff stub with the [a^1_{RRM}] outgoing path) and takes over all the scenarios in execution (E^3_{RRM} that corresponds to the [a6] zig-zag path shown in Figure 62). On the other hand, the RelRes stub is responsible for releasing the network resources that are no longer necessary and it handles the outcomes from the Handoff and HFailure stubs ([a^2_{RRM}] and [a^3_{RRM}] paths, respectively).

Figure 64a depicts the Communication Management plug-in bound to the CM static stub. As stated earlier, functionally, users are not identical and we make the distinction between the user who requested the establishment of the communication (originating party) and the user who answers the call (terminating party) with the plug-ins bound to the Party stub. The start points of the plug-ins bound to the dynamic Party stub are triggered differently according to their pre-conditions. This flow is triggered (S^1_{CM} start point) after either a user dials a number (the terminating party number), which corresponds to an outgoing call, or an incoming call request is issued. These two different user's behaviors are expressed in terms of the plug-ins bound to the Party dynamic stub, as follows: originating party (Figure 64b) and terminating party (Figure 64c).

Figure 65a illustrates how the call is routed to the terminating party. Since the dialed number does not refer to the terminating party current location, the iD responsibility interrogates the terminating home database about the current location and the status of the terminating party. Based on the current location, the cR responsibility checks the route to connect the two users and accepts the call based on the availability of resources (e.g., free interworking devices, or a free circuit with the external network). This responsibility generates successful or unsuccessful outcomes (respectively, E^1_{Rout} or E^2_{Rout} end points).

After finishing the routing function, the current status of the terminating party is analyzed. Figure 65b shows the Status map that is triggered by the resulting event of the E^2_{Rout} end point. The first responsibility checks the terminating party status (cS responsibility) and returns one of three different outcomes. The call request is denied (dR responsibility in the [b1] path) as a result of the terminating party subscription capability (incompatible with the originating party). The call is established ([b2] path) in case users are compatible and the terminating user is idle. Last, a busy tone is played at the originating user side (bS responsibility in the [b3] path), if the terminating user is currently busy.

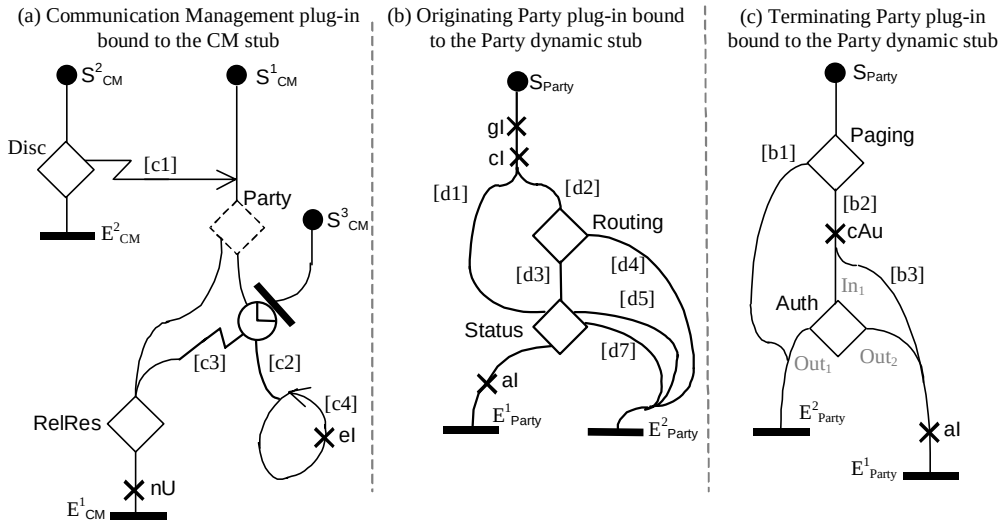


Figure 64 Plug-in bound to the CM stub and Plug-ins bound to the Party Stub

Figure 65c shows the Disconnection map that is responsible for releasing the network resources (rR responsibility) after a user hangs up the phone or presses the end key. The and-fork notation splits the main path into [c1] and [c2] sub-paths that are performed in parallel. The [c1] sub-path indicates to the other related maps in execution at this time that a disconnection has occurred and that the disconnection map has replaced the current UCM routes. The [c2] sub-path finishes the actions of the Disconnection map. The other user is notified about this disconnection with the nU responsibility.

Unsuccessful outcomes can occur either from the originating plug-in or from the terminating plug-in. In addition, they can occur after the timer expires as a result of the

terminating party not answering the call in a certain amount of time ([c3] zig-zag timeout path). After one of these unsuccessful outcomes, the actions of releasing resources (rR responsibility) and notifying the user (nU responsibility) are performed by the Communication Management plug-in. The E^1_{CM} end point represents these unsuccessful attempts of establishing a call between two users.

If the terminating party answers the incoming call, the call is finally established ([c2] path) and users are able to exchange information (voice, data, fax, or video), which is represented by the eI responsibility. The loop depicted by the [c4] path is performed until one of the users disconnects. The disconnection event can happen at any time and it is specified by the Disconnection plug-in (see Figure 65c) that is bound to the Disc stub.

Figure 64b shows the originating party plug-in that has alternative paths labeled by [d1] (a fixed user is the terminating party) and [d2] (a mobile user is the terminating party). These paths are chosen according to the following responsibilities: get the terminating user's number information (gI responsibility) and check this information (cI responsibility). The other alternative sub-paths describe successful and unsuccessful outcomes caused by the plug-ins bound to the Routing and Status stubs. For example, [d4], [d5], and [d7] are unsuccessful attempts to set up a call (Status stub) and [d3] is a successful outcome of the Routing stub. This originating plug-in is suitable for a scenario where the user tries to make a call soon after powering on the mobile station. In case of changing location, another plug-in with the MM stub should be available.

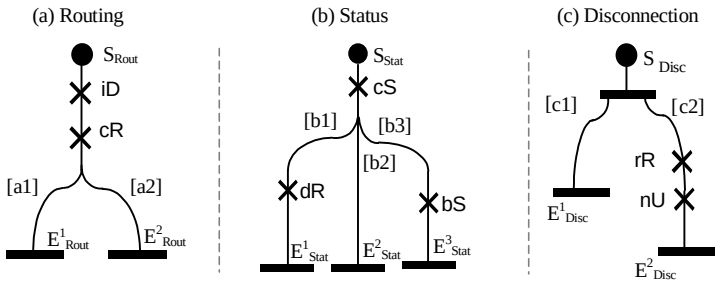


Figure 65 Unbound Plug-ins for Routing, Status, and Disc Stubs

After the unsuccessful outcomes, the Communication Management plug-in expresses the actions of releasing resources (rR responsibility) and notifying the user (nU responsibility). After the successful outcome of the Status stub, the aI responsibility (alerting indication) is played at the originating and terminating user sides, a timer starts and waits for the terminating user's answer (represented by the S^3_{Party} start point). The E^1_{Party} and E^2_{Party} end points represent, respectively, successful and unsuccessful establishment of calls between two users.

An alternative for the Party dynamic stub is illustrated in Figure 64c as follows: the mobile station behaves as the terminating party. The behavior of the terminating mobile user is different from the originating mobile user as follows: first, the Paging stub has to be performed and, then, the Auth stub can be performed. The paging plug-in, which is depicted in Figure 34b of Chapter 4, is bound to the stub. This plug-in corresponds to the

graphical specification of the *paging* pattern (see Section Figure 44 and Figure 45 of Chapter 5).

After describing these functionalities with unbound UCMs, the structural patterns, which represent the network entities (*security database*, *home and visitor database*, and *anchor mobile switching center*), and the other common architectural elements presented in Chapter 4 are mapped to these unbound UCMs yielding bound UCMs. The next section describes this mapping and illustrates a scenario where the originating and terminating are successfully engaged in a communication.

7.2.2 Mapping of the Functional Behavior to Structural Patterns and Network Reference Model

As mentioned earlier, the framework is a generic scenario ready to be re-used by mobile systems at the requirements and analysis development stages (see also the proposed approach in Chapter 6). At the requirements stage, the focus is on the system functional behavior. The system structural model comes into play at the analysis stage when details regarding which component performs each functional behavior become an issue. Figure 66 shows a set of UCM components that correspond to the network reference model generated from the common architectural elements presented in Figure 39 of Chapter 4. The components that are shown in gray in the figure are described as structural patterns in Chapter 5. The environment shown in Figure 39 of Chapter 4 is the foundation for the network reference model that is a reusable element for the mobile system development at the analysis stage.

The mapping of this UCM substrate to the unbound UCMs is presented in this research at the plug-in level. For instance, the responsibilities within the Routing, Status, Disconnection, Paging, Location Registration, and Authentication plug-ins are performed by the following components: originating mobile station, terminating mobile station, (anchor) mobile switching center, home database, visitor database, and security database.

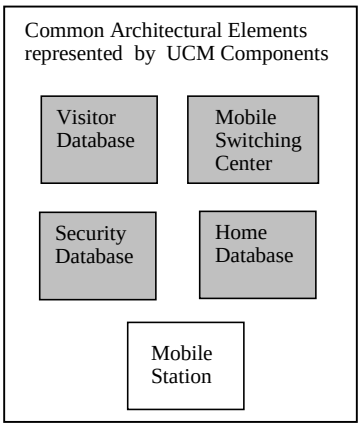


Figure 66 Network Reference Model represented by UCM Components

Chapter 5 presents the graphical specification of the following behavioral patterns with bound UCMs: *authentication*, *paging*, *location registration*, *inter-system handoff execution*, and *handoff failure actions*. As stated earlier, these UCMs correspond to the plug-ins that are bound to the following stubs in the framework: Auth, Paging, Loc, Handoff, and HFailure. Furthermore, Figure 67 adds components to the Routing, Status, and Disconnection plug-ins mentioned earlier.

Figure 68 depicts the execution of scenarios that are part of the proposed framework. The whole map describes the call establishment between the originating and the terminating parties. This map represents the binding of the plug-ins to their respective stubs that are depicted in Figure 64a, Figure 64b, and Figure 49 of Chapter 5. The UCM routes shown in gray are not performed in this map.

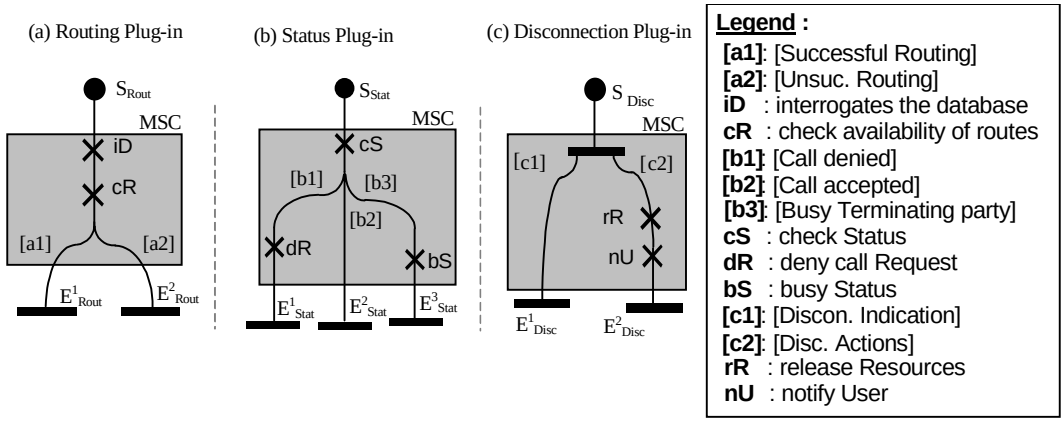


Figure 67 Routing, Status, and Disconnection Bound Plug-ins

First, the map depicted in Figure 68, which describes the originating party's execution, is triggered after satisfying the following pre-conditions (S^1 start point): the originating mobile user has already powered on and the mobility management functions have been successfully performed. After reaching the and-fork, radio resource and communication management functions are performed in parallel and independently.

The communication management flow starts when the originating user dials a number (the terminating party number) as follows: S^1_{CM} start point shown in Figure 64a followed by gI and cI responsibilities. In this scenario, the originating party's handoff is successfully performed (<aC, tNC, vC, uP, uTP, E^1_{Hand}> path). Furthermore, the originating network entities successfully perform the routing, the terminating party is not busy, and the originating party waits for the terminating user to answer the call (<iD, cR, cS, aI> path).

Meanwhile, the terminating party is not roaming (RRM functions are not performed at the terminating party's map). The communication flow starts at the terminating party's side when a paging is issued (S^1_{CM} start point shown in Figure 64a followed by pU, cA, cAu and aI responsibilities depicted in Figure 45a of Chapter 5 and in Figure 64c). The

terminating user answers the call (S^3 start point) that leads to the successful establishment of the call.

These scenarios are merged for the exchange of information (eI responsibility). When one of the users hangs up, the call finishes (as shown in Figure 64a by the Disc stub, whose plug-in is illustrated in Figure 65c).

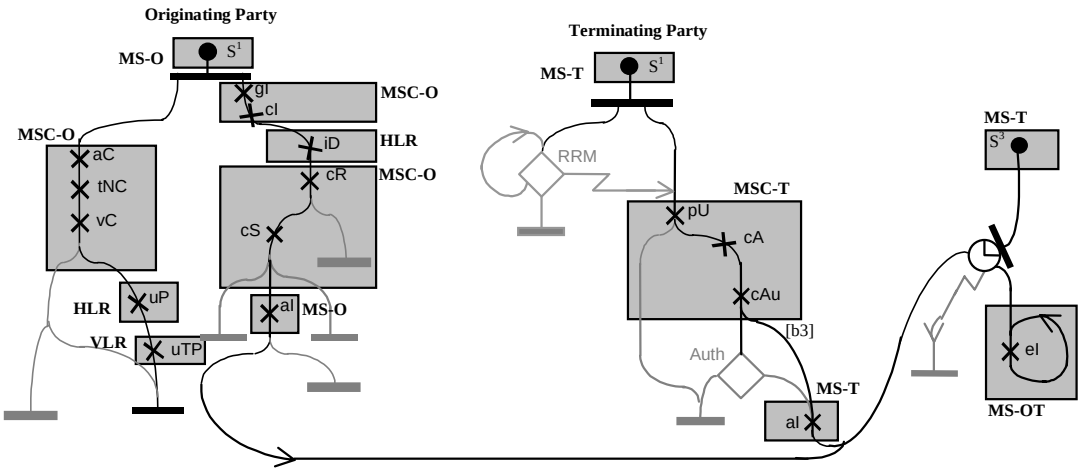


Figure 68 Integration of the Originating and Terminating Scenarios

The following components (network entities) perform the responsibilities depicted in the figure: Originating Mobile Station (MS-O), Originating Mobile Switching Center (MSC-O), Home Database (HLR), Visitor Database (VLR), Terminating Mobile Station (MS-T), Terminating Mobile Switching Center (MSC-T), Originating and Terminating Mobile Stations (MS-OT).

The framework intends to show the feasibility of the proposed approach for the early development stages. At the analysis stage, the mapping of functional behaviors to the network reference model can be done. This mapping allows the designer to do early simulations of the whole system. This framework is suitable to be re-used by existing systems such as the Global System for Mobile Communications (GSM) [139] and the American National Standards Institute – 41 (ANSI-41) [25] based systems (e.g., respectively, GSM-1800 and D-AMPS) when adding new features. Design decisions regarding messages, parameters, authentication algorithms, and interfaces among components are taken at later stages. The next sections present the application of this framework to the development of two systems.

7.3 A Family Member of IMT-2000 Systems: Second Case Study

This section presents a case study that integrates a service, which is described in the Wireless Intelligent Network (WIN) [174], in the framework presented in the last section.

The IMT-2000 generic reference model is used instead of the network reference model presented in Figure 66.

The development of this system begins with the addition of Incoming Call Screening (ICS), which is a variability for the communication management functions described in the framework. At the design stage (not shown here), other variabilities can be included in the models according to the IMT-2000 system documents [112][113][114].

As shown in Chapter 4, the behavioral and structural patterns are representative of the commonalities between third (e.g., IMT-2000 systems) and second (e.g., GSM) generation systems. Furthermore, ICS is part of WIN that is a second generation system. In this context, we consider that this prototype is a seamless environment for diverse mobile users and a family member of IMT-2000 systems whose capabilities should be compatible with second generation systems.

The next sub-sections show part of the requirements and analysis models of a simplified IMT-2000 family member in which mobile users can subscribe to the ICS service. According to our approach, a prototype is specified with UCMs and LOTOS, and then validated with LOTOS validation test cases.

7.3.1 ICS at the Framework Requirements Model

The overall functional behavior of a simplified IMT-2000 family member can be described using the requirement model shown in Figure 62 (the UCM framework root map). Then, variabilities are added according to the IMT-2000 stage 1 documents. For instance, besides mobility, communication and radio resource management functions described in the pattern language, the IMT-2000 requirement model consists of the addition of the ICS feature.

In [12] and [23], we present the graphical specification of the WIN ICS with unbound and bound UCMs. UCM components represent the mapping of the WIN distributed functional model to the network reference model. In our case study, only the unbound UCM that describes ICS is reused. Figure 69b illustrates the addition of ICS feature to the IMT-2000 originating party plug-in. At the analysis stage (see next sub-section), we map the IMT-2000 generic reference model presented in [113] to the unbound ICS plug-in as illustrated in Figure 70.

The Party dynamic stub shown in Figure 69a is bound to the Originating Party or Terminating Party plug-ins. In case a user has subscribed to the ICS service, the ICS plug-in (bound to the ICS stub that is described inside the CM stub) is performed (see Figure 69 and Figure 70). The behavior of the terminating mobile user (not shown here) is different from the one of the originating mobile user: paging and authentication are performed before sending the alerting indication to the user.

The cSF responsibility in Figure 69b is activated along the path to decide whether the mobile station has subscribed to the ICS feature or not. The alternatives sub-paths

(labeled [d1] and [d2]) are generated after this decision. The plug-in for the Routing stub in Figure 65 contains actions for the case where the called party is located in a place other than its home location.

The LOTOS requirements specification is generated from these unbound UCMs and validated with validation test cases. Section 6.4.2 and Section 6.4.3 of Chapter 6 describes these steps.

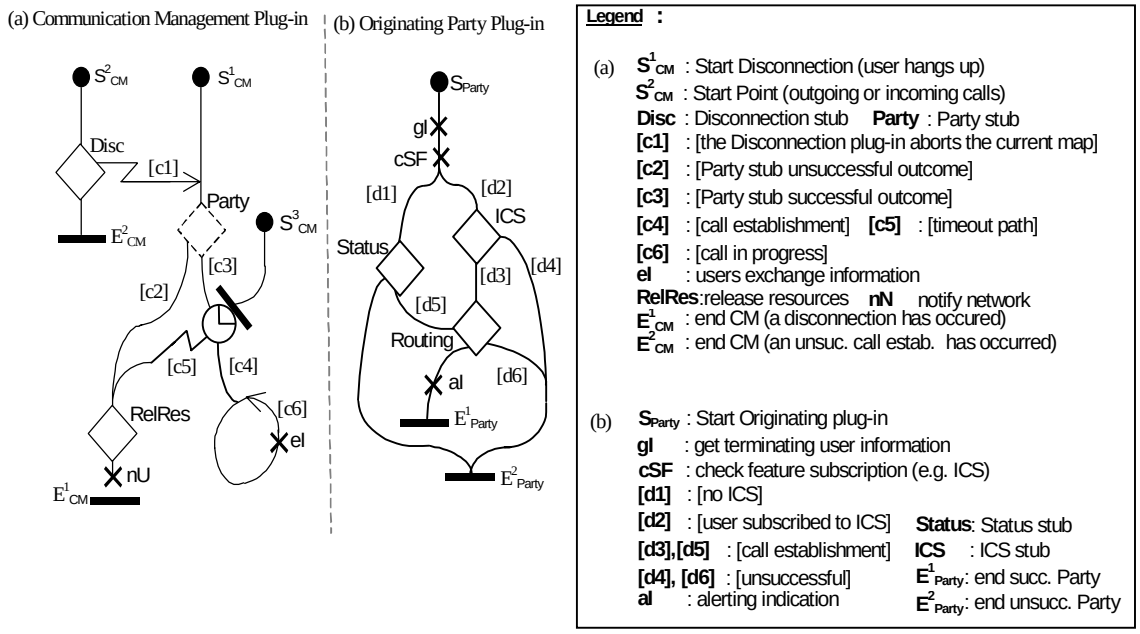


Figure 69 Communication Management plug-in (bound to CM stub) and Originating Party plug-in (bound to Party stub)

7.3.2 Analysis Model with bound UCMs

As mentioned in Chapter 4, the level of abstraction in which UCMs describe the pattern solutions makes them reusable at the requirement stages. As a result of these graphical specifications, decisions regarding architectural elements such as the IMT-2000 network entities depicted in Figure 15 of Chapter 2 that perform these functional behaviors can be taken at the analysis stage. Bound UCMs are then generated based on the combination of the unbound UCMs described at the requirements model and of the architectural elements described with the UCM component notation. The generic reference model of IMT-2000 systems includes all the structural patterns presented in Chapter 5.

In our case study, after validating the requirement model against the validation test cases derived from the commonalities (i.e., the behavioral patterns), the unbound UCMs are mapped to the IMT-2000 generic reference model.

A bound ICS plug-in, which describes the mapping of the WIN network reference model to the WIN distributed functional model, is presented in [12] and [23]. At the

analysis stage, the WIN reference model is replaced by the IMT-2000 generic reference model and the original ICS functional behavior is slightly modified (e.g., Loc responsibility and Routing stub are moved to the Originating Party plug-in depicted in Figure 69b). Figure 70 depicts the IMT-2000 network entities involved with ICS. In order to compare the differences between the WIN model and the IMT-2000 model, the reader may refer to Chapter 2.

The UCM components depicted in Figure 70 represent the mapping of the FEs described in the IMT-2000 Distributed Functional Model to the Network Entities (NEs) [113]. The following FEs are illustrated in the figure: Service Data Function (SDF), Service Control Function (SCF), and Specialized Resource Function (SRF). In addition, the following NEs incorporate one or more of these FEs: Intelligent Peripheral (IP) and Service Control Point (SCP).

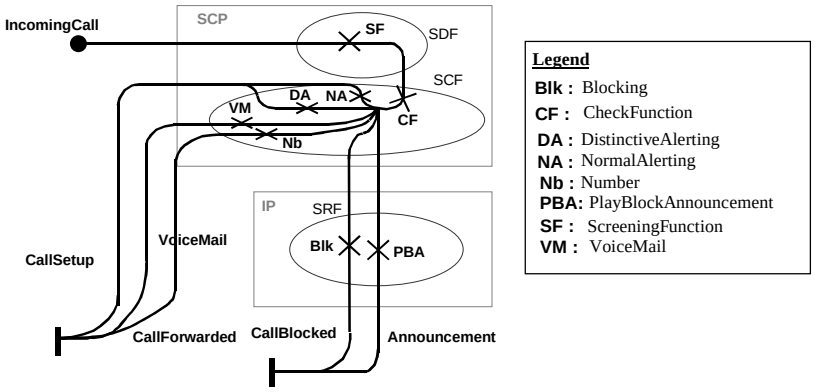


Figure 70 ICS plug-in mapped to the IMT-2000 generic reference model

Without getting into the details and subtleties of the analysis model for the ICS plug-in, its interpretation is essentially as follows. An incoming call attempt causes the request to be analyzed (see also Figure 69b). This causes the execution of a check function to determine whether the ICS service is active or not (cSF is performed by the LMF_H in the terminating subscriber's HLR). If it is inactive, then the regular call setup scenario continues: the location of the called party is determined, the call is routed (Routing stub), and this results in the continuation of the call setup.

When ICS is active, the screening function is checked (SF, by the SDF in SCP) and performed (CF, by the SCF). This can result in the continuation of the call setup with or without a distinctive alerting (DA or NA), in a redirection to the voice mail (through VM, by the SCP's SCF), in the forwarding of the call to another subscriber (through Nb), in a specific announcement (through PBA in the intelligent peripheral's SRF), or in the call being blocked (through Blk). These five end points are the possible outcomes of ICS and they are bound to their respective outgoing paths in the originating party plug-in (see [d3] and [d4] ICS stub outcomes in Figure 69b).

7.3.3 Specification and Validation with LOTOS

LOTOS specifications are first generated from the unbound UCMs and then from the bound maps. Section 6.4.2 of Chapter 6 explains this process in detail.

Figure 71 depicts part of the LOTOS formal model that is generated from the root map illustrated in Figure 63. The UCM casual paths between the stubs are represented by the following LOTOS operators: enable (sequence of stubs), choice (alternative outgoing paths), and disable (abort path).

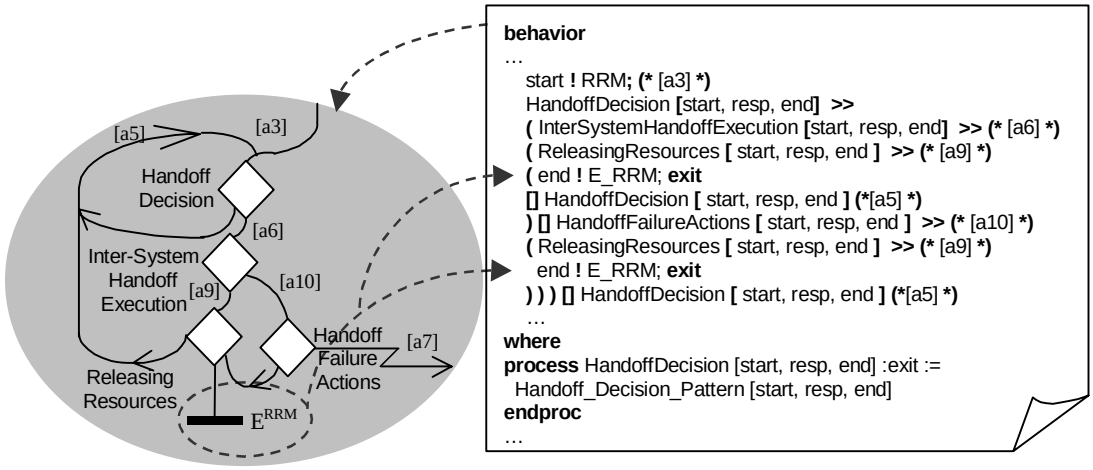


Figure 71 From Unbound UCMs to LOTOS

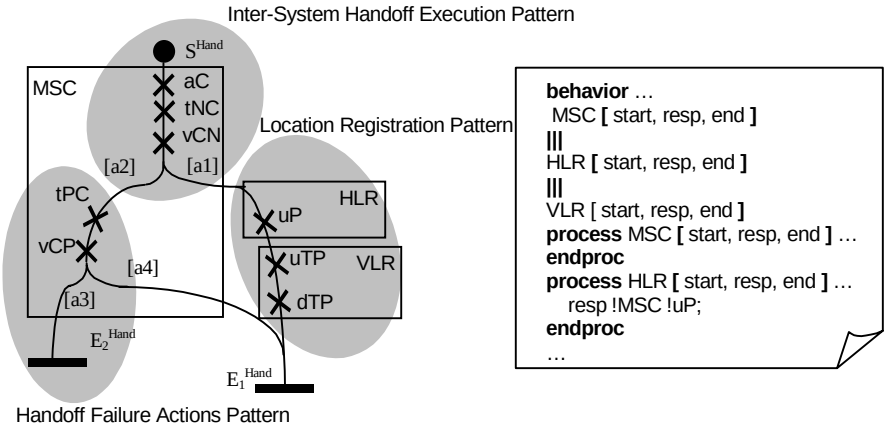


Figure 72 From Bound UCMs to LOTOS

Figure 72 depicts part of the LOTOS specification of the analysis model, to address the addition of MSC, HLR, and VLR processes to the LOTOS requirements specification. At the design stage (not developed in this case study), the LOTOS analysis specification is modified with details about exchanged messages, interfaces (represented by LOTOS

gates) between processes, parameters, and data types. The inter-system handoff execution pattern solution in modified in our case study to include part of the location registration pattern solution and another verification of connection. These are variabilities added to the requirements model.

For the validation part, the UCM validation scenarios are translated into LOTOS validation test cases as shown in Figure 73 (see Section 6.4.3 of Chapter 6).

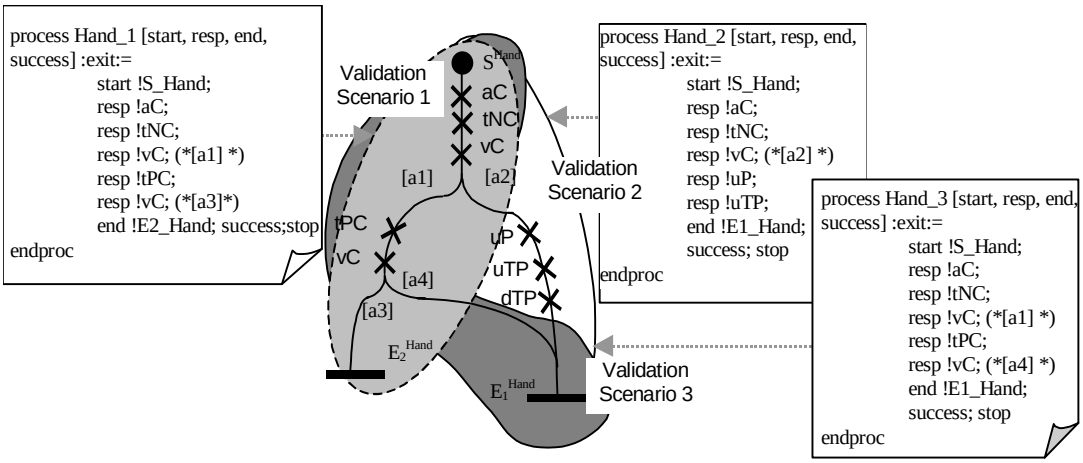


Figure 73 Validation Test Cases

7.4 Wireless Mobile ATM Networks: Third Case Study

In this research, requirements and analysis models for overlay signaling alternatives of wireless mobile ATM networks were developed using the reverse and forward engineering approaches presented in Chapter 4. These models were investigated and *common* functional behaviors and architectural elements between these models and the other chosen systems were identified, captured, and documented as patterns in this research. However, commonalities between WmATM networks and the other systems were not always identified (see Appendix A). For instance, WmATM networks do not use the *temporary identification*, *visitor database* and *security database* patterns.

In [21], we show how a simplified Wireless mobile ATM network (WmATM) can be specified with UCMs and LOTOS as well as validated using LOTOS techniques. In addition, we present how the validation results can be translated to the MSC notation. The requirements and analysis models mentioned earlier were used to generate the WmATM network design model. In the case study that we present in this section, this design model is modified to include pattern solutions.

First, this section introduces the addition of two patterns to these requirements and analysis models and then presents a behavior model, which is specified in LOTOS, and a

scenario model, which is described with MSCs. Validation scenarios are provided with the reuse of the pattern solutions and are validated against the LOTOS design specification (see Figure 57 in Chapter 6). Besides providing a better and more precise description of the system at the early stages in comparison with the original documents, our goal is to show how the proposed approach helps to improve the existing signaling protocol alternatives with pattern reuse and validation. In Section 7.4.4, we discuss the identification of design problems such as inconsistencies and ambiguities and errors in the original documents and how we solve them in the prototype.

As mentioned earlier, the WmATM network features were reverse-engineered from the description of WmATM overlay signaling protocols presented in [4] and [5]. The WmATM network procedures are gradually modified in terms of sequential actions with *unbound* UCMs (the requirements model) followed by more details about the system behavior and the addition of the *visitor database* structural pattern to the WmATM reference architecture with *bound* UCMs (the analysis model).

The next sub-sections presents the evolution of the WmATM prototype mentioned in Chapter 4 with the modification of the *authentication* function and the addition of the *visitor database* patterns. During the development of the WmATM network environment, each component of the reference architecture is specified with its corresponding functional behavior related to mobility management functions. Modifications related to the other functions (communication and radio resource management functions) as well as exceptions (such as network failure, lack of network resources, and database failure) follows the same steps presented in the next sections, but they are not considered in this research. The next sub-sections also show these procedures in the root map, but our focus is on the mobility management function. Furthermore, we concentrate on the description of the design stage for this second case study.

7.4.1 Requirements Model: Unbound UCMs

At the requirements capture stage, the modification is done in the original authentication procedure on the basis of the *authentication* pattern solution. Our goal is to improve the security in the air interface. Instead of sending the authentication key through the radio ports, the new requirements model assumes that the authentication key has been already stored in the mobile station and in the network side. The result of the user authentication calculation is then sent through the air interface with the mobile station's identity.

The WmATM network root map is similar to the first case study root map (see Figure 62). Figure 74 depicts the second level of the requirements model when mobility management procedures are decomposed into small units (a plug-in for the MM stub). These small units are represented by the Auth and Update *stubs* and they are grouped into the Location Registration *plug-in* that partially corresponds to the plug-in bound to the MM stub depicted in Figure 35 of Chapter 4.

The Location Registration *plug-in* is in turn bound to the MM *stub* in the WmATM root map. The cR responsibility point is activated along the [b2] path to decide whether

the mobile station is registered or not in the current location area. The alternative sub-paths (labeled [b3] and [b4]) are generated after this decision. Auth stub has two outgoing paths labeled [b1] and [b2] that correspond to end points of the authentication *plug-in* (respectively, unsuccessful and successful outcomes). The Update stub groups all the functions related to updating user information.

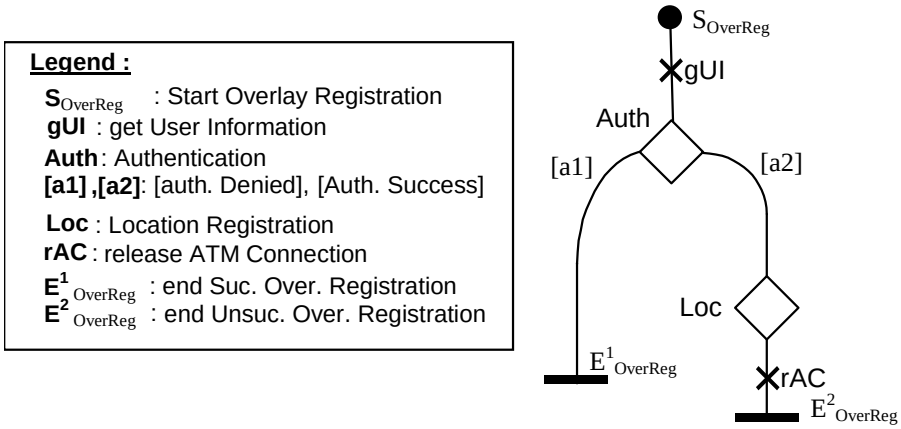


Figure 74 (a) Location Registration *Plug-in* for MM *Stub*

The WmATM environment considered in this thesis is illustrated in Figure 16 of Chapter 2. The WmATM reference architecture includes *mobile stations* (called portable in [4] and [5]), *WmATM switches* and *databases*. An *ATM network* composed of *ATM switches* and *fixed stations* is also described to allow the communication between fixed and mobile stations.

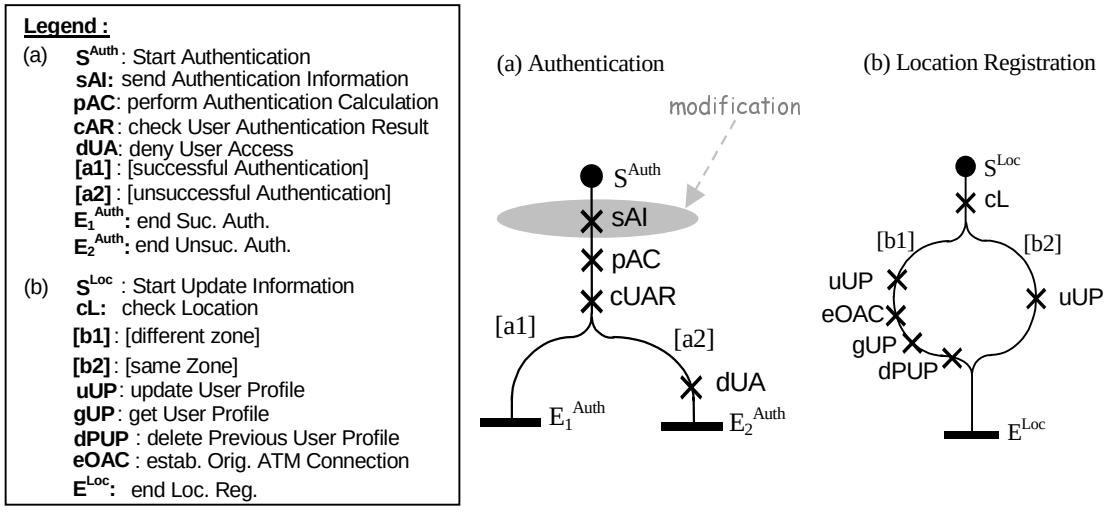


Figure 75 Unbound UCMs: Authentication and Location Registration *Plug-ins*

Since our focus is on signaling protocols for the application layer, *base stations* are not considered. Mobile stations communicate directly with WmATM switches and mobility occurs every time the mobile station changes a location area (called zone in [4] and [5]).

The user roaming is represented by changing the WmATM switch. With respect to the communication management functions, this case study concentrates only on the specification and validation of connections between mobile users and between mobile and fixed users.

At the *analysis* stage (see Figure 56 in Chapter 6), details regarding the WmATM architectural elements and their functional behaviors are added to UCMs. Figure 75 and Figure 76 depicts unbound (requirements model) and bound plug-ins (analysis model) for the authentication and location registration procedures. The WmATM achitectural elements that are mapped to the *unbound UCMs* described at the requirements model represent the network reference model depicted in Figure 16 of Chapter 2. The following WmATM *components* are involved in the authentication and update information functions: *Mobile Stations*, *WmATM switch*, ATM switch, *Home Database* and *Visitor Databases*.

Figure 75a depicts the unbound Authentication *plug-in* that is modified on the basis of the *authentication* pattern solution presented in Chapter 5. The modification consists of replacing the gAI responsibility (gAI stands for get authentication information) by the sAI responsibility, which avoids sending the authentication key through the air interface. sAI is a *common* functional behavior among GSM/GPRS/UMTS, ANSI-41/WIN and IMT-2000 systems (see a list of commonalities in Appendix A). This modification is done at the requirements stage.

The functional behavior shown in Figure 75a is now similar to the authentication description presented in Chapter 4 and Chapter 5. First, the mobile station calculates the user authentication and sends the authentication result to the WmATM network (sAI responsibility). Then, the pAC responsibility performs the same authentication calculation on the network side. The cUAR responsibility generates the successful or unsuccessful outcomes (respectively, E_1^{Auth} or E_2^{Auth} end points). In case of denied authentication, the mobile user is notified (dUD responsibility). Figure 76a illustrates the mapping of the WmATM architectural elements to this unbound UCM.

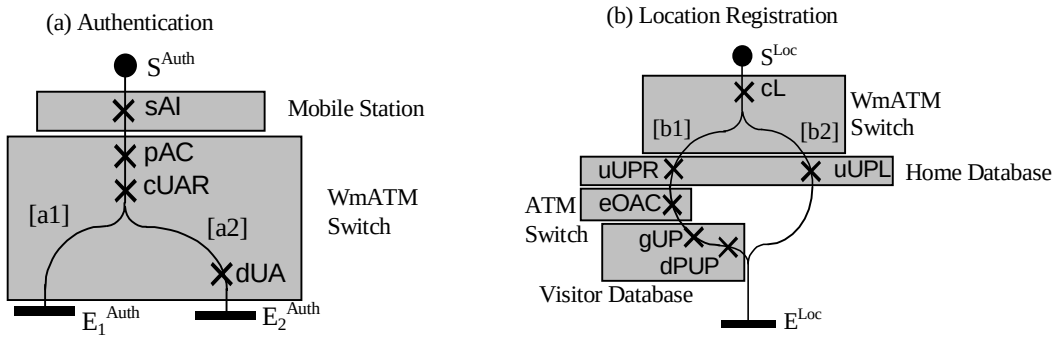


Figure 76 Bound UCMs: Authentication and Update Information *Plug-ins*

Figure 75b shows what happens inside the Update stub that partially corresponds to the *location registration* pattern solution (see also Appendix A) and Figure 76b depicts the architectural elements mapped to these functional behaviors. The cL responsibility generates different outcomes according to whether the mobile user is roaming or not. uUPL, uUPR, gUP and uTP responsibilities are operations on home and visitor database items. Sub-paths labeled [b1] and [b2] are concatenated after these operations. When the user is roaming, an ATM connection is established between the home network and the serving network (the eOAC responsibility).

The bound UCMs are translated into a LOTOS analysis specification following the construction rules presented in Section 6.4.2 of Chapter 6. Validation test cases are generated from the authentication pattern and from other functional behaviors described in the WmATM network according to the proposed approach (see Section Figure 60 of Chapter 6).

```
process Scenario1_Auth [start, resp, end,
Scenario1_Auth] :exit:=
        start !MS !S_Auth;
        resp !MS !sAI;
        resp !WSH !pAC;
        resp !WSH !cUAR;
        end !E1_Auth; (*[a1] *)
        Scenario1_Auth;stop
endproc
```

```
process Scenario2_Auth [start, resp, end,
Scenario1_Auth] :exit:=
        start !MS !S_Auth;
        resp !MS !sAI;
        resp !WSH !pAC;
        resp !WSH !cUAR;
        resp !WSH !dUA; (*[a2] *)
        end !E2_Auth;
        Scenario2_Auth; stop
endproc
```

Figure 77 LOTOS Validation Test Cases for Authentication

7.4.2 Design Model: WmATM Network Specification and Validation with LOTOS

At the design stage, a formal model is manually generated on the basis of the previous models described in the last section.

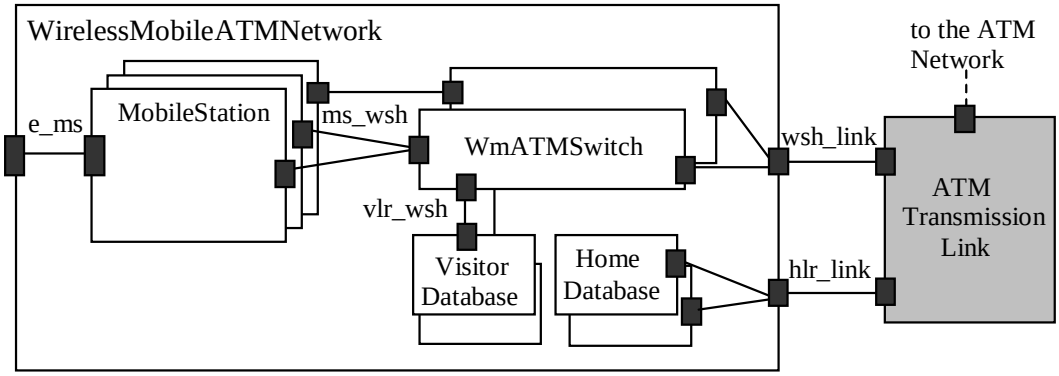


Figure 78 Graphical Representation of the LOTOS Specification Architecture

Figure 78 depicts the highest level of abstraction of the LOTOS specification, which is composed of *Wireless mobile ATM Network, ATM Transmission Link* (depicted in gray in

the figure to differentiate from the processes described also as UCM system components at the previous stages) and *ATM Network processes* (not shown in the figure).

Wireless mobile ATM Network includes *mobile stations* (originating and terminating sides), *WmATM switch*es (same process for previous and current *WmATM switches*), *home databases* (referred to Home Location Register - *HLR* in the specification), and *visitor databases* (referred to Visitor Location Register - *VLR*) *sub-processes.* These processes are synchronized through the following gates: ms_wsh, vlr_wsh, wsh_link, and hlr_link. *ATM network process* contains *fixed stations* and *ATM switches* connected through gate fs_sh. *ATM transmission link* process is responsible for the communication between the *home database* and the *WmATM switch*. In addition, this process provides the communication among *WmATM switch processes* (through gate wsh_link) and among *ATM switches processes* (through gate sh_link). Gates e_ms and e_fs provide the interaction of mobile and fixed stations with the environment (for simulation purposes). *ATM Network* process includes the ATM switch component depicted in Figure 77.

```
behavior
hide ms_to_wsh, vlr_to_wsh, hlr_to_link, wsh_to_link, fs_to_sh, sh_to_link
in
(( WirelessMobileATMNetwork [e_to_ms, ms_to_wsh, wsh_to_link, vlr_to_wsh,
hlr_to_link] ||| ATMNetwork [e_to_fs, fs_to_sh, sh_to_link]    )
     |[wsh_to_link, sh_to_link, hlr_to_link]|
    ATMTransmissionLink [wsh_to_link, sh_to_link, hlr_to_link] )
where
process WirelessMobileATMNetwork [e_to_ms, ms_to_wsh, wsh_to_link,
vlr_to_wsh, hlr_to_link]: exit :=
   ( (* users power on the mobile station *)
   ( MobileStation [e_to_ms, ms_to_wsh] (user_A, info_A, zone_1, hlr_1, 0)
 ||| MobileStation [e_to_ms, ms_to_wsh] (user_B, info_B, zone_1, hlr_1, 0)
 ||| MobileStation [e_to_ms, ms_to_wsh] (user_C, info_C, zone_2, hlr_2, 0)
)
 |[ms_to_wsh]|
   ((WmATMSwitch [ms_to_wsh, wsh_to_link, vlr_to_wsh] (zone_1)
 |[vlr_to_wsh]| VLR [vlr_to_wsh] (vlr_1, InitialVLRSet1))
 |||(WmATMSwitch [ms_to_wsh, wsh_to_link, vlr_to_wsh] (zone_2)
 |[vlr_to_wsh]| VLR [vlr_to_wsh] (vlr_2, InitialVLRSet2))     )
 |||(HLR [hlr_to_link] (hlr_1, InitialHLRSet1)
 |||HLR [hlr_to_link] (hlr_2, InitialHLRSet2) )   ) …
```

Figure 79 Highest Level of Abstraction of the LOTOS Specification

Figure 79 depicts how these processes synchronize through the gates (||| represents the interleaving operator and |[gate list]| the selective interleaving parallel operator). The use of these LOTOS operators allows process synchronization and the ability to simulate and test the whole system behavior as shown in the next section (see details about the LOTOS notation in Section 6.3.1 and about LOTOS validation techniques in Section Figure 60 of Chapter 6).

Data types direct information exchange among processes. In particular, each MobileStation is identified by its identification number (user_A, user_B, and User_C in the figure), electronic serial number, random variable, secret key (these identifiers are represented by info_A, info_B and info_C), home database (hlr_1 and hlr_2) and current

zone (zone_1 and zone_2). Each WmATM switch has its identification (zone_1 and zone_2 in the figure). HLR and VLR processes keep the identity and information about mobile stations in a set of database record (called HLRRecSet and VLRRecSet, respectively).

As mentioned earlier, the WmATM functional behavior modifications start at the requirements and analysis stages with the modification of the original authentication procedure (getAuthInfo responsibility is replaced by sAI responsibility) and the addition of *visitor database*. Unbound and bound UCMs are the resulting models of these stages together with the LOTOS requirements and analysis specifications. Figure 75, Figure 76, Figure 77, and Figure 80 depict these steps partially. After this, both informal descriptions and information flows presented in [4] and [5] are considered to add design details such as data types and parameters to the LOTOS analysis specification that evolves to the LOTOS design specification.

The system behavior starts at any time after the user powers on the mobile station. Figure 80 depicts part of the behavior of the MobileStation process when a mobile user powers on and authentication and update information plug-ins are triggered (see also Figure 77).

```
process  MobileStation [e_to_ms, ms_to_wsh] (usrid: UserIDN, userInfo:
InfoIDN, myzoneid: ZoneIDN, hlrid:DatabaseIDN, n: Nat) :exit :=
( … e_to_ms !usrid ?czid:ZoneIDN;  (* change location area *)
    ( [h(myzoneid) ne h(czid)] ->  (* registration begins *)
      (  ms_to_wsh !usrid !czid !InitiateRegREQ;
         ms_to_wsh !usrid !czid ?M:Message [h(M) eq h(InitiateRegCONF)];
            (* authentication process takes place *) …
         ms_to_wsh !usrid !czid !hlrid !r !AuthUserResult;
         ms_to_wsh !usrid !czid ?M:Message;
         ( [h(M) eq h(AuthSuccess)] ->
           MobileStation [e_to_ms, ms_to_wsh] (usrid, userInfo, czid,
hlrid, 0)
           [] [h(M) eq h(AuthDenied)] ->
            … MobileStation [e_to_ms, ms_to_wsh] (usrid,…,czid, hlrid,
n)))) … )
      [>  e_to_ms !usrid !myzoneid ?M:Message[h(M) eq h(PowerOff)]; stop
      >> MobileStation [e_to_ms, ms_to_wsh] (usrid, myzoneid, hlrid, 0)
endproc (* MobileStation *)
```

Figure 80 Partial Behavior of the *MobileStation* Process

As mentioned in Chapter 6, LOLA provides the following tools to validate a LOTOS specification: simulator (or debugging) tools to simulate the behavior step by step and to evaluate data value expressions; and testing tools to calculate the response of a system specification to a test according to testing equivalence [36]. These tools are applied to the WmATM network specification. The validation test cases are generated not only from the authentication pattern but also from the location registration plug-in. The testing tools include an expansion transformation tool that is used to generate successful or unsuccessful LOTOS trace. The next sub-section presents MSC scenarios that are automatically generated from such traces.

7.4.3 Scenarios with Message Sequence Charts

MSCs scenarios are generated automatically from the LOLA validation traces using the LOTOS2MSC converter. According to the restrictions of this converter related to the direction of message exchanges, the gate names are replaced by process1_to_process2 instead of process1_process2 as shown in Figure 78 (see details in Section 6.4.4 of Chapter 6).

Figure 81 depicts a validation test case for a registration request sent by the mobile station to the WmATM switch through ms_to_wsh and wsh_to_ms gates. These gates represent the air interface. The LOTOS behavior model is composed in parallel with this validation test case using the LOTOS interleaving operator (see Section 6.3.1 of Chapter 6). LOLA reports whether the execution reaches the success event (scenario1 in the figure) or not. Our goal is to test whether the behavioral model conforms to the initial registration requirements described by the authentication behavioral pattern.

```
process Scenario1[e_to_ms, ms_to_wsh, wsh_to_ms, hlr_to_link,
wsh_to_link, vlr_to_wsh, scenario1] :noexit :=

      e_to_ms !user_a !roaming !zone_2;
      ms_to_wsh ! user_a ! zone_2 ! initiateregreq;
      wsh_to_ms ! user_a ! zone_2 ! initiateregconf;
      ms_to_wsh ! user_a ! zone_2 ! hlr_1 ! authuserresult;
      scenario1; stop   (* success *)
   endproc
```

Figure 81 A LOTOS Validation Test Case for Registration Request

Figure 82 illustrates a *scenario model* that represents the new authentication functional behavior of the WmATM overlay signaling alternative. This MSC is generated from the results of the LOTOS *behavioral model* validation. LOLA tools generate trace files that show the sequence of actions executed during the validation part. In case of rejected execution, these sequences can be analyzed to discover errors in the specification. In case of positive results, scenario models are ready to be reused for the implementation. For the sake of clarity, we represent WmATM switch process as current WmATM switch in the figure.

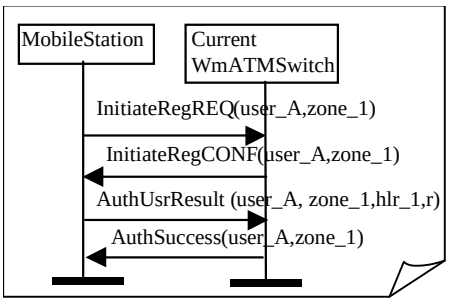


Figure 82 Scenario Models of a Successful Authentication Outcome

7.4.4 WmATM Signaling Protocol Improvements

As discussed in Section 4.2 of Chapter 4, the notations used in the development of WmATM networks are inappropriate to iteratively and gradually add details during different development stages while checking for ambiguities, inconsistencies, and undesirable interactions. The reuse of good solutions, which are described in the behavioral and structural patterns, to solve certain design problems of WmATM networks (e.g., security problems) is one of the contributions achieved in this case study. Validation test cases derived from the chosen pattern solutions provide confidence in the reuse of these patterns.

The improvements in the overall quality of the existing WmATM signaling protocols are also noticed with the detection of incomplete and undesirable behaviors in the original system (see Figure 83). As proposed in [10], [13] and [18], the combination of different techniques at the appropriate stages of the WmATM system evolution is the reason for these protocol enhancements. Figure 83a depicts the original information flows presented in [4] for the description of the registration procedure, which includes authentication. Figure 83b illustrates the MSC generated from the validation results of the LOTOS design specification for the description of the new authentication procedure after the reuse of the *authentication* pattern solution.

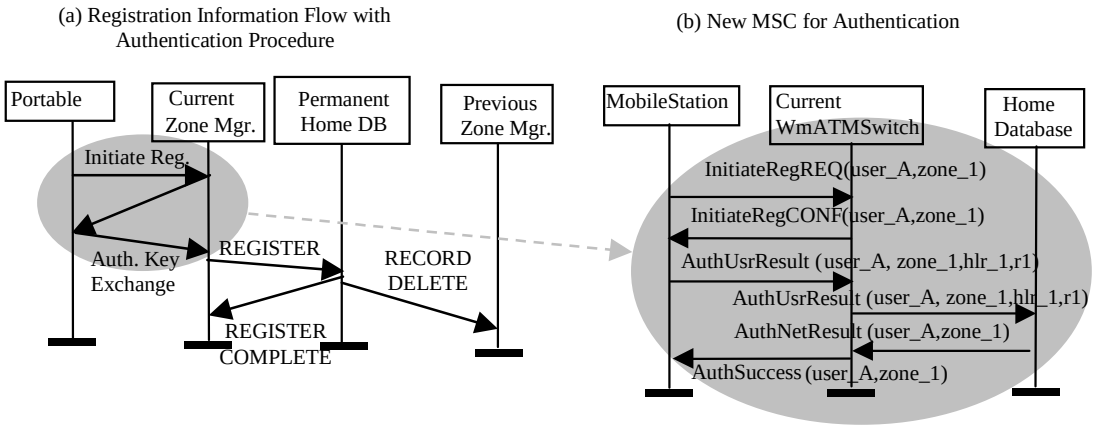


Figure 83 Original WmATM Information Flows from [4] and Generated MSCs

The registration information flow presented in [4] and [5] with the MSC generated in our case study are depicted in Figure 83a and Figure 83b, respectively. By comparing them, certain design decisions make the latter scenarios clearer and more complete. For instance, the authentication result is initially calculated at the MS side, sent to the current switch, and then calculated and compared at the network side thereby guaranteeing security through the air interface. The MSC also shows more details about parameters, which makes the protocol implementation easier.

Design decisions such as the use of the same procedure for connection establishment between two mobile users and between mobile and fixed users are also taken when

maintaining the WmATM network models. These design decisions minimize duplicated behaviors in the original documents (e.g., three similar connection establishment procedures are described separately for the communication between wireless users, wireless and fixed users, and fixed and wireless users). These specification errors are detected and corrected using the LOLA simulator and testing tools at different development stages. We believe that a better design decision would have been to represent these procedures using the IN conceptual model (originating and terminating parties) as shown in the first case study.

Finally, some inconsistencies related to the call establishment, which were found when reverse engineering information from the existing WmATM signaling protocols, are corrected in this case study. For instance, the existing signaling protocol mixes the location resolution with the status functions (e.g., call blocking due to routing problems and call blocking due to terminating user's preferences).

7.5 Discussion

With the reusability and the framework concepts in mind, this research proposes a requirements application framework for mobile systems. This framework is developed according to the MoRaR pattern language introduced in Chapter 5 and the remaining common functional behavior and architectural elements (also called network entities in this work).

The MoRaR pattern language has informally described the relationships among patterns (see Figure 40 of Chapter 5). The framework mentioned above graphically specifies these relationships with Use Case Maps (UCMs). Each pattern solution is considered as a single framework building block, which is represented by UCM plug-ins bound to stubs. When these patterns are combined with the common functional behavior and the network entities that are not described as patterns in Chapter 5, they constitute the requirements application framework.

This framework offers not only the reusable single building blocks but also the graphical specification of the relationships among them. Furthermore, this framework provides the mapping of the functional behaviors to the network reference model. This is our simplified view of frameworks that represent a repository of common good solutions for similar design problems among mobile systems. At the requirements and analysis stages, these solutions are ready to be reused by designers when building new mobile systems or adding new features to existing ones.

As a case study, the proposed requirements application framework has been already applied to develop a wireless mobile ATM network and the results are presented in Chapter 7 and published in [19]. As mentioned earlier, the proposed framework is suitable for requirements capture and analysis development stages.

The application of the proposed approach for reuse and validation of the MoRaR pattern language to other domains such as robotics, which also includes mobility and radio resources procedures in their system design, is left as an open area of research.

There are efforts to make the Internet also useful for mobile users and to enable IP to support voice in addition to data. For instance, mobile IP is used to support mobility in the Internet as well as data [149]. Recently, a new group called Mobile Wireless Internet Forum (MWIF) has been working on the integration of mobile IP and voice.

Mobile IP was proposed to support mobility functions for portable computers enabling them to support Internet when people move from place to place with their laptop. The main idea is to allow a laptop to be carried anywhere inside a wireless LAN and be able to access a personal Internet account without need to change manually its configuration. In addition, a multi-modal physical layer will provide independence of the transmission medium (infrared, radio LAN, mobile phone or even fixed medium such as ethernet and token ring).

On the other hand, voice over IP is an attempt to integrate the Internet with existing fixed telecommunication networks. The lack of communication among different designer groups has caused the incompatibility between these two systems. We believe that it is feasible to improve the efforts to build a mobile wireless Internet capable to support voice and data on the basis of the MoRaR pattern language.

Furthermore, another case study that is left for future research consists of the development of a family member of the IMT-2000 systems that includes IP networks (the so-called all over-IP IMT-2000 family member). The term hybrid networks has been used in the literature to describe this combination of mobile wireless networks, fixed telecommunication networks, and IP networks [97]. This proposal is under development within The ITU-T subgroup that has been developing IMT-2000 systems.

7.6 Conclusion

A seamless environment is developed in the first case study with the reuse of the MoRaR pattern language at the requirements (reuse of the behavioral patterns) and the analysis (reuse of the structural patterns) stages. These requirements and analysis models are also validated. In addition, this chapter shows how the reuse and the validation of behavioral and structural patterns (respectively, temporary identification and visitor database patterns) improve the design of an overlay signaling alternative for WmATM networks, which is presented in [4].

The goal of the first case study is to show the feasibility of developing a seamless system on the basis of the MoRaR pattern language. At the *requirements capture* stage, the *unbound* UCMs that represent the behavioral patterns are reused. As mentioned in Chapter 6, this model focuses on the causality relationship between responsibilities, without considering components. At the *analysis* stage, the system components that are

the structural patterns combined with a third generation network reference model and other behavior details (i.e., variabilities) are added to these maps generating the *bound* UCMs.

This case study indicates how the patterns and their relationships can be reused to facilitate the integration of second and third generation systems. UCMs graphically represent a high-level solution for the integration of an IMT-2000 family member and the WIN ICS service. This map is formally specified and validated using LOTOS at the requirement and analysis stages. The LOTOS specifications are validated against the validation test cases that are generated from the pattern solutions.

The second case study adds new functional behavior and new architectural elements, which are part of the MoRaR pattern language, to the WmATM network requirements and analysis models that are described using the approach presented in Chapter 4. Furthermore, a formal specification is developed with LOTOS at the *design* stage. The design model not only shows an example of the validation part but also how many possible behaviors can be described concurrently with details such as data types, parameters, and specific events. A set of LOTOS tools is used to validate the completeness of the system and verifies correctness and consistency properties. MSC scenarios are automatically generated from the results of the LOTOS validation in order to facilitate future protocol implementation as well as to analyze the benefits of the pattern reusability in comparison with the original documents.

The motivation for choosing third generation systems and WmATM networks as case studies resides in two factors as follows: they are still under development and they provide a reasonable amount of information about their protocol alternatives. These factors make the production of the design prototype feasible. As a result, this chapter presents the usefulness of patterns in the development and evolution of mobile systems.

Chapter 8 Conclusions and Open Areas of Research

This thesis provides an approach for the capture, the reuse, and the validation of behavioral and structural patterns for mobility and radio resource management. These patterns, which are grouped into the MoRaR pattern language, are captured from the commonalities identified among second and third generation systems. In addition, different case studies address the application of the proposed approach for the reuse of the MoRaR pattern language and for the validation of the pattern solutions at the early stages of system development and evolution. This chapter presents a summary of our contributions and of the open areas of research that arise from this thesis.

8.1 Introduction

This research aims at providing designers of mobile wireless communication systems a foundation for their requirements and analysis models when they are either building new systems or maintaining existing systems with the addition of new features.

This foundation is offered with the set of patterns that capture common problems and solutions (presented in **Chapter 5**) of second and third generation systems such as GSM-900 and IMT-2000 (details are shown in **Chapter 2**). These patterns give the designers a way to identify similarities and differences among these systems. As a result of the recognition of the commonalities among existing systems, the designers can iron out differences and either enable a new system to interwork with existing systems or facilitate the addition of new features in existing systems.

The pattern language, which groups the behavioral and structural patterns, helps the development and evolution of systems by showing the relationships among the patterns. These relationships are also described in terms of a Use Case Map (UCM) framework for mobile system requirements and analysis models presented in Chapter 7. An approach for the reuse and validation of the pattern solutions is also proposed. Case studies shows the reuse and validation of commonalities starting from the early stages of the system development and evolution is a promising way to overcome potential incompatibilities between existing and new systems.

In addition, we believe that another application of this research for the mobile system domain is to offer means for the development of the global roaming capability and seamless mobile wireless services as discussed in [86].

The next section summarizes the contributions of this thesis and Section 8.3 introduces the open areas of research generated from this work.

8.2 Contributions

This research investigates the mobile wireless communication systems discussed in Chapter 2 and captures their common design problems and respective solutions as shown in Chapter 4. Chapter 5 shows the commonalities related to mobility and radio resource management and presents their documentation as behavioral and structural patterns. In addition, a pattern language is introduced to show how the functional behaviors (behavioral patterns) and the architectural elements (structural patterns) interact. At a high level of abstraction (requirements and analysis stages), a mobile system can be developed or can evolve from the pattern language as shown in Chapter 7. This thesis proposes an approach for the reuse and validation of these patterns.

These results bring new contributions to the mobile domain as discussed in this section. Figure 84 depicts a UCM with a summary of our main contributions. The achieved results are described on the top of the map (responsibility names) and the main contributions on the bottom of the map.

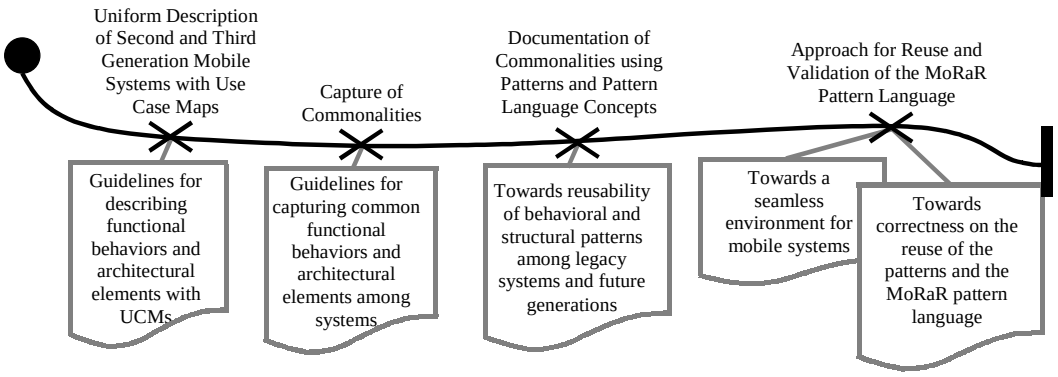


Figure 84 Contributions of the Thesis

In order to achieve the results depicted in the figure, the first contribution of this thesis is the reverse and forward engineering approach used to describe the chosen systems with **U**se **C**ase **M**ap**s** (UCMs) [48][50]. This description is done on the basis of the documentation available in the standards (stage 1, stage 2, and stage 3 documents) and in the literature [4][35][79][139]. We provide guidelines to describe mobile systems with UCMs.

The following mobile systems are described with UCMs: GSM/GPRS/UMTS [139], ANSI-41/WIN [25], IMT-2000 [112][113][114][118] and WmATM [4][54][191]. The graphical specification describes their functional behavior at stage 1 (requirements model) followed by the mapping of the architectural elements at stage 2 (analysis model). As a result, the UCM description reduces the gap between stage 1 and stage 2 documents in the current development approaches applied to the chosen systems. In addition, it is possible to visualize the interaction of mobility, communication, and radio resource management functions in each system at the early development stages (see Section 4.5 of Chapter 4).

The second contribution towards the main goal of this research is the investigation of these UCM descriptions and the identification of common functional behaviors and architectural elements on the basis of responsibilities, start points, and end points (see Section 4.6 of Chapter 4). We provide rules on how to extract common functional behavior and architectural elements among the UCM system descriptions.

As mentioned earlier, the behavioral and the structural patterns are extracted from these commonalities while looking at how these systems solve specific design problems (see **Chapter 5**). It is important to stress that the pattern concept presented in **Chapter 3** is followed when documenting our patterns. We consider to be potential patterns those that specify good solution for design problems and have forces proving that the solution is appropriate. Furthermore, patterns must have at least three known uses. In this thesis, a potential pattern solution is specified with bound UCMs.

Another contribution is the approach for reuse and validation of the patterns grouped within the MoRaR pattern language. This approach combines the use of semi-formal and formal techniques with the reuse and validation of pattern solutions at the requirements and analysis stages. Besides the UCM descriptions, a formal method, LOTOS [98][100], is used to specify the system prototype and validate each pattern solution. Another formal method, MSCs [106], is used to show the validation results.

Patterns allow the application of the software reusability concept to the mobile system development process and evolution. The combination of different techniques at the appropriate development stages and the reuse of patterns can help designers to generate systems that are easy to maintain and to augment with new features. We believe that our method leads to a reduction of development and evolution costs and also leads to the improvement of the overall quality of these systems.

In **Chapter 7**, we present three case studies as an attempt to show the practical use of the proposed approach. A framework for mobile system requirements and analysis models is graphically specified with UCMs and describes the relationships shown by the pattern language and the mapping of commonalities, which include behavioral and architectural patterns, to the network reference model (see **Chapter 4**). This framework is our second case study and it is proposed as reusable requirements and analysis models in **Chapter 6**. The last case study shows how the proposed approach can improve the evolution of signaling protocol alternatives for WmATM networks.

Finally, this research also provides secondary contributions that refer to UCMs and to their relevance for the mobile communication development process and evolution. For instance, we have shown that the UCM notation is powerful enough to capture common functional behavior and structural elements (see details in **Chapter 4**). Furthermore, UCMs are an appropriate tool to describe commonalities at the early development stages (requirements and analysis stages) as shown in **Chapter 6**. For instance, when the unbound maps are combined to a structure of network entities, they are still reusable even if the underlying structure is modified at later stages (see IMT-2000 systems with ICS feature and WmATM networks case studies in **Chapter 7**).

As mentioned in Section 8.1, these contributions are relevant for the mobile communication domain. With the pattern documentation, we organize a repository of good solutions that have been recurrently applied to this domain to solve common design problems. This repository also works as a vocabulary that teaches newcomers about recurring problems and solutions in the mobile system domain.

8.3 Open Areas of Research

This thesis generates open areas of research and this section provides a list of the main issues that can be further explored.

The capture of commonalities among communication management functions and their documentation as patterns is an issue that is left open. These patterns can be integrated with the MoRaR pattern language (see Figure 40 in Chapter 5) in a new category called communication management.

Another issue that is not tackled in this thesis is the identification of architectural elements that are not common to all mobile systems but provide good solutions for design problems. For example, the Service Control Point (SCP) is a network entity that separates the call processing activities from the switching tasks. Although this entity is used only in IN-based systems, it represents a good solution for the separation of call processing from switch services. We believe that the SCP concept has potential to be described as a structural pattern for mobile systems.

It is also important to provide a more complete identification of the relationship among telecommunication design and analysis patterns, which have been presented in the literature [160], and the set of behavioral and structural patterns, which are presented in Chapter 5. This combination can generate a pattern language that improves the development of new telecommunication systems as illustrated in Section 0 of Chapter 5.

Another application of the reuse and validation of the behavioral and structural patterns is the development of a mobile wireless Internet capable of supporting voice and data. As mentioned in Chapter 7, this system is called *all over IP* and is a recent proposal of the IMT-2000 system standardization group.

Furthermore, it is possible to generate *design patterns* using the object-oriented paradigm (e.g., objects and classes definitions, inheritance, polymorphism, and dynamic binding as discussed in Section 3.2 of Chapter 3) from the description of the *requirements and analysis patterns* for mobility and radio resource management. These patterns are described in a functional-oriented approach. However, This generation is an open area of research.

Another open area that also refers to object-oriented concepts is the investigation of the *design patterns* presented in [52] and [80] in order to evaluate their potential application in the mobile wireless communication domain. Once this investigation is

done, these existing design patterns can be added to the MoRaR design patterns, which are derived from the requirements and analysis patterns proposed in this research.

The detection and avoidance of call-related feature interactions have been studied in the literature [50][184]. However, non-call related feature interactions should also be a concern mainly regarding mobility management functions, such as the interaction between handoff, location registration, and authentication. We believe that the MoRaR pattern language and the UCM framework for mobile system requirements and analysis models can help the designers to identify these feature interactions starting from the early stages of the development process. This is also left as an open area of research.

This research provides a method to compare existing systems at a high-level of abstraction. For this reason, interfaces, parameters, and physical entities are not taken in consideration. Thus, an open area of research towards the design and implementation stages is the investigation of problems that make the integration of these systems difficult. Examples are:

- while focusing on requirements and analysis stages, the authentication procedure appear similar for many mobile systems. However, at the implementation stage, different procedures may have different encryption algorithms;

- communication among databases in different mobile wireless networks is another concern. For instance, depending on how these databases are implemented, the location registration procedure can be completely incompatible among these systems.

The investigation of these differences is an important issue for developing a seamless wireless environment, but it is not tackled in this research. In order to investigate these design and implementation constraints, the framework proposed in Figure 62 of Chapter 7 can be implemented with CORBA or Java as mentioned in Chapter 3. These technologies can be used to implement the case studies presented in Chapter 7 and to test the applicability of this work to generate a prototype of a seamless environment.

Finally, the method of capturing, documenting, reusing, and validating the MoRaR pattern language that has been developed in this work has a broad range of application. For example, the robotics and the air traffic control domains include similar mobile procedures in system designs. Thus, the application of the results of this thesis in other domains is a research item that is also left open.

References

[1] Acampora, A., "Wireless ATM Networks: Architecture, System Design and Prototyping", *IEEE Personal Communications*, Vol. 3, No. 4, pp. 42-49, August 1996.

[2] Acampora, A., "Wireless ATM: A Perspective on Issues and Prospects", *IEEE Personal Communications*, Vol. 3, No. 4, 8-17, August 1996.

[3] Adams, M., Coplien, J., Gamoke, R., Hanmer, R., Keeve, F., and Nicodemus, K., "Fault-Tolerant Telecommunication System Patterns," AT&T Bell Laboratories, 1995. Available at http://www.bell-abs.com/people/cope/patterns/telecom/PloP95.telecom.html

[4] Akyol, Bora A., "Signaling Alternatives in a Wireless ATM Network," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 1, January 1997.

[5] Akyol, B. A. and Cox, D. C., "Rerouting for Handoff in a Wireless ATM Network," *IEEE Personal Communications*, Vol. 3, No. 5, 26-33, October 1996.

[6] Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S., *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press, New York, NY, 1977.

[7] Alexander, C., *The Timeless Way of Building*, Oxford University Press, New York, NY, 1979.

[8] Amyot, D., *Specification and Validation of Telecommunication Systems with Use Case Maps and LOTOS*, Ph.D Thesis in Computer Science, Ottawa-Carleton Institute for Computer Science, submitted in January 2001.

[9] Amyot, D., and Eberlein, A., "An Evaluation of Scenario Notations for Telecommunication Systems Development," In: 2001. Available at www.usecasemaps.org/pub/cc99.pdf

[10] Amyot, D. and Logrippo, L., "Use Case Maps and LOTOS for the Prototyping and Validation of a Mobile Group Call System," *Computer Communications 23*, Special Issue on FDTS, 1135-1157, 2000.

[11] Amyot, D., and Mussbacher, G., "On the Extension of UML with Use Case Maps Concepts," In: *3rd International Conference on the Unified Modeling Language (UML2000)*, LNCS 1939, 16-31, York, UK, October 2000. Available at www.usecasemaps.org/pub/cc99.pdf

[12] Amyot, D., Andrade, R., "Description of Wireless Intelligent Networks with Use Case Maps", *Proc. Brazilian Symposium on Computer Networks (SBRC'99)*, Salvador (BA), Brazil, 418-433, May 25-28, 1999.

[13] Amyot, D., Andrade, R., Logrippo, L., Sincennes, J., and Yi, Z. (1999) "Formal Methods for Mobility Standards". *IEEE 1999 Emerging Technology Symposium on Wireless Communications & Systems*, Dallas, USA, April 12-13, 1999.

[14] Amyot, D., Hart, N., Logrippo, L., and Forhan, P., "Formal Specification and Validation using a Scenario-Based Approach: The GPRS Group-Call Example," In: *ObjecTime Workshop on Research in OO Real-Time Modeling*, Ottawa, Canada, January 1998. Available at http://www.csi.uottawa.ca/~damyot/wrroom98/wrroom98.pdf

[15] Amyot, D., Logrippo, L., and Buhr, R.J.A., "Spécification et conception de systèmes communicants : une approche rigoureuse basée sur des scénarios d'usage," In: *CFIP 97*, Ingénierie des protocoles, Liège, Belgique, September 1997. Available at http://www.csi.uottawa.ca/~damyot/cfip97/cfip97.pdf
[16] Amyot, D., "Some Ideas for the Early Phases of the WIN Standardization Process," November 1997. Presentation available at http://www.csi.uottawa.ca/~lotos
[17] Amyot, D., Bordeleau, F., Buhr, R. J. A., and Logrippo, L., "Formal support for design techniques: a Timethreads-Lotos approach," In: von Bochman, G., Dssouli, R., and Rafiq, O. (Eds.), *FORTE VIII, 8th International Conference on Formal Description Techniques*, Chapman & Hall, 57-72, 1996. Available at http://www.csi.uottawa.ca/~damyot/phd/forte95/forte95.pdf
[18] Amyot, D., *Formalization of Timethreads Using Lotos*, M.Sc. Thesis, Dept. of Computer Science, University of Ottawa, Ottawa, Canada, 1994. Available at http://www.csi.uottawa.ca/~damyot/phd/msctheses.pdf
[19] Andrade, R., Bottomley, M., Logrippo, L., Coram, T., "A Pattern Language for Mobility Management", In: *Proc. of the 7th Conference on the Pattern Languages of Programs* (PLoP 2000), Monticello, Illinois, August 2000.
[20] Andrade, R., and Logrippo, L., "Reusability at the Early Development Stages of the Mobile Wireless Communication Systems," In: *Proceedings of the 4th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2000)*, Vol. VII, Computer Science and Engineering: Part I, Orlando, Florida, pp. 11-16, July 2000.
[21] Andrade, R., "Applying Use Case Maps and Formal Methods to the Development of Wireless Mobile ATM Networks", In: *Proc. of the Fifth NASA Langley Formal Methods Workshop*, Williamsburg, Virginia, June 2000.
[22] Andrade, R., "Redes Inteligentes sem Fio e a Terceira Geração de Padrões para Telecomunicações,"[2] In: R. Rios (ed.) Handbook of *II Escola de Informática da SBC – Edição Nordeste*, 140-161, (EINE'99), Fortaleza, Ceará, Brazil, November 1999.
[23] Andrade, R., "Description of WIN Incoming Call Screening Service using Use Case Maps," CSI 5761 Course Project, December 1997. Available at http://www.csi.uottawa.ca/~lotos
[24] Andrade, R., "Design Patterns for Telecommunications," 95514 Course Project, Carleton University, December 1997.
[25] Andriantsiferana, L., Ghribi, B., Logrippo, L., "Prototyping and Formal Requirement Validation of GPRS: A Mobile Packet Radio Service for GSM," In: *Proc. of the 7th Int. Working Conference on Dependable Computing for Critical Application (DCCA-7)*, San Jose, California, USA, 99-118, January 1999.
[26] ANSI/TIA/EIA ANSI-41-D, Cellular Radiotelecommunications Intersystem Operations, 1997.
[27] Ardis, M., Daley, N., Hoffman, D., Siy, H., Weiss, D., "Software Product Lines: A case Study," In: *Software – Practice and Experience*, 30, 825-847, 2000.
[28] Ardis, M. A., Chaves, J.A., Jagadeesan, L. J., Mataga, P., Puchol, C., Staskauskas, M.G., and Olnhausen, J.V., "A Framework for Evaluating Specification Methods for

[2] "Wireless Intelligent Networks and the Third Generation Standards for Telecommunications." Paper available in Portuguese.

Reactive Systems - Experience Report," In: *IEEE Transactions on Software Engineering*, 22 (6), 378-389, 1996.
[29] Ayanoglu E. et al., "Wireless ATM: Limits, Challenges, and Proposals", *IEEE Personal Communications*, Vol. 3, No. 4, 18-36, August 1996.
[30] Beck, K., "Using a Pattern Language for Programming," during Norm Kerth's Workshop on Specification and Design, Addendum to Proceedings of OOPSLA, 1987.
[31] Beck, K., Coplien, J., Crocker, R., Dominick, L., Meszaros, G., Paulisch, F., and Vlissides, J., "Industrial Experience with Design Patterns," *Proceedings of the 10th Int'l Conference on Software Engineering (ICSE'96)*, IEEE Computer Soc. Press, Los Alamitos, California, 1996. Available at http://www.bell-labs.com/~cope/Patterns/ICSE96/icse.html
[32] Bekkers, R., Smits, J., *Mobile Telecommunication: Standards, Regulation, and Applications*, Boston: Artech House, 1999.
[33] Berruto, E. et al., "Research Activities on UMTS Radio Interface, Network Architectures, and Planning," *IEEE Communications Magazine*, 82-94, February 1998.
[34] Black, Uyless D., *The Intelligent Network*, Prentice Hall, Inc., 1998.
[35] Black, Uyless, *Second Generation Mobile & Wireless Networks*, Prentice Hall Series in Advanced Communication Technologies, Prentice-Hall, Inc., 1999.
[36] Bolognesi, T., and Brinksma, E., "Introduction to the ISO Specification Language LOTOS," *Protocol Specification, Testing and Validation VIII*, North-Holland, 23-73, 1988.
[37] Bolognesi, T., De Frutos, D., Langerak, R., and Latella, D., "Correctness Preserving Transformations for the Early Phases of Software Development," In: Bolognesi, T., van de Lagemaat, J., and Vissers, C., *Lotosphere: Software Development with Lotos*, Kluwer Academic Publishers, The Netherlands, 1995.
[38] Bolognesi, T., van de Lagemaat, J., and Vissers, C., *Lotosphere: Software Development with Lotos*, Kluwer Academic Publishers, The Netherlands, 1995.
[39] Booch, G., *Object-Oriented Design*, Redwood City, CA: Benjamin/Cummings, 1991.
[40] Bordeleau, F., Buhr, R.J.A., "The UCM-ROOM Design Method: from Use Case Maps to Communicating State Machines," In*: Proc. Conference on the Engineering of Computer-Based Systems*, Monterey, USA, March 1997.
[41] Boudol, Gérard, "Flow Event Structures and Flow Nets," *Lecture Notes in Computer Science 469*, Springer-Verlag, 62-95, 1990.
[42] Braek, R., "SDL Basics," In: *Computer Networks and ISDN Systems*, 28, 1585-1602, 1996.
[43] Braek, R., and Haugen, O., *Engineering Real Time Systems, an object oriented methodology using SDL*, Prentice-Hall, Hermel Hempstead, 1993.
[44] Braga, A. M., Rubira, C. M. F., Dahab, R., "Tropyc: A Pattern Language for Cryptographic Software," Pattern Languages of Program Design 4 (PLoPD4), N.D. Harrison, B. Foote, and H. Rohnert, eds., Addison-Wesley, 337-371, 2000.
[45] Brown, W. J., McCormick III, H. W., Thomas, S. W., *Antipatterns and Patterns in Software Configuration Management*, John Wiley & Sons, Inc., 1999.

[46] Brown, F. L. Jr., DiVietri, J., Villegas, G. D., Fernandez, E. D., "The Authenticator Pattern", In: *Proceedings of Pattern Language of Programs (PloP'99)*, August 15-18, 1999.
[47] Brown, K., and Whitenack, B., "A Pattern Language for Smalltalk & RDBs," *Object Magazine*, 6(7) September, 50-55, 1996.
[48] Buhr, R. J. A. "Use Case Maps as Architectural Entities for Complex Systems". In: *IEEE Transactions on Software Engineering, Special Issue on Scenario Management*. Vol. 24, No. 12, December 1998. Available at http://www.UseCaseMaps.org/UseCaseMaps/pub/tse98final.pdf
[49] Buhr, R. J. A., "Scenario-Path Signatures as Architectural Entities for Complex Systems," In: *ObjectTime Workshop on Research in OO Real-Time Modeling*, Ottawa, Canada, January 1998. Available at http://www.sce.carleton.ca/ftp/pub/UseCaseMaps/ucmUpdate.pdf
[50] Buhr, R.J.A., Amyot, D., Elammari, M., Quesnel, D., Gray, T., and Mankovski, S., "Feature-Interaction Visualization and Resolution in an Agent Environment." In: *FIW'98, Fifth International Workshop on Feature Interactions in Telecommunications Software Systems*, IOS Press, 135-149, 1998. http://www.UseCaseMaps.org/UseCaseMaps/pub/fiw98.pdf
[51] Buhr, R. J. A. and Casselman, R.S., *Use Case Maps for Object-Oriented Systems*, Prentice-Hall, USA, 1995. Available at http://www.UseCaseMaps.org/UseCaseMaps/pub/UCM_book95.pdf
[52] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M., *Pattern-Oriented Software Architecture*, John Wiley and Sons, New York, NY, 1996.
[53] Chen, X., and Logrippo, L., "Deriving Use Cases for Distributed Systems from Knowledge Requirements," *Annals of Telecommunications*, 55, 1-2, 45-57, 2000.
[54] Cheng, F.-, Holtzman, J. M., "Wireless Intelligent ATM Network and Protocol Design for Future Personal Communication Systems," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, September 1997.
[55] Coad, P. and Yourdon, E., *Object-Oriented Analysis*, Englewood Cliffs, NJ: Yourdon Press, 1990.
[56] Coplien, J. O., Hoffman, D. M., Weiss, D. M., "Commonality and variability in software engineering, *IEEE Software*, 15(6), 37-45, 1998.
[57] Coplien, J. O., *Software Patterns*, SIGS books and Multimedia, June 1996.
[58] Coplien, J. O., "Generative Pattern Languages: An Emerging Direction of Software Design," In: *Proc. of 5th Annual Borland Int. Conf.*, 1994.
[59] Coplien, J. O., and Schmidt, D. C., eds., *Pattern Languages of Program Design*, Addison-Wesley, Reading, MA, Proceedings of the PLOP '94 Conference, 1994.
[60] Corriveau J.,-P., Nel, D. N., "Introduction to Object-Oriented Software Engineering Version 1.0," 95514 Course Notes, School of Computer Science, Carleton University, 1997.
[61] Courtiat, J.-P., Dembinski, P., Holzmann, G., Logrippo, L., Rudin, H., and Zave, P., "Formal methods after 15 years: Status and trends - A paper based on contributions of the panelists at the FORmal Technique `95 Conference," Montreal, October 1995". In: *Computer Networks and ISDN Systems*, 28, Elsivier Science, B.V., 1845-1855, 1996.

[62] Craigen, D., Gerhart, S., and Ralston, T., "Industrial applications of formal methods to model, design, and analyze computer systems: an international survey," Noyes Data Corporation (Publisher), USA, The Cryptomathic ASN.1 Tool Kit, 1994. Available at http://www.cryptomathic.dk/newsletter/9707/frames/asn1.html

[63] DeBruler, D. L., "A Generative Pattern Language for Distributed Processing," Pattern Languages of Program Design 1 (PLoPD1), J. O. Coplien and D. C. Schmidt (Eds.), Addison-Wesley, 69-89, 1995. Available at http://www.bell-labs.com/people/cope/Patterns/DistributedProcessing/DeBruler/index.html

[64] Demestichas, P. P. et al, "Issues in the Design of Future Mobile Communications Systems," IEEE International Conference on Communications, Vol. 1 of 3, 364-368, Vancouver, Canada, June, 1999.

[65] Demeyer, S., Ducasse, S., Tichelar, S., "A Pattern Language for Reverse Engineering," In: Proceedings of the EuroPLoP'99 conference, 1999.

[66] D'Souza, D. F., Wills, C. A., *Objects, Components, and Frameworks with UML: the Catalysis Approach*, Addison-Wesley Longman, Inc., 1999.

[67] Duell, M., "Managing Change with Patterns," *IEEE Communications Magazine*, Vol. 37, No. 4, April 1999.

[68] ETSI, Digital Cellular Telecommunications System (Phase 2+), *General Packet Radio Service (GPRS): Service Description Stage 1 (GEM 02.60), Version 2.0.0*, November 1996.

[69] Faci, M., Logrippo, L., Stepien, B., "Structural Models for Specifying Telephone Systems," *Computer Networks and ISDN Systems 29*, 501-528, 1997.

[70] Faynberg, I., Gabuzda, L.R., Jacobson, T., "The Development of the Wireless Intelligent Network (WIN) and its Relation to the International Intelligent Network Standards," *Bell Labs Technical Journal*, Volume 2, Number 3, Summer 1997.

[71] Farooqui, K., and Logrippo, L., "Architecture for Open Distributed Software Systems," Chapter 11 of: A. Zomaya (ed.) Handbook of *Parallel and Distributed Computing*, Mc. Graw-Hill, 303-329, 1996.

[72] Farooqui, K., Logrippo, L., and deMeer, J., "The ISO Reference Model for Open Distributed Processing: an introduction," *Computer Networks and ISDN Systems 27*, 1215-1229, 1995.

[73] Fernandez, J-C., Garavel, H., Kerbrat, A., Mateescu, Mounier, L., and Sishireanu, M., "CADP (CÆSAR/ALDEBARAN Development Package): A Protocol Validation and Verification Toolbox," In: *Proceedings of CAV'96*, Rajeev A. and Thomas A. H., editors, Vol. 1102 of LNCS, 437-440, Springer-Verlag, August 1996.

[74] Foster, W., Casselman, R.S., Buhr, R. J. A., "From Timethreads to Petri Nets: A First Pass," TR-SCE-93-26, Depto. Of Systems & Computer Engineering, Carleton University, Ottawa, Canada, 1993.

[75] Fowler, Martin, Scott, Kendall, *UML Distilled: a brief guide to the standard object modeling language*, Second Edition, Addison-Wesley, 2000.

[76] Fowler, M., *Analysis Patterns: Reusable Object Models*, Addison-Wesley, Reading, MA, 1997.

[77] Fowler, M., "Implementing Analysis Patterns," presentation at OOP '97, 1997.

[78] Francis, J. C., Herbrig, H., and Jefferies, Nigel, "Service Provision of UMTS Services over Diverse Access Networks," *IEEE Communications Magazine*, 128-136, February 1998.
[79] Gallagher, Michael D., Snyder, Randall A., *Mobile Telecommunications Networking with IS-41*, McGraw-Hill, 1997.
[80] Gamma, E., Helm, R., Johnson, R., and Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.
[81] Geppert, B., and Röbler, F., "Pattern-based Reuse with SDL," In: *9th SDL Forum Tutorial*, June 21-29, Montreal, Canada, 1999.
[82] Graham, I., *Requirements Engineering and Rapid Development an object oriented approach*, Acm Press Prentice-Hall, 1998.
[83] Greenberg, E., *Network Application Frameworks: Design and Architecture*, Addison-Wesley Inc., 1999.
[84] Ghribi, B., and Logrippo, L., "Understanding GPRS: The GSM Packet Radio Service," To appear in *Computer Networks*.
[85] Ghribi, B. and Logrippo, L., "A Validation Environment for LOTOS," In: A. Danthine, G. Leduc, and P. Wolper (Eds), *Protocol Specification, Testing and Verification*, XIII, North-Holland, 1993.
[86] Grinberg, A., *Seamless Networks: Interoperating wireless and wireline networks*, Addison-Wesley, 1996.
[87] Haj-Hussein, M., Logrippo, L. and Sincennes, J., "Goal Oriented Execution for Lotos," In: M. Diaz and R. Groz (Eds), *Formal Description Techniques*, V, North-Holland, 311-327, 1993.
[88] Halsall, F., *Data Communications, Computer Networks and Open Systems*, Addison-Wesley Ltd., 1996. ISBN 0-201-42293-X.
[89] Händel, R., Huber, M. N., Schröder, S., *ATM Networks: Concepts, Protocols, Applications*, Addison-Wesley Publishing Company, 1994.
[90] Hanmer, R., Stymfal, G., "An Input and Output Pattern Language," *Pattern Languages of Program Design 4* (PLoPD4), N. B. Harrison, B. Foote, and H. Rohnert (Eds.), Addison-Wesley, 503-536, 2000.
[91] Hlavacek, D. M., Zurawski, R. J., "Alternative Methods for Introducing New Wireless Intelligent Network Services Using Triggers and Queries," *Bell Labs Technical Journal*, Volume 2, Number 3, Summer 1997.
[92] Hoare, C. A. R., *Communicating Sequential Processes*, Prentice-Hall International, 1985. ISBN: 0131532715.
[93] Hodges, J., and Vissers, J., "Accelerating Wireless Intelligent Network Standards through Formal Techniques", In: *Proc. of the Vehicular Technology Conference (VTC'99)*, Houston, TX, May 17-19, 1999.
[94] Holz, E., "Application of UML in the SDL Design Process," In: *Proceedings of the 1st Workshop of the SDL Forum Society on SDL and MSC*, Berlin, Germany, June 29th – July 1st, Volume II, 143-150, 1998.
[95] Holzmann, G., J., "Formal Methods for Early Fault Detection," In: *Proc. of Formal Techniques for Real-Time and Fault Tolerant Systems*, LNCS Vol. 1135, 40-54, 1996.

[96] Holzmann, G. J., *Design and Validation of Computer Protocols*, Prentice Hall Software Series, 1991.
[97] Hubaux, J-P., Nagel, D., Kienurtz, B., Special Issue on Provision of Communication Services over Hybrid Networks, *IEEE Communications Magazine*, Vol. 37, No. 7, July 1999.
[98] Huber, J. F., Weiler, D., and Brand, H., "UMTS, the Mobile Multimedia Vision for IMT-2000: A Focus on Standardization," *IEEE Communications Magazine*, Vol. 38, No. 9, 129-136, September 2000.
[99] ISO, Information Processing Systems, Open Systems Interconnection, "Lotos - A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour," IS 8807, Geneva, 1988.
[100] ISO/IEC JTC1/SC21/WG1, "Enhancement to Lotos (1.21.20.2.3)," January 1997.
[101] ISO/EIC, "OSI CTMF Part 3: The Tree and Tabular Combined Notation - Second Edition," IS 9646-3: 1997, Geneva, 1997.
[102] ISO/EIC, Proposed ITU-T Z.500 and Committee Draft on "Formal Methods in Conformance Testing (FMCT)," ISO/EIC JTC1/SC21/WG7, ITU-T SG 10/Q.8, CD-13245-1, Geneva, 1996.
[103] ISO/EIC, "OSI CTMF Part 3: The Tree and Tabular Combined Notation," IS 9646-3: 1992, Geneva, 1992.
[104] ISO/EIC, Information Technology, Open Systems Interconnection, "Conformance Testing Methodology and Framework (CTMF)," IS 9646, ISO, Geneva. Also: CCITT X.290-X.294, 1991.
[105] ISO/ITU-T – *Open Distributed Processing, Reference Model*, ISO 10746, ITU Recommendation X.901-904, Geneva, 1995.
[106] ITU, "Recommendation Z. 120: Message Sequence Chart (MSC)," Geneva, 1996.
[107] ITU, "Recommendation Z.105, SDL Combined with ASN.1 (SDL/ASN.1)," Geneva, 1995.
[108] ITU, "Recommendation X.680-683, Abstract Syntax Notation One (ASN.1)," Geneva, 1994.
[109] ITU, "Recommendation Z.100, Specification and Description Language (SDL)," Geneva, 1994.
[110] ITU-T - *Q.1220 Series, Intelligent Network Capability Set-2 Recommendation Structure*. Geneva, 1997.
[111] ITU-T - *Q.1200 General Series, Intelligent Networks Recommendation Structure.* Geneva, 1995.
[112] ITU-T, "Q1701 Recommendation: IMT-2000 Systems Framework," March 1998.
[113] ITU-T, "Q1711 Recommendation: IMT-2000 Systems Signaling Requirements," 1998.
[114] ITU-T, "Q1721 Draft Recommendation: IMT-2000 Systems Information Flows," 1999.
[115] ITU-T, "Q.2931 Recommendation: ATM Network Signaling Specification."
[116] Jacobson, I., Christerson, M., Jonsson, P., and Övergaard, G., *Object-Oriented Software Engineering A Use Case Driven Approach*, Addison-Wesley Pub., ACM Press, 1992.

[117] Jacobson, I., Ericsson, M., Jacobson, A., *The Object Advantage Business Process Reengineering with Object Technology*, Addison-Wesley Pub., ACM Press, 1995.
[118] Kelly, Dean, *Automata and Formal Languages: An Introduction*, Prentice-Hall, Inc., 1995. ISBN 0-13-497777-7.
[119] Krechmer, K., "Recommendations for the Global Information Highway: A Matter of Standards," *StandardView ACM Perspectives on Standardization*, Vol. 4, No. 1, 24-28, March 1996.
[120] Kriaras, I. N., Jarvis, A. W., Phillips, V. E., Richards, D. J., "Third-Generation Mobile Network Architectures for the Universal Mobile Telecommunications System (UMTS)," *Bell Labs Technical Journal*, 99-117, Summer 1997.
[121] Kristoffersen, F. and Walter, T., "TTCN: Towards a formal semantics and validation of test suites," In: *Computer Networks and ISDN Systems*, 29 (01), 15-48, 1996.
[122] Laitinen, M., Rantala, J., "Integration of Intelligent Network Services into Future GSM Networks," *IEEE Communication Magazine*, June 1995.
[123] Langerak, R., "Bundle event structures: a non-interleaving semantics for LOTOS," In: M. Diaz and R. Groz (Eds.) *Formal Description Techniques, V (C-10)*, Elsevier Science Publisher B.V., North Holland, 331-346, 1993.
[124] Leach, Ronald F., *Software Reuse: methods, models, and costs*, Computing McGraw-Hill, 1997.
[125] Leon, G. et al, "An Industrial Experience on Development with LOTOS and SDL," *Formal Description Techniques*, VI, 1994.
[126] Lim, W. C., *Managing Software Reuse: A comprehensive Guide to Strategically Reengineering the Organization for Reusable Components*, Prentice Hall, ISBN 0-13-552373-7, 1998.
[127] Logrippo. L., "Immaturity and Potential of Formal Methods: A Personal View," *Feature Interactions in Telecommunications and Software Systems VI*, M. Clader and E. Magill (eds.), IOS Press, 9-13, 2000.
[128] Logrippo. L., Faci, M., Haj-Hussein, M., "An Introduction to LOTOS: Learning by Examples," *Computer Network and ISDN Systems 23*, 325-342, 1992.
[129] Löwis of Menar, Martin, Schröder, "Object Oriented Data Concepts for SDL," In: *Proceedings of the 1st Workshop of the SDL Forum Society on SDL and MSC*, Berlin, Germany, June 29th – July 1st, Volume I, 45-54, 1998.
[130] Lu Zhong-wan, *Mathematical Logic for Computer Science*, World Scientific Publishing Co. Pte. Ltd., 1989. ISBN 997102518.
[131] Mauw, S. and Reniers, M., A., "High-Level Message Sequence Charts."
[132] Meszaros, G., "Design Patterns in Telecommunications System Architecture," *IEEE Communications Magazine*, Vol. 37, No. 4, April 1999.
[133] Meszaros, G., Doble, J., "A Pattern Language for Pattern Writing," *Pattern Language of Program Design 3*, edited by Robert C. Martin, Dirk Riehle, and Frank Buschmann, Addison-Wesley (Software Patterns Series), 1997.
[134] Meszaros, G., "A Pattern Language for Improving the Capacity of Reactive Systems," *Pattern Languages of Program Design 2* (PLoPD2), J. O. Coplien and D. C. Schmidt (Eds.), Addison-Wesley, 575-591, 1996.

[135] Miga, A., *Application of Use Case Maps to System Design with Tool Support*, M.Eng. Thesis, Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada, 1998.
[136] Milner, R., *Communication and Concurrency*, Prentice-Hall, New York, 1989. ISBN 0131149849.
[137] Misra, P., "Routing Protocols for Ad Hoc Mobile Wireless Networks," Courses Notes, available at http://www.cis.ohio-state.edu/~jain/cis788-99/adhoc_routing/index.html
[138] Moreira, A. M. D. and Clark R. G., "Adding rigour to object-oriented analysis," In: *Software Engineering Journal*, 11(5), 270-280, September 1996.
[139] Mouly, M. and Pautet, M.-B., *The GSM System for Mobile Communications*, 1992.
[140] Mowbray, T. J., and Malveau, R., *CORBA Design Patterns*, John Wiley & Sons, New York, NY, 1997.
[141] Muller, N. J., *Mobile Telecommunications Factbook*, McGraw-Hill Telecommunications, 1998. ISBN 0-07-044461-7.
[142] Munro, A., et al., "Services and Applications: Requirements and Realizations in the UMTS Era," *IEEE Communications Magazine*, 118-126, February 1998.
[143] Myers, G. J., *The Art of Software Testing*, Wiley, 1979.
[144] Nicola, R., and Hennessey, M., "Testing Equivalence for Processes," *Theoretical Computer Science 34 (1,2)*, 83-133, November 1984.
[145] OSI - IS 8807, "Information Processing Systems - Open Systems Interconnection - LOTOS - A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour", International Standard IS 8807 (E. Brinksma, ed.), 1989.
[146] Paetsch, M., *Mobile Communications in the US and Europe: Regulation, Technology, and Markets*, Mobile Communication Series, Artech House, Inc., 1993.
[147] Parnas, D. L., "On the Criteria to be used in decomposing systems into modules," *Communications of the ACM*, 15(12), 1972.
[148] Pávon, S., Larrabeiti, D., and Rabay, G., "Lola-User Manual, version 3.6," DIT, Universidad Politécnica de Madrid, Spain, Lola/N5/V10, February 1995.
[149] Perkins, Charles E., Mobile IP Design Principles and Practices, Addison-Wesley Wireless Communication Series, 1998.
[150] Pires, L. F., Architectural Notes: A Framework for Distributed Systems Development, Ph.D. Thesis, Centre for Telematics and Information Technology, 1994.
[151] Pree, W., "Essential Framework Design Patterns," *Object Magazine 7(1)*, 34-37, 1997.
[152] Pree, W., *Framework Patterns*, SIGS Books, New York, NY, 1996.
[153] Pree, W., *Design Patterns for Object-Oriented Software Development*, Addison-Wesley Publishing Company, ACM Press, 1995.
[154] Pressman, Roger S., *Software Engineering, A Practioner's Approach*, fourth edition, McGraw-Hill, 1997.
[155] Probert, R.L. and Monkewich, O., "TTCN: the international notation for specifying tests of communications systems," In: *Computer Networks and ISDN Systems*, 23 (05), 417-438, 1992.

[156] Prycker, M., *Asynchronous Transfer Mode: solution for broadband ISDN*, Prentice Hall International (UK) Limited, 1995.
[157] Quemada, J., "Working Draft on Enhancements to LOTOS," 1997.
[158] Raychaudhuri, D., "Wireless ATM Networks: Architecture, System Design and Prototyping," *IEEE Personal Communications*, Vol. 3, No. 4, 42-49, August 1996.
[159] Rada, R., Software Reuse: Principles, Methodologies, Intellect Ltd., 1995.
[160] Redl, S. H., Weber, M. K., Oliphant, W., *An Introduction to GSM*, Artech House, Inc., 1995. ISBN 0-89006-785-6.
[161] Rising, Linda, *The Pattern Almanac 2000*, Software Pattern Series, Addison-Wesley, 2000. ISBN 0-201-61567-3.
[162] Rising, Linda, "Patterns: A Way to Reuse Expertise," *IEEE Communications Magazine*, Vol. 37, No. 4, April 1999.
[163] Rudolph, E., Graubmann, P., and Grabowski, J., "Tutorial on Message Sequence Charts," In: *Computer Networks and ISDN Systems*, 28, 1629-1641, 1996.
[164] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., *Object-Oriented Modeling and Design*, Prentice Hall, Inc., 1991.
[165] Schmidt, D., "Tutorial about Design Patterns," Available at http://www.cs.wustl.edu/~schmidt/patterns.html
[166] Schmidts, H., and Vissers, J., "Framework for IMT-2000 Networks," To appear in *Computer Networks and ISDN Systems*, 2000.
[167] Selic, B., Gullekson, G., and Ward, P. T., *Real-Time Object-Oriented Modeling*, John Wiley & Sons, Inc., 1994.
[168] Sommerville, I., *Software Engineering*, fourth edition, Addison-Wesley, 1992.
[169] Steedman, D., "Abstract Syntax Notation One ASN.1: The Tutorial & Reference," Technology Appraisals, Twickenham, UK, 1990.
[170] Stepien, B., "Lotos2MSC Converter – User's Manual," University of Ottawa LoTOS Group, January 2000.
[171] Stepien 98.
[172] Suzuki, J. and Yamamoto, Y., "OpenWebServer: An Adaptive Web Server Using Software Patterns," *IEEE Communications Magazine*, Vol. 37, No. 4, April 1999.
[173] Tanenbaum, A. S., *Computer Networks, 3rd Ed.*, Prentice Hall, 1996.
[174] TIA/EIA *Wireless Intelligent Networks (WIN)*, Additions and modifications to ANSI-41 (Phase 1), TR-45.2.2.4, PN-3661 Ballot Version, May 1998.
[175] TIA/EIA *Wireless Intelligent Networks (WIN)*, Additions and modifications to ANSI-41 (Phase 2), TR-45.2, Draft Version, July 1998.
[176] TIA/EIA Documents: WIN ANSI-41.3 C Additions, WIN ANSI-41.1 C Modifications, and WIN Stage 1 Modifications.
[177] Tuok, R., *Modeling and Derivation of Scenarios for a Mobile Telephony System in Lotos*, M.Sc. Thesis, Dept. of Computer Science, University of Ottawa, Ottawa, Canada, 1996.
[178] Turner, K.J., *Using Formal Description Techniques; An Introduction to ESTELLE, LOTOS and SDL*, Wiley Publishers, U.K., 1992.
[179] Ulema, Mehmet, "Wireless Intelligent Networking," Tutorial number 4, *Global Telecommunications Conference* (Globecom'99), Rio de Janeiro -RJ, Brazil, 5-9 December 1999.

[180] Umehira, M., et al., "ATM Wireless Access for Mobile Multimedia: Concept and Architecture", *IEEE Personal Communications*, Vol. 3, No. 5, 39-48, October 1996.
[181] *Use Case Maps Web Page and UCM User Group*: http://www.UseCaseMaps.org, since 1999.
[182] Use Case Maps Handouts. Available at Use Case Maps User Group http://www.UseCaseMaps.org
[183] Utas, Greg, "An Overview of Selected Call Processing Patterns," In: *IEEE Communications*, 64-69, April 1999.
[184] Utas, Greg, "A Pattern Language of Feature Interaction," In: *Feature Interactions in Telecommunications and Software System V*, K. Kimbler and W. Bouma, Editors, IOS Press, Amsterdam, September 1998. ISBN 90-5199-431-1.
[185] Varshney, U., "Supporting Mobility with Wireless ATM," *Computer*, Vol. 30, No. 1, 131-137, January 1997.
[186] Varshney, U., and Vetter, R., "Emerging Mobile and Wireless Networks," *Communication of the ACM*, Vol. 43, No. 6, 73-81, June 2000.
[187] Veeraraghavan, M. et al., "Mobility and Connection Management in a Wireless ATM LAN," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 1, 50-68, January 1997.
[188] Vlissides, J., "Seven Habits of Successful Pattern Writers," *C++ Report*, November/December 1996. Available at http://hillside.net/patterns/papers/
[189] Vlissides, J., "Patterns: The Top Ten Misconceptions," *Object Magazine*. Available at http://hillside.net/patterns/papers/
[190] Weidenhaupt, K., Pohl, K., Jarke, Matthias, and Haumer, P., "Scenarios in System Development: Current Practice," In: *IEEE Software*, March/April 1998, 34-45 W3 Consortium (1998), Extensible Markup Language (XML) 1.0. W3C Recommendation, 10 February 1998. Available at http://www.w3.org/TR/REC-xml
[191] Wesel, K. E., *Wireless Multimedia Communications: Networking Video, Voice, and Data*, Addison-Wesley Wireless Communications Series, 1998.
[192] White, C. M., *Data communications and computer networks: an OSI approach*, Boyd & Fraser Publishing Company, 1995.
[193] Wilson, David R., "Signaling System No. 7, IS-41, and Cellular Telephony Networking," In: *Proceedings of the IEEE*, Vol. 80, No. 4, April 1992.
[194] Winskel, G., "An introduction to event structures," Lecture Notes in Computer Science 354, Springer-Verlag, pp. 364-397, 1989.
[195] Wirfs-Brock, R. J., and Johnson, R. E., "Surveying Current Research in Object-Oriented Design," *Communications of ACM*, 33(9), 1990.
[196] Wirth, N., "Program Development by stepwise refinement," *Communications of ACM*, 14(4), 1971.
[197] Wirth, N., "On the composition of well structured programs," *ACM Computing Surveys*, 6(4), 1974.
[198] Wireless Intelligent Network Tutorial. Available at http://www.webproforum.com/dsc
[199] Worm, T., "Using Metapatterns with SDL," In: *SDL'99: The next Millenium*, Dssouli, R., Bochmann, G.v., and Lahav, Y., eds., Elsevier Science B.V., 1999.

[200] Yi, A. Z., "CNAP Specification and Validation: A Design Methodology using LOTOS and UCM," M.Sc. Thesis, Dept. of Computer Science, University of Ottawa, Ottawa, Canada, 2000.

Appendix A **Common Functional Behaviors**

This appendix summarizes the common functional behaviors that this research have identified among second and third generation systems. Each table represents a common functional behavior and its common responsibilities with pre-conditions and post-conditions and common end points with post-conditions. Chapter 4 and Chapter 5 present more details about these commonalities that are graphically specified with UCMs in Chapter 4.

ANSI/WIN Specifications	**GSM/GPRS/UMTS Specifications: Location Management Procedures**	**IMT-2000 Systems: TMUI Assignment**	***Common* Functional Behavior: Temporary Identification Assignment**
P_aNTMSI: a TMSI assignment has been requested	P_aTMSI: a TMSI assignment has been requested	P_rATMUI: the network has verified the identity of the user and it has requested a TMUI	P_aTID: a temporary identification assignment has been requested
assignNewTMSI	assignTMSI	ReqAssignTMUI	aTID: assign a Temporary Identification
Q_aNTMSI: a TMSI has been assigned	Q_aTMSI: a TMSI assignment has been assigned	Q_ rATMUI: The TMUI request assignment has been retrieved	Q_aTID: Temporary Identification has been assigned
-	P_ wTMUIA => Q_aTMSI	P_aR => Q_ rATMUI	P_ cATID => Q_aTID
-	waitTMUIAck	analyzeReport	cATID: check Assigned Temporary identification
-	Q1_ wTMUIA: TMUI has been acknowledged Q2_ wTMUIA: TMUI hasn't been acknowledged	Q1_aR: TMUI has been successfully assigned Q2_aR: TMUI hasn't been successfully assigned	Q1_ cATID: TMUI has been successfully assigned Q2_ cATID: TMUI hasn't been successfully assigned
E1_TMSI	E1_TMUI	E1_TMUIAssign	E1_TMUIAssign
Q_E1_TMSI => Q_aNTMSI	Q_E1_TMUI => Q1_ wTMUIA: TMUI	Q_E1_TMUIA => Q1_aR	Q_E1_TMUIAssign => Q1_ cATID
-	E2_TMUI	E2_TMUIAssign	E2_TMUIAssign
-	Q_E2_TMUI => Q2_ wTMUIA: TMUI	Q_E2_TMUIA => Q2_aR	Q_E2_TMUIAssign => Q2_ cATID

Table 10 Functional Behavior: Temporary Identification

ANSI-41/WIN Specifications: Registration and Unique Challenge/Response Authentication	**WmATM Networks: Authentication**	**GSM/GPRS and UMTS Authentication**	**IMT-2000 Systems: User Authentication**	***Common*** **Functional Behavior: Authentication**
P_oRI: an user authentication has been requested	-	P_sAI: an authentication has been requested	P_tSI: an authentication has been requested and the user authentication information is not complete in the visited network	P_ sAI: an authentication has been requested
obtainRandInfo	-	sendAuthInfo	transfSecInfo	sAI
Q_oRI: information has been obtained	-	Q_sAI: information has been sent	Q_tSI: security information has been transferred	Q_ sAI: security information has been sent
P_pCN => Q_oRI	P_pAC: authentication information is received	P_pAC => Q_sAI	P_pC => Q_tSI	P_aAA => Q_sAI
perfCalcNet	perfAuthCalc	perfAuthCalc	performCalculation	aAA
Q_pCN: an authentication calculation has been performed	Q_pAC: authentication result is generated	Q_pAC: an authentication calculation has been performed	Q_pC: an authentication calculation has been performed	Q_aAA: an authentication calculation has been performed
P_cAR => Q_pCN	P_cUAR => Q_pAC	P_cR => Q_pAC	P_cR => Q_pC	P_cAR => Q_aAA
chkAuthResult	checkUsrAuthRes	compareRes	checkResult	cAR
Q1_cAR: the authentication results have matched Q2_cAR: the authentication results haven't matched	Q1_ cUAR: authentication is successful Q2_ cUAR: authentication is unsuccessful	Q1_ cR: authentication is successful Q2_ cR: authentication is unsuccessful	Q1_cR: the results are similar Q2_cR: the results are different	Q1_cAR: the authentication results are similar Q2_cAR: the authentication results are different
P_aD => Q2_cAR	P_dUA => Q2_ cUAR	P_dA => Q2_cR	Q_ E2_Auth => Q2_cR	Q_ nAD => Q2_cAR
applyDenial	denyUsrAccess	denyAccess	E2_Auth	nAD
Q_aD: a denial treatment has been applied	Q_dUA: a denied authentication treatment has been done	Q_dA: a denied authentication notification has been sent	-	Q_dA: a denied authentication treatment has been done

Table 11 Functional Behavior: Authentication

ANSI-41/WIN Specifications: Authentication and Signaling Message Encryption	**WmATM Networks: Overlay Registration**	**GSM/GPRS/UMTS Specifications: Ciphering**	**IMT-2000 Systems: Authentication Management: Authentication Calculation and Ciphering Data**	***Common*** **Functional Behavior: Ciphering**
P_sEI: a location registration has been requested and a MS is subscribed to voice privacy	P_eEI: a registration process has started	P_cCK: a location registration process has been requested	P_sC: a location registration process has been requested and an authentication calculation has been successfully performed	P_cCK: a location registration has been requested
sendEncrypInfo	excEncrypInfo	creatingCipherKey	storeCiph	sCK: store ciphering key
Q_sEI: the authentication request has included the encryption information	Q_eEI: an exchange of encryption keys has occurred	Q_cCK: the authentication procedure has returned a ciphering key	Q_sC: a ciphering key has been created and stored	Q_cCK: the authentication procedure has returned a ciphering key
P_sE => Q_sEI	P_sE => Q_eEI	P_sCT => Q_cCK	P_sE: A request for ciphering has been received	P_sC => Q_cCK
startEncryption	startEncryption	startCipherTraffic	startEncryption	sC: start ciphering
Q_sE: voice/traffic channel have their information encrypted	Q_sE : messages are encrypted	Q_sCT : traffic is encrypted	Q_sE: ciphering control over the radio interface is active	Q_sC : messages are encrypted

Table 12 Functional Behavior: Ciphering

ANSI-41/WIN Specifications: Paging	**WmATM Networks: Overlay Originating Party - Connection**	**GSM/GPRS/UMTS Specifications: Paging**	**IMT-2000 Systems: Paging**	***Common*** **Functional Behavior: Paging**
P_pLA: the location area ID is known and a page has been requested P_pSA: the location area ID is not known and a page has been requested	P_p: the calling user's number is not in the blocking list and a page has been requested	P_pMS: a paging request has been received	P_pU: a paging request has been received	P_pU: a paging request has been received
pLA or pSA	paging	pageMS	pagingUser	pU
Q_pLA: a paging has been issued within the user's location area Q_pSA: a paging has been issued within the user's entire service area	Q_p: the called user has been paged	Q1_pMS: a user has been successfully paged Q2_pMS: otherwise	Q1_pU: a user has been successfully paged Q2_pU: otherwise	Q1_pU: a user has been successfully paged Q2_pU: otherwise
P_aC => Q_pLA or P_aC => Q_pSA	P_eC => Q_p	P_aC => Q1_pMS	P_aC => Q1_pU	P_aCMS => Q1_pU
assigChannel	estabConn	assignChannel	assignChannel	ACMS: assign channel to the mobile station
Q_aC : a channel has been assigned	Q_eC : an ATM connection has been established between the users	Q_aC : a channel has been assigned	Q_aC : a channel has been assigned	Q_aCMS : a channel has been assigned to the mobile station
P_nMSU : paging hasn't reached the user	-	Q_ E2_Pag => Q_aC	P_nMSU => Q2_pU	P_nUU => Q2_pU
notifyMSUnreach	-	E2_Pag	notifyMSUnreachable	nUU
Q_ nMSU : a notification has been sent	-	-	Q_ nMSU: a notification of MS unreachable has been sent	Q_ nUU: a notification of user unreachable has been sent

Table 13 Functional Behavior: Paging

ANSI-41/WIN Specifications: Registration Notification and Location Cancellation	**WmATM Networks: Overlay Registration**	**GSM/GPRS/UMTS Specifications: Location Management**	**IMT-2000 Systems: Roaming Registration**	***Common*** **Functional Behavior: Location Registration**
P_sI: location information has been requested	P_gUI: user has changed zone	P_gLI: the mobile station's location has changed	P_rRR: user has changed location	P_sLI: the mobile station's location has changed
sendInfo	getUserInfo	getLocInfo	reqRoamReg	sLI
Q_sI: information has been sent	Q_gUI: user's identification number and previous zone identification are known	Q_rRR: a roaming request registration has been sent with the location information	Q_rRR: a roaming request registration has been sent with the location information	Q_sLI: location information has been sent
P1_cR: a user has powered on a mobile station and authentication hasn't been performed P2_cR: a user has tried to make a call and authentication hasn't been performed	P_cL => Q_gUI	P_cL => Q_rRR	-	P_cL => Q1_sLI
checkRoaming	checkLocation	checkLoc	-	cL
Q1_cR: the user is roaming and it is the first time in the serving system Q2_cR: the user is roaming and it isn't the first time in the serving system Q3_cR: the user isn't roaming	Q1_cL: current zone identification and last zone identification are different Q2_cL: current zone identification and last zone identification are the same	Q1_cL: current location is different from previous location Q2_cL: current location is the same as previous location	-	Q1_cL: current location is different from previous location Q2_cL: current location is the same as previous location
P_rNL: Q1_cR and a registration cancellation confirmation has been received and	P_uUP: Q1_cL and authentication is successful	P_uL => Q1_cL	P_uA => Q_rRR	P_uPR => Q1_cL P_uPL => Q2_cL
recordNewLoc	updtUsrProfile	UpdateLocation	updateAdd	uPL (or uPR)
Q_rNL: new location information has been recorded	Q_uUP: user profile is updated with the new location information	Q_uL: user profile is updated with the new location information	Q_uA: the new location address has been updated	Q_uP: user profile is updated with the new location information

ANSI-41/WIN Specifications: Registration Notification and Location Cancellation	**WmATM Networks: Overlay Registration**	**GSM/GPRS/UMTS Specifications: Location Management**	**IMT-2000 Systems: Roaming Registration**	***Common*** **Functional Behavior: Location Registration**
P_rI => Q_sI	P_gLUP => Q_uUP	P_iD => Q_uL	P_tUP => Q_uA	P_uTP => Q_uP
recordInfo	getLocUsrProf	insertData	transfUsrProf	uTP
Q_rI: information has been recorded	Q_gLUP: user profile is in the current location	Q_iD: information has been recorded in the visiting system	Q_tUP: the user profile has been stored in the visiting network	Q_uTP: information has been recorded in the visiting system
P_dPR => Q_rI	P_dPUP => Q_gLUP	P_c => Q_iD	P_dR => Q_tUP	Q_dTP => P_uTP
deletePrevRecord	DelPrvUsrProf	cancellation	de-Register	dTP
Q_dPR: the user's profile in the previous zone has been deleted	Q_dPUP: the user's profile in the previous zone is deleted	Q_c: the user's profile in the previous zone has been deleted	Q_dR: the user profile has been deleted from the previously visited network	Q_dTP: the user's profile in the previous zone has been deleted

Table 14 Functional Behavior: Location Registration

WmATM Networks: Handoff Failure Actions	**GSM/GPRS/UMTS Specifications: Call Re-Establishment**	***Common*** **Functional Behavior: Handoff Failure Actions**
P_tPC: a request for tuning to the previous channel has been sent	P_rOC: a request for tuning to the old connection has been sent	P_tPC: a request for tuning to the previous channel has been sent
tunesPrevChn	returnOldConn	tPC
Q_tPC: the portable has tuned back to the previous channel	Q_rOC: the mobile station has tuned back to the old connection	Q_tPC: the mobile station has tuned back to the previous channel
P_dAC => Q_tPC	P_dC => Q_rOC	P_rRA => Q_tPC
deallocAssignChan	deallocChan	rRA
Q_dAC: the assigned new channel has been deallocated	Q_dC: the assigned new channel has been deallocated	Q_rRA: the assigned new channel has been deallocated

Table 15 Functional Behavior: Handoff Failure Actions

ANSI-41/WIN Specifications: Handoff Measurements	**WmATM Networks: Overlay Inter-Zone Handoff through the Previous Port**	**GSM/GPRS/UMTS Specifications: Handover Decision**	***Common*** **Functional Behavior: Handoff Decision**
P_eC: a handoff measurement has been requested	P_mP: a monitoring process is on	P_tM: a handoff measurement has been requested	P_tM: a handoff measurement has been requested
electCandidate	monitorPort	takeMeasurements	tM
Q_eC: a candidate cell has been elected	Q_mP: signal power has been received	Q_tM: a candidate cell has been selected	Q_tM: a candidate cell has been selected
P_cCQ => Q_eC	P_cP => Q_mP	P_cM => Q_tM	P_cM => Q_tM
checkCellQuality	comparePower	checkMeasurements	cM
Q1_cCQ: the current cell hasn't met the quality criteria for communication Q2_cCQ: the current cell has met the quality criteria for communication	Q1_cP: a link to another port has become stronger Q2_cP: a link to another port has not become stronger	Q1_cM: a handoff is necessary Q2_cM: a handoff is not necessary	Q1_cM: the current connection is ok Q2_cM: the current connection is not ok

Table 16 Functional Behavior: Handoff Decision

ANSI-41/WIN Specifications: Handoff Back, Handoff Forward, and Handoff to Third Procedures and Call Release Procedure	**WmATM Networks: Overlay Registration, Call Setup and Handoff**	**GSM/GPRS Specifications: Handover Execution**	***Common*** **Functional Behavior: Releasing resources**
P_rC: a release resource request has been received	P_rC: transaction is complete	P_rR: a release resource request has been received	P_rR: a release resource request has been received
Release Call, release Channel, and releaseResources	releaseConnection	releaseResources	rR
Q_rC: the network resources have been released	Q_rC: the connection has been released	Q_rR: the network resources have been released	Q_rR: the network resources have been released

Table 17 Functional Behavior: Releasing Resources

ANSI-41/WIN Specifications: Handoff Back, Handoff Forward, and Handoff to Third	**WmATM Networks: Overlay Handoff Previous – Alternative 1**	**GSM/GPRS/UMTS Specifications: Handover Procedures**	***Common*** **Functional Behavior: Inter-System Handoff Execution**
P_sVC: a handoff request has been received	P_aC: the user profile has been transfer to the current zone and a handoff has been requested	P_aC: a handoff request has been received (P_aC =>Q1_vC)	P_aC: a handoff request has been received
selectVoiceChannel	allocChan	allocateChannel	aC
Q_sVC: a voice channel has been selected	Q_aC: a channel between the user and the candidate zone manager (CZM) has been assigned	Q_aC: a channel has been allocated	Q_aC: a channel has been allocated
P_tNC => Q1_cVC	P_tNC: the channel information has been relayed to the portable	P_tNC => Q_aC	P_tNC => Q_aC
tunesNewChannel	tunesNewChannel	tunesNewChannel	tNC
Q_tNC: the MS has tuned to the new channel	Q_tNC: the portable has tuned to the new channel	Q_tNC: the MS has tuned to the new channel	Q_tNC: the MS has tuned to the new channel
P_cVC => Q_sVC	P_vC => Q_tNC	P_vC : a handoff has been requested	P_vC => Q_tNC
checkVoiceChannel	verifyConn	verifyConnection	vC
Q1_cVC: the channel is available Q2_cVC: the channel is not available	Q1_vC: the connection is successfully verified Q2_vC: the connection is not verified	Q1_vC: the connection is successfully verified Q2_vC: the connection is not verified	Q1_vC: the connection is successfully verified Q2_vC: the connection is not verified

Table 18 Functional Behavior: Inter-System Handoff Execution

Appendix B **Common Architectural Elements**

This appendix summarizes the common architectural elements that this research have identified among second and third generation systems. Each table represents a set of common architectural elements and their common responsibilities and common end points. Chapter 2, Chapter 4 and Chapter 5 present more details about the architectural elements of the chosen systems and their common functional behaviors. A common network reference model is graphically specified with UCM components in Chapter 4.

ANSI-41/WIN Specifications	**GSM/GPRS/UMTS Specifications: Location Management Procedures**	**IMT-2000 Systems: TMUI Assignment**	***Common*** **Architectural Element**
assignNewTMSI: VLR	assignTMSI: new MSC/VLR and MS	reqAssignTMUI: RAN+CNv (SACF and LMFv)	sTID: VLR
-	waitTMUIAck: MS and MSC/VLR	analyzeReport: RAN+CNv (SACF and LMFv)	cATID: VLR
E1_TMSI: MSC	E1_TMUI: MSC/VLR	E1_TMUIAssign: RAN+CNv (SACF)	E1_TMUIAssign: MSC
E2_TMSI: MSC	E2_TMUI: MSC/VLR	E2_TMUIAssign: RAN+CNv (SACF)	E2_TMUIAssign: MSC

Table 19 Common Architectural Elements for Temporary Identification

Note: LMFv stands for Location Management Function in a visiting system. This is an IMT-2000 functional entity (FE) that corresponds to the VLR network entity (NE). The SACF FE stands for Service Access Control Function and corresponds to the MSC NE (see Chapter 2).

ANSI-41/WIN Specifications: Authentication and Signaling Message Encryption	**WmATM Networks: Overlay Registration**	**GSM/GPRS/UMTS Specifications: Ciphering**	**IMT-2000 Systems: Authentication Management: Authentication Calculation and Ciphering Data**	***Common*** **Architectural Element**
sendEncrypInfo: AC	excEncrypInfo: portable and radio port	creatingCipherKey: AC	storeCiph: CNh (LMF and AMF)	sCK: AC
startEncryption: MSC	startEncryption: radio port	startCipherTraffic: MSC and MS	startEncryption: MT and RAN+CNv (SACF)	sC: MSC

Table 20 Common Architectural Elements for Ciphering

Note: AMF stands for Authentication Management Function and it is an IMT-2000 FE that corresponds to the AC NE.

ANSI-41/WIN Specifications: Registration and Unique Challenge/Response Authentication	**WmATM Networks: Authentication**	**GSM/GPRS and UMTS Authentication**	**IMT-2000 Systems: User Authentication**	***Common*** **Architectural Element**
obtainRandInfo: serving system (MSC) and MS	-	sendAuthInfo: MSC/VLR	TransfSecInfo: SCAF, AMFh and LMFv	sAI: MSC
perfCalcNet: serving system (AC)	perfAuthCalc: zone manager of the user's current zone	perfAuthCalc: AC	performCalculation: CNh (AMF/LMF)	aAA: AC
chkAuthResult: serving system (AC)	checkUsrAuthRes: zone manager of the user's current zone	compareRes: AC	checkResult: CNh (AMF/LMF)	cAR: AC
applyDenial: serving system (MSC)	denyUsrAccess: zone manager of the user's current zone	denyAccess: MSC/VLR	E2_Auth: RAN+CN (SACF/LMF)	nAD: MSC

Table 21 Common Architectural Elements for Authentication

ANSI-41/WIN Specifications: Paging	**WmATM Networks: Overlay Originating Party - Connection**	**GSM/GPRS/UMTS Specifications: Paging**	**IMT-2000 Systems: Paging**	***Common*** **Architectural Element**
pLA or pSA: serving MSC	paging: destination zone manager	pageMS: BTS and MSC	pagingUser: RAN+CN (SACF) to MT (MCF)	pU: MSC
assigChannel : serving MSC	estabConn : destination zone manager	assignChannel: MSC	assignChannel : RAN+CN (SACF)	aCMS: MSC
MSUnreach: serving MSC	-	E2_Pag: MSC	notifyMSUnreachable: RAN+CN (SACF)	nUU: MSC

Table 22 Common Architectural Elements for Paging

ANSI-41/WIN Specifications: Handoff Measurements	**WmATM Networks: Overlay Inter-Zone Handoff through the Previous Port**	**GSM/GPRS/UMTS Specifications: Handover Decision**	***Common*** **Architectural Element**
electCandidate: serving MSC	monitorPort: portable	takeMeasurements: BS and MSC	tM: MSC
checkCellQuality: candidate MSC	comparePower: portable	checkMeasurements: MS and MSC	cM: MSC and MS

Table 23 Common Architectural Elements for Handoff Decision

ANSI-41/WIN Specifications: Handoff Back, Handoff Forward, and Handoff to Third	**WmATM Networks: Overlay Handoff Previous – Alternative 1**	**GSM/GPRS/UMTS Specifications: Handover Procedures**	***Common*** **Architectural Element**
selectVoiceChannel: target MSC	allocChan: candidate zone manager	allocateChannel: MSC-A and MSC-B	aC: MSC
tunesNewChannel: MS and MSC	tunesNewChannel: portable	tunesNewChannel: MS and MSC	tNC: MS and MSC
checkVoiceChannel: target MSC	verifyConn: the candidate zone manager and the portable	verifyConnection: MSC-A and MSC-B	vC: MSC

Table 24 Common Architectural Elements for Inter-System Handoff Execution

ANSI-41/WIN Specifications: Registration Notification and Location Cancellation	**WmATM Networks: Overlay Registration**	**GSM/GPRS/UMTS Specifications: Location Management**	**IMT-2000 Systems: Roaming Registration**	***Common*** **Architectural Element**
sendInfo: serving system (MSC)	getUserInfo: new zone manager	getLocInfo: MS and new MSC/VLR	reqRoamReg: from the RAN+CNv to the CNh (LMFh)	sLI: new MSC
checkRoaming: serving system (MSC)	checkLocation: portable	checkLoc: HLR and MSC	-	cL: MSC
recordNewLoc: HLR	updtUsrProfile: home database	updateLocation: HLR and new VLR	updateAdd: LMFh	UPL (or UPR): HLR
recordInfo: serving system (VLR)	getLocUsrProf: current zone	insertData: HLR and new VLR	transfUsrProf: current LMFv	uTP: new VLR
deletePrevRecord: previous serving system (VLR)	delPrvUsrProf: previous zone	cancellation: HLR and old VLR	de-register: previous LMFv	dTP: old VLR

Table 25 Common Architectural Elements for Location Registration

WmATM Networks: Handoff Failure Actions	**GSM/GPRS/UMTS Specifications: Call Re-Establishment**	***Common*** **Architectural Element**
returnOldConn: the portable	tunesPrevChn: MS and MSC	tPC: MS and MSC
deallocAssignChan: candidate zone manager	releaseRes: MSC	rRA: MSC

Table 26 Common Architectural Elements for Handoff Failure

ANSI-41/WIN Specifications: Handoff Back, Handoff Forward, and Handoff to Third Procedures and Call Release Procedure	**WmATM Networks: Overlay Registration, Call Setup and Handoff**	**GSM/GPRS Specifications**	***Common*** **Architectural Element**
Release Call, release Channel, and releaseResources: MSC	ReleaseConnection: zone manager	releaseResources: MSC	rR: MSC

Table 27 Common Architectural Elements for Releasing Resources

Appendix C Summary of the MoRaR Pattern Language

The following table summarizes the behavioral and structural patterns presented in **Chapter 5**. For more information about each pattern, the reader should refer to the respective sections that are also shown in the table.

Section	Problem	Solution	Pattern Name
5.4.1	How does one ensure privacy of the subscriber's identity when sending it on the radio path?	Assign a temporary identification to the mobile user in order to avoid exchanging the subscriber's real identity and the electronic serial number of the mobile station over a non-*ciphering* (Section 5.4.3) radio path.	*Temporary identification*
5.4.2	How does one handle the mobile user's sensitive information while assuring its protection on the network side?	Create a repository of the user's sensitive information that is only accessed by functions involved in the security management process.	*Security Database*
5.4.3	How does one protect the privacy of the communication over an insecure wireless communication channel?	Apply digital cryptography mechanisms to the communication when the mobile subscriber is using a digital traffic channel in the dedicated mode.	*Ciphering*
5.4.4	How does one prevent unauthorized or fraudulent access to cellular networks by mobile stations illegally programmed with counterfeit identification and electronic serial number?	Perform an authentication operation in both the mobile station and the network sides based on an encryption algorithm and a secret key number.	*Authentication*
Figure 44	How does one reach the terminating mobile user and route the call to the user's actual location?	Send a paging message to reach the terminating mobile user only in the cells within the user's current location area (for example, several dozen cells).	*Paging*
5.4.6	How does one enable a user's mobility between location areas of the same provider or between location areas of different providers?	Create two types of repositories to handle the mobile user's information: one is the home database that is responsible for mobile users permanently registered in a location area; and the other is the visitor database that takes care of mobile users currently visiting a particular location area.	*Home and Visitor Databases*

Section	Problem	Solution	Pattern Name
5.4.7	How does one keep up to date information about a mobile user's location every time the user changes location area?	Perform a location registration procedure that consists of updating and inserting the mobile user's location, respectively, in the *home and* current *visitor databases* (Section 5.4.6) every time the mobile user changes location area.	*Location Registration*

Table 28 Patterns related to Mobility Management Functions

Section	Problem	Solution	Pattern Name
5.5.1	How does one control the quality of the radio communication link between the mobile user and the network?	Take measurements on the quality of the transmission for ongoing dedicated radio connections, check the transmission quality on the neighbor cells, and verify the load of the base station transceiver.	*Handoff Decision*
5.5.2	How does one handle the call processing information during an inter-system handoff?	Choose the first mobile switching center serving the dedicated channel as an anchor that is in charge of the communication and keeps the control during the dedicated mode.	*Anchor Mobile Switching Center*
5.5.3	How does one guarantee user's communication service assessment continuously?	Identify the candidates considered for handoff purposes and evaluate them before executing the handoff. Then, select the one that is most suitable to handle the call.	*Inter-system handoff execution*
5.5.5	How does the network minimize the use of resources that are no longer needed, such as the inter-mobile switching center circuits?	The *anchor mobile switching center* requests the previous mobile switching centers to release the unnecessary inter-MSC circuits. .	*Releasing Resources*
5.5.4	How does the network handle a handoff failure?	When a handoff failure occurs, the *anchor mobile switching center* requests the release of all resources in use. This anchor also requests the notification and disconnection of the other party in case a call was in progress.	*Handoff Failure Actions*

Table 29 Patterns related to Radio Resource Management Functions

Table of Acronyms and Terminology

3G	Third Generation	**MSC**	Message Sequence Chart (interaction diagram)
3GPP	3rd Generation Partnership Project	**MSC**	Mobile Switching Center
AC	Authentication Center	**NE**	Network Entity
ACF	Authentication Control Function	**NRM**	Network Reference Model
AMPS	Analog Mobile Phone System	**ODP**	Open Distributed Processing
ANSI	American National Standards Institute	**PCS**	Personal Communication System
AVR	Available Bit Rate	**PDC**	Personal Digital Cellular
BCSM	Basic Call State Model	**PSTN**	Public Switched Telephone Network
BS	Base Station	**LAN**	Local Area Network
BSC	Base Station Controller	**LRFv**	Location Registration Function - VLR
BST	Base Station Transceiver	**LRFSM**	LRF State Model
Camel	Customized applications for mobile network enhanced logic	**MACF**	Mobile Station Access Control Function
CAVE	**C**ellular **A**uthentication and **V**oice **E**ncryption	**MAP**	Mobile Application Part or Mobile Application Protocol
CBR	Constant Bit Rate	**OSI**	Open System Interconnection
CCF	Call Control Function	**RACF**	Radio Access Control Function
CCH	Control Channel	**RCF**	Radio Control Function
CDMA	Code Division Multiple Access	**RF**	Radio Frequency
D-AMPS	Digital AMPS	**RTF**	Radio Terminal Function
ETSI	European Telecom. Standard Institute	**SCA**	Selective Call Acceptance
FDMA	Frequency Division Multiple Access	**SCEF**	Service Creation Environment Function
FE	Functional Entity	**SCF**	Service Control Function
GPRS	General Packet Radio Services		
GSM	Global System for Mobile Communications	**SCP**	Service Control Point
HLR	Home Location Register	**SMAF**	Service Management Access Function
ICS	Incoming Call Screening	**SME**	Short Message Entity
IE	Information Element	**SMF**	Service Management Function
IF	Information Flow	**SN**	Service Node
IMT-2000	International Mobile Telecommunications Systems 2000	**SS7**	Signaling System 7
IN	Intelligent Network	**SSF**	Service Switching Function
INAP	Intelligent Network Application Protocol	**SRF**	Specialized Resource Function
INCM	IN Conceptual Model	**TDMA**	Time Division Multiple Access
IP	Intelligent Peripheral	**TCH**	Traffic Channel
IS	Interim Standard	**TIA**	Telecommunications Industry Association
ISDN	Integrated Services Digital Network	**TTC**	Telecommunications Technology Committee
ISO	International Standards Organization	**VBR**	Variable Bit Rate
ITU	International Telecommunications Union	**VLR**	Visitor Location Register
ITU-T	ITU-Telecommunications Standardization Sector	**WIN**	Wireless Intelligent Network
JDC	Japanese Digital Cellular Systems	**WmATM**	Wireless mobile ATM Network
LOTOS	Language of Temporal Ordering Specifications	**VBR**	Variable Bit Rate
LRFH	Location Registration Function - HLR	**UBR**	Undefined Bit Rate
MC	Message Center	**UCM**	Use Case Map
MoRaR	Mobility and Radio Resource Management	**UMTS**	Universal Mobile Telecom. System
MS	Mobile Station	**UTRA**	Universal Terrestrial Radio Access

Index