Reusability at the Early Development Stages of Mobile Wireless Communication Systems

Rossana Andrade[1]
TSERG, School of Information Technology and Engineering, University of Ottawa
150 Louis Pasteur, MCD 409, Ottawa, Ontario, K1N 6N5, Canada
E-mail: randrade@site.uottawa.ca

and

Luigi Logrippo
TSERG, School of Information Technology and Engineering, University of Ottawa
150 Louis Pasteur, MCD 310, Ottawa, Ontario, K1N 6N5, Canada
E-mail: luigi@site.uottawa.ca

ABSTRACT

Development productivity and quality improvements, as well as cost reduction, are the main factors 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 the budget make the reusability concept attractive. In this context, this work provides a framework for mobile wireless communication systems that allows designers to apply reusability to the early stages of the development process. In order to achieve this, firstly, existing systems are investigated and common network entities and their respective functionalities are identified. Secondly, the framework is generated based on these commonalities with a technique called Use Case Maps.

Keywords: Mobile Wireless Communication Systems, Software Reuse, Development Process, Use Case Maps, Network Reference Model, Framework.

1. INTRODUCTION

According to [13], software reuse improves development productivity and quality by means of saving time when reusing parts from other projects that have previously been developed, tested and debugged.

With the reusability concept in mind, this work concentrates on the mobile wireless communication domain where different systems provide support for mobile users. Although the differences among the existing systems make them incompatible, they use common network entities that perform the same basic procedures [7], such as: authentication, location registration, connection establishment, paging, and handoff.

First, the similarities among these systems are identified. Then, their common functional behaviors are described with a technique called **U**se **C**ase **M**aps (UCMs) [5] and a reference model is generated based on the common network entities. After this, the functional behavior is mapped into the reference model and a framework is proposed. This framework represents a repository of functionalities and components ready to be reused by designers in building new mobile systems. As a case study, the high-level framework has been already applied to develop a wireless mobile ATM network and the results are presented in [3]. In addition, the authors are recently applying the software pattern concept to this framework in order to develop patterns for mobile systems.

The proposed framework is suitable for requirements capture and analysis development stages [9]. Design, implementation, and testing stages are left for future work.

---

[1] Ph.D. Candidate at the SITE, Assistant Professor at the Computer Science Department, Federal University of Ceará, Brazil, and sponsored by CAPES (Brazilian Federal Agency for Graduate Studies).

2. USE CASE MAP NOTATION

UCM is a visual notation that helps to express the general structure of a system in terms of causal relationships between responsibilities. UCMs explain the system functionalities and allow designers to think about the overall system behaviour at the early stages of the development process. This section includes examples referring to the specific UCM notation and concepts used to describe the common functionalities and network entities presented in Sections 3 and 4.

Figure 1 shows a UCM scenario that is called *root map* and Figure 2 presents its sub-maps. This root map starts with a *triggering event* or a satisfied *pre-condition* (filled circle labeled StartPoint) and ends with one or more *resulting events* or satisfied *post-conditions* (bars labeled EndPoint_1, EndPoint_2, and EndPoint_3). *And-forks* represent composite UCMs that split a path into parts (sub-paths) that proceed concurrently ([label_1] and [label_2] sub-paths).

A *stub* (such as StaticStub and DynamicStub in Figure 1) identifies places where details are delayed to a sub-map, called *plug-in*. A static stub is represented by a plain diamond and it is bound to only one plug-in. 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 2a and Figure 2b) according to a selection policy described in the pre-conditions. 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) in order to enable the 2^{nd} plug-in represented by Figure 2b.

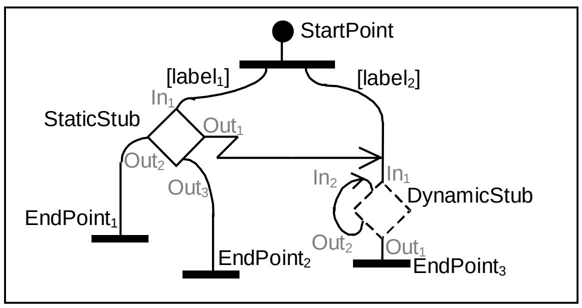


Figure 1. The Root Map

The zig-zag UCM notation (Out_1 path in Figure 1 and [timeout] path in Figure 2b) is used to describe an exception or an abort path that can terminate the execution of another path. This notation can also be used to describe an interaction between a stub, such as the StaticStub, and another path. For instance, a plug-in is statically bound to the StaticStub when a certain pre-condition is satisfied and it replaces the [label_2] path on the right-hand side of the figure.

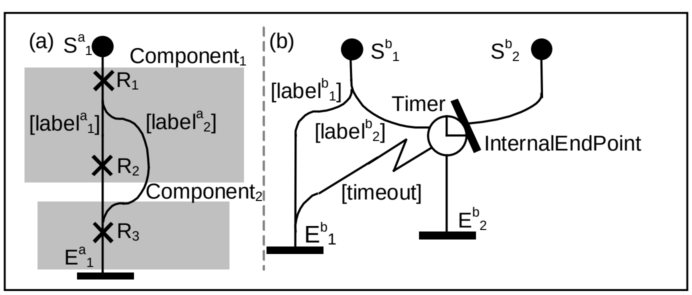


Figure 2. (a)1st and (b)2nd Plug-ins for DynamicStub

A *route* is a path that links an initial cause (start point) to a final effect (end point) as shown in Figure 2a where *responsibilities* represented by crosses are activated along the path to form routes. These *responsibility* points can identify tasks to be performed, events or actions to occur, or operations on data items. *Or-forks* and *Or-joins* represent composite UCMs that, respectively, can be split into two different paths (such as [label^a_1] and [label^a_2]) or can be joined into only one path (such as [label^a_1]

joining [$label^a{}_2$]). There is no level of concurrency associated with *Or-forks* and *Or-joins*. For instance, one of the alternative routes <$S^a{}_1$, R_1, [$label^a{}_1$], R_2, R_3, $E^a{}_1$> or <$S^a{}_1$, R_1, [$label^a{}_2$], R_3, $E^a{}_1$> is chosen depending on the result of a specific condition described in R_1.

When UCMs combine paths and responsibilities without components, they are called *unbound maps*. On the contrary, when they represent scenarios as causal paths cutting across organizational structures of components, they are called *bound maps.* The *component* notation describes runtime entities of systems, such as, objects or a group of them, functional elements, network entities, and processes. These entities ($Component_1$ and $Component_2$ in Figure 2a) form a component substrate.

The notation also allows for timers as shown in Figure 2b. Timers are special waiting places triggered by the arrival of a specific event. In case where the event does not arrive in time, a zig-zag path is triggered ([timeout] label in the figure).

A UCM drawing tool called UCM Navigator is available and was applied to this work to edit the maps as well as to check for duplication among responsibility names. For a detailed description of the UCM notation, the interested reader may refer to [5]. For information about the current status of the tool and references about UCMs refer to [15].

3. DESCRIPTION OF COMMON FUNCTIONAL BEHAVIOR WITH UNBOUND UCMS

The functional behaviors of mobile wireless communication systems are often grouped in three different categories [11], as follows: mobility, communication, and radio resource management functions. Based on these functions, this work identifies similarities among the systems at a high level of abstraction.

For instance, every mobile system performs mobility management functions to update information about a mobile user location, to locate a user, and to guarantee location access security. A location cancellation function is also invoked by the network to delete information from the previous location area of the mobile station. Furthermore, all systems perform communication management functions for routing, for establishment of the communication among mobile users, or mobile and fixed users, and for clearing connections. The identified common behavior among the radio resource functions is the handoff function performed to maintain the quality of link when the user moves from one location to another. This work concentrates on the handoff involving an exchange of mobile switching centers, since this is the only case that affects the signalling at the upper layers. This function also handles handoff failures which occur due to the lack of good links surrounding the mobile station.

The next sub-section describes the common functionalities among mobile wireless communication systems with unbound UCMs. First, the common functional behaviour is presented in terms of separated maps. After this, the UCMs are represented by *plug-ins* bound to *stubs* that hide unecessary details to show how these maps work together. By using the stub notation, the designer is also able to see the global picture of the system at a high level of abstraction.

Common Functional Behavior with UCM Plug-ins

The visual representation of the common functionalities without considering which network entity performs them is the main advantage of applying *unbound* UCMs. At the beginning, the focus is on the UCM description of every common functionality as an isolated map. These maps are bound to high-level stubs as shown in the next sub-section.

Figure 3 and Figure 4 illustrate UCMs related to the mobility management functions. The Authentication plug-in depicted in Figure 3a is triggered when a mobile station powers on, changes location area, or when the network requests the authentication (S^{Auth} start point). First, the sAI (send Authentication Information) responsibility that contains the result of an authentication operation performed by the mobile station is sent to the network. 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 outcome of the comparison between the respective results. In case of denied authentication, the mobile user is notified ([a2] path with nAD responsibility). Otherwise, the network is notified ([a1] path with nN responsibility).

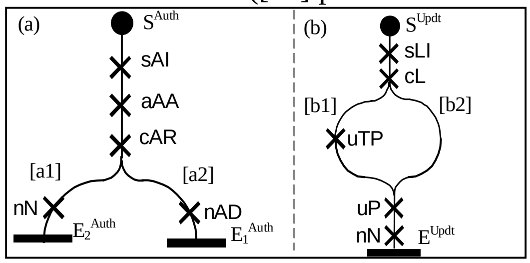


Figure 3. (a) Authentication and (b) Updating Information Plug-ins

Figure 3b describes the Updating Information plug-in 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 whether the mobile user is visiting a new location area (both visitor and home databases are updated) or not (just the home database is updated). uP (update home user Profile) and uTP (update Temporary user Profile) responsibilities are operations on the home and visitor database items. Sub-paths labeled [b1] (visiting location area) and [b2] (home location area) are joined and the network is notified (nN responsibility) about the successful operation. Unsuccessful outcomes are not shown in Figure 3b, but they can occur due to network or database failures as presented in the next sub-section (see the root map).

Figure 4a represents the Location Cancellation plug-in that is triggered after the network updates the information about the current location of the mobile station (S^{Cancel} start point). Its purpose is to delete the user profile in the previous visiting location area. As depicted in the figure, cPI checks the profile information. When the previous location is a visiting area, the temporary profile is deleted from the previous visiting database (dTP responsibility in the [a1] path) and the network is informed (nN responsibility). Otherwise, the home database has already been updated and no action is performed.

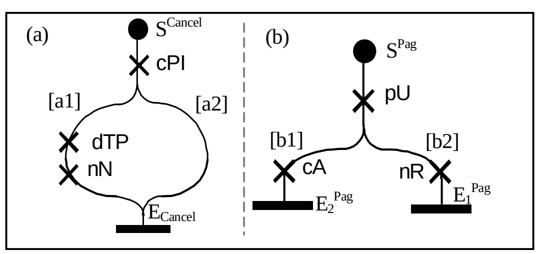


Figure 4. (a) Location Cancellation and (b) Paging Plug-ins

The last plug-in related to mobility is shown in Figure 4b. The Paging plug-in is responsible for the location of a mobile station within the whole location area where this mobile station is registered. The pU responsibility issues the paging. Paging starts when an incoming call arrives to an idle terminating party (S^{Pag} start point), and it ends after either the channel is assigned (successful outcome) or the mobile station is unreachable (unsuccessful outcome) what is represented by, respectively, cA and nR responsibilities.

Figure 5 and Figure 6 depict the communication management plug-ins. Functionally, users are not identical and we have to distinguish between the user who requested the establishment of the communication (originating party) and the terminating party. Figure 5a 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 the successful or unsuccessful outcomes (respectively, E_2^{Rout} or E_1^{Rout} end points).

After finishing the routing function, the current status of the terminating party is analyzed. Figure 5b shows the Status plug-in 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.

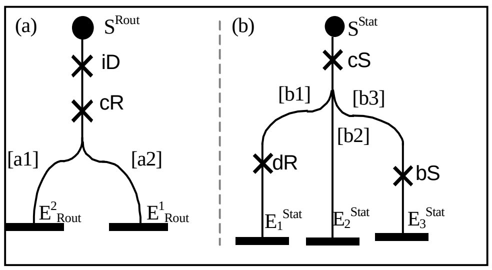


Figure 5. (a) Routing and (b) Status Plug-ins

Figure 6a illustrates the Terminating Party plug-in which S_1^{Term} start point follows the E_2^{Stat} end point of the Status plug-in. After enabling the first start point, an alerting indication (aI responsibility) is played at the originating and terminating user sides, a timer starts and waits for the terminating user's answer that triggers the S_2^{Term} start point and, consequently, the E_2^{Term} end point. If the user answers the incoming call in a certain amount of time, the call is finally set up ([a2] path). Otherwise, the timeout path (zig-zag path labeled [a1]) is performed. If the call is established, the users are able to exchange information (voice, data, fax, or video) as depicted in Figure 6b. The loop represented by the [b1] path is performed until one of the users disconnects (as shown in the next section in the root map).

Figure 6c shows the Disconnection plug-in responsible for releasing the network resources (rR responsibility) after a user hangs up the phone or presses the *end* key. The other user is notified about this event with the nU responsibility. 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 plug-ins in execution at this time that a disconnection has occurred and that the disconnection plug-in has replaced the current UCM routes. The [c2] sub-path finishes the actions of the Disconnection plug-in.

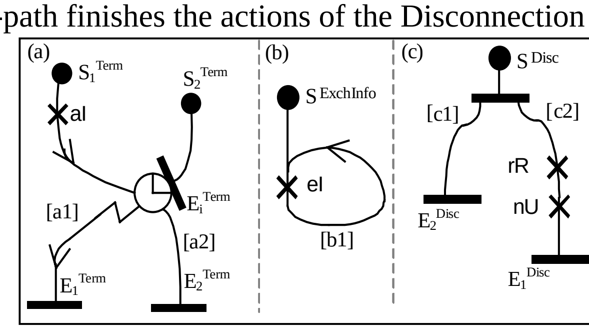
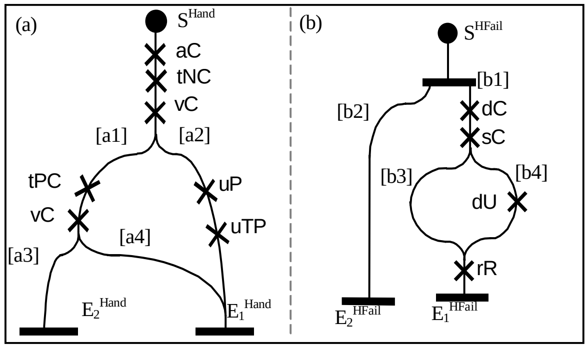


Figure 6. (a) Terminating Party, (b) Exchange Information, and (c) Disconnection Plug-ins

Finally, Figure 7 represents the last set of common functionalities tackled in this work. These functions describe inter-system handoffs and the actions performed by the network when a handoff failure occurs. Figure 7a 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), and according to the communication status (whether a communication between two users is occuring or not), the call is also rerouted. 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. The profile is updated (uP and uTP responsibilities) in case of positive result. Otherwise, the mobile station tunes to the previous channel that is checked again to verify whether the link quality is satisfactory or not (respectively, [a4] and [a3] sub-paths).

Figure 7. (a) Handoff and (b) Handoff Failure Actions Plug-ins



If the handoff fails ([a3] sub-path in Figure 7a leads to the Figure 7b), the [b2] sub-path indicates to the other plug-ins in execution at this time that a problem has occurred and that the Handoff Failure Actions plug-in has replaced the current UCM routes. The [b1] sub-path performs the proper actions. First, the channel is de-allocated (dC responsibility). Second, in case the user is involved in communication (status of the Communication - sC responsibility), both users are disconnected (dU responsibility in the [b4] sub-path). Otherwise, no action is performed. At the end, all network resources are released (rR responsibility).

Plug-ins Encapsulated into Stubs

After identifying all common behaviors and representing them with unbound UCMs, the next step is to show an example of how these sub-maps work together. Figure 8 provides this example by encapsulating the plug-ins presented in Figure 3, Figure 4, Figure 5, Figure 6, and Figure 7 into static and dynamic stubs.

Informally, the root map illustrates the “big picture” of a simplified mobile wireless communication system. The relationships among the common functionalities are understood by following the flow of the root map shown in Figure 8. First of all, the mobility management functions (MM stub) are performed. After this, the communication management functions (CM stub) and the handoff procedures (HP stub) are triggered at the same time. Handoff failure actions (HFA stub) are also handled at this level. At any time, a user is able to power off the mobile station, a network failure or a database failure can occur, and a user is able to disconnect. These four behaviors are represented by different plug-ins bound to the dynamic stub labeled FourBs. Only the Disconnection plug-in is shown in this paper for lack of space.

A plug-in for the MM dynamic stub is the Mobility Management illustrated in Figure 9a, as follows: the Auth stub is bound to the Authentication plug-in, the Update stub is bound to the Updating Information plug-in, and the Cancel stub is bound to the Location Cancellation plug-in. The Auth stub has two outgoing paths labeled [b1] and [b2] that correspond to the end points of the Authentication plug-in (Out_1 corresponds to E_1^{Auth}, and Out_2 corresponds to E_2^{Auth}). The cR (check registration) responsibility is activated along the [b2] sub-path to decide whether the mobile station is registered at the current location area or not. One of the alternatives sub-paths ([b3] or [b4] labels) is followed after this decision.

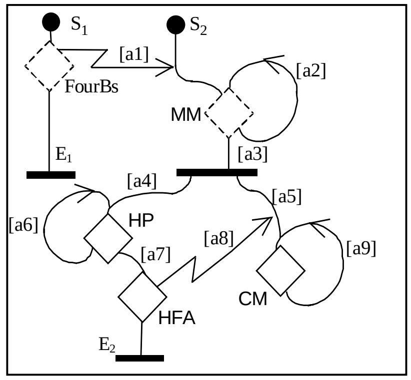


Figure 8. The Mobile System Root Map

Figure 9b depicts the Communication management plug-in bound to the CM static stub. This scenario is composed of Paging, Routing, Status, Term, and ExchInfo stubs that are bound to the plug-ins presented in Figure 4b, Figure 5a, Figure 5b, Figure 6a and Figure 6b, respectively.

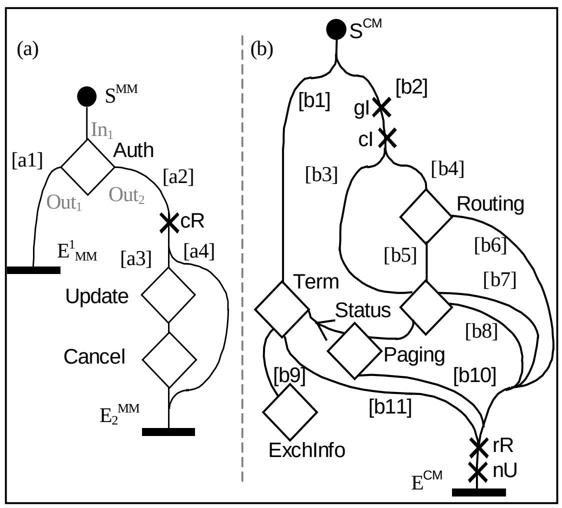


Figure 9. Plug-ins bound to the (a) MM and (b) CM Stubs

The flow is triggered (S^{CM} start point) after either a user dials a number or a user receives a call. These two different behaviors are expressed in terms of an or-fork that splits the main path into [b1] and [b2] sub-paths, respectively. Alternative paths labeled by [b3] (the terminating party is a fixed user) and [b4] (the terminating party is a mobile user) are chosen according to the following sequence of 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. For example, [b6], [b7], [b8], [b10], and [b11] are unsuccessful attempts to set up a call and, then, resources are released (rR responsibility) and the user is notified (nU responsibility). The [b9] sub-path represents a successful establishment of a call between two users.

This flow finishes either with an unsuccessful communication attempt (E^{CM} end point), or with a user disconnection. At any time, one of the users is able to hang up or press the end key. The Disconnection plug-in (shown in Figure 6c) bound to the FourBs dynamic stub in the root map expresses this behavior.

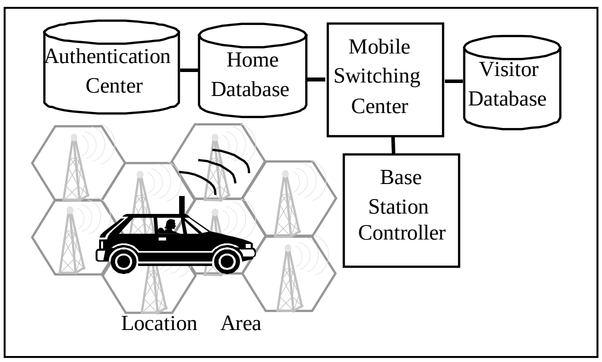
After describing these functionalities with unbound UCMs, common network entities are identified among the existing systems. The next section describes the network reference model, and the mapping of the unbound UCMs to this model that constitutes the proposed high-level framework.

4. MAPPING OF THE FUNCTIONAL BEHAVIOR TO THE NETWORK REFERENCE MODEL

Most mobile wireless communication systems use similar components to perform their functional behavior. These components are usually expressed in terms of network entities that are an abstraction of the physical components often called physical entities in the literature [4][8]. The concept of network entities comes from the WIN standard [14]. Even though they determine a specific structure for a mobile system, these entities are still one step away from the physical implementation. However, implementations require ultimately that the network entities be allocated to specific physical entities. In short, one or more network entities are grouped into a physical entity at the implementation stage.

Figure 10 illustrates a simplified mobile wireless environment with the following common network entities identified among existing systems: mobile station (represented as a car), base station controller, home location database, visitor location database, authentication center, and mobile switching center. As shown in the figure, the mobile wireless environment is divided into cells. 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 base station controller is responsible for a set of base station transceivers and the connection between them is done through the radio access ports. Many mobile stations share the capacity of each base station transceiver. Databases are responsible for keeping information about mobile users' location, services, and equipment. The wireless backbone communicates with the fixed backbone (PSTN and ISDN systems) using the wired access ports (not shown in the figure).

Figure 10. A Simplified Mobile Wireless Environment



The network reference model that is proposed in this work as a reusable design element for the mobile system development is based on this environment. However, since this paper focuses on functions performed by the upper layers, base station transceiver and base station controller network entities are not considered. A location area is composed of several cells, a switch and its respective databases. Mobile stations communicate directly with the mobile switching centers and the users' movements are represented by the change of location area.

The Proposed High-Level Framework

According to [12], frameworks are ready-to-use and semi-finished building blocks. A well-designed framework also defines the composition and interaction among its components what constitutes the overall system architecture. This work considers frameworks as collections of functions working together and their respective mappings into the network structure. The main difference from the traditional framework concept is the focus on the requirements and analysis levels. In short, developers often re-use either design elements or code, the framework that is proposed in this work includes reusable requirements and analysis models.

This work applies functional modeling that is a valuable tool for identifying the functions to be performed by the network entities without restricting possible implementations. In this context, the

UCM notation is appropriate to show how the common behaviour and the reference model interact and the UCM components represent the network entities.

Since the proposed framework is a generic scenario ready to be re-used at the early development stages, it is called high-level framework. Figure 11 depicts the combination of two bound UCMs that are part of the proposed framework. These scenarios are integrated when the call between the originating and the terminating parties is established. The UCM paths shown in gray are not performed.

First, the map on the left-hand side is able to be triggered after satisfying the following pre-conditions: the originating mobile user has already powered on, and the mobility management functions have been successfully performed. Only when the user dials a number (the terminating party number), the flow starts. After reaching the and-fork, handoff and communication functions are performed in parallel and independently. In this case, the originating handoff is successful as well as the attempt of establishing a call, since the terminating user answers the call. The terminating party is not roaming (handoff is not performed at the other side). These scenarios are merged for the exchange of information. When one of the users hangs up, the call finishes (as shown in the disconnection plug-in and root map, respectively, Figure 6c and Figure 8).

The following components perform the responsibilities depicted in the figure: Originating Mobile Station (MS-O), Originating Mobile Switching Center (MSC-O), Home Location Register (HLR), Visitor Location Register (VLR), Terminating Mobile Station (MS-T), Terminating Mobile Switching Center (MSC-T), Originating and Terminating Mobile Stations (MS-OT).

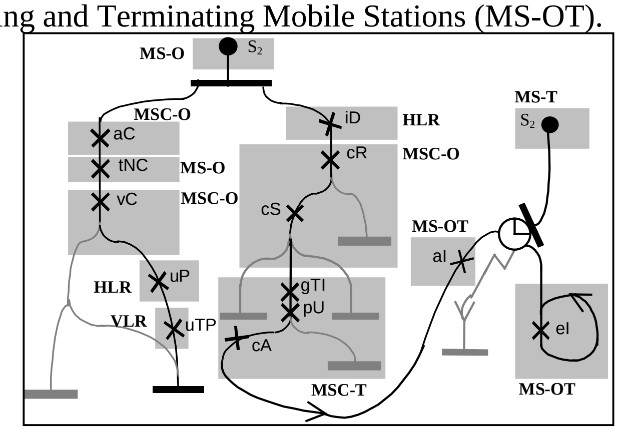


Figure 11. Integration of the Originating and Terminating Scenarios

A possible approach for re-using the proposed framework at the requirements and analysis stages is presented, as follows: first, the focus is on re-using the UCM description of the common functionalities without considering which network entity performs them. At this stage, the designer adds new functionalities to build her specific system. At the *analysis* stage, the designer concentrates on re-using the network reference model and when it is necessary, new network entities are included. The last step before the design stage is to join the unbound UCMs with the reference model. As mentioned in [6], the best place to select components is at the early stages of the development life cycle. Under such circumstances, part of the drawbacks of choosing components as an implementation decision are solved. Furthermore, inconsistencies between design decisions without reusable components and implementation decisions with components are avoided. Component deficiencies and new functionalities that require extra coding are also reduced at later stages.

For lack of space, scenarios that show the mapping of all functional behaviors mentioned earlier to the network reference model are omitted. However, the previous sections convey the main goal of this work that is to show the feasibility of applying the reusability concept to the mobile wireless communication domain. This high-level framework is suitable to be re-used by the **G**lobal **S**ystem for **M**obile Communications (GSM) [11] and by the **A**merican **N**ational **S**tandards **I**nstitute – 41 (ANSI-

41) [4] based systems when adding new features. Design decisions regarding messages, parameters, authentication algorithms, and interfaces among components are taken at later stages.

5. CONCLUSION

We have outlined concepts that can enable designers to re-use common network entities and their respective functional behaviors starting from the early stages of the development process of a mobile wireless communication system. The proposed high level framework is reusability driven, applicable to attend the designer needs in the development of new systems or in the maintenance of existing ones. A new system developed on the basis of this framework would be more suitable to be integrated with existing systems. Furthermore, the concept of framework is a promising way to overcome potential incompatibilities between existing and new systems. As a result, global roaming and seamless mobile wireless services as discussed in [7] can be developed more easily.

When reusability is taken into consideration to the early stages, it saves time and costs by eliminating the effort to understand the requirements. It also reduces the activities associated with the analysis stage [10], such as describing how behavior and structure models work together, and it reduces the changes necessary to the reusable code at later stages. Although software reuse has many advantages, the investment to make the software reusable by ensuring its quality, flexibility and providing more documentation is higher than the one required for non-reusable systems, and its benefits come in a long term [6][10][13].

The UCM notation is powerful enough to capture common knowledge and when combined to a structure of network entities, these maps are still reusable even if the underlying structure is modified at the later stages. Thus, UCMs are a proper software artifact to represent the high-level framework. The University of Ottawa LOTOS Group has successfully applied UCMs to the description of Wireless Intelligent Network standards [14] as presented in [1] and the combination of UCMs and formal methods can be found in [2] and [3].

6. ACKNOWLEDGMENTS

I would like to thank the UofO's LOTOS Group for their support. Many thanks to the graduate students of the CSI 5171 course for their judicious comments. Finally, I acknowledge CAPES for its financial support.

7. REFERENCES

[1] Amyot, D., Andrade, R., "Description of Wireless Intelligent Networks with Use Case Maps", *Proc. Brazilian Symposium on Computer Networks (SBRC'99)*, Salvador (BA), Brazil, 25-28 May 1999, pp. 418-433.

[2] Amyot, D., Andrade, R., Logrippo, L., Sincennes, J., and Yi, Z., "Formal Methods for Mobility Standards", *IEEE 1999 Emerging Technology Symposium on Wireless Communications & Systems*, Dallas, US, April 1999.

[3] Andrade, R., "Applying Use Case Maps and Formal Methods to the Development of Wireless Mobile ATM Networks", to appear in the Proc. of the Fifth NASA Langley Formal Methods Workshop, Williamsburg, Virginia, June 2000.

[4] ANSI/TIA/EIA ANSI-41-D, Cellular Radiotelecommunications Intersystem Operations, 1997.

[5] 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, pp. http://www.UseCaseMaps.org/UseCaseMaps/pub/tse98final.pdf

[6] Graham, Ian, Requirements Engineering and Rapid Development an object oriented approach, Acm Press Prentice-Hall, 1998.

[7] Grinberg, Arkady, Seamless Networks: Interoperating Wireless and Wireline Networks, Addison-Wesley, 1996.

[8]ITU-T Q.1200 General Series, Intelligent Networks Recommendation Structure, Geneva, 1995.
[9]Jacobson, Ivar et. al., Object-Oriented Software Engineering (A Use Case Driven Approach), ACM Press, Addison-Wesley, 1992.
[10]Leach, Ronald F., Software Reuse: methods, models, and costs, Computing McGraw-Hill, 1997.
[11]Mouly, Michel, and Pautet, Marie-Bernadette, The GSM System for Mobile Communications, M. Mouly et M. Pautet, 1992.
[12]Pree, Wolfgang, Design Patterns for Object-Oriented Software Development, Addison-Wesley Publishing Company, ACM Press, 1995.
[13]Rada, Roy, Software Reuse: Principles, Methodologies, Intellect Ltd., 1995.
[14]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.
[15]Use Case Maps Web Page: http://www.UseCaseMaps.org since 1999.