Applying Use Case Maps and Formal Methods to the Development of Wireless Mobile ATM Networks

Rossana M. C. 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

Abstract. Over the last few years, several alternatives for adding mobility to Asynchronous Transfer Mode (ATM) signaling protocols have been presented in the literature. However, most of the current approaches for wireless mobile ATM (WmATM) network development include only text and information flows. As a result of the complexity involved in handling mobility, communication, and handoff procedures for WmATM networks, current approaches can lead to ambiguities, gaps, inconsistencies and undesirable interactions at the later stages of the development process where changes can be costly and provoke backward incompatibility. With these problems in mind, this work proposes a development approach that includes a technique called Use Case Maps (UCMs), and the following formal methods: Language of Temporal Ordering Specifications (LoTOS) and Message Sequence Charts (MSCs). UCMs are applied at the *requirements capture* and *analysis* stages, followed by LoTOS and MSCs at the *design* stage. Besides providing a better and more precise description of the system at the early stages, our main goal is to combine these techniques and to help solve design problems like the ones mentioned above. As a case study, WmATM network procedures are specified using the proposed approach.

Keywords. Causal Scenarios, Use Case Maps, Formal Techniques, Language of Temporal Ordering Specifications, Message Sequence Charts, Wireless Mobile ATM Networks.

1. INTRODUCTION

Although **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, most current development approaches for **W**ireless **m**obile **ATM** (WmATM) networks include basically text and information flows at the early stages. As a result of the complexity involved in handling mobility, communication and handoff procedures, these approaches can lead to ambiguities, gaps, inconsistencies and undesirable interactions at the later stages.

FDTs, such as LoTOS [21] (the **L**anguage **o**f **T**emporal **O**rdering **S**pecifications) and MSCs [18] (**M**essage **S**equence **C**hart**s)** have not only shown resiliency in the usability, but have also improved in tool support and training over the last 15 years [4][12][13]. Even though LoTOS and MSCs can be used at different levels of abstraction, they require precision on 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, visual techniques such as **U**se **C**ase **M**ap**s** (UCMs) [8][9] give to the designer the capability to work with whatever amount of detail is available. Thus, they are appropriate for the early stages.

In this context, we propose the application of UCMs, LoTOS, and MSCs at different stages of the system development process. UCMs are applied at the *requirements capture* and *analysis* stages, followed by LoTOS and MSCs at the *design* stage. The proposed approach is applied to the development of a prototype for WmATM networks. Mobility, communication and handoff procedures are firstly described with UCMs and, in conformance to that, formally specified and validated with LoTOS. MSC scenarios are automatically generated from the LoTOS specification in order to represent the results of the validation and to facilitate the implementation of protocols.

This paper is divided into 7 sections. An overview of the WmATM networks is given in Section 2. Section 3 illustrates the proposed development approach. A big picture of the relationship among mobility, communication and handoff procedures for WmATM networks is described with UCMs in Section 4. After that, the corresponding LoTOS specification is presented in Section 5. Section 6 shows

---

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

the generated MCS scenarios and lastly, Section 7 discusses our main contributions. Related works are also mentioned in Sections 4, 5 and 6.

2. CASE STUDY: 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 [23]. Nowadays, ATM is viewed as a strong candidate to extend these services to portable systems using wireless technologies [1][28][30]. Accordingly, several alternatives for adding mobility to ATM signaling protocols have been presented in the literature [5][6][7][10][24][31]. For example, [7] and [28] present WmATM networks as a wireless extension of ATM networks with mobility and no modification in the existing ATM signaling protocols. On the contrary, [24], [30] and [31] believe that minimum changes should be done in the ATM networks to support mobility and to achieve a global WmATM network environment. In [5] and [6], the authors present two different signaling protocols to support both alternatives.

2.1. A Typical WmATM Network Environment

Figure 1 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. The wireless backbone can communicate with the *ATM network backbone* using *wired access ports*.

We choose a simplified wireless mobile ATM network as a case study, since it is representative of large and complex systems and touches upon common problems in the development process of these systems. The WmATM reference architecture considered in our work includes *mobile stations*, *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. Since we focus on signaling protocols for upper layers, *base stations* are not considered. Mobile stations communicate directly with WmATM switches and mobility occurs every time the mobile station changes a location area (represented by changing the WmATM switch).

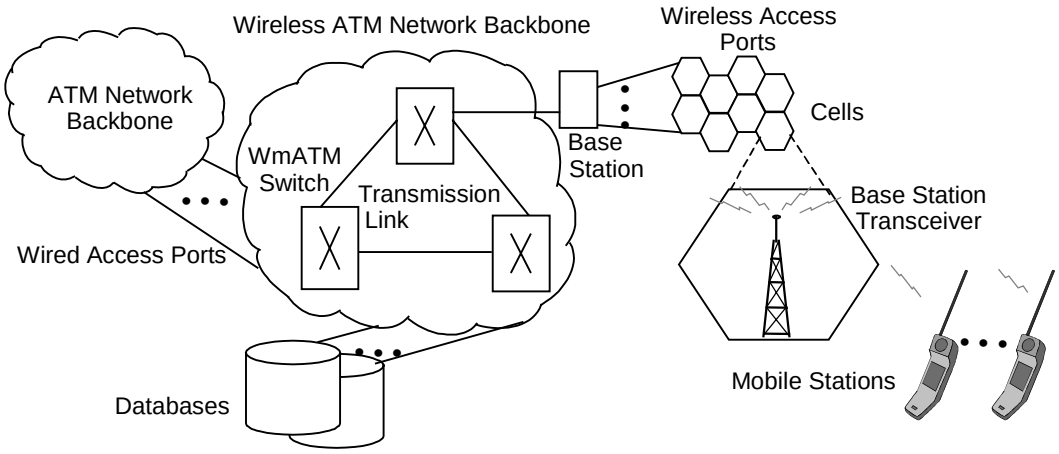

Figure 1. A Possible Wireless Mobile ATM Network Environment

(adapted from Figures 2, 3 and 4 of [24][7] and [1], respectively)

In the ATM fixed networks [17], there is no need for databases since each fixed station has a user's identification that determines where the user is and how to route a call to the user. Our work focuses only

on the specification and validation of connections between mobile users and between mobile and fixed users.

During the development of the WmATM network environment, each component of the reference architecture is specified with its corresponding protocols related to mobility, communication, and handoff. Informally, *mobility management functions* provide a secure environment for mobile users, update location information and perform the user de-registration in an old location area when a mobile user roams and registers in a new location area. *Communication management functions* are used to establish, release and maintain calls between two mobile users, from mobile to fixed users, and from fixed to mobile users at their request. Meanwhile, *handoff functions* give the mobile user the freedom of motion beyond a wireless coverage area by maintaining the quality of a link whenever a user moves from one location to another.

2.2. Current Development Approaches

Several signaling protocol alternatives for wireless mobile ATM networks have been presented in the literature. Their development approaches involve *informal descriptions* as text at the early stages followed by *flow charts* [7][31], *state models* [10], or *information flows* [5][6][24] as shown in Figure 2.

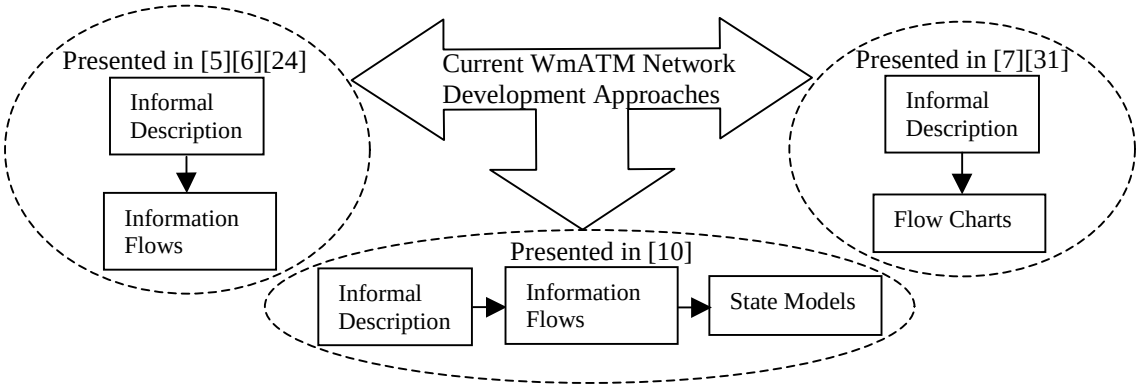

Figure 2. Different Development Approaches for WmATM Networks

When signaling protocol requirements are described only with text, they contain redundancies and become cumbersome to read, understand and manage at the later stages. An attempt to solve these problems is usually done following *informal description* by *information flows* (also known as sequence diagrams or message sequence charts). However, they are only necessary for detailed designs where decisions about messages, parameters, data, and system components need to be taken. *State models* are also suitable for later stages since they demand full precision during the definition of each state and underlying architecture. As a result, from the informality of the text to these formal models, a description gap can be identified that leads to protocol inconsistencies and undesirable interactions at the 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.

3. THE PROPOSED DEVELOPMENT APPROACH

In order to overcome the problems mentioned earlier, this work proposes the combination of techniques such as UCMs, LoTOS and MSCs in the system development process. The proposed approach splits this process into a number of steps, called *stages*, each of which produces a more detailed view of the system. By decomposing a system into manageable units, we are applying a strategy used by object-and function-oriented software community when dealing with the complexity of large systems [11]. Besides this, we are adding rigor to the approach by using formal techniques. Figure 3 depicts the proposed development approach with *requirements capture*, *analysis* and *design* stages. Arrows show how these stages interact and represent the relation of *dependency*. The approach allows several development cycles which

represent the gradual and iterative characteristics of the approach. Since implementation and testing are not considered in our work, we omit these stages in the figure.

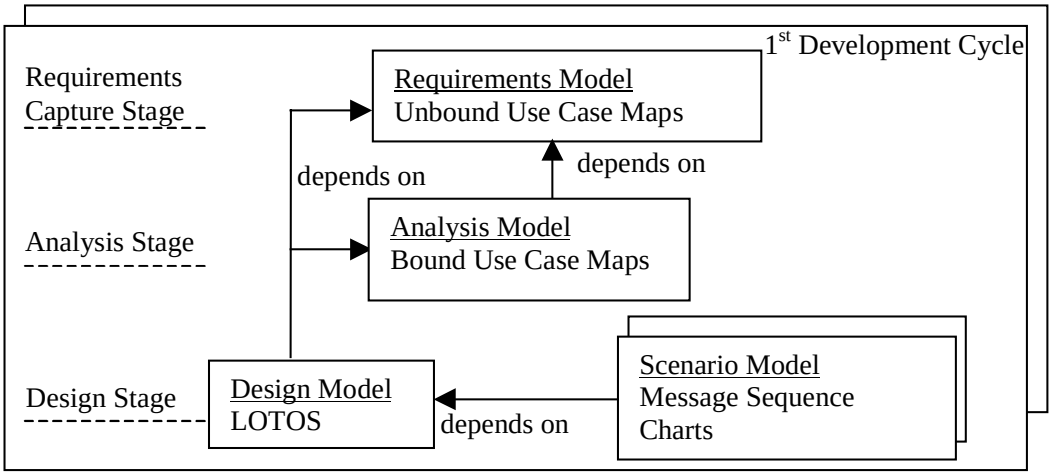

Figure 3. Proposed Development Approach

The *requirements capture* stage is the first place in the development of a system. The elicitation of meaningful requirements identifies and documents what the system is supposed to do and which are the main functions to be described. Use cases are used in the software domain to describe the sequence of events an actor (an external agent) uses in a system to complete a process [19]. To avoid ambiguities caused by narrative documents such as simple text or even textual use cases, these informal descriptions are replaced by a *requirements model* developed with UCMs. Since the notation is informal and intuitive, it is suitable for the early stages when the user needs are described in a high-level of abstraction and designers are discussing, visualizing, and explaining the overall behavior of a system. At the beginning, when organizational structure details are not available, this visual technique describes high-level scenarios in terms of *causal relationships* between *responsibilities* (called *unbound UCMs*). The *stub* notation is used to hide functions that are detailed at later stages.

Design decisions regarding which system component is responsible for a specific action, event or transaction are taken during the *analysis* stage. The functional behavior is further investigated and mapped to system components (part of the reference architecture). The *analysis model* is generated with *bound UCMs*. Detailed descriptions about what the system does are represented in terms of UCM notation: *plug-ins*, detailed *responsibilities*, detailed *pre-* and *post-conditions*.

Even though UCMs are supported by a drawing tool (the UCM Navigator) [20][29], due to their informality, validation and verification techniques are not possible. Two formal methods are included at the *design* stage, LoTOS [21] and MSCs [18], to describe how system components communicate or interact in order to fulfill the analysis model. Details regarding data types, parameters and exchange messages are introduced in a *design* model (behavior and reference architecture are described with LoTOS) and successful and unsuccessful outcomes are shown in several MSCs (*scenario* models).

LoTOS specifications represent a system prototype by describing temporal relations with externally observable behaviors. Abstract data types are also included in this formal technique. Details about LoTOS, standardized by ISO, can be found in [21]. This FDT is supported by tools that offer ways of checking completeness and consistency. For instance, **LO**tos **LA**boratory (LOLA) is a set of tools developed by the Department of Telecommunication Engineering (ETSIT) of the University of Madrid [22] that includes: a step-by-step executor, a tool for obtaining the labeled transition system, and a tool for testing.

MSCs [18], standardized by ITU-T, describe interactions between system components. Each MSC represents exactly one scenario by focusing on the communication behavior of system components and their environment through message exchanges. We use MSCs to represent the results of the LoTOS validation and these scenarios are used as input to the implementation and testing stages. Recently, a

Lotos2MSC Converter is being developed [25] in order to generate MSCs directly from LoTOS traces. The first version is already available and is applied to our work.

As a case study, we iteratively and gradually specify and validate a simplified WmATM network environment. The system behavior increases with designer and user needs. Each development cycle brings more details regarding new functional requirements as well as new system components. At the beginning, mobility management functions are described (development cycle 1), followed by communication and handoff functions (development cycle 2 and 3). In the following sections, the first development cycle is presented in more details.

4. WmATM DESCRIPTION WITH USE CASE MAPS

This section presents the *requirements capture* and *analysis* stages of the proposed development approach depicted in Figure 3. These models are based on the description of WmATM signaling protocols presented in [5][6] as well as on our experience with wireless network standards [3][4]. By focusing on the functional requirements with the UCM notation, firstly, it is possible to describe the whole scenario of how the simplified WmATM network environment works. The system is decomposed in the following functions: mobility (authentication, registration and de-registration), communication (connection establishment and disconnection) and handoff functions. These functions are gradually described in terms of sequential actions with *unbound* UCMs (the requirements model) followed by more details about the system behavior and the addition of the reference architecture with *bound* UCMs (the analysis model). We present the first development cycle related to mobility management functions. Scenarios related to the other functions as well as exceptions (such as network failure, lack of network resources, database failure and so on) are left to the next development cycles.

4.1. Requirements Model: Unbound UCMs

The whole behavior of the system and, consequently, the relationship among the functions mentioned earlier are better understood by following the UCM flows shown in Figure 4. Based on the root map, users and designers can consider early decisions regarding the sequence in which these functions are performed. This map describes the system behavior that starts when a pre-condition is satisfied, for example, the user powers on a mobile station (filled circle labeled S). A *stub* (such as MM, HP and CM in the figure) identifies places where details are delayed to a sub-UCM, called *plug-in*. The *stub* notation is applied to our work not only to hide details, but also to decompose the system into small manageable units. In this paper, we focus on the MM Stub to show the development approach step-by-step. Communication management and handoff functions are not described due to space limitation.

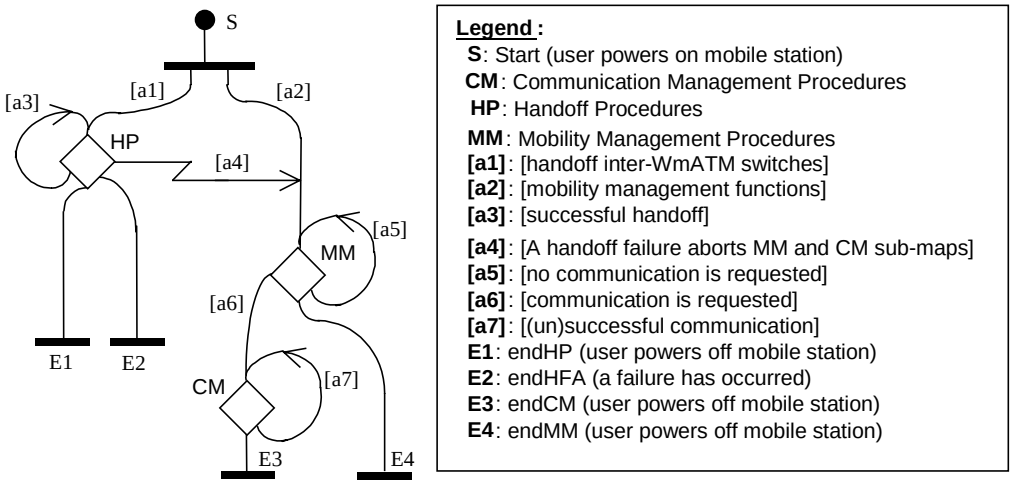

Figure 4. WmATM Network Root Map: *Unbound* UCMs

This scenario ends when one or two of the following triggering events occur: user powers off mobile station (bars labeled E1, E3 and E4) or a handoff failure occurs (bar labeled E2). These events are represented by post-conditions in the UCMs. A *route* is a path that links an initial cause to a final effect. For example, <S, [a2], MM, E4> represents a *route* for registration followed by an user powers off event. The zig-zag notation ([a5] path) exists in the current UCM notation to describe exception paths, however, we propose its use also to describe synchronous interactions between stubs such as HP and CM (HP replaces CM in case of handoff failure). Direction arrows help designers visualize the UCM flow as in the HP *stub* where an outgoing path returns to the same stub in order to trigger the *plug-in*. *And-forks* represent composite UCMs that split a path into parts (sub-paths) that proceed concurrently ([a1] and [a2] in the figure). *Or-joins* represent composite UCMs that can be concatenated in only one path (represented by [a3] joining [a1], [a5] joining [a2], and [a7] joining [a6]). There is no concurrency associated with *OR-joins*.

Figure 5 depicts the second level of the requirements model when mobility management procedures are decomposed into small units. These small units are represented by the Auth and Update *stubs* in the Location Registration *plug-in.* This *plug-in* is in turn bound to the MM *stub* in the root map. Alternative paths (called *OR-forks*) represent composite UCMs that can be split into two different paths (no level of concurrency is associated with them). For instance, a responsibility point (cross labeled cR in the figure) 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.

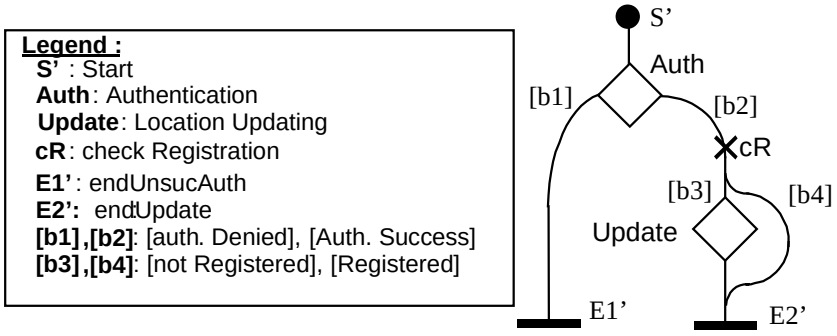

Figure 5. (a) Location Registration *Plug-in* for MM *Stub*

The main advantage of applying *unbound* UCMs in comparison to informal descriptions is the visual representation of the overall system behavior right from the early development stages. Under such circumstances, the system description becomes more readable and design decisions regarding the mapping of the reference architecture, exchanged messages and data types are easier to handle at the later stages.

4.2. Analysis Model: Bound UCMs

At the *analysis* stage, the previous *stubs* are detailed with r*esponsibility* points along the paths that identify actions, events, or operations on data items. Figure 6(a) details the Auth *stub*. First, the mobile station processes the user authentication and sends the authentication result to the network (sI responsibility). Then, the aAA responsibility performs the same authentication operation on the network side. The cAR responsibility generates the successful or unsuccessful outcomes (respectively, E2'' or E1'' end points). In case of denied authentication, the mobile user is notified (nAD responsibility). Otherwise, the network is notified (nN responsibility).

Figure 6(b) shows what happens inside the Update stub. For example, cL generates different outcomes according to whether the mobile user is roaming or not. uP and uTP responsibilities are

operations on database items. Sub-paths labeled [c1] and [c2] are concatenated after the network is notified (nN responsibility) about the successful operation.

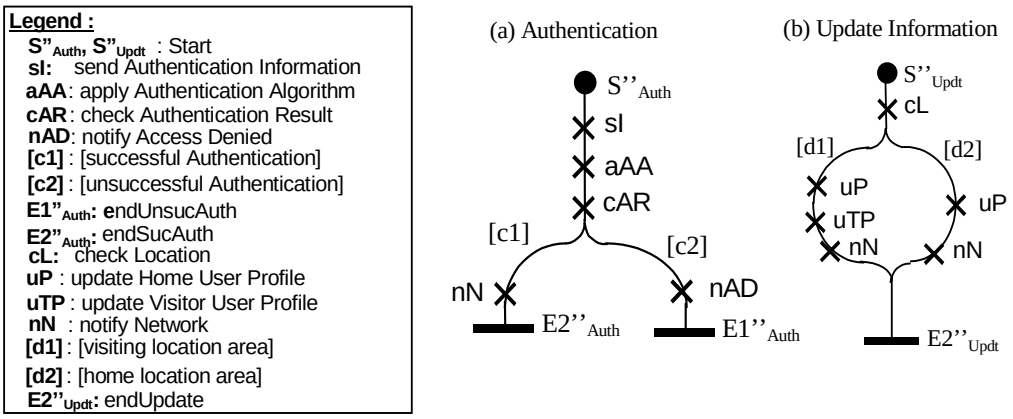

Figure 6. (a) Authentication and (b) Update Information *Plug-ins*

Besides describing detailed scenarios as causal paths with new plug-ins bound to the stubs at this stage, organizational structures of system *components* (represented by rectangular boxes as shown in Figure 7) are added. For instance, WmATM *components* involved in the authentication and update information functions, *Mobile Stations*, *WmATM switch* and *Home and Visitor Databases* are mapped to the *unbound UCMs* described at the requirements model.

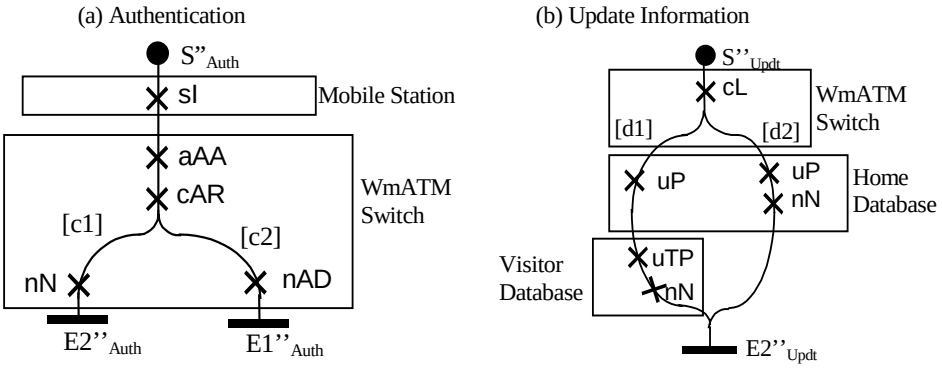

Figure 7. *Bound* UCMs: Authentication and Update Information *Plug-ins*

Related and Future Work. With the increasing popularity of the Unified Modeling Language (UML) for modeling systems using object-oriented concepts, UCMs are currently being investigated as another UML artifact to help the system development process. According to [2], UCMs can help bridge the gap between the use case model and the analysis and design models represented by behavioral diagrams (sequence, state charts, and activity diagrams) in the UML. We intend to apply UCMs as an alternative for the early stages of the function-oriented development process combined with strong formal methods such as LoTOS and MSCs. In short, our approach brings more powerful tools to tackle the verification problem (how can a designer solve a given problem systematically so that requirements are realized) in large systems. Object-oriented analysis and design that are the subject of UML are one step further and are considered as future work.

UCMs resembles Petri Nets at first sight. For example, their graphical notations look alike and both are based on causality events. However, many differences can be quickly perceived, such as semantics are not defined for UCMs. This notation is often applied to the early stages of the development process to give a global picture of the system. In addition, UCMs are light-weight, easy to learn, 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. Furthermore, they are more appropriate to the design stage and, therefore, should be compared to formal languages like LoTOS and Specification and Description Language (SDL). A mapping of UCMs to Petri Nets can be investigated as future work.

5. WMATM SPECIFICATION AND VALIDATION WITH LOTOS

At the design stage, a formal model is generated based on the *bound* UCMs described earlier. LoTOS has many advantages in specifying and validating complex and large systems. For example, different levels of abstraction can be used to describe functional behavior at different development stages, not to mention the LoTOS ability of process instantiation and parallel composition to specify the system reference architecture with the sequence of responsibilities defined previously at the requirements and analysis models. LoTOS tools like the LOLA environment are available to automatically support validation and verification methods. These methods allow the detection of design errors, inconsistencies and incompleteness at the time in which the LoTOS specification is being developed.

Since the behavior and structure models are iterative and incrementally generated at the *requirements* and *analysis* stages, the LoTOS specification becomes easier to develop. In addition, the gap between stages is also reduced by moving from *bound UCMs* (such as Figure 7) to LoTOS *processes* and *gates* shown in Figure 8. Even though the processes are derived from the UCM system components, design decisions related to how they communicate through gates are not always straightforward and depend on real interfaces and synchronization needs.

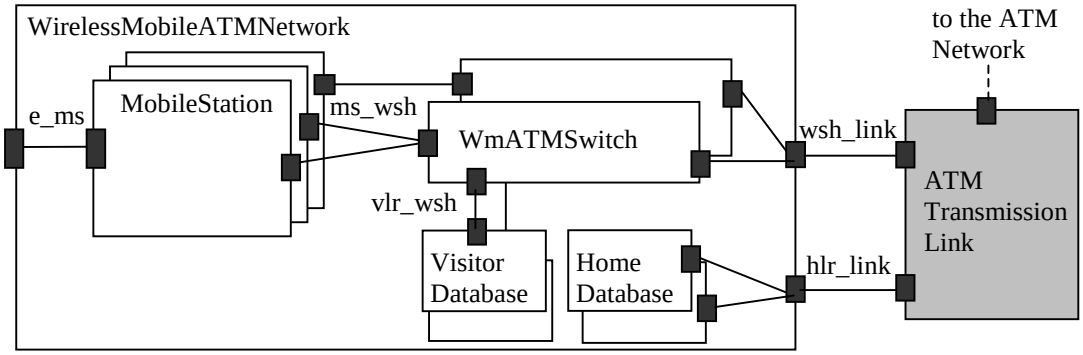

Figure 8. Graphical Representation of the LoTOS Specification Architecture

At the highest level of abstraction, the specification 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 for lack of space). *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 overcomes a LoTOS limitation of not allowing instances of the same process to communicate directly and 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 purpose).

Figure 9 depicts how these processes synchronize through the gates (||| represents the interleaving operator and |[gate list]| the selective parallel operator). The use of these LoTOS operators allows process synchronization and the ability to simulate and test the whole system behavior. Data types are designed to guarantee 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).

```
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 9. Highest Level of Abstraction of the LoTOS Specification

The behavior of each process is first generated on the basis of the sequence of UCM responsibilities (for instance, from Figure 4, Figure 5, and Figure 7 to Figure 10). After that, both informal descriptions and information flows presented in [5] and [6] are considered to add more detail to the specification such as data types and specific messages. During this stage, duplicate behavior and incomplete scenarios related to the signaling protocols are detected and corrected using the simulation and testing Lola tools. For example, in our specification the same procedure is used for connection establishment between two mobile users, as well as between mobile and fixed users, minimizing duplicated behaviors. Also, more unsuccessful scenarios are described, such as power off, handoff failure and disconnection (represented by the [> disable operator). These scenarios occur at any time after the user powers on or after the connection establishment. Figure 10 depicts part of the behavior of the *MobileStation* process when a mobile user powers on and authentication and update information plug-ins are triggered as shown in Figure 7.

```
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 10. Partial Behavior of the *MobileStation* Process

LOLA is a transformational and state exploration tool that supports execution and testing of LoTOS specification. To do this, LOLA provides a set of tools that help designers analyze the behavior of a system before the implementation stage. The following tools are applied to our specification: *simulation* 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. This expansion transformation tool is also used to generate a trace (one possible scenario). The next sub-section presents MSC scenarios that are automatically generated from such traces.

6. SCENARIOS WITH MESSAGE SEQUENCE CHARTS

MSC is the favorite notation to describe scenarios of current systems. For example, basic sequence diagrams are used at the early phases of the development of large systems. Wireless mobile standards documents describe protocols using information flows. As well, they are used to represent early behavior models in object-oriented approaches. Nevertheless, these diagrams are static and disjoint, and only one sequence of events can be observed in each of them. Due to these characteristics, validation and verification techniques are not possible and in [4], we propose their use as a complement of formal methods such as LoTOS and SDL [16]. Recently, High-Level MSCs include control structures that can combine several MSCs representing more than one scenario. However, such MSCs are not considered in our work. MSCs enable us to represent clearly the results of the LOLA validation activities. Successful and unsuccessful MSC scenarios can be more readable and attractive than LoTOS traces and they can be used for implementers to generate the protocols.

The generation of MSCs is done automatically with the Lotos2MSC converter tool. 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. The converter restricts the LoTOS capability of full-duplex communication through gates (no direction is associated to the execution of LoTOS actions among processes) by demanding that gates represent directions and components. Since the LoTOS generic concept of action defines messages and parameters implicitly in terms of abstract data types, as mentioned above, the tool can only recognize messages and parameters when they are described in the LoTOS action. A direct mapping of LoTOS to MSC concepts is not done due to the LoTOS synchronization of many simultaneous actions between processes in contrast to the MSC exchange of asynchronous messages between components. This 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 [25].

Figure 11 illustrates two disjoint scenarios that represent specific behaviors of the system in conformance not only with the LoTOS specification (Figure 10) but also with the *bound* UCMs (Figure 7(a)). For sake of clarity, we represent *WmATM switch process* as current WmATM switch.

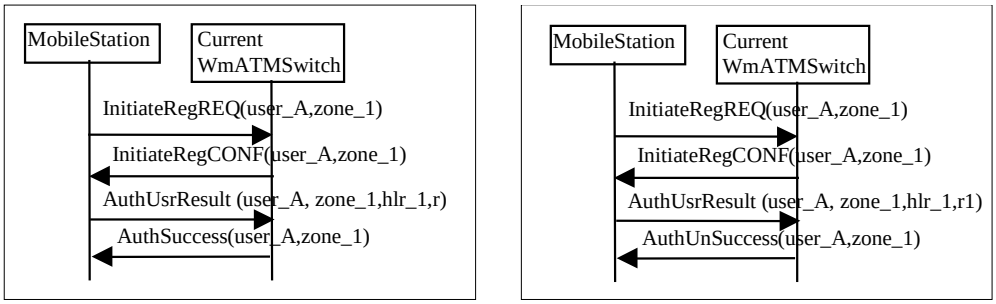

Figure 11. (a) Successful Authentication (b) Unsuccessful Authentication

By comparing the MSCs depicted above with some of the protocols presented in the literature, certain design decisions make the former clearer and more complete. For instance, Figure 11 shows that 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. Also, these MSCs show more details about parameters that make the implementation easier.

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) and the SPIN tool 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 more formal verification techniques such as the ones supported by the SPIN model checker [15] to these diagrams is presented in [14]. This process is also a research interest for the SDL and LoTOS communities. We believe that these solutions 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.

7. CONCLUSION

Current development approaches for wireless mobile ATM (WmATM) networks describe all specific information related to the signaling protocols at once. However, a good approach should be iterative and gradually add details during different development stages and life cycles, while checking for ambiguities, inconsistencies, and undesirable interactions. In this context, the main contribution of this work is to introduce the combination of different techniques at appropriate stages of the system development process. As a case study, mobility, communication and handoff procedures for WmATM networks are developed using the proposed approach.

In short, at the *requirements capture* stage, *unbound* Use Case Maps (UCMs) are used as first scenarios by focusing on the causality relationship between responsibilities, without any concern about components. At the *analysis* stage, system components and more behavior details are added to these maps, generating *bound* UCMs. This notation provides a better human understanding of the system and it helps network designers to produce descriptions of the requirements more legibly as well as it facilitates the system development and maintenance. At the *design* stage, a formal specification is developed with LoTOS. This adds rigor to the approach and many possible behaviors are described concurrently with details such as data types, parameters, and specific events. A set of LoTOS tools assures 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.

The proposed approach improves the existing current development process for wireless mobile ATM networks in different ways as follows: by achieving a better model, by helping human understanding and by reaching technical quality with the formal specification for future maintenance. Using our approach, inconsistencies of parameters and incompleteness of the informal description are detected and corrected. In addition, the UCM technique can reduce the gap between early and later stages. Our results also intend to show how the combination of informal and formal techniques at the appropriate development stages can really aid designers to generate good systems, ready to be reused and easy to maintain, and add new features.

The motivation for choosing WmATM networks resides in the fact that they are still under development and also on the amount of information available about the signaling protocol alternatives. This makes the production of the design prototype feasible. Our approach can also be applied to other wireless mobile communication systems. 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) [27], and UCMs to the description of Wireless Intelligent Network standards [26] as presented in [3]. Currently, the combination of these techniques is one of the main research topics of our group.

8. ACKNOWLEDGMENTS

I would like to thank the UofO's LoTOS Group for their support, especially Luigi Logrippo, Jacques Sincennes, Masahide Nakamura, Daniel Amyot and Leila Charfi. Many thanks to the anonymous reviewers for their judicious comments. Finally, I acknowledge CAPES for its financial support.

9. REFERENCES

[1] Acampora, A., "Wireless ATM: A Perspective on Issues and Prospects", IEEE Personal Communications, Vol. 3, No. 4, pp. 8-17, August 1996.

[2] Amyot, Daniel, Use Case Maps and UML for Complex Software-Driven Systems, Technical Report, August 1999. www.usecasemaps.org

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

[4] 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 1999.

[5] Akyol, B. A., "Signaling Alternatives in a Wireless ATM Network", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 1, pp. 35-49, January 1997.

[6] Akyol, B. A. and Cox, D. C., "Rerouting for Handoff in a Wireless ATM Network", *IEEE Personal Communications*, Vol. 3, No. 5, pp. 26-33, October 1996.

[7] Ayanoglu E. et al., "Wireless ATM: Limits, Challenges, and Proposals", *IEEE Personal Communications*, Vol. 3, No. 4, pp. 18-36, August 1996.

[8] Buhr, R.J.A. and Casselman, R.S., *Use Case Maps for Object-Oriented Systems*, Prentice-Hall, USA, 1995. http://www.UseCaseMaps.org/UseCaseMaps/pub/UCM_book95.pdf

[9] Buhr, R.J.A. (1998) "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. http://www.UseCaseMaps.org/UseCaseMaps/pub/tse98final.pdf

[10] Cheng, Fang-Chen, Holtzman, Jack M., "Wireless Intelligent ATM Network and Protocol Design for Future Personal communication Systems", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, Sept. 1997.

[11] Corriveau J.,-P., Nel, D. N., "Introduction to Object-Oriented Software Engineering", Version 1.0, Course Notes, School of Computer Science, Carleton University, 1997.

[12] Courtiat, J.,-P., Dembinski, P., Holzmann, G., J., Logrippo, L., Rudin, H., Zave, P., "Formal Methods after 15 years: Status and trends", *Computer Networks and ISDN Systems 28*, pp. 1845-1855, 1996.

[13] Faci, M., Logrippo, L., Stepien, B., "Structural Models for Specifying Telephone Systems", *Computer Networks and ISDN Systems 29*, pp. 501-528, 1997.

[14] Holzmann, Gerard, J., Formal Methods for Early Fault Detection, Proc. Of Formal Techniques for Real-Time and Fault Tolerant Systems, LNCS Vol. 1135, pp. 40-54, 1996.

[15] Holzmann, Gerard, J., Design and Validation of Computer Protocols, Prentice Hall Software Series, 1991.

[16] ITU-T, *Recommendation Z. 100: Specification and Description Language (SDL)*. Geneva, 1994.

[17] ITU-T, *Recommendation Q.2931*: ATM Network Signaling Specification, 1995.

[18] ITU-T, *Recommendation Z. 120: Message Sequence Chart (MSC)*. Geneva, 1996.

[19] Jacobson, Ivar et. al, Object-Oriented Software Engineering (A Use Case Driven Approach), ACM Press, Addison-Wesley, 1992.

[20] 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.

[21] 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.

[22] Pávon, S., Larrabeiti, D., and Rabay, G., "LOLA – User Manual", version 3.6. DIT, Universidad Politécnica de Madrid, Spain, LOLA/N5/V10, February 1995.

[23] Prycker, Martin de, Asynchronous Transfer Mode: solution for broadband ISDN, Prentice Hall International (UK) Limited, 1995.

[24] Raychaudhuri, D., "Wireless ATM Networks: Architecture, System Design and Prototyping", *IEEE Personal Communications*, Vol. 3, No. 4, pp. 42-49, August 1996.

[25] Stepien, Bernard, Lotos2MSC Converter – User's Manual, University of Ottawa LoTOS Group, January 2000.

[26] TIA/EIA (1998) *Wireless Intelligent Networks (WIN).* Additions and modifications to ANSI-41 (Phase 1). TR-45.2.2.4, PN-3661 Ballot Version, May 1998.

[27] Tuok, R., *Modelling and Derivation of Scenarios for a Mobile Telephony System in LOTOS*, M.Sc. Thesis, Computer Science Program, SITE, University of Ottawa, Canada, 1996.

[28] Umehira, M., et al., "ATM Wireless Access for Mobile Multimedia: Concept and Architecture", *IEEE Personal Communications*, Vol. 3, No. 5, pp. 39-48, October 1996.

[29] *Use Case Maps Web Page*: http://www.UseCaseMaps.org , since 1999.
[30] Varshney, U., "Supporting Mobility with Wireless ATM", *Computer*, Vol. 30, No. 1, pp. 131-137, Jan. 1997.
[31] Veeraraghavan, M. et al., "Mobility and Connection Management in a Wireless ATM LAN", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 1, pp.50-68, January 1997.