

Florida International University

School of Computing and Information Science



Modeling and Realization of Mobile Multimedia Communications (MMMMC)

Validation Document

CEN 5064 – Software Design (Section U01)

November 3, 2010

Prepared for: Professor Peter Clarke

Prepared by Group 1:

Xabriel J. Collazo-Mojica

Karl Morris

Abstract

Communications are vital across all domains of our lives. With the advent of various computer-based applications, multimedia communications have proliferated. This domain of communications includes n-way voice calls, n-way chat messaging, n-way video calls, file sharing and others. Typically, not all these multimedia communication features are provided by a specific software application, and if they are all provided, the vendor does not necessarily support all the popular computer platforms. In the area of mobile devices, we have a similar landscape: disparate native features or applications are available for subsets of the aforementioned multimedia communications.

In our Software Design Document [SSD], we delineated the proposed design for our solution, and also presented the details of the underlying platform such as the Metamodel and the UML Profile. In this document, we focus on the final validation of our proposed solution. We present both Software Inspection analysis as well as Software Testing analysis. In the case of Software Inspection, we present validations in regards to our models syntaxes, semantics, and aesthetics. In the case of the dynamic Software Testing, we present system test cases based on stimulus-response methodology. We argue that our design is complete and coherent, although a few test cases do fail.

Table of Contents

Abstract	2
Introduction	6
Purpose of the system	6
Scope of the system	6
Limitations of the current system	6
Analysis and design methodology	7
Software Process Model	7
Modeling Language	8
Analysys Model	9
Definitions, acronyms, and abbreviations	9
Overview of document	10
Project Organization	12
Hardware and software requirements	13
Work Breakdown	14
Overview	15
Functional Requirements	15
Nonfunctional Requirements	16
Use case model description	18
Validation of the use case model	20

Test cases	20
Use case validation	33
Structured Walkthrough	34
Requirement Analysis	35
User interfaces	35
Validation of the Analysis model	35
Structured Walkthrough	36
Overview	38
Metamodel.....	40
UML Profile for the Architecture	41
Generative Architecture	41
Validation of the system model.....	42
Structured Walkthrough	43
Overview	45
Object Interaction	47
State Machines for Main control objects	47
Detailed Class Design	48
Validation	49
Structured Walkthrough	50
Implementation	52

Description of the PSM	52
Validation	53
Glossary	65
References	66
Appendix	67
Appendix A – Complete use case diagram, use case diagram for the use cases to be implemented.....	67
Appendix B – Use Cases to be implemented	69
Appendix C - Object diagrams for the analysis model.....	92
Appendix D – Sequence Diagrams for the analysis model	101
Appendix E – User interfaces	106
Appendix F – Detailed Class Diagrams	108
Appendix G – Class interfaces.....	109
Appendix H – Project Schedule	111
Appendix I – Diary of meeting and tasks	112

Introduction

Communications are vital across all domains of our lives. With the advent of various computer-based applications, multimedia communications have proliferated. This domain of communications includes n-way voice calls, n-way chat messaging, n-way video calls, file sharing and others. Typically, not all these multimedia communication features are provided by a specific software application, and if they are all provided, the vendor does not necessarily support all the popular computer platforms. In the area of mobile devices, we have a similar landscape: disparate native features or applications are available for subsets of the aforementioned multimedia communications.

Purpose of the system

The proposed system's purpose is to provide a way for communication experts to abstract away the details of mobile communications needs of end-users. That is, the system will provide a means for specifying communication schemas in terms of mobile device capabilities, and when these schemas are deployed to mobile devices, end-users will only have to specify with whom they want to communicate and not how they want to communicate.

Scope of the system

The proposed system will implement a modeling front-end through which communications experts will be able to realize mobile communication models. The system will include a modeling language in the domain of mobile device communications. However, it will not include a modeling framework, it will reuse existing software for this purpose. The proposed system will also implement application modules for each of the supported mobile devices.

Limitations of the current system

Currently, from a modeling perspective, an implementation of a Communication Modeling Framework, namely the Communication Virtual Machine (CVM), is in place. This framework permits its users to create communications models, which rely on fully-fledged computers, to realize computer-to-computer communications. However, this system neither permits its users

to model computer-to-mobile nor mobile-to-mobile communications. Additionally, this system's implementation is not suitable for the limited resources of a mobile device.

Similarly, from a mobile device perspective, the only way to realize communications is for all the participants of a communication to know which specific features or applications each other participant has in their respective mobile devices. For example, if a participant of a communication wants to send a file from a his/hers mobile device to another participant's mobile device, he/she has to know whether the other participant's device has a native 'receive file' feature or whether it has an installed application; in both cases the originator also needs to know if that feature is compatible to the originator's 'send file' feature or application. Clearly, this current system is not scalable because of the ever-growing market share of mobile devices. This tendency continues to add variability of features, and the availability of applications for each mobile device is diverse.

Analysis and design methodology

In this section we elaborate on the chosen software process model, and the types of models that we are using to represent the proposed system. Additionally, we present our analysis methodology for the validation of our models.

Software Process Model

The Software Process Model chosen for this project is the Unified Software Development Process [USDP]. Figure 1 shows a representation of this approach, but including only the parts significant to the design process. We chose USDP for three main reasons. First, the whole process is driven by the use cases. This allows our project to have strong traceability. Note in the figure that the use cases are used not only to produce a design, but all the steps that this process entails.

Second, this process presents each work product as a model, and typically expects the use of a modeling language, such as UML. This ties well with our workflow, as we have decided to use UML for most of the phases of our project.

Third, USDP allows us to break down a complex project into phases. Each phase must go through four states: inception, elaboration, transition, and construction. Each phase must also pass thru all the models presented in Figure 1. From the beginning of our project we knew that because of time constraints we would not be able to implement the full project specified in the SRAD. Therefore, the phases in USDP model our approach of only implementing part of the use cases, and specifically, on this first phase, we will implement the core use cases, as defined on USDP.

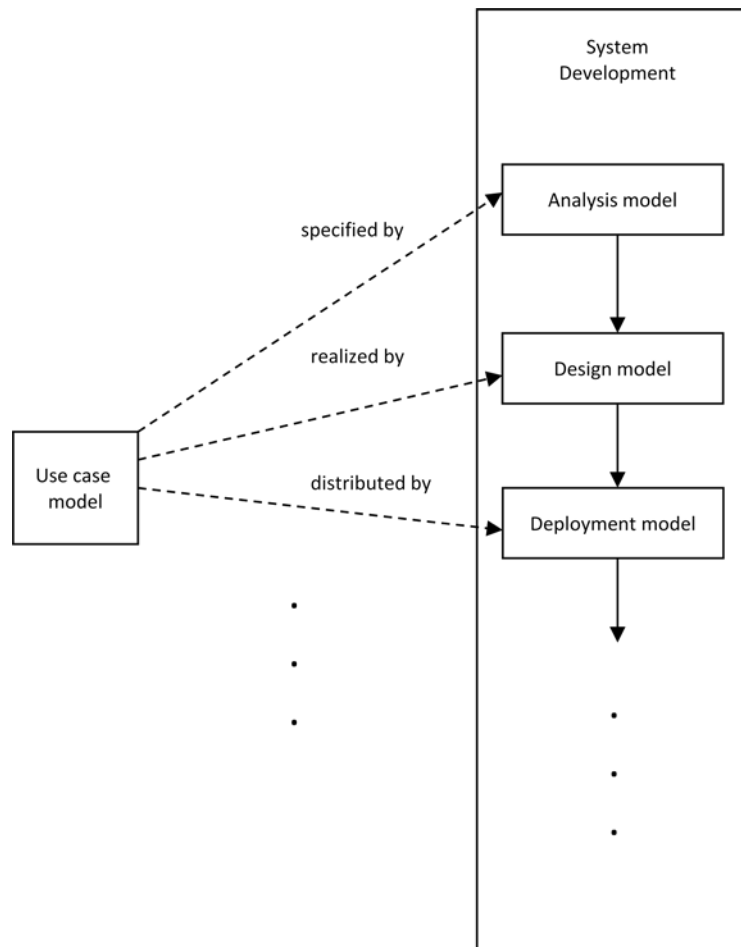


Figure 1

Modeling Language

The chosen modeling language is the Unified Modeling Language (UML). Specifically, we are using use case diagrams, class diagrams, UML Profiles, sequence diagrams, and state machines,

among others. The UML Language is quite comprehensive in its analysis and design components, and it complements our chosen process model. Clearly, using a standardized language for discussion of our design ideas is much more easier than to come up with our own.

Analysys Model

Our analysis model includes two main broad approaches. First, Software Inspection, which entails the static analysis. In these inspections we focus on three correctness aspects: the syntactical, the semantical, and the aesthetical. In the syntactical correctness, we try to answer questions regarding the well-formedness of a model given the model's rules. For example, if we have a use case diagram, is that particular diagram using the correct associations and figures? For the semantical correctness, we focus on the information we want to present with the model. For example, given a use case scenario and an object diagram, can we say that the diagram follows from the scenario? Are we representing all the significant objects? In the aesthetical correctness checks the focus is on more subjective measures such as readability of the model. For example, is the diagram too big to comprehend easily? Should we partition it in two? For all these checks we present tables with the criteria being addresses, and a pass or fail measure. If the model fails a certain criteria, we present our understanding of why it is so, and some workaround strategy.

The second analysis approach is that of the software testing. In this dynamic checks for the implementation of our models, we prepare a set of inputs and hypothesize on the output according to what we modeled. If the output is indeed equal to the expected one, then the test passes. Otherwise, we present our understanding of why the test failed, and some workaround strategy.

Definitions, acronyms, and abbreviations

CVM – Communications Virtual Machine. Refers to an already implemented system for modeling and realizing computer-to-computer communication schemas.

Mobile device – Throughout this document we use this term to represent the group of mobile devices with computer-like capabilities. These devices typically have limited computing power

in terms of available CPU speed and RAM size. Present-day examples include Apple's iPhone, Apple's iPod Touch, and Google's Android-based telephones.

Fully-fledged computer – Throughout this document we use this term to represent the group of personal desktop or laptop computers. These devices typically have abundant computing power in terms of available CPU speed and RAM size. Present-day examples include Apple's iMac, Apple's MacBook series, and Dell's Inspiron series.

Overview of document

In the rest of this document, we present the validation analysis of the proposed system. In specific, on the Proposed System Software Architecture section, we first detail the major subsystems and give an overview of each subsystem. We also give reasons for the selected architecture and identify the architectural patterns used. With this, we run a validation steps on each one of the artifacts.

We then continue on to the metamodel, where we present our understanding of the underlying mobile platforms with a fully annotated class diagram. We also present a UML Profile for the architecture, a generative architecture diagram, which shows the transformations expected from the architecture to the mobile platform. We finish this section with a subsystem decomposition where we use UML component diagrams to model the different component and frameworks that constitute our proposed system. Then we apply the validation steps to each one of these artifacts.

We follow with an object design, where we first present a minimal class diagram for the subsystems we will implement. We include a brief description of each class and we mention the design patterns that we deemed appropriate for the solution, while we also include our main control objects for the two major subsystems. To end this section, we present a detailed class design where we explain the purpose of each class and reference the appropriate class diagrams, and we present OCL constraints for the static model of the main controller in the mobile platform. After this, we validate the artifacts each one of these artifacts.

Finally, we present a glossary of terms as an aid for the reader, to finish up with an appendix including details for the use case diagrams, the use cases with nonfunctional requirements that we will implement, detailed class diagrams showing attribute and methods for each class, and a diary of meeting and tasks completed as of today.

Project Plan

In this section we present information regarding the organization and requirements and responsibilities for our group to be able to complete the proposed system on time. We also present our hardware and software requirements, that is, the artifacts needed for our solution to be useful. Finally, we also make reference to the GANTT Chart produced for visualizing the timeline of the project.

Project Organization

The responsibilities associated with the Mobile Communication Model Deployment project are rotated in order to provide exposure to all facets of the development process to all members of the project team. Said responsibilities are outlined below:

Project Leader – The de facto Project Manager, is responsible for ensuring that all deadlines are met, and that all deliverables are at an acceptable level of completeness with full adherence to documented requirements.

Scribe – Responsible for the documenting of all project team interactions, decisions, deadlines, responsibilities and meeting minutes both scheduled and ad-hoc.

Modeling Environment Developer – This person is responsible for the specification and development of all use cases related to Model design up to the point of deployment.

Model Instantiation Developer – This person is responsible for the specification and development of all use cases related to model instantiation on the mobile platform including the development of the relevant components of the CVM model.

Deployment Service Developer – This developer is responsible for managing the transport of deployed models from the modeling platform, to the mobile platform via a Deployment Service, which will also be developed by this individual.

For the first rotation, we had the following assignments:

- Karl – Project Leader, Model Instantiation Developer

- Xabriel – Modeling Environment Developer, Deployment Service Developer
- Zhimin – Scribe

For the second rotation, which persisted throughout the System Design phase, we have the following assignments:

- Karl – Model Instantiation Developer, Scribe, Deployment Service Developer
- Xabriel – Project Leader, Modeling Environment Developer

For the third rotation, which will last throughout the end of the project, we have the following assignments:

- Karl – Project Leader, Model Instantiation Developer
- Xabriel – Modeling Environment Developer, Scribe, Deployment Service Developer

Hardware and software requirements

Hardware requirements:

- PC with a minimum of a 2 GHz Processor speed, 512 MBs of RAM, and 20 GB of storage for the development, testing and deployment of the design environment.
- Android mobile device running Android version 2.0 capable of connecting to a 3G mobile network.

Software requirements:

- Rational Rose 7.0.0
- Eclipse Helios Release
- Eclipse Modeling Framework 2.6.0
- Graphical Editing Framework 2.3.0
- Android emulator 0.9.7
- Android SDK Version 2.2

Platform Specific Model requirements:

- An Android device, such as the Nexus One phone.
- Android 2.2 with Google extensions

- N-way chat service

Work Breakdown

For this project, we have developed a GANTT chart, available in Appendix A under Figure 18.

This GANTT chart states the 5 broad milestones that we need to accomplish: The Project Definition, the Requirement Elicitation process, the Requirement Analysis process, the Detailed Software Design, and the Prototype Implementation. On Figure 18, we can see the timeline that we expect to follow for the project. Note that, as of now, we are done with the Project Definition phase, and we are just to finish the Requirement Elicitation and Analysis phase. Each one of our identified milestones are also the deliverables we need to present to our customer.

Requirements elicitation and analysis

Overview

The proposed system's purpose is to provide a way for communication experts to abstract away the details of mobile communications needs of end-users. That is, the system will provide a means for communication experts for specifying communication schemas in terms of mobile device capabilities, and when these schemas are deployed to mobile devices, end-users will only have to specify with whom they want to communicate and not how they want to communicate.

Functional Requirements

The system shall provide means for the following high-level requirements. Each requirement is further explained on the use case scenarios referenced, and the use case diagram can be found on Appendix A. Note that here we mention all related use cases, which includes uses cases which will not be implemented. For a complete list of the elucidated use cases refer to [SRAD].

1. The system shall allow domain experts to design models for the case of mobile-to-mobile communications.

Related use cases: MMMC-2, MMMC-3, MMMC-9, MMMC-11

2. The system shall allow domain experts to design models for the case of mobile-to-computer communications.

Related use cases: MMMC-2, MMMC-3, MMMC-9, MMMC-11

3. The system shall be able to deploy mobile-to-mobile communication models to the respective mobile participants

Related use cases: MMC-12, MMMC-13

4. The system shall be able to deploy mobile-to-computer communication models to the respective mobile and computer participants.

Related use cases: MMC-12, MMMC-13

5. The system shall allow an already deployed communication model to be edited in the mobile device.

Related use cases: MMMC-3, MMMC-7

6. The system shall allow an already deployed communication model to be instantiated from a mobile device.

Related use cases: MMMC-1, MMMC-4, MMMC-5, MMMC-6, MMMC-8, MMMC-10, MMMC-14

Nonfunctional Requirements

What follows is an overview of the functional and nonfunctional requirements of our proposed system that will be implemented as part of this phase of the project. After each requirement, we present any applicable constraints.

1. The system shall allow domain experts to design models for the case of mobile-to-mobile communications.

Related use cases that will be implemented: MMMC-2, MMMC-3, MMMC-9, MMMC-11.

Synthesized constraints for this requirement:

- Usability: The domain expert should be able to validate a model with only one click. The interface must be familiar to users of the available computer-to-computer CVM implementation.

- Reliability: This use case should present neither false positives nor false negatives. Functionality that relies on the mobile platform's native address book will be as reliable as the platform's implementation.
- Performance: Executing this requirement should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: This requirement will only support the Eclipse platform for designing models.

2. The system shall be able to deploy mobile-to-mobile communication models to the respective mobile participants

Related use cases that will be implemented: MMMC-12, MMMC-13.

Synthesized constraints for this requirement:

- Performance: Receipt of notification of availability of the model on a mobile device should be completed within 30 seconds of deployment. This requirement must download the model within 20 seconds of availability on the deployment server.
- Usability: This requirement should utilize the default notification system of the Android platform as to present the user with a familiar interface.

3. The system shall allow an already deployed communication model to be edited in the mobile device.

Related use cases that will be implemented: MMMC-3.

Synthesized constraints for this requirement:

- Usability: This requirement should be started automatically.

- Reliability: This requirement should be as reliable as the native address book of the mobile device.
 - Performance: Executing this requirement should take no more than 60 seconds, discounting the time waiting for user input.
 - Implementation: Only the Android platform will be support for deployment.
4. The system shall allow an already deployed communication model to be instantiated from a mobile device.

Related use cases that will be implemented: MMMC-1, MMMC-4, MMMC-8.

Synthesized constraints for this requirement:

- Usability: Initiating this use case should take no more than 4 clicks on the mobile interface. Once the user chooses the communication service to be realized, all other steps to realize the communication should be automatic.
- Reliability: If all validation steps are passed, this use case should function at least 90% of the time. This requirement should be as reliable as the underlying Android platform communication services.
- Performance: Executing this requirement should take no more than 60 seconds, discounting the time waiting for user input.
- Implementation: Only one mobile device platform will be supported, this being Android 2.2.

Use case model description

On Figure 2, we present the four main components of our system in terms of a use case diagram. As can be noted, we propose to have a way for users the Design Mobile Communication Models. This entails having a Visual Language and an integrated development

environment. We also include a way for users to deploy these models. This entails having some way for them to specify which mobile device they are targeting and, from the point of view of the device, to have a way to receive the device. Similarly, we present a way to be able to edit those models in the mobile device, as being able to instantiate the models. For this we include the Edit and Instantiate packages of functionality. This is only a bird's eye view of the system, and more details of the breakdown of use cases can be seen on Appendix A, specifically on Figure 16.

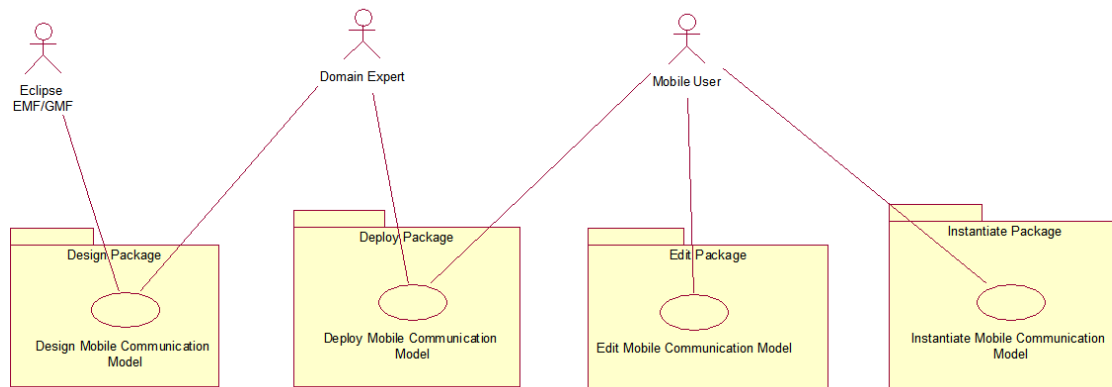


Figure 2

Because of the time constraints of the semester, we will not be able to implement all the functionality presented on these use cases. For this, we have chosen the most important functionality blocks, as presented on Figure 17. These are the use cases that we have designed and developed in the course's timeframe.

Validation of the use case model

Test cases

Test Case ID	MMMC-SystemTest-1
Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System accepted the model as valid.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-2
Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 services and 3 participants.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System accepted the model as valid.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-3
Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed an invalid MCM with 2 services and 3 participants, but one participant is lacking a mobile device.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System rejects the model and explains the errors found.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-4
Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 1 service and 2 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-5
---------------------	--------------------------

Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 2 services and 3 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-6
Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 3 services and 3 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-7
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-8
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 services and 3 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-9
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 3 service and 3 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-10
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants. There is connectivity to the internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System deploys the model to the server.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-11
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 service and 3 participants. There is connectivity to the internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System deploys the model to the server.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-12
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants. There is no connectivity to the internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System is unable to deploy the model.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-13
---------------------	---------------------------

Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 1 service and 2 participants has been deployed and is waiting on the server. There is connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device automatically polls the server and eventually gets the model.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-14
Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 2 services and 2 participants has been deployed and is waiting on the server. There is connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device automatically polls the server and eventually gets the model.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-15
Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 1 service and 2 participants has been deployed and is waiting on the server.

	There is no connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device is unable to fetch the MCM.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-16
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model.
Stimulus	The user selects Run from the presented menu.
Expected Response	System will load the mobile communication model and launch the required services.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-17
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model. Mobile model no longer present on device.
Stimulus	The user selects Run from the presented menu.
Expected Response	The system informs the user that the model cannot be found.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-18
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model. Model is malformed.
Stimulus	The user selects Run from the presented menu.
Expected Response	System informs user that the model is invalid.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-19
Purpose	To validate Use Case MMMC-4 (Do N-way chat) in the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device has chat capabilities.
Stimulus	N-way chat service found in model.
Expected Response	N-way chat service launched
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-20
Purpose	To validate Use Case MMMC-4 (Do N-way chat) in the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device contains multiple chat capabilities.
Stimulus	N-way chat service found in model.
Expected Response	User is presented with option to select application.
Actual Response	Undefined.
Result	

Test Case ID	MMMC-SystemTest-21
---------------------	---------------------------

Purpose	To validate Use Case MMMC-4 (Do N-way chat) in the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device does not contain chat capabilities.
Stimulus	N-way chat service found in model.
Expected Response	System responds with message indicating no appropriate capability present
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-22
Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request. The mobile device has call capabilities.
Stimulus	N-way call service found in model
Expected Response	N-way chat service launched using native voice call infrastructure.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-23
---------------------	---------------------------

Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request for multiple participants. The mobile device has call capabilities.
Stimulus	N-way call service found in model
Expected Response	N-way chat service launched using native voice call infrastructure for each participant and allow user to merge calls.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-24
Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request. The mobile device does not have call capabilities.
Stimulus	N-way call service found in model
Expected Response	System responds with message indicating no appropriate capability present
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-25
---------------------	---------------------------

Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with some participants empty.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays local contact list to populate missing participant fields.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-26
Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with all participant fields missing.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays local contact list to populate all participant fields.
Actual Response	
Result	

Test Case ID	MMMC-SystemTest-27
---------------------	---------------------------

Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with all participant fields missing. Local contact list is unavailable.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays message indicating unavailability of contact list.
Actual Response	
Result	

This use case was not implemented.

Use case validation

Criteria	Pass/Fail
Permissible UML objects used	PASS
Notation for diagram elements	PASS
Boundary present and correct	FAIL
Correct naming of Actors	PASS
Actors are true representation of problem space	PASS
Representation of each Use Case is clear	PASS
Alternative flows present and correct	PASS
Use cases ranked and prioritized	PASS
Minimal number of Actors	PASS
Actors interact with more than one use case	PASS
Use case document is of reasonable size	PASS
Diagram(s) sufficiently describe the system	PASS

We were not able to add a boundary for the use case diagram as it is not supported by the Rational Rose 7.0 software.

Structured Walkthrough

Criteria for the Use Case Diagram	Pass/Fail
There is a use case in the model for every use case.	PASS
Every actor interacts with the specified use case.	PASS
Every Use Case extends the required use case in the model.	PASS
Every Use Case includes the required use case in the model.	PASS
Every Use Case name in the diagram is relevant to the use cases.	PASS

Requirement Analysis

User interfaces

In this section we present screenshots of the system's main functionality. Note that this functionality is captured with the first level use cases. All the referenced mockups are to be found in Appendix E.

Figure 27 presents an actual screenshot of the design environment for the mobile communication models. Note that the Domain Expert will be able to draw a communication using a specialized Visual Language. This language will be an extended version of an already available communication language. Once the domain expert is satisfied with the model, he/she would proceed to deploy the model, where a wizard will be presented as to send the model to a mobile device.

A Mobile User will be notified if a Mobile Communication Model has been deployed to his/her mobile device. A notification similar to that of Figure 28 will appear. If the user wants to verify the authenticity of the model he will be allowed to do so. Similarly, if the user chooses to Run the model, an Instantiation Application will be loaded. The user can also choose to save the model for later usage.

Another important aspect of the system is when an incomplete model is deployed to the device. In this case, the user will be presented with a view similar to the one shown in Figure 29. The mobile user will then be able to specify the participants in the communication model.

Validation of the Analysis model

Criteria for the Sequence diagrams	Pass/Fail
There is a sequence diagram for every use case.	PASS
Every noun maps to an object in the sequence	PASS
Every verb maps to an interaction in the sequence diagram.	PASS

Every path was followed.	PASS

Criteria for the Object diagrams	Pass/Fail
Objects and links correlate	PASS
One object per rectangle	PASS
No operations shown	PASS
No multiplicities shown	PASS
Object diagrams explain links	PASS
Objects related to other diagrams that require clarification	FAIL
Not too many object diagrams	PASS

Structured Walkthrough

In this section we do test our requirement analysis against the test cases to measure if we believe we will be able to comply with them with our choice. See the test cases defined in the Test cases section of the Use Case analysis chapter.

Test Case ID	Pass/Fail
MMMC-SystemTest-1	PASS
MMMC-SystemTest-2	PASS
MMMC-SystemTest-3	PASS
MMMC-SystemTest-4	PASS
MMMC-SystemTest-5	PASS
MMMC-SystemTest-6	PASS
MMMC-SystemTest-7	PASS
MMMC-SystemTest-8	PASS

MMMC-SystemTest-9	PASS
MMMC-SystemTest-10	PASS
MMMC-SystemTest-11	PASS
MMMC-SystemTest-12	PASS
MMMC-SystemTest-13	PASS
MMMC-SystemTest-14	PASS
MMMC-SystemTest-15	PASS
MMMC-SystemTest-16	PASS
MMMC-SystemTest-17	PASS
MMMC-SystemTest-18	PASS
MMMC-SystemTest-19	PASS
MMMC-SystemTest-20	PASS
MMMC-SystemTest-21	PASS
MMMC-SystemTest-22	PASS
MMMC-SystemTest-23	PASS
MMMC-SystemTest-24	PASS
MMMC-SystemTest-25	PASS
MMMC-SystemTest-26	PASS
MMMC-SystemTest-27	PASS

Proposed System Software Architecture

In this section we elaborate on the proposed system software architecture. First, we present an overview of the system with a package diagram of the major subsystems and also give a brief description of each one of them. We then present the metamodel for the underlying mobile platform with a fully annotated class diagram. A UML profile for the mobile platform architecture is also presented, as well as a generative architecture, which shows the transformations expected from the metamodel to the actual Android platform. Finally, we present the subsystem decomposition, describing each of the major subsystems in more detail.

Overview

Figure 3 presents the major subsystems in our architecture. The architecture is bifurcated into two major components with a transient deployment component shared between them. The first component is responsible for communication modeling and is comprised of the Communication Modeling, Schema Transformation Environment, Model Repository (which uses the repository pattern because of the shared nature of the models) and the UCI-SE Interface subsystems. Its purpose is to allow the domain expert to model a communication workflow for deployment to a mobile device. Note that we use a Repository Pattern, as various subsystems utilize the Communication Model.

The second major component is the Instantiation Application, which is responsible for the execution of a communication model. It is comprised of the Synthesis Engine and NCB subsystems, which will execute and hand off communication to the various frameworks resident on a device as outlined in the communication model.

The deployment component straddles both major subsystems and addresses the issue of lack of addressability of mobile devices in the field. It employs a Client-Server Pattern and will accept communication models deployed by the communication modeling component, and deliver it to the instantiation component on a mobile device upon request by said device.

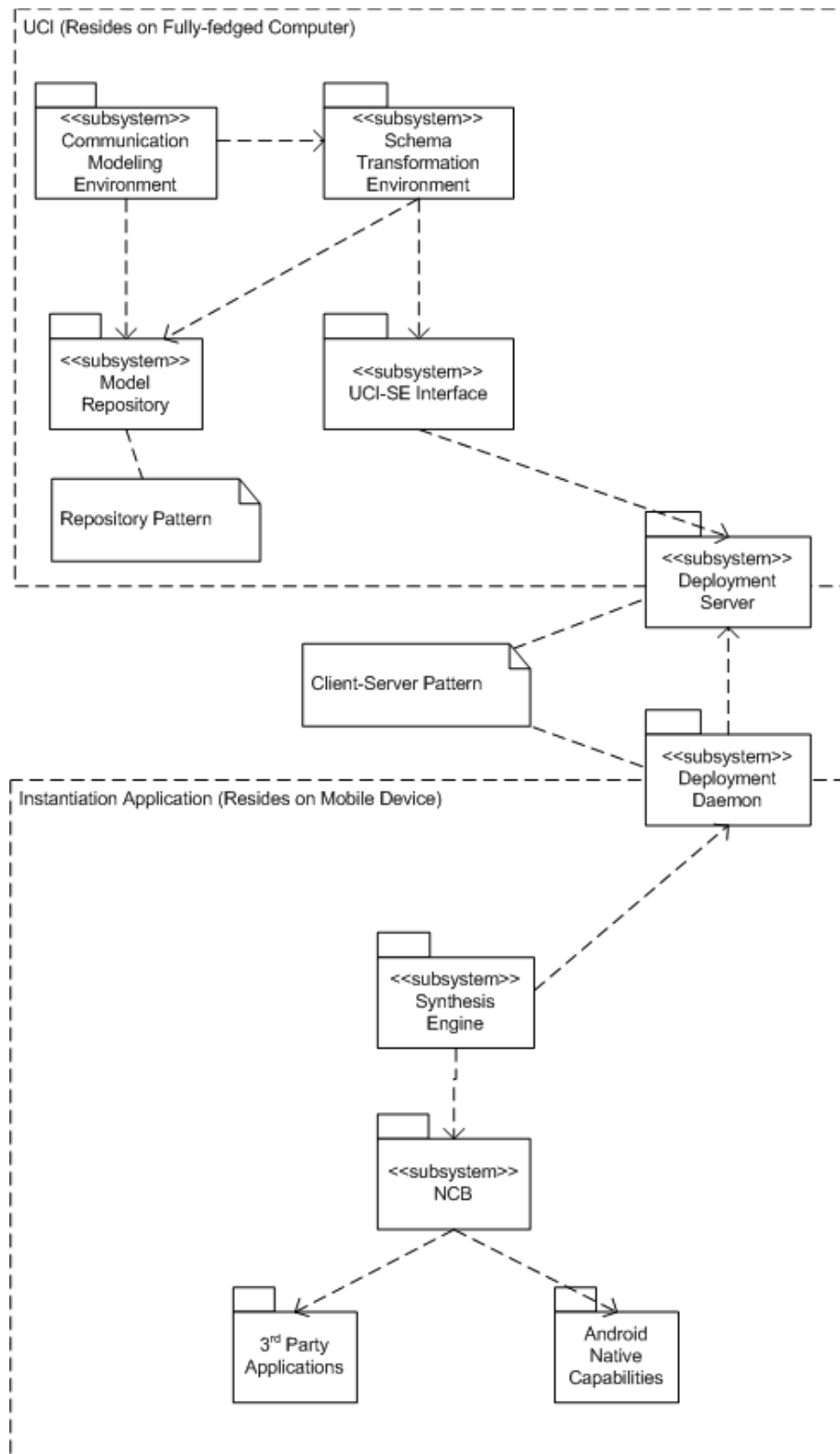


Figure 3

Metamodel

The Metamodel outlined in Figure 3 details the architecture of a mobile device capable of executing a developed communication model. It is abstracted into 4 major layers: Kernel, Libraries and Runtime, Framework, and Application layer. These layers, through exposed APIs, allow various communication processes to be established via the execution of a communication model. Feature variants are inherent to this model based on the platform used for implementation; such as Android or iOS. This project will be using the Android platform for implementation.

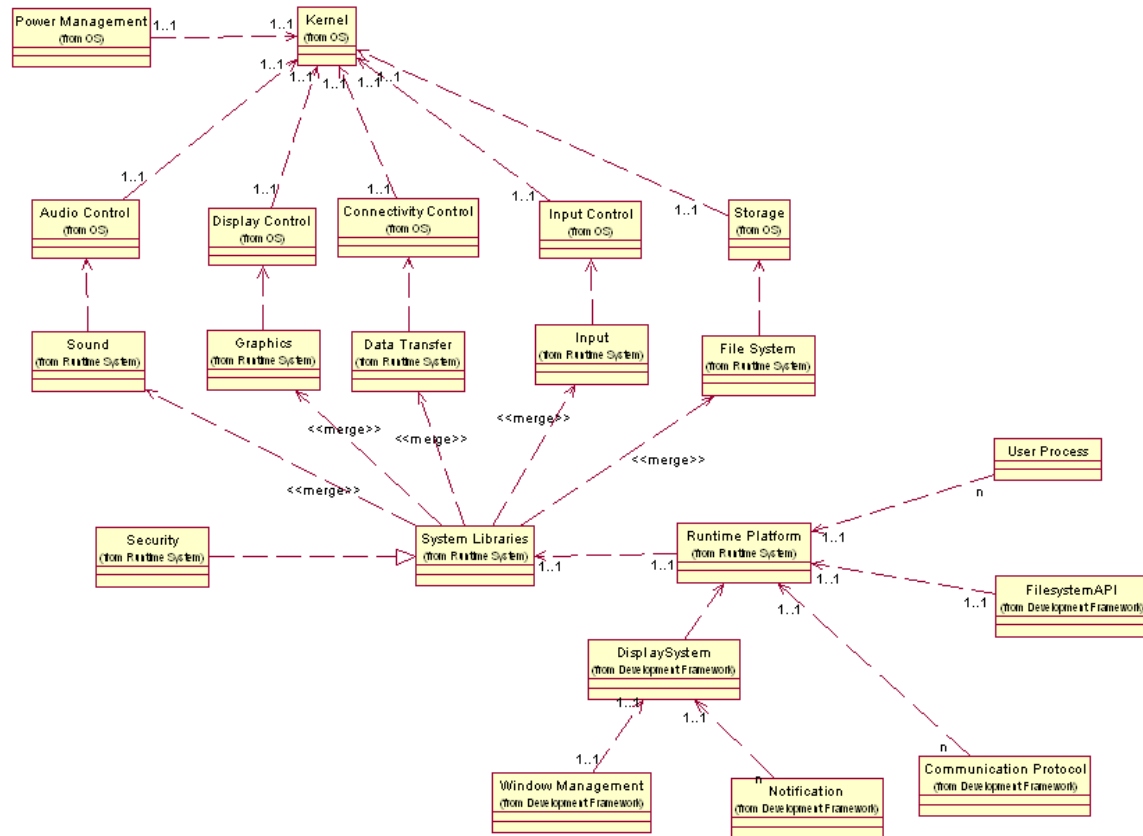


Figure 4

UML Profile for the Architecture

The UML Profile in the figure below outlines the constraints applied to the Mobile Architecture Metamodel. It shows the dependencies and extensions of MOF, which form the lineation of the various components of a mobile device. In summary, a platform extends from the ExecutionEnvironment UML metaclass as a platform such as Android is exactly just that: a constrained execution environment. Similarly, we extend the component metaclass for our framework, namely the Dalvik VM. A userProcess depends on both the framework as well as the execution platform, and includes a unique identification number as a special attribute. Finally, we have various libraries and kernel level artifacts that extend the Class metaclass because of the constrained kernels in mobile devices.

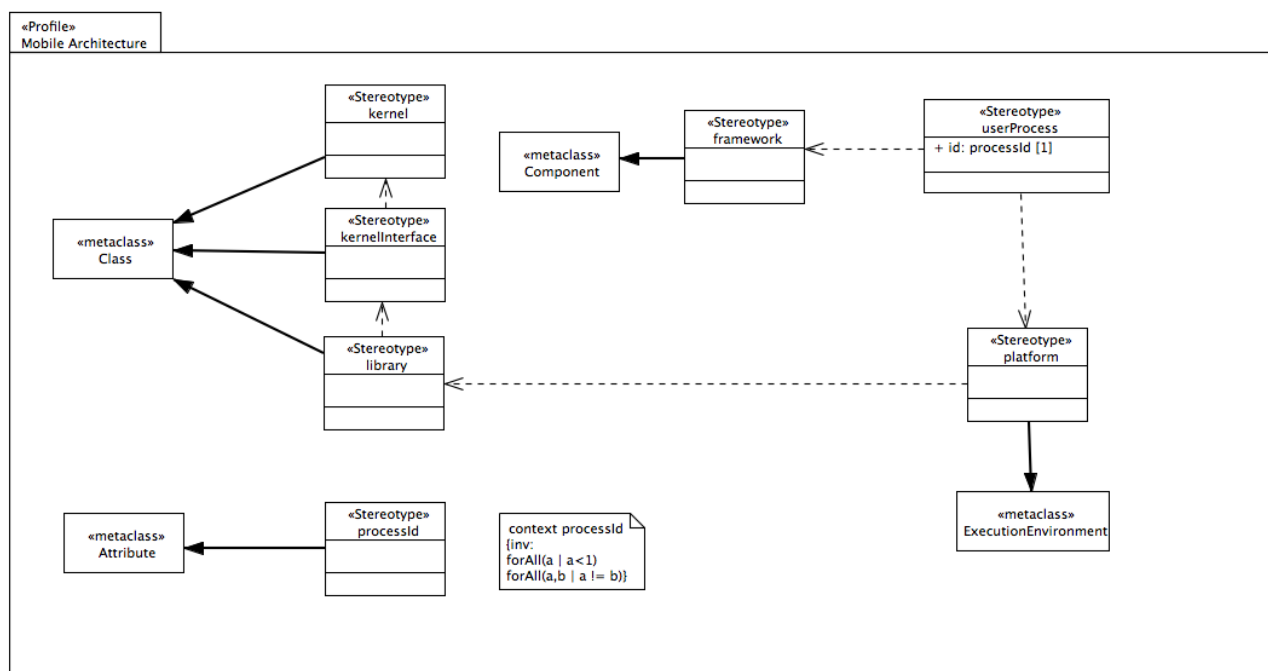


Figure 5

Generative Architecture

The Generative Architecture shown in Figure 5 shows the transformations from the high level architectural concepts found in the UML Meta Model to the implementation on the chosen Platform, Android.

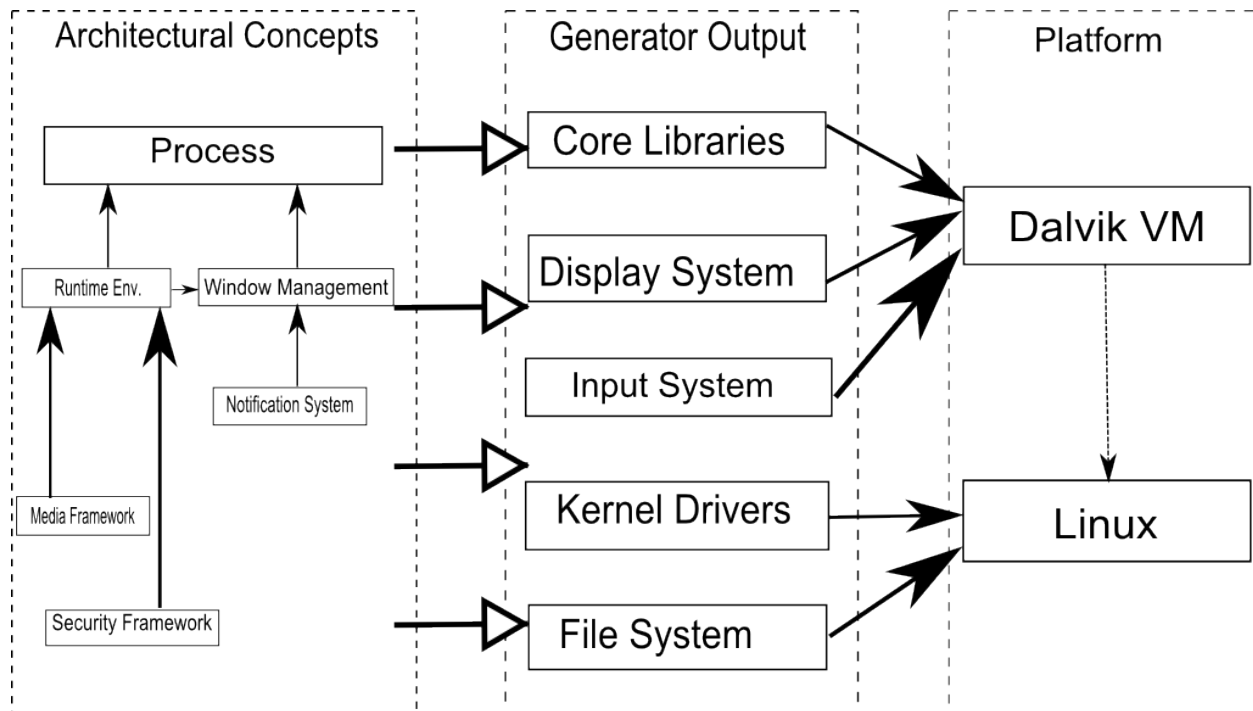


Figure 6

Validation of the system model

What follows is the system validation matrix.

Criteria	Pass/Fail
Every subsystem was touched by at least one use case.	PASS
Each subsystem has annotations for clarification.	FAIL
Check What does an operation mean? Ensure that the meaning of the operation is reflected in its name and format.	PASS
All sequence diagrams touch all the subsystems.	PASS
Subsystems compliant with each of the profiles.	PASS
At least one of the subsystems depicts the architecture pattern used in the application.	PASS
Architecture is annotated in the subsystems with notes.	PASS
Check the name of the subsystems matches the specified name in the architecture diagram.	FAIL

Check for dependencies among subsystem	PASS
Check for relations between subsystems.	PASS

In this validation we got two related failures. First, the validation dictates that each subsystem should have annotations for clarification, and the second states that the name of the subsystems should match the actual names in the implementation. In our implementation environment we don't get to specify these parameters, as the code is automatically generated for us. Actually, we see as a good thing that we failed these criteria, as in our chosen implementation we don't care about those details anymore.

Structured Walkthrough

In this section we do test our software architecture against the test cases to measure if we believe we will be able to comply with them with our choice. See the test cases defined in the Test cases section of the Use Case analysis chapter.

Test Case ID	Pass/Fail
MMMC-SystemTest-1	PASS
MMMC-SystemTest-2	PASS
MMMC-SystemTest-3	PASS
MMMC-SystemTest-4	PASS
MMMC-SystemTest-5	PASS
MMMC-SystemTest-6	PASS
MMMC-SystemTest-7	PASS
MMMC-SystemTest-8	PASS
MMMC-SystemTest-9	PASS
MMMC-SystemTest-10	PASS
MMMC-SystemTest-11	PASS
MMMC-SystemTest-12	PASS
MMMC-SystemTest-13	PASS

MMMC-SystemTest-14	PASS
MMMC-SystemTest-15	PASS
MMMC-SystemTest-16	PASS
MMMC-SystemTest-17	PASS
MMMC-SystemTest-18	PASS
MMMC-SystemTest-19	PASS
MMMC-SystemTest-20	PASS
MMMC-SystemTest-21	PASS
MMMC-SystemTest-22	PASS
MMMC-SystemTest-23	PASS
MMMC-SystemTest-24	PASS
MMMC-SystemTest-25	PASS
MMMC-SystemTest-26	PASS
MMMC-SystemTest-27	PASS

Object Design

In this section we detail the underpinnings of our proposed solution by going over the classes included in each subsystem, as well as the interaction between instances of these classes by the use of UML Sequence and State Machine diagrams.

Overview

Figure 6 we present the minimal class diagram for our proposed design. What follows is a brief description of the diagram, with emphasis on the design patterns used. The class discussion goes from the left to the right of the diagram. Our design allowed us to naturally apply four main design patterns: Observer, Visitor, Singleton, and Strategy patterns. As we can note from the diagram, we utilized the Observer pattern twice to model the interactions between the Views of the system and the models of the systems. This pattern permits the Views to ‘subscribe’ to the models. Whenever a model changes, it simply calls back its subscriber to notify them.

Similarly, we were able to apply the visitor pattern twice. Because our model for a communication, a MCM, is basically a tree-like data type, it is a natural candidate for this pattern. The Visitor pattern allows an entity to ‘accept’ a visitor while this entity does not need to have specific methods for its visit. This permits a more elegant design by allowing us to add more visitors without modifying the entity.

The singleton pattern was also used twice, for the `Deployment_Daemon`, and the `MCM_Instantiation_App` classes. The singleton pattern permits only one instance of a class at runtime. This is needed in the case of these two classes, as we cannot allow two communication models to be running at the same time.

Finally, the Strategy pattern is used for the deployment of Communication models. Because we have two ways of deployment, namely the Push and Pull models, we use this pattern to delegate that design to runtime, without affecting the caller.

More details the Class Diagram can be seen in the Detailed Class Design section.

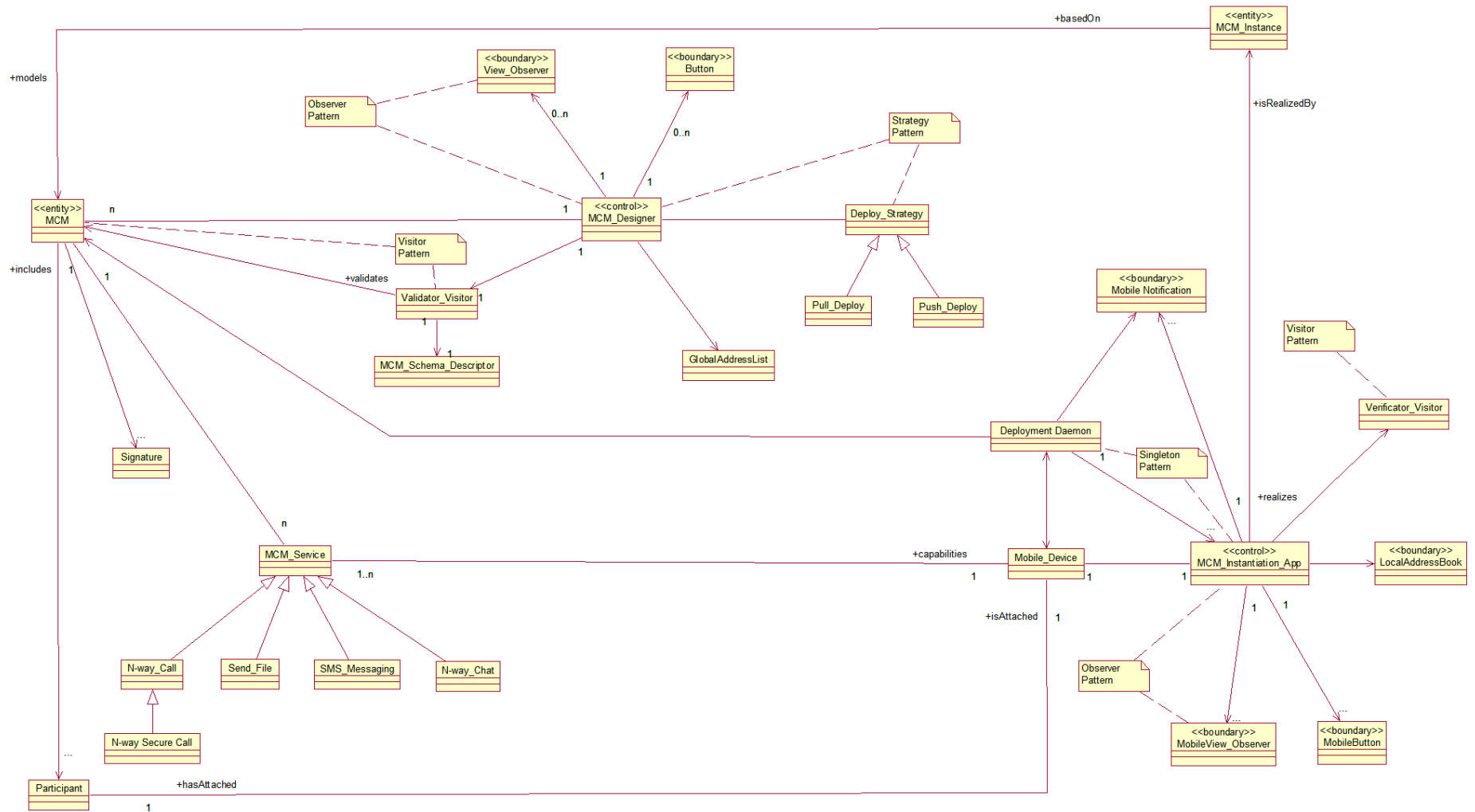


Figure 7

Object Interaction

The object interactions are presented using Sequence Diagrams, which detail the control and information flow between objects as identified from the scenarios, which were outlined in the SRAD. All necessary objects and actors present in the system are depicted in one or more of the Sequence Diagrams presented. They can be seen at Appendix C and D respectively.

State Machines for Main control objects

Figure 15 presents the state machine for the main control object in the mobile platform, which is the Application Instantiation control class. Basically, whenever the platform receives a model, this class gets initialized and the model given as a parameter. After this, the model is parsed and verified. Whenever the user is ready, the model is executed. Because a communication model can include one or more communication services, the Application Instantiation object will, one by one, hand of the services included in the model to the Android platform, and then wait until it is done. Whenever the communication is done, then the object will either terminate itself, or realize the next communication model.

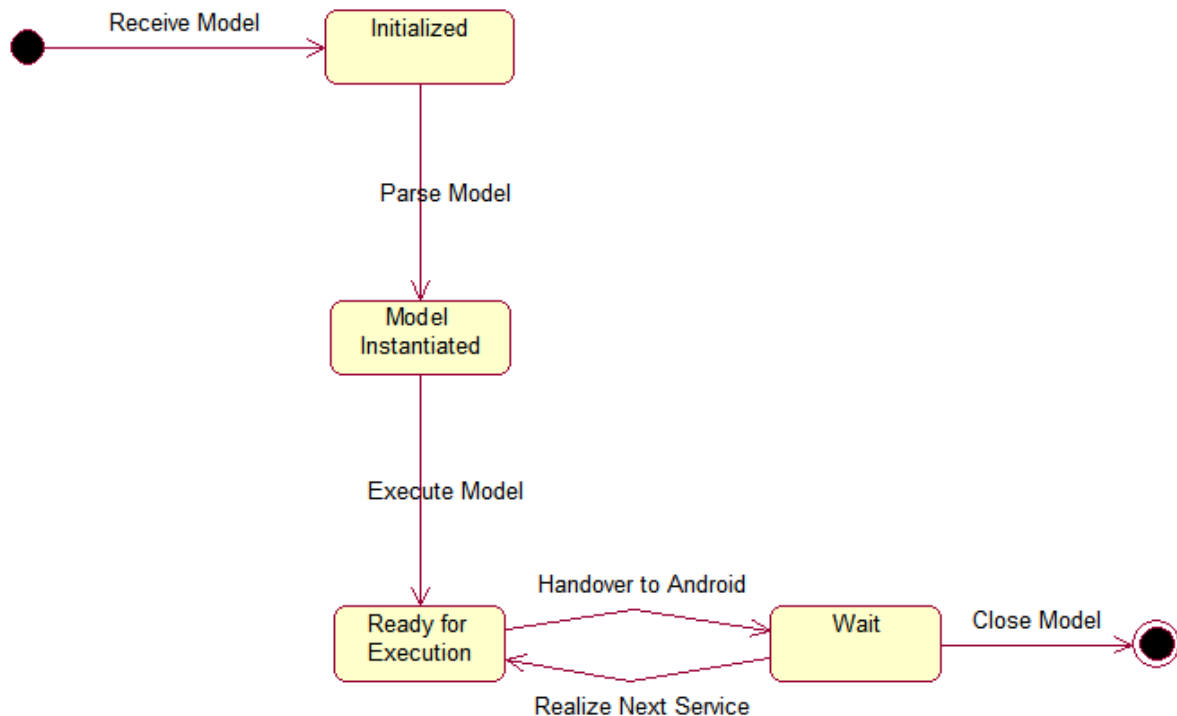


Figure 8

Detailed Class Design

What follows is a class-by-class description of the system with respect to the complete Class Diagram presented on Appendix C – Detailed Class Diagrams.

First, we have the entity class MCM, which stands for mobile communication model. This class represents an abstract data type, that contains all the information regarding a specific communication. As it can be seen, this may include a digital signature. A MCM includes one or more MCM_Services, which represent the different communication services supported by the majority of mobile platforms. MCM_Service is a general class, which is subclassed by N-Way_Call, Send_File, SMS_Messaging, and N-way_Chat, which are all typical communication services. We envision the possibility of having N-way_Secure_Calls, so we also include that as a subclass of N-Way_Call. A MCM also includes Participants, which are represented with their own class, as they are persisted independently from the MCM, and added to the MCM whenever needed. A MCM must be validated before being deployed. For this, we have a Validator_Visitor class, which follows the visitor pattern jointly with the MCM. This is needed as various managing objects may transform the MCM, so it is a natural candidate for this pattern. Note that to be able to validate, we need the MCM_Schema_Descriptor that provides the set of rules for the validation. As the main controller for the designer, we have the MCM_Designer class. This designer includes a reference to a GlobalAddressList from where it extracts Participants. As it also represents a Graphical User Interface, we include the Button class, and also the View_Observer boundary class, which follows the Observer pattern. The idea is to have the MCM model edited on the MCM_Designer, and whenever that happens, the View_Observer will update because it is subscribed to the MCM_Designer update events, as defined on the Observer pattern. The final part, which resides on the designer, is the Deploy_Strategy class, with the Pull_Deploy and the Push_Deploy subclasses. This part of the system follows the Strategy pattern; it presents the same interface, but hides away the algorithm for the deployment to go thru.

On the mobile device, we start by having a Deployment_Daemon and MCM_Instantiation_App classes. These classes follow the singleton pattern, which mean that only one instance of the

class is permitted at runtime. This limitation is needed, as a communication model can come in while another communication model is being realized, and a mobile device typically can handle only one communication function at a time. The `Deployment_Daemon` class depends on the availability of a `Mobile_Notification` from the mobile platform, which will notify the system of an incumbent communication model. All of these classes depend on the `Mobile_Device` class, which uses the "platform" stereotype to represent all the frameworks and functionality available in a mobile platform, as per the UML Profile defined. Other supporting classes are also included in the mobile device, such as `MobileView_Observer` (which follows an Observer Pattern) and `Mobile_Button`, for user interaction. Similarly, we also have a `LocalAddressBook` class to support adding participants to the model at runtime, and finally, a `Verificator_Visitor`, which follows the Visitor Pattern, to verify the Signature, if any, on the MCM whenever it is deployed to the device.

Validation

Criteria - Syntax Checks for Classes	Pass/Fail
Check that multiplicity on an association is correctly represented on the class diagram.	PASS
Ensure that stereotypes are represented by << >> on classes, attributes, operations and relationships on a class diagram.	PASS
Check to see if a class is an exception class.	PASS
Check how error handling is modeled and implemented in a class.	PASS

Criteria - Semantic Checks for Classes	Pass/Fail
Check direction for association.	PASS
Check the meaning of the relationships on a class diagram.	PASS
Check for collection of classes.	PASS
Check the business rules behind the multiplicity.	PASS
Check for association classes.	PASS
Check if the operations of a class that has been specialized are overloaded.	PASS
Check for encapsulation.	PASS

Ensure that language constructs subject to interpretations are checked for their implied meaning.	PASS
---	------

Criteria - Aesthetic Checks for Classes	Pass/Fail
Check number of attributes.	PASS
Check the number of operations.	PASS
Check the load on operations.	PASS
Check the load on the class.	PASS

Structured Walkthrough

In this section we do test our detailed class diagram against the test cases to measure if we believe we will be able to comply with them with our choice. See the test cases defined in the Test cases section of the Use Case analysis chapter.

Test Case ID	Pass/Fail
MMMC-SystemTest-1	PASS
MMMC-SystemTest-2	PASS
MMMC-SystemTest-3	PASS
MMMC-SystemTest-4	PASS
MMMC-SystemTest-5	PASS
MMMC-SystemTest-6	PASS
MMMC-SystemTest-7	PASS
MMMC-SystemTest-8	PASS
MMMC-SystemTest-9	PASS
MMMC-SystemTest-10	PASS
MMMC-SystemTest-11	PASS
MMMC-SystemTest-12	PASS
MMMC-SystemTest-13	PASS
MMMC-SystemTest-14	PASS
MMMC-SystemTest-15	PASS

MMMC-SystemTest-16	PASS
MMMC-SystemTest-17	PASS
MMMC-SystemTest-18	PASS
MMMC-SystemTest-19	PASS
MMMC-SystemTest-20	PASS
MMMC-SystemTest-21	PASS
MMMC-SystemTest-22	PASS
MMMC-SystemTest-23	PASS
MMMC-SystemTest-24	PASS
MMMC-SystemTest-25	PASS
MMMC-SystemTest-26	PASS
MMMC-SystemTest-27	PASS

Implementation

In this chapter, we first make a description of the platform specific model in the form of a component diagram. Then we validate our systems against the test cases defined in the Use case Analysis.

Description of the PSM

In Figure 9, we present our platform-specific implementation of our Metamodel. Specifically, we use the extensions defined in our UML Profile to instantiate an Android platform as an execution environment. Similarly, we use the UML Profile to instantiate the Dalvik Virtual Machine and the Instantiation Application as frameworks, which as per our profile extend the components UML metaclass. The Deployment Daemon is simpler, and thus does not use directly our profile.

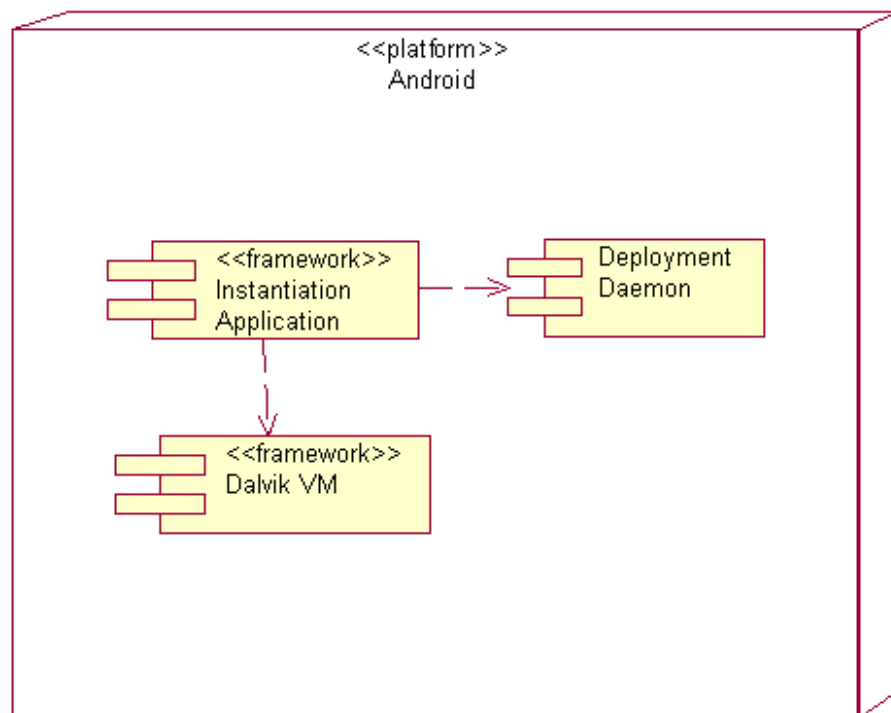


Figure 9

Validation

Test Case ID	MMMC-SystemTest-1
Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System accepted the model as valid.
Actual Response	System accepted the model as valid.
Result	PASS

Test Case ID	MMMC-SystemTest-2
Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 services and 3 participants.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System accepted the model as valid.
Actual Response	System accepted the model as valid.
Result	PASS

Test Case ID	MMMC-SystemTest-3
---------------------	--------------------------

Purpose	To validate Use Case MMMC-2 (Validate model) functionality
Preconditions	User has launched the Design environment and has designed an invalid MCM with 2 services and 3 participants, but one participant is lacking a mobile device.
Stimulus	User clicks on the validate feature of the model.
Expected Response	System rejects the model and explains the errors found.
Actual Response	System rejects the model and explains the errors found.
Result	PASS

Test Case ID	MMMC-SystemTest-4
Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 1 service and 2 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-5
---------------------	--------------------------

Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 2 services and 3 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-6
Purpose	To validate Use Case MMMC-11 (Design Mobile Communication model) functionality
Preconditions	User has launched the Design environment.
Stimulus	User designs a MCM with 3 services and 3 participants.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-7
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-8
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 services and 3 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-9
Purpose	To validate Use Case MMMC-9 (Sign Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 3 service and 3 participants.
Stimulus	User adds a signature to the model.
Expected Response	System accepts the model and saves to disk.
Actual Response	System accepts the model and saves to disk.
Result	PASS

Test Case ID	MMMC-SystemTest-10
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants. There is connectivity to the

	internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System deploys the model to the server.
Actual Response	
Result	FAIL

Test Case ID	MMMC-SystemTest-11
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 2 service and 3 participants. There is connectivity to the internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System deploys the model to the server.
Actual Response	
Result	FAIL

Test Case ID	MMMC-SystemTest-12
Purpose	To validate Use Case MMMC-12 (Deploy Mobile Communication Model) functionality
Preconditions	User has launched the Design environment and has designed a valid MCM with 1 service and 2 participants. There is no connectivity to the internet from both the fully-fledged computer internet, and the mobile device.
Stimulus	User chooses the deploy functionality.
Expected Response	System is unable to deploy the model.

Actual Response	System is unable to deploy the model.
Result	PASS

Test Case ID	MMMC-SystemTest-13
Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 1 service and 2 participants has been deployed and is waiting on the server. There is connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device automatically polls the server and eventually gets the model.
Actual Response	The device automatically polls the server and eventually gets the model.
Result	PASS

Test Case ID	MMMC-SystemTest-14
Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 2 services and 2 participants has been deployed and is waiting on the server. There is connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device automatically polls the server and eventually gets the model.
Actual Response	The device automatically polls the server and eventually gets the model.
Result	PASS

Test Case ID	MMMC-SystemTest-15
Purpose	To validate Use Case MMMC-13 (Pull Model) functionality
Preconditions	A MCM with 1 service and 2 participants has been deployed and is waiting on the server. There is no connectivity to the internet from the mobile device.
Stimulus	None.
Expected Response	The device is unable to fetch the MCM.
Actual Response	The device is unable to fetch the MCM.
Result	PASS

Test Case ID	MMMC-SystemTest-16
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model.
Stimulus	The user selects Run from the presented menu.
Expected Response	System will load the mobile communication model and launch the required services.
Actual Response	System will load the mobile communication model and launch the required services.
Result	PASS

Test Case ID	MMMC-SystemTest-17
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile

	Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model. Mobile model no longer present on device.
Stimulus	The user selects Run from the presented menu.
Expected Response	The system informs the user that the model cannot be found.
Actual Response	The system informs the user that the model cannot be found.
Result	PASS

Test Case ID	MMMC-SystemTest-18
Purpose	To validate Use Case MMMC-1 (Instantiate Mobile Communication Model) in the actual implementation.
Preconditions	The mobile device has found and downloaded a new mobile communication model. Model is malformed.
Stimulus	The user selects Run from the presented menu.
Expected Response	System informs user that the model is invalid.
Actual Response	No response.
Result	FAIL

Test Case ID	MMMC-SystemTest-19
Purpose	To validate Use Case MMMC-4 (Do N-way chat) in

	the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device has chat capabilities.
Stimulus	N-way chat service found in model.
Expected Response	N-way chat service launched
Actual Response	N-way chat service launched
Result	PASS

Test Case ID	MMMC-SystemTest-20
Purpose	To validate Use Case MMMC-4 (Do N-way chat) in the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device contains multiple chat capabilities.
Stimulus	N-way chat service found in model.
Expected Response	User is presented with option to select application.
Actual Response	Undefined.
Result	FAIL

Unable to validate this use case as multiple chat clients were not available for testing.

Test Case ID	MMMC-SystemTest-21
Purpose	To validate Use Case MMMC-4 (Do N-way chat) in the actual implementation.
Preconditions	The currently executing model contains a chat service request. The mobile device does not contain chat

	capabilities.
Stimulus	N-way chat service found in model.
Expected Response	System responds with message indicating no appropriate capability present
Actual Response	No response.
Result	FAIL

Test Case ID	MMMC-SystemTest-22
Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request. The mobile device has call capabilities.
Stimulus	N-way call service found in model
Expected Response	N-way chat service launched using native voice call infrastructure.
Actual Response	N-way chat service launched using native voice call infrastructure.
Result	PASS

Test Case ID	MMMC-SystemTest-23
Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request for multiple participants. The mobile device has call capabilities.
Stimulus	N-way call service found in model
Expected Response	N-way chat service launched using native voice call infrastructure for each participant and allow user to merge calls.

Actual Response	N-way chat service launched using native voice call infrastructure for each participant and allow user to merge calls.
Result	PASS

Test Case ID	MMMC-SystemTest-24
Purpose	To validate Use Case MMMC-8 (Do N-way call) in the actual implementation.
Preconditions	The currently executing model contains a call service request. The mobile device does not have call capabilities.
Stimulus	N-way call service found in model
Expected Response	System responds with message indicating no appropriate capability present
Actual Response	No response.
Result	FAIL

Test Case ID	MMMC-SystemTest-25
Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with some participants empty.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays local contact list to populate missing participant fields.
Actual Response	Undefined.
Result	FAIL

This use case was not implemented.

Test Case ID	MMMC-SystemTest-26
Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with all participant fields missing.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays local contact list to populate all participant fields.
Actual Response	Undefined.
Result	FAIL

This use case was not implemented.

Test Case ID	MMMC-SystemTest-27
Purpose	To validate Use Case MMMC-3 (Add or Specify Participants) in the actual implementation.
Preconditions	The currently executing model contains a call service request with all participant fields missing. Local contact list is unavailable.
Stimulus	User selects “Run” from the presented menu.
Expected Response	System displays message indicating unavailability of contact list.
Actual Response	Undefined.
Result	FAIL

This use case was not implemented.

Glossary

CVM – Communications Virtual Machine. Refers to an already implemented system for modeling and realizing computer-to-computer communication schemas.

Communication Schema – A blueprint of a communication, including the type of services used and the participants. It is normally persisted as an XML file.

Mobile device – Throughout this document we use this term to represent the group of mobile devices with computer-like capabilities. These devices typically have limited computing power in terms of available CPU speed and RAM size. Present-day examples include Apple's iPhone, Apple's iPod Touch, and Google's Android-based telephones.

Fully-fledged computer – Throughout this document we use this term to represent the group of personal desktop or laptop computers. These devices typically have abundant computing power in terms of available CPU speed and RAM size. Present-day examples include Apple's iMac, Apple's MacBook series, and Dell's Inspiron series.

Android Activities – An activity is a single, focused thing that the user can do. Almost all activities interact with the user, so the Activity class takes care of creating a window for you in which you can place your User Interface. More info at [AActivity].

Android Intents – An intent is an abstract description of an operation to be performed. It provides a facility for performing late runtime binding between the code in different applications. More info at [AIntent].

References

- [USDP] Unified Software Development Process. As defined on: Object-Oriented Software Engineering, Third Edition, by B. Bruegge, and A. Dutoit. Pages 641-644. Prentice Hall. 2010.
- [SRAD] Software Requirement and Analysis Document for the Modeling and Realization of Mobile Multimedia Communications Project. X. Collazo-Mojica, K. Morris, and Z. Yuan. 2010.
- [SDD] Software Design Document for the Modeling and Realization of Mobile Multimedia Communications Project. K. Morris and X. Collazo-Mojica. 2010.

Appendix

Appendix A – Complete use case diagram, use case diagram for the use cases to be implemented

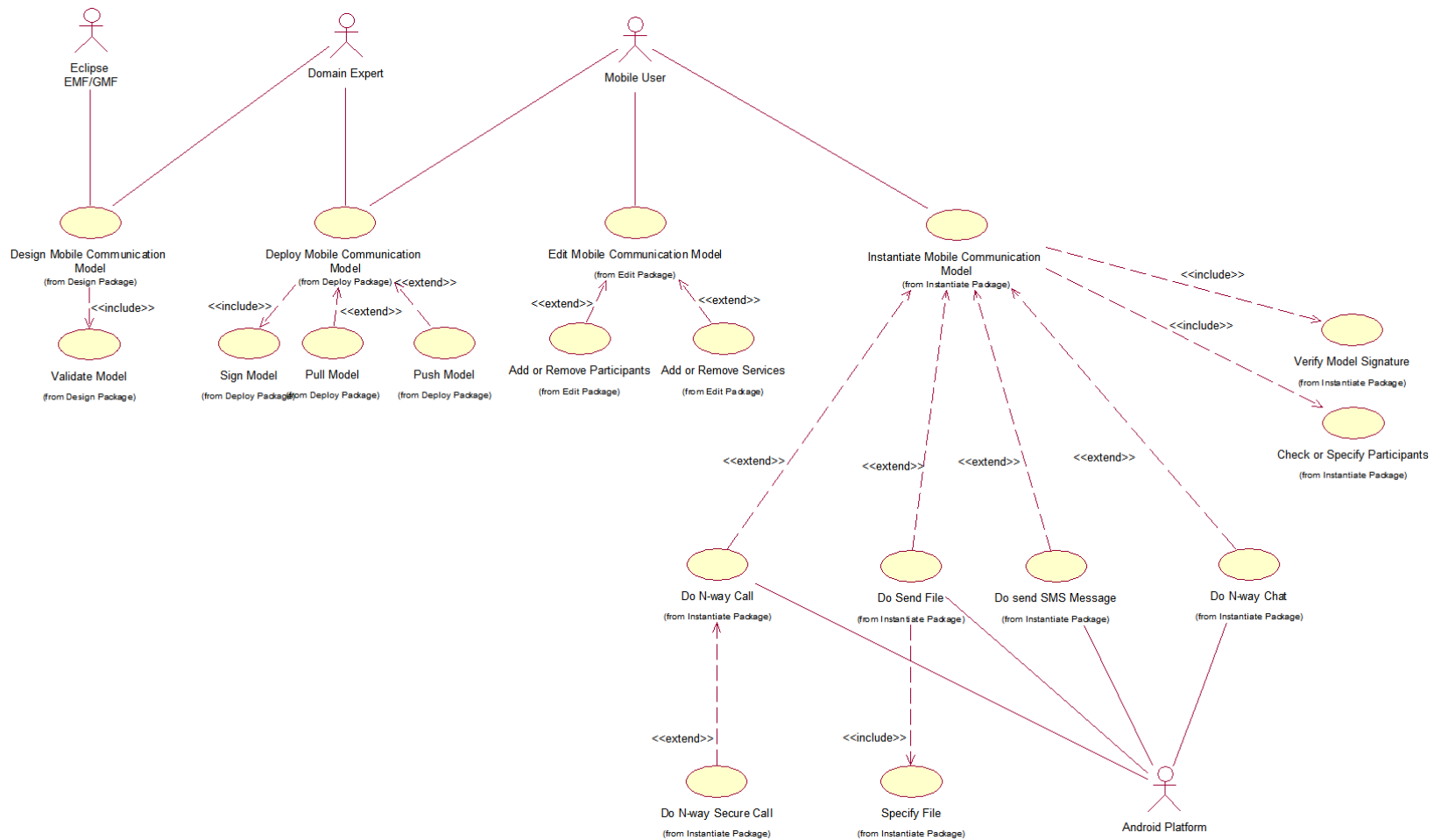


Figure 10

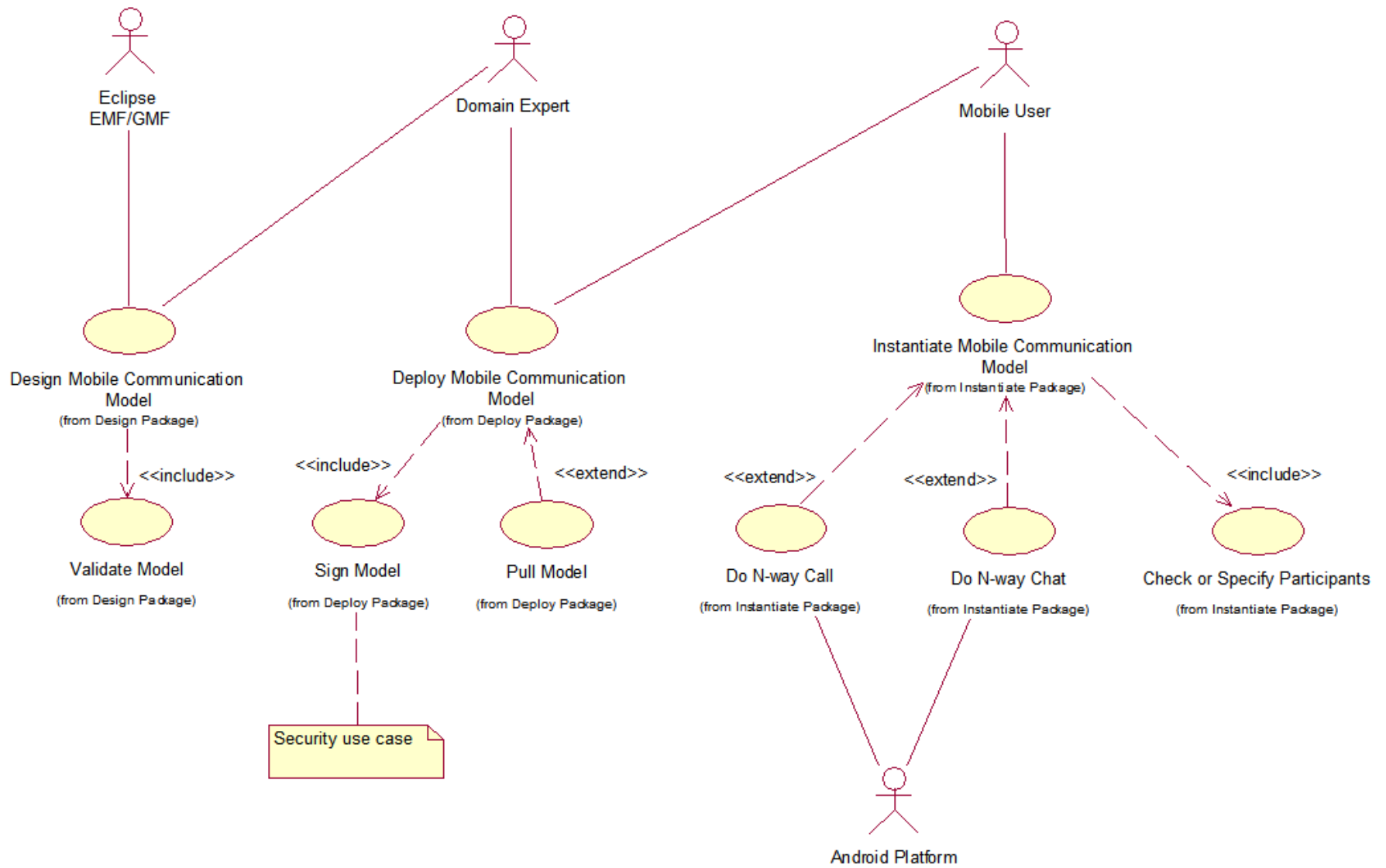


Figure 11

Appendix B – Use Cases to be implemented

Use Case name: Instantiate Mobile Communication Model

Use Case ID: *MMMC-1*

Use Case Level: *High-level*

Details:

Actor(s): Mobile User

Pre-conditions: The device must be properly booted and ready to be used. The Mobile Communication Model should be already downloaded to the device. The device must have connectivity (I.e. good reception from the access point or cellular tower).

Description:

Trigger: The user initiates an action by selecting to run an already deployed communication model.

The system responds by:

1. Running the Verify Model Signature use case.
2. The system runs the Specify Participants use case.
3. Finally, following the Mobile Communication Model, the system responds by presenting any combination of the extended use cases.

Relevant requirements: *None.*

Post-conditions: *The mobile user was able to instantiate a communication model. The user is presented with a notification that he will be transferred to the relevant application for the realization of the communication.*

Alternative Courses of Action: *If any of the steps above fail, this actions will be done.*

1. If the included Verify Model Signature use case fails, the system will present the mobile user with a notification of the issue.
2. If the included Specify Participants use case fails, the system will present the mobile user with a notification of the issue.
3. If for some other reason the system is not able to instantiate the communication, the system will present a relevant notification.

Extensions: *The mobile communication model can present any combination of the following extended use cases:*

- Do N-way call
- Do Send File
- Do send SMS Message
- Do N-way chat

Exceptions: If no communication model is present on the device, then a notification will be presented and the use case aborted. If there is no connectivity, then a notification will be presented and the use case aborted.

Concurrent Uses: *None.*

Related Use Cases:

Just before: Deploy Mobile Communication Model

Just after: None.

Decision Support

Frequency: 5 times daily. *Every time the system is used in a mobile device, this use case will be executed.*

Criticality: *High. The system will not function properly if this use case is not available or implemented.*

Risk: *High. This use case belongs to the critical path of the project.*

Constraints:

- Usability: Initiating this use case should take no more than 4 clicks on the mobile user interface.
- Reliability: If all validation steps are passed, this use case should function at least 90% of the time.
- Performance: Executing this use case should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: Only one mobile device platform will be supported, this being Android 2.2.
- Implementation: Must be implemented in the mobile device's native development platform.

Modification History – Initial Version (September 28, 2010)

Owner: Xabriel J. Collazo-Mojica

Initiation date: September 28, 2010

Date last modified: September 28, 2010

Use Case name: Validate Model

Use Case ID: *MMMC-2*

Use Case Level: *Functional sub-use case*

Details:

Actor(s): Mobile User

Pre-conditions: The Design Mobile Communication Model use case should be in progress.

Description:

Trigger: The expert domain initiates this action by pressing the validate button in the Designing environment.

The system responds by:

1. Validating the Mobile Communication Model. This entails comparing the model schema with an already defined schema descriptor.

Relevant requirements: *The schema descriptor will be produced as part of this project. Once produced, it will be used on this use case.*

Post-conditions: *The mobile user was able to validate the communication model and was presented with a notification that the model passed the validation.*

Alternative Courses of Action:

1. If the Communication Model fails the validation step, a notification of the error will be presented to the user, and the use case will be aborted.

Extensions: *None.*

Exceptions: If the underlying EMF and/or GMF modeling platform aborts, then this use case will also abort with no notifications to the user.

Concurrent Uses: *None.*

Related Use Cases:

Just before: Deploy Mobile Communication Model

Just after: Specify Participants use case.

Decision Support

Frequency: 5 times daily. *Everytime the system is used to design a communication model, this use case will be executed.*

Criticality: *High. The system will not function properly if this use case is not available or implemented.*

Risk: *High. This use case belongs to the critical path of the project.*

Constraints:

- Usability: The domain expert should be able to validate a model with only one click.
- Reliability: This use case should present neither false positives nor false negatives.
- Performance: Executing this use case should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: This use case will only support Eclipse .
- Implementation: Must be implemented on top of Eclipse's EMF/GMF platform.

Modification History – Initial Version (September 28, 2010)

Owner: Xabriel J. Collazo-Mojica

Initiation date: September 28, 2010

Date last modified: September 28, 2010

Use Case name: Check or Specify Participants

Use Case ID: *MMMC-3*

Use Case Level: *Functional sub-use case*

Details:

Actor(s): Mobile User

Pre-conditions: The Instantiate Mobile Communication Model use case should be in progress.

Description:

Trigger: The system initiates this action automatically if a user already chose the parent use case.

The system responds by:

1. Presenting the user with the mobile device's native address book interface so that he/she can choose the participants of the communication.

Relevant requirements: *Each communication model presents different number of participants. That requirement will be embedded on each communication model.*

Post-conditions: *The mobile user was able to specify the communication model participants.*

Alternative Courses of Action:

1. *If the participants were already defined in the model creation, then there is no need to re-specify them on this use case.*

Extensions: *None.*

Exceptions: If the user chooses to cancel the action, the use case will be aborted.

Concurrent Uses: *This use case is included in the Instantiate Mobile Communication Model use case.*

Related Use Cases:

Just before: Deploy Mobile Communication Model

Just after: Any of the extensions for the Instantiate Mobile Communication Model use case.

Decision Support

Frequency: 2 time daily. *Everytime the system is used in a mobile device, this use case will be executed.*

Criticality: *High. The system will not function properly if this use case is not available or implemented.*

Risk: *High. This use case belongs to the critical path of the project.*

Constraints:

- Usability: This use case should be started automatically.
- Reliability: This use case should be as reliable as the native address book of the mobile device.
- Performance: Executing this use case should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: At least one mobile device platform should be supported.
- Implementation: Must be implemented in the mobile device's native development platform.

Modification History – Initial Version (September 28, 2010)

Owner: Xabriel J. Collazo-Mojica

Initiation date: September 28, 2010

Date last modified: September 28, 2010

Use Case name: Do N-way chat

Use Case ID: *MMMC-4*

Use Case Level: *Functional sub-use case*

Details:

Actor(s): Mobile User

Pre-conditions: For any given Mobile Communication Model, the Instantiate Mobile Communication Model use case should be in progress. The device should have connectivity.

Description:

Trigger: The user initiates an action by selecting the N-way chat option from the presented list of available communication types of the currently running communication instance.

The system responds by:

1. Presenting the user with a relevant communication application that allows N-way chat with the participants of the communication instance.

Relevant requirements: *Each communication model presents different number of participants. That requirement will be embedded on each communication model.*

Post-conditions: *The mobile user was able to initiate an N-way chat with all the participants specified in the communication instance.*

Alternative Courses of Action:

1. In the event that no common or compatible N-way chat software is found between all the participants in the communication instance, a notification will be presented and the use case will be aborted.

Extensions: *None.*

Exceptions: In the event that there is no connectivity, the use case will notify the user and abort.

Concurrent Uses: *This use case is included in the Instantiate Mobile Communication Model use case.*

Related Use Cases:

Just before: Deploy Mobile Communication Model

Just after: None.

Decision Support

Frequency: 3 times daily. *Frequency depends on the definition of the communication models.*

Criticality: *Low. The system does not depend on this use case.*

Risk: *Low. This use case does not belong to the critical path of the project.*

Constraints:

- Usability: Once the user chooses the N-way chat option, all other steps to realize the communication should be automatic.
- Reliability: This use case should be as reliable as the native N-way chat application being used to support the communication in the mobile device.
- Performance: Executing this use case should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: At least one mobile device platform should be supported.
- Implementation: Must be implemented in the mobile device's native development platform.

Modification History – *Initial Version (September 28, 2010)*

Owner: Xabriel J. Collazo-Mojica

Initiation date: September 28, 2010

Date last modified: September 28, 2010

Use Case name: Do N-way Call

Use Case ID: *MMMC-8*

Use Case Level: *Functional sub-use case*

Details:

Actor(s): Mobile User

Pre-conditions: For any given Mobile Communication Model, the Instantiate Mobile Communication Model use case should be in progress. The device should have connectivity.

Description:

Trigger: The user initiates an action by selecting the N-way Call option from the presented list of available communication types of the currently running communication instance.

The system responds by:

1. Presenting the user with a relevant communication application that allows to N-way Call with the participants of the communication instance.

Relevant requirements: *Each communication model presents different number of participants. That requirement will be embedded on each communication model.*

Post-conditions: *The mobile user was able to initiate a N-way Call with all the participants specified in the communication instance.*

Alternative Courses of Action:

1. In the event that no common or compatible N-way Call software is found between all the participants in the communication instance, a notification will be presented and the use case will be aborted.

Extensions: *None.*

Exceptions: In the event that there is no connectivity, the use case will notify the user and abort.

Concurrent Uses: *This use case is included in the Instantiate Mobile Communication Model use case.*

Related Use Cases:

Just before: Deploy Mobile Communication Model

Just after: None.

Decision Support

Frequency: 3 times daily. *Frequency depends on the definition of the communication models.*

Criticality: *High. The system depends on this use case.*

Risk: *High. This use case belongs to the critical path of the project.*

Constraints:

- Usability: Once the user chooses the N-way Call option, all other steps to realize the communication should be automatic.
- Reliability: This use case should be as reliable as the native N-way Call application being used to support the communication in the mobile device.
- Performance: Executing this use case should take no more than 60 seconds, discounting the time waiting for user input.
- Supportability: At least one mobile device platform should be supported.
- Implementation: Must be implemented in the mobile device's native development platform.

Modification History – *Initial Version (September 28, 2010)*

Owner: Xabriel J. Collazo-Mojica

Initiation date: September 28, 2010

Date last modified: September 28, 2010

Use Case name: Sign Model

Use Case ID: *MMMC-9*

Use Case Level: *System-level*

Details:

Actor(s): *Domain Expert*

Pre-conditions: The Deploy Mobile Communication Model use case is in progress.

Description:

1. The system signs the model using the private key of the user.

Relevant requirements: *This Use Case depends on the availability of a Public/Private key encryption mechanism.*

Post-conditions: *The model to be deployed will have a signature attached.*

Alternative Courses of Action: *1. If no Private key is available, the system will not sign the model. No error messages are generated.*

Extensions: *None*

Exceptions: Private key invalid

Related Use Cases: Deploy Mobile Communication Model, *Verify Model Signature*

Decision Support

Frequency: *Twice weekly*

Criticality: *Not necessary for the functioning of the system.*

Risk: Medium risk. Strong skill set not available in-house, however many external resources exist.

Constraints: None

Modification History – *Initial Version (2010-09-26)*

Owner: Karl Morris

Initiation date: 2010-09-26

Date last modified: 2010-09-26

Use Case name: Design Mobile Communication Model

Use Case ID: *MMMC-11*

Use Case Level: *High-level*

Details:

Actor(s): Domain Expert, Eclipse EMF/GMF

Pre-conditions: *The user clicks on the “Design Model”button.*

Description:

1. The system loads the design environment
2. The user designs the communication workflow and clicks on “Save”
3. The system presents the user with dialog to save the model.
4. The user selects the location to save the model and clicks on “Save”.

Relevant requirements: *None*

Post-conditions: *The system saves a model to an available file system.*

Alternative Courses of Action: *None*

Extensions: *None*

Exceptions: *None*

Concurrent Uses: *None*

Related Use Cases: Deploy Mobile Communication Model

Decision Support

Frequency: *Use case occurs when initiated by the user.*

Criticality: *This is a critical use case. If unavailable, the user's ability to use the system will be severely impacted.*

Risk: *This is a high-risk use case. No internal experts are available and external resources are limited.*

Constraints:

- The interface must be clear and intuitive allowing the user to manipulate the various tools to create a model in a short amount of time.

Modification History – Initial Version (2010-09-26)

Owner: Karl Morris

Initiation date: 2010-09-26

Date last modified: 2010-09-26

Use Case name: Deploy Mobile Communication Model

Use Case ID: *MMMC-12*

Use Case Level: *High-level*

Details:

Actor(s): Domain Expert, Mobile User

Pre-conditions: *The user has designed or loaded a model to be deployed.*

Description:

1. The user clicks on “Deploy Model”
2. The system prompts the user to identify the recipient
3. The user confirms the recipient
4. The system initiates the Sign Model use case if a private key is available. When completed the system deploys the model to the selected Mobile User.
5. The Mobile User will receive a notification that a model has been received and be prompted to verify it if a signature is attached.

Relevant requirements: *None*

Post-conditions: *The Mobile User will have a model deployed to their mobile device.*

Alternative Courses of Action: *None*

Extensions:

Exceptions: *Unsuccessful deployment – This can arise because the mobile device or deployment service is unavailable. If this happens the user will attempt to deploy the model at a later time.*

Concurrent Uses: *Sign Model*

Related Use Cases: *Design Mobile Communication Model*

Decision Support

Frequency: Twice per week.

Criticality: *This is a critical use case. The system will not function without it being present.*

Risk: *This is a high risk use case. No internal experts available and limited external resources.*

Constraints:

- All necessary steps required to deploy a model and verify its signature should be presented in a continuous and intuitive manner.
- Receipt of the model on a mobile device should be completed within 30 seconds of deployment.
- This Use Case should whenever possible utilize the default notification system of the mobile operating system on which it is deployed.

Modification History – Initial Version (2010-09-26)

Owner: Karl Morris

Initiation date: 2010-09-26

Date last modified: 2010-09-26

Use Case name: Pull Model

Use Case ID: *MMMC-13*

Use Case Level: *High-level*

Details:

Actor(s): Mobile User

Pre-conditions: *A model has been deployed and is awaiting acceptance by the mobile device.*

Description:

- 1. The system establishes a connection with the deployment service and checks for the availability of a model.*
- 2. If a model is found the system initiates steps to download the model.*
- 3. The system notifies the user that a model has been received.*

Relevant requirements: *None*

Post-conditions: *The system has received the deployed model.*

Alternative Courses of Action

Extensions: *None*

Exceptions: *Unable to establish a connection with the deployment service.*

Concurrent Uses: *None*

Related Use Cases: *Deploy Mobile Communication Model*

Decision Support

Frequency: 288 times per day

Criticality: *This is a critical use case.*

Risk: Low risk. *Skill set is available in-house to realize this use case.*

Constraints:

- The use case must download the model within 20 seconds of availability on the deployment server.

Modification History -- Initial Version (2010-09-026)

Owner: Karl Morris

Initiation date: 2010-09-26

Date last modified: 2010-09-26

Appendix C - Object diagrams for the analysis model

Object models depicted with UML Object Diagrams are shown below. Please note that to make sense out of these models, they should be analyzed jointly with the respective scenarios from the [SRAD] document.

Figure 11 and Figure 13 present the scenarios for Instantiate Mobile Communication Model.

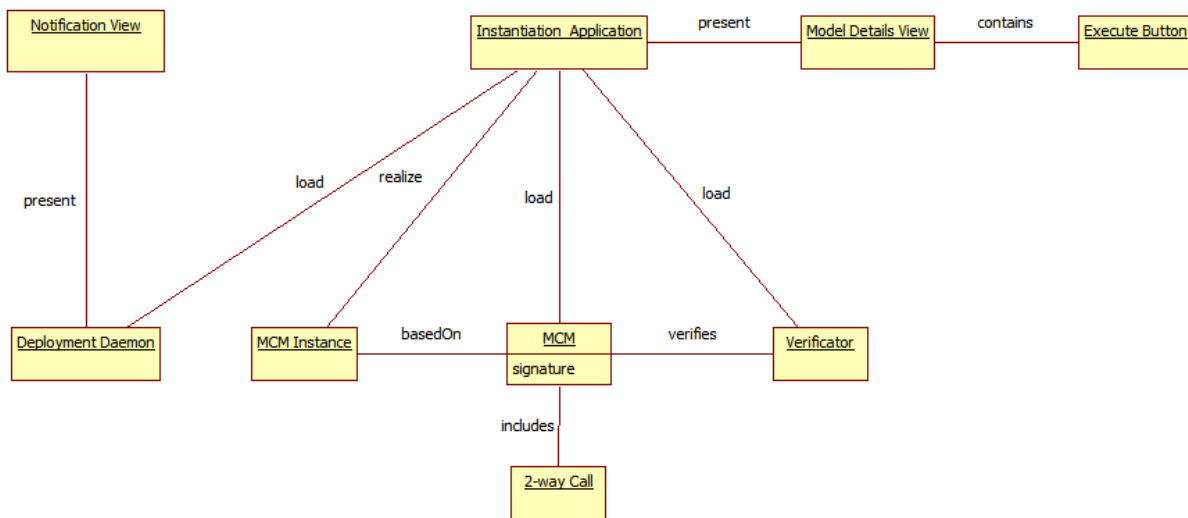


Figure 12

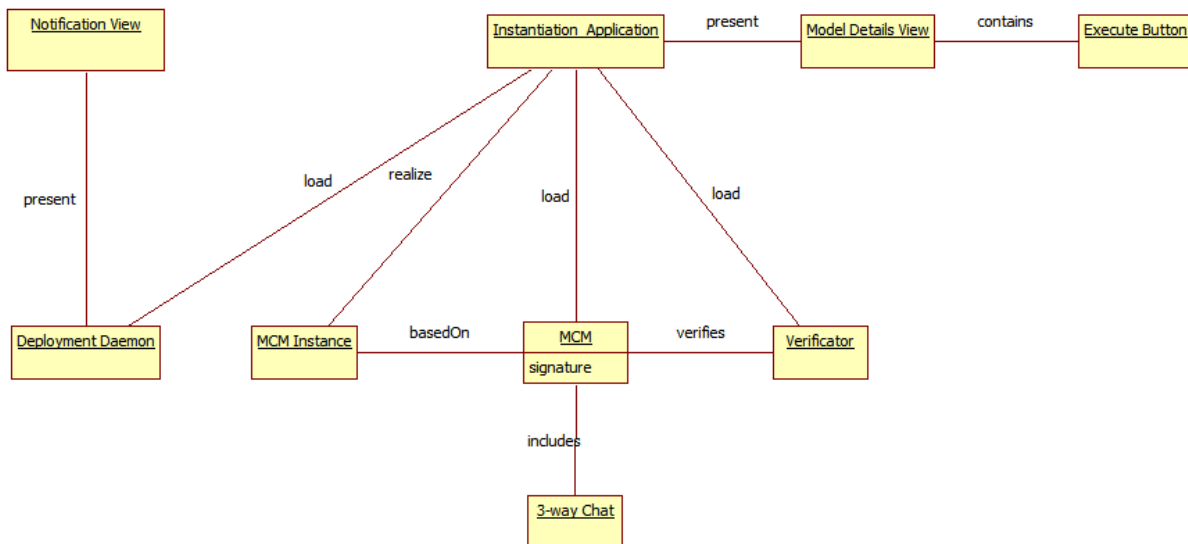


Figure 13

Figure 15 and Figure 17 present the Check or Specify Participants scenarios.

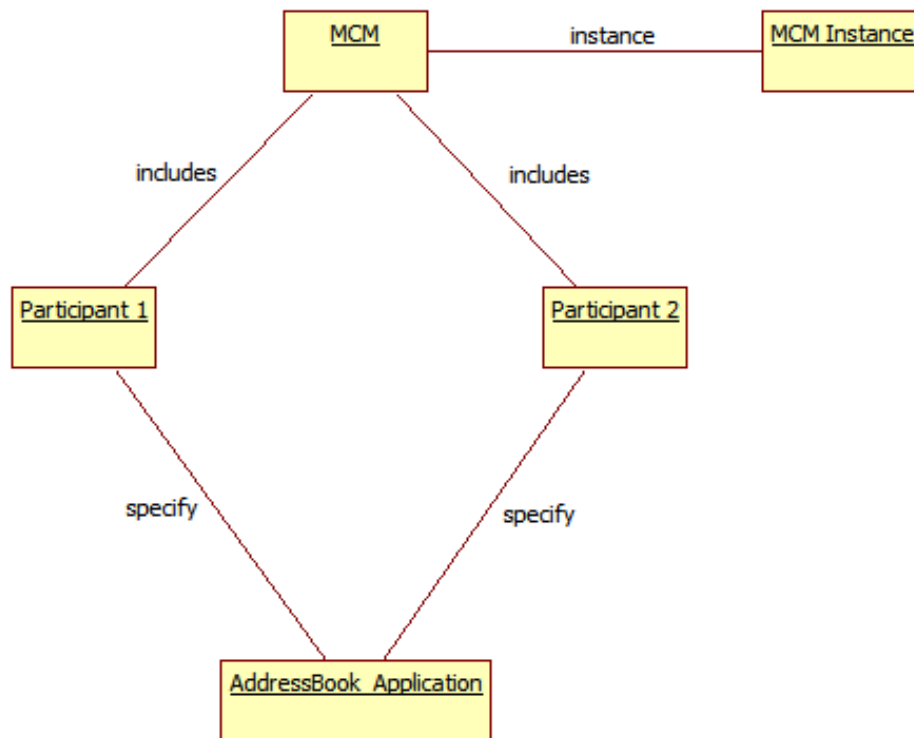


Figure 14

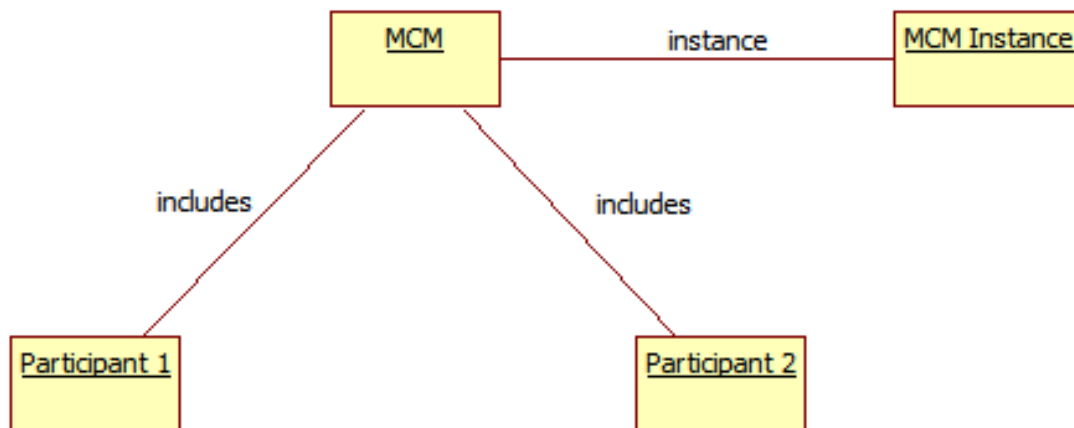


Figure 15

Figure 19 and Figure 21 present the N-way Call scenarios.

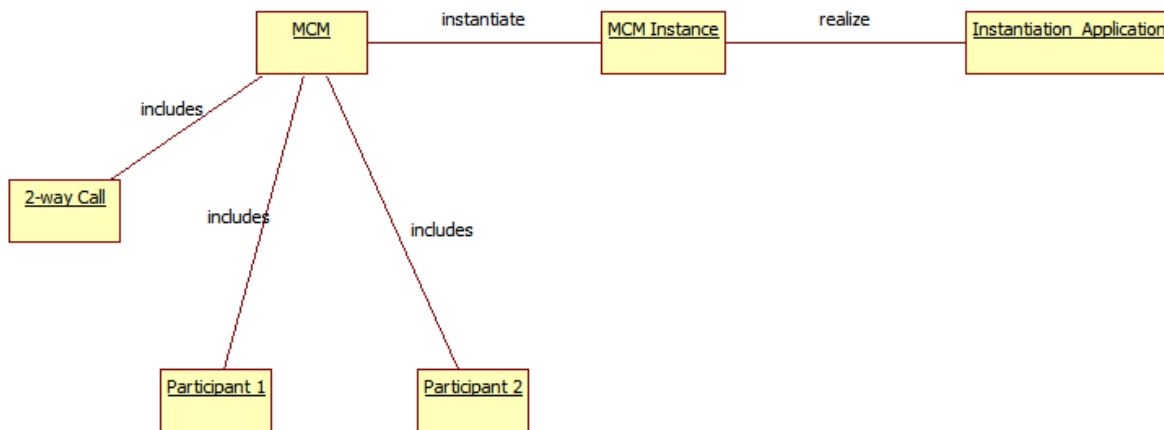


Figure 16

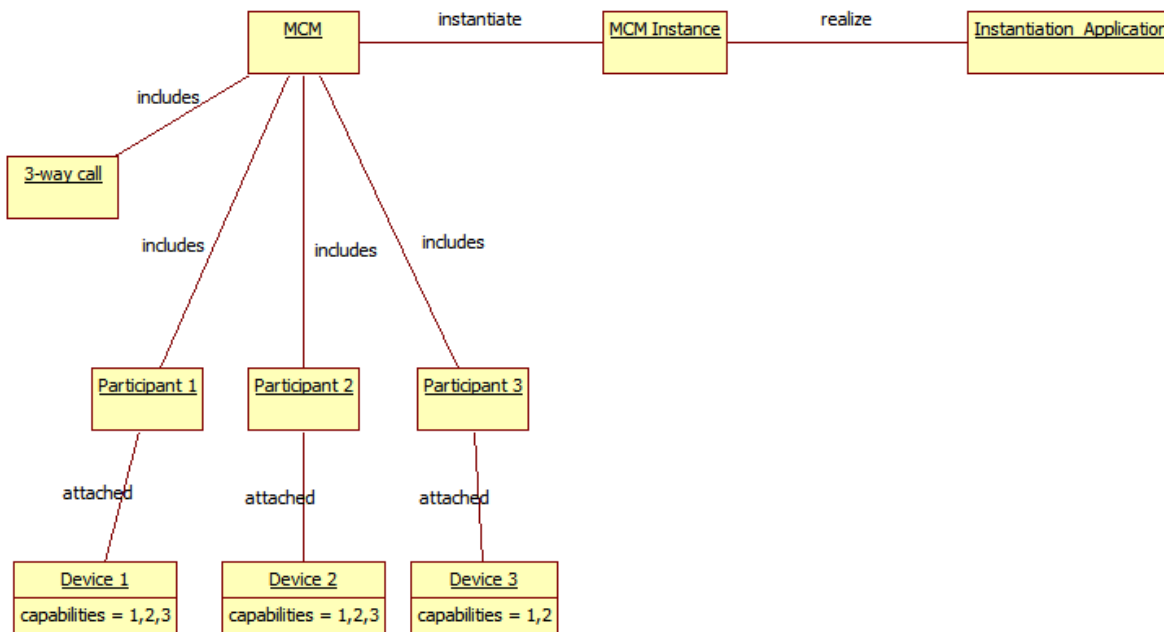


Figure 17

Figure 23 and Figure 25 present the N-way Chat scenarios.

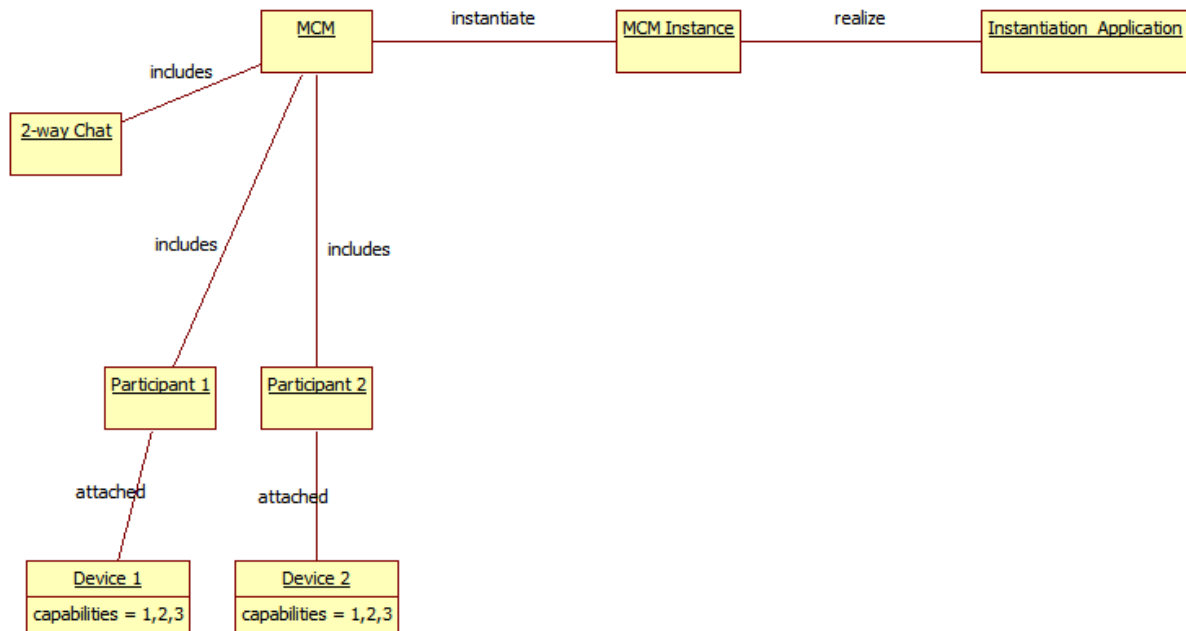


Figure 18

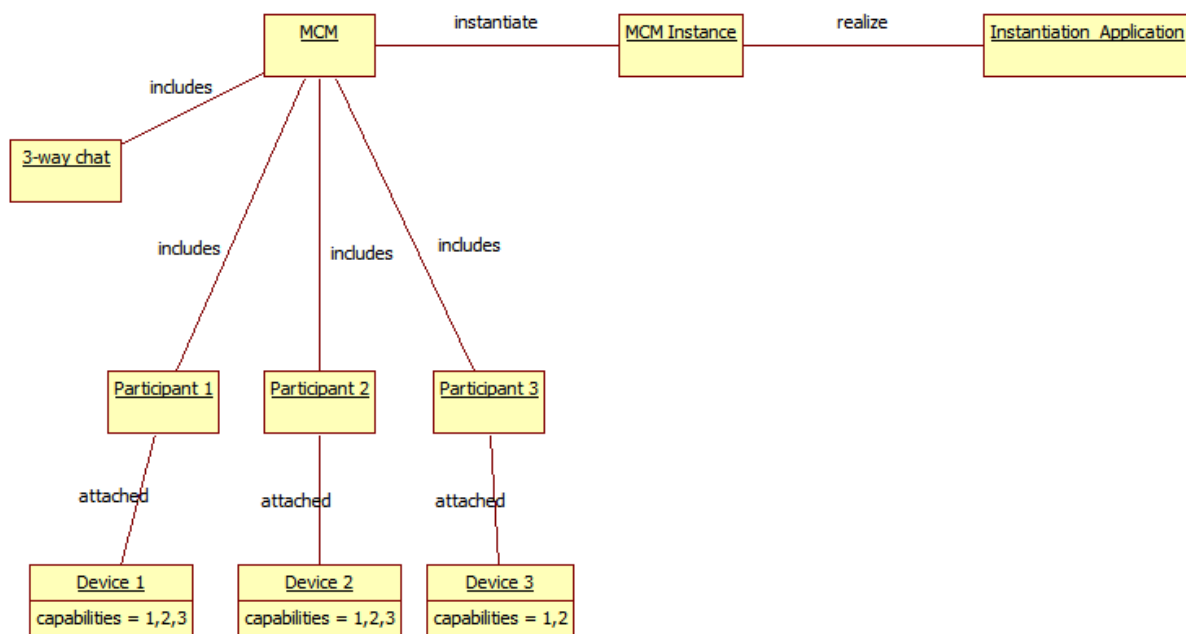


Figure 19

Figure 26 and Figure 27 present the Deploy Mobile Communication scenarios.

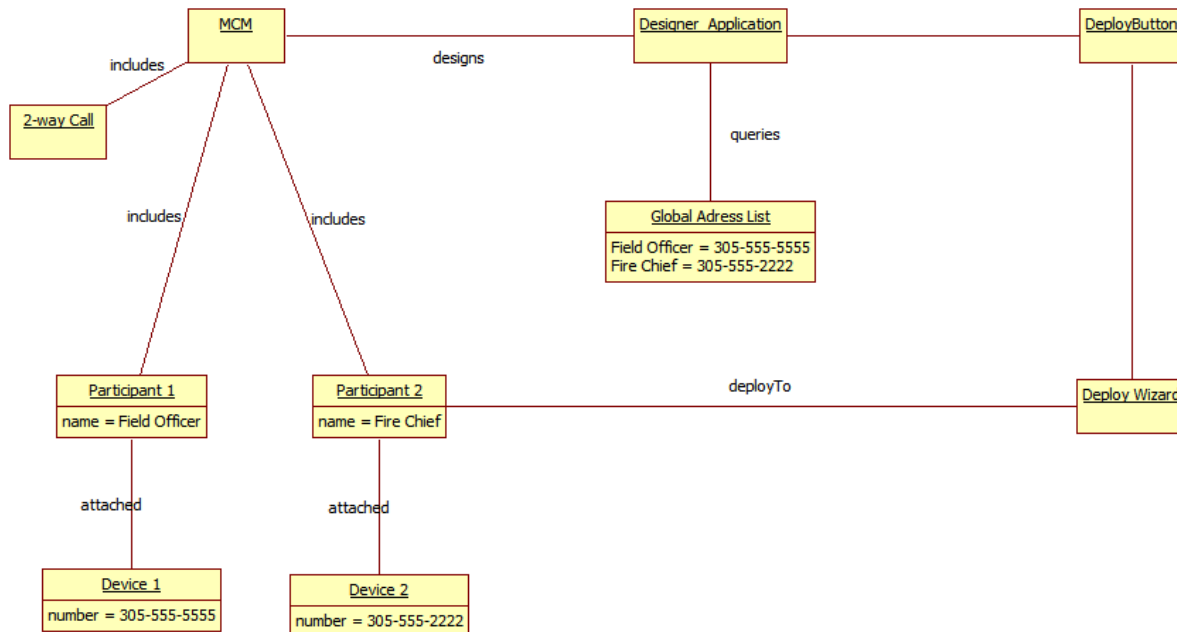


Figure 20

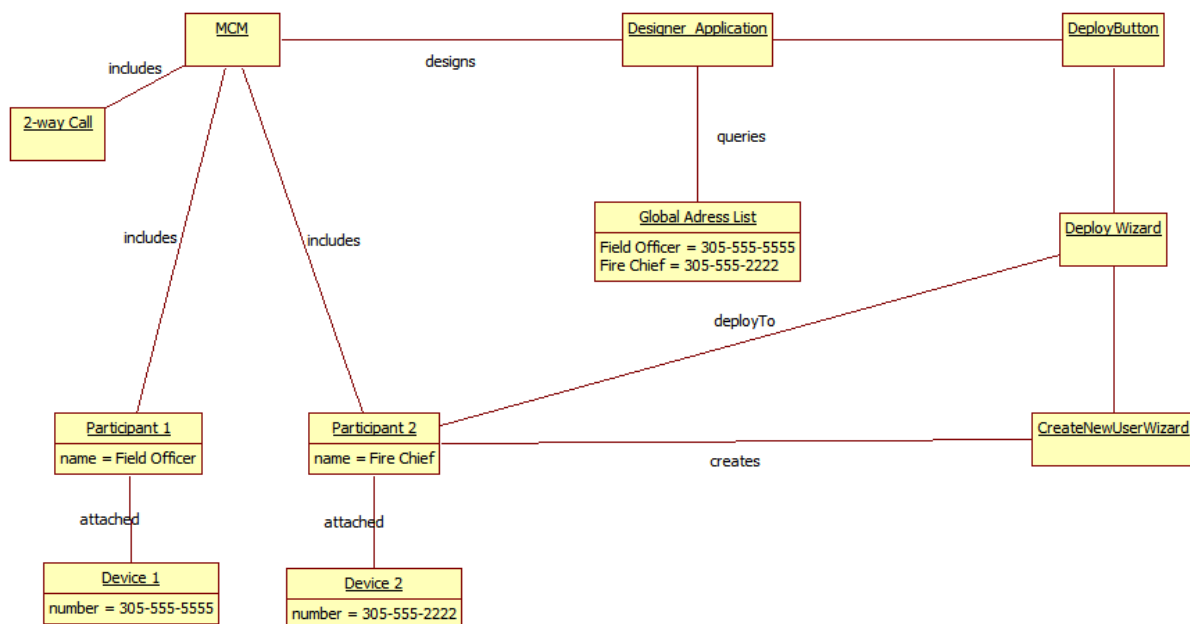


Figure 21

Figure 28 and Figure 29 present the Design Mobile Communication scenarios.

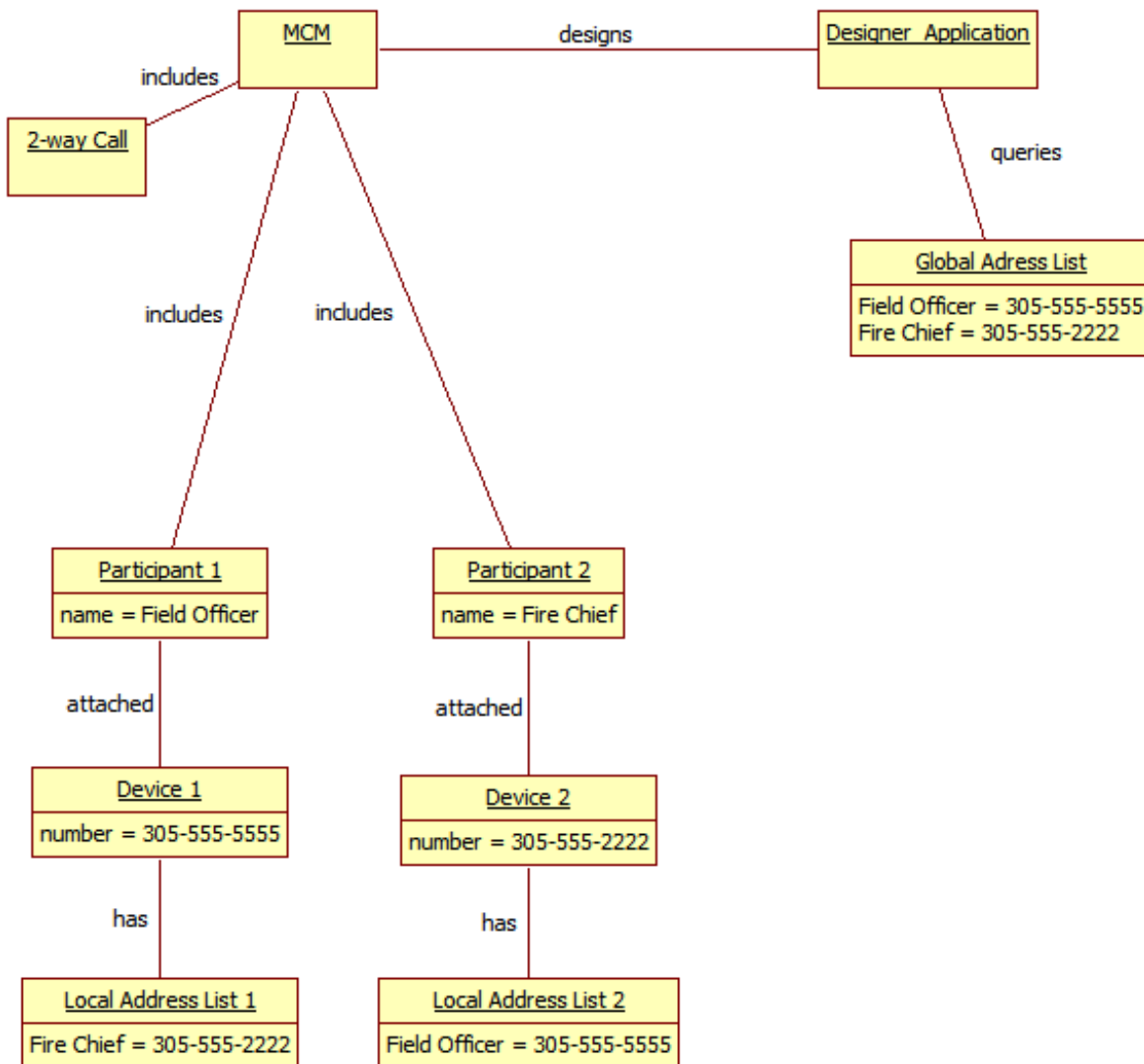


Figure 22

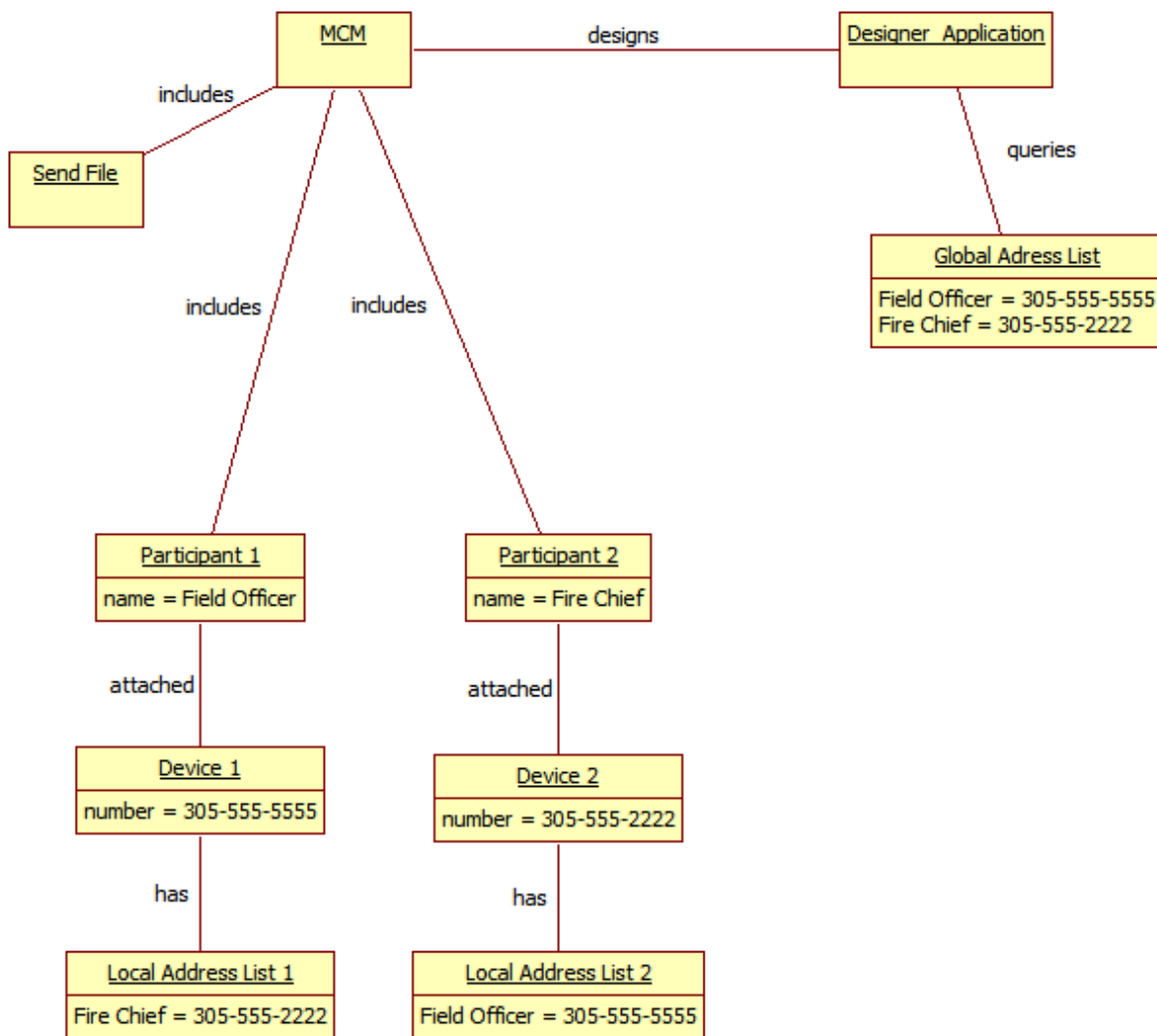


Figure 23

Figure 30 and Figure 31 present the Pull model scenarios.

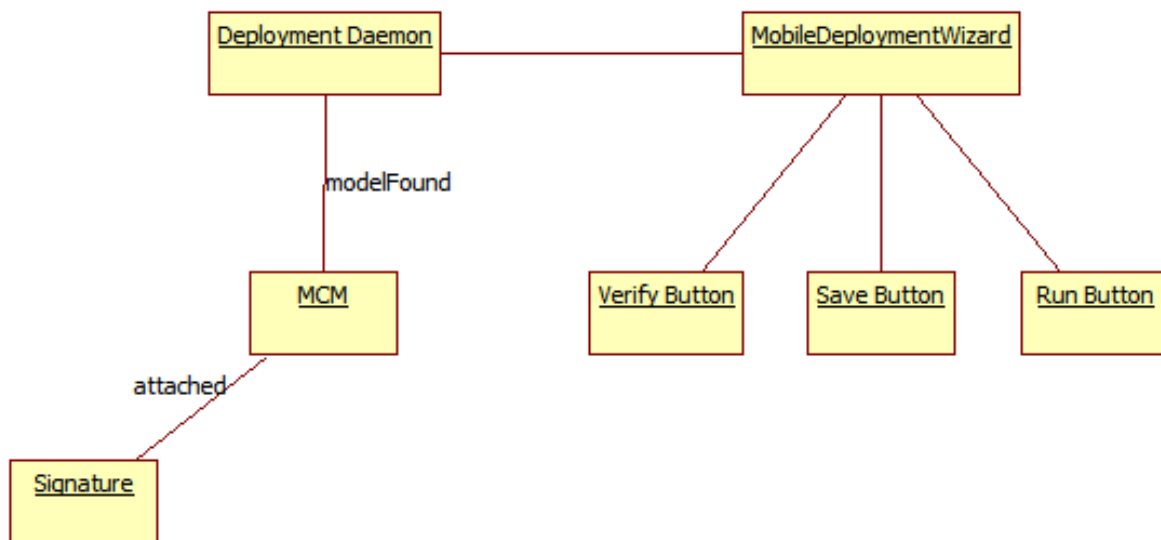


Figure 24

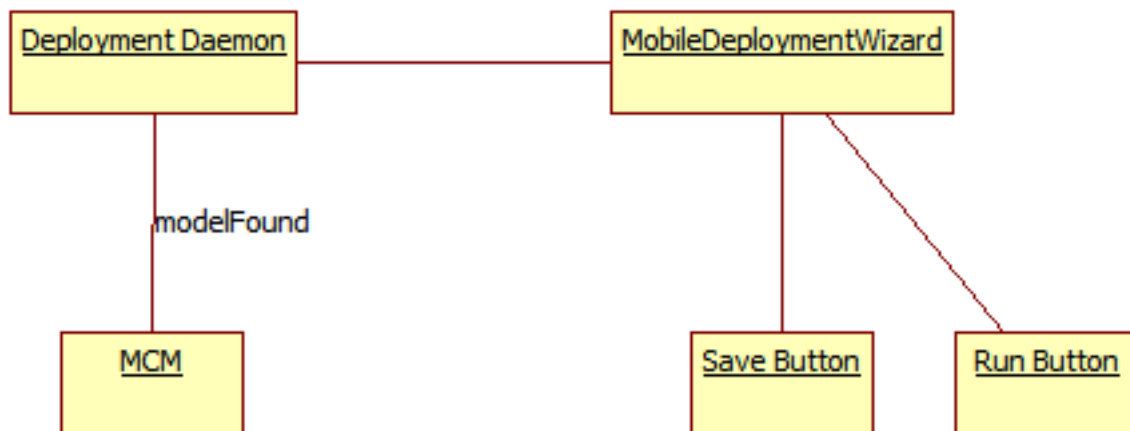


Figure 25

Figure 32 and Figure 33 present the Validate Model scenarios.

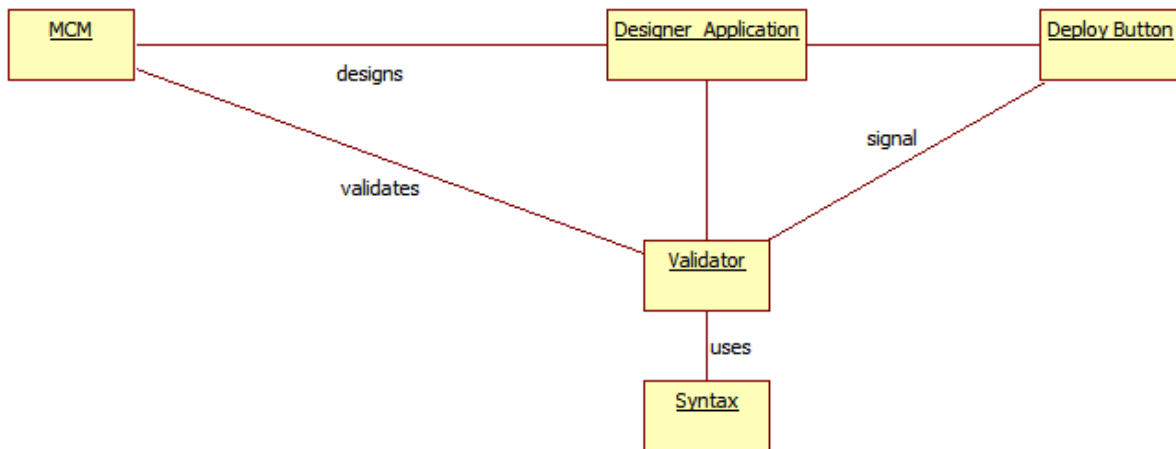


Figure 26

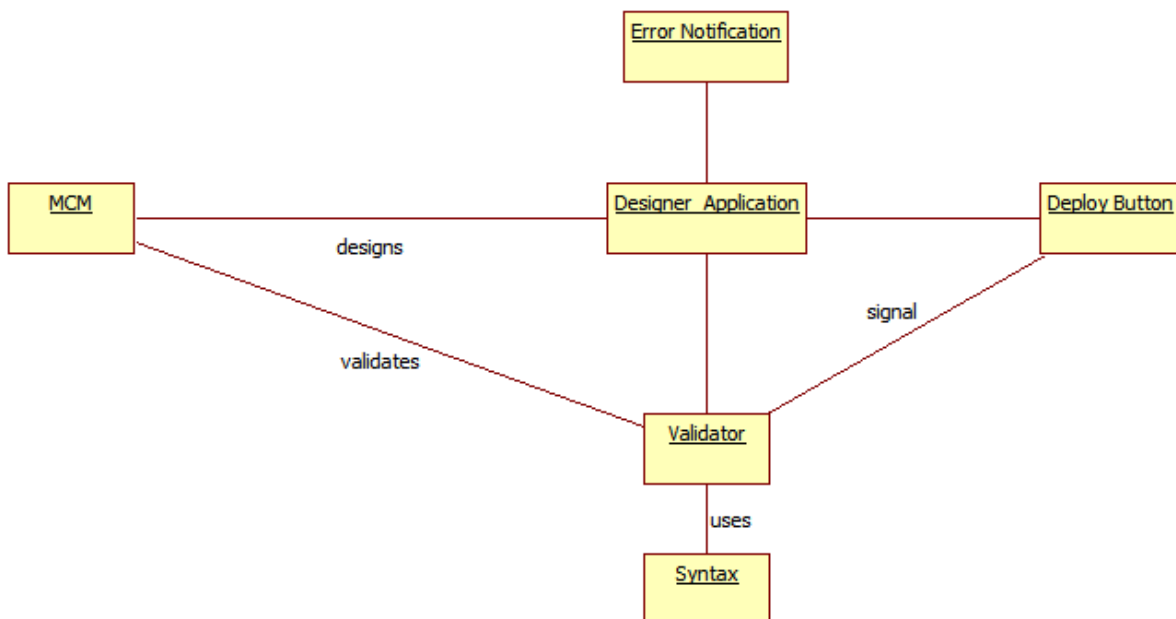


Figure 27

Appendix D – Sequence Diagrams for the analysis model

Figure 19 presents the sequence diagram for the Instantiate Mobile Communication Model use case scenarios.

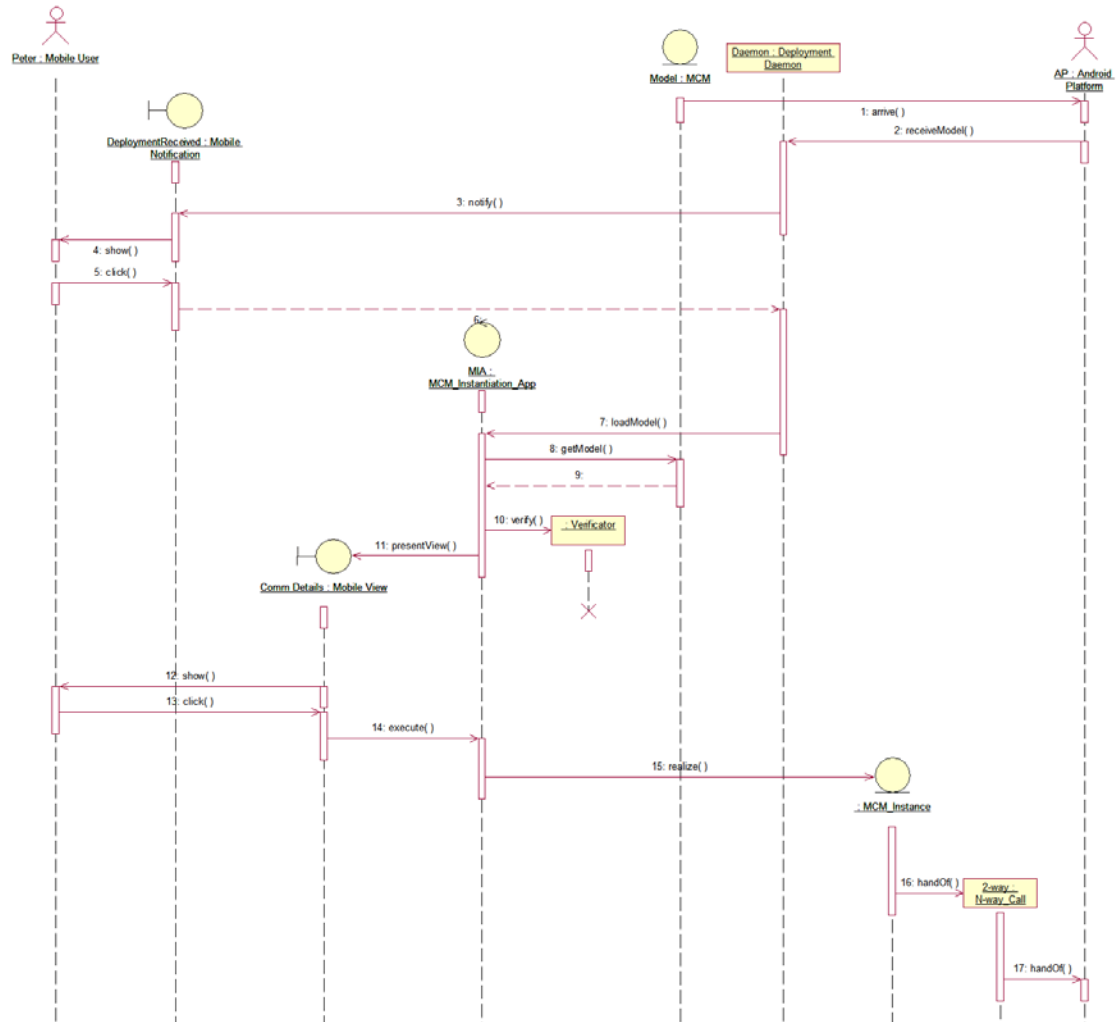


Figure 28

Figure 8 presents the sequence diagram for the Check Or Specify Participants use case scenarios.

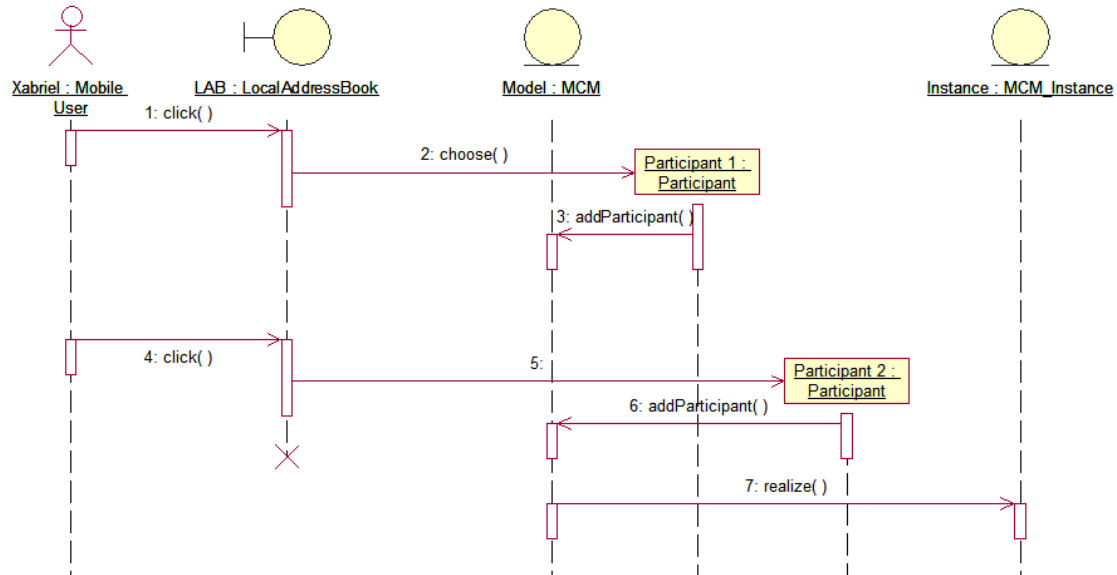


Figure 29

Figure 9 presents the sequence diagram for the N-Way Call use case scenarios.

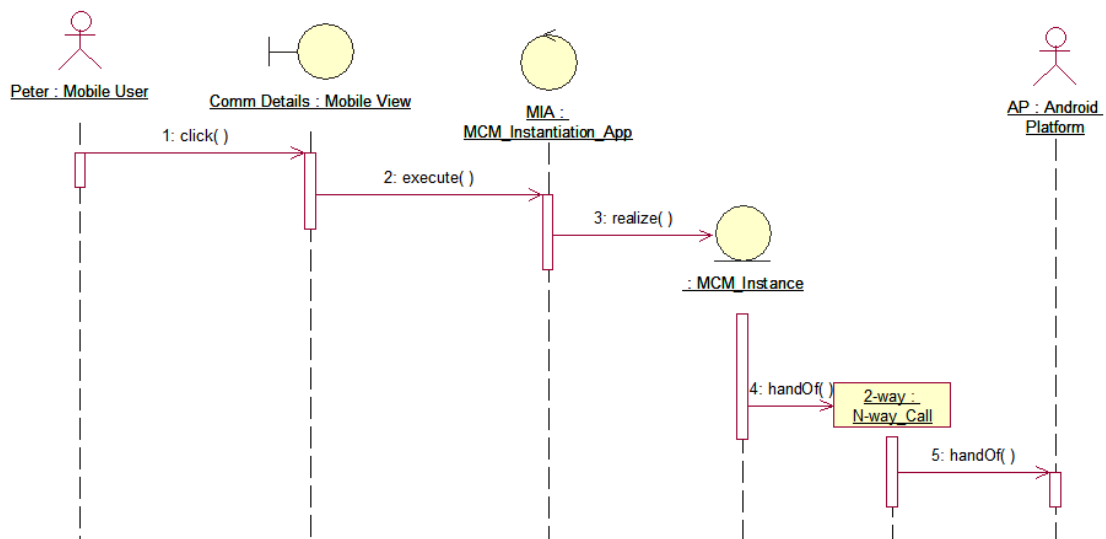


Figure 30

Figure 10 presents the sequence diagram for the N-way Chat use case scenarios.

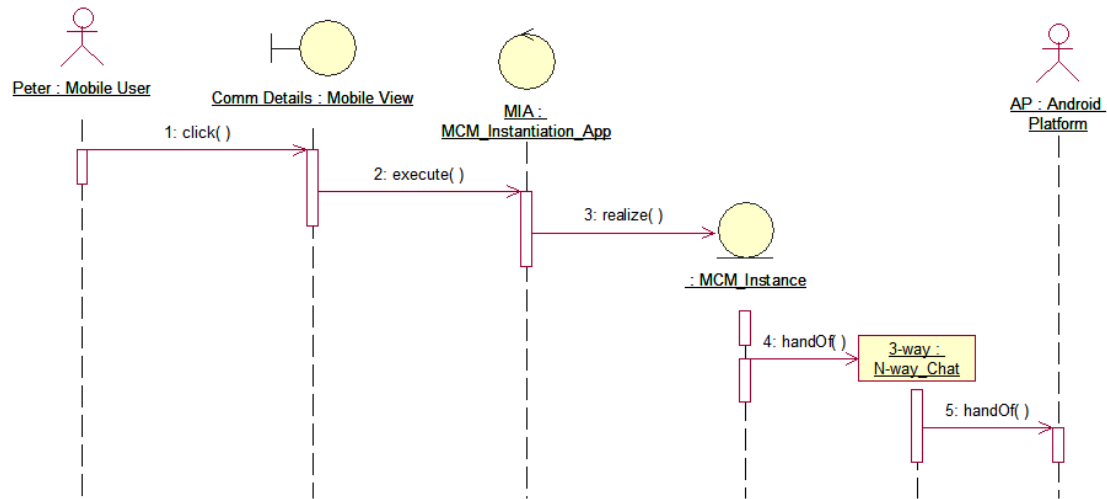


Figure 31

Figure 11 presents the sequence diagram for the Deploy Mobile Communication use case scenarios.

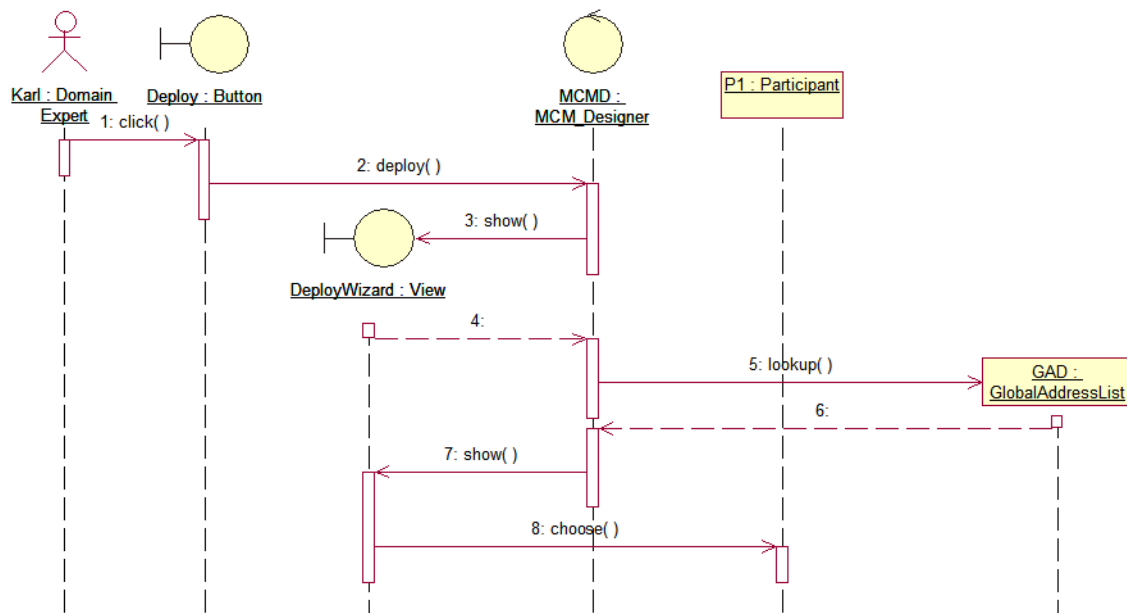


Figure 32

Figure 12 presents the sequence diagram for the Design Mobile Communication Model use case scenarios.

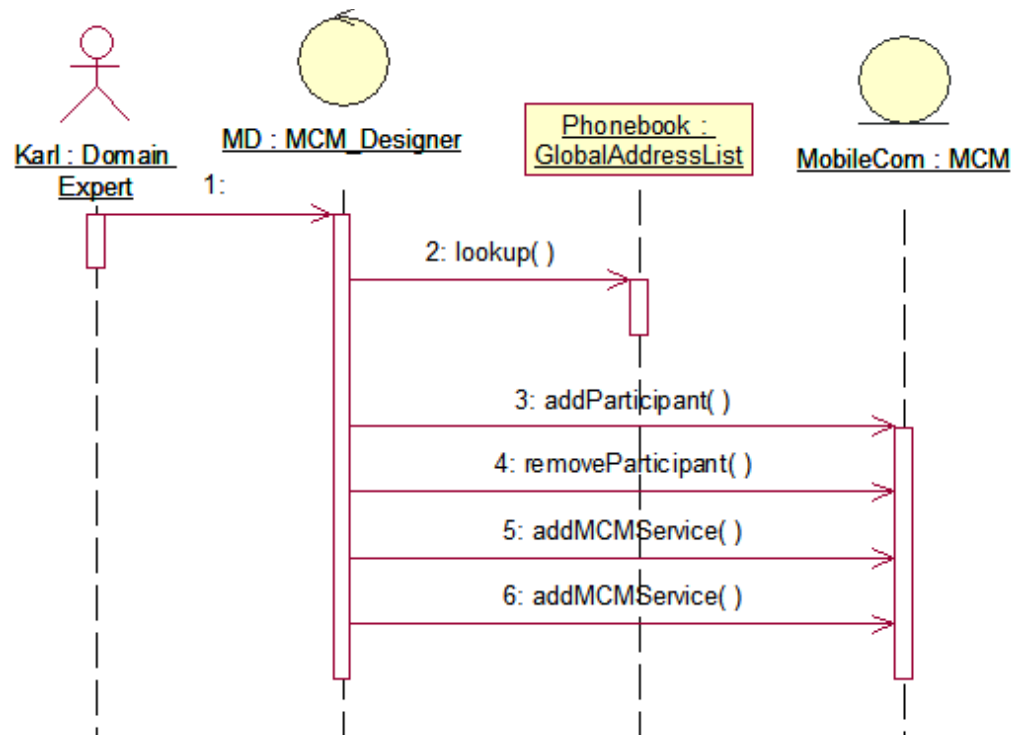


Figure 33

Figure 13 presents the sequence diagram for the Pull Model use case scenarios.

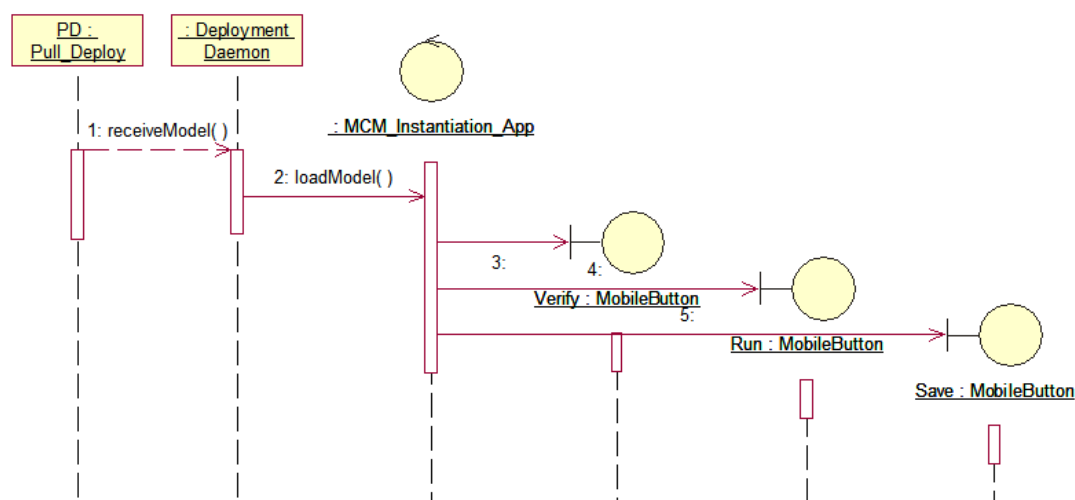


Figure 34

Figure 14 presents the sequence diagram for the Validate Model use case scenarios.

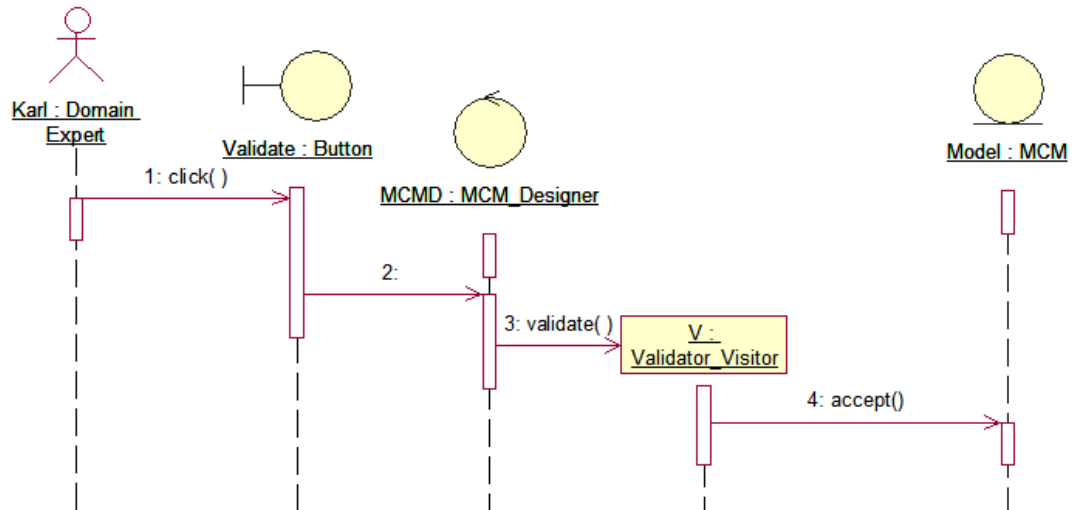


Figure 35

Appendix E – User interfaces

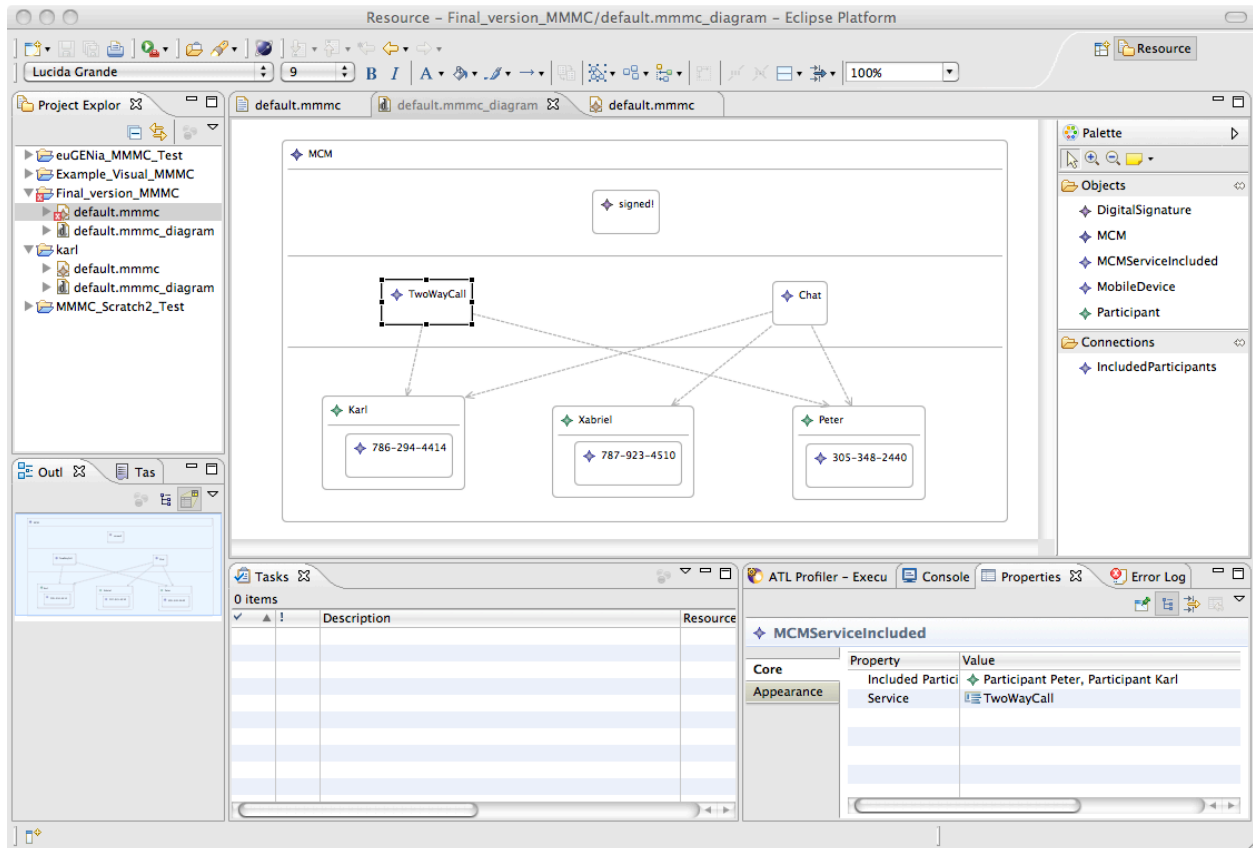


Figure 36

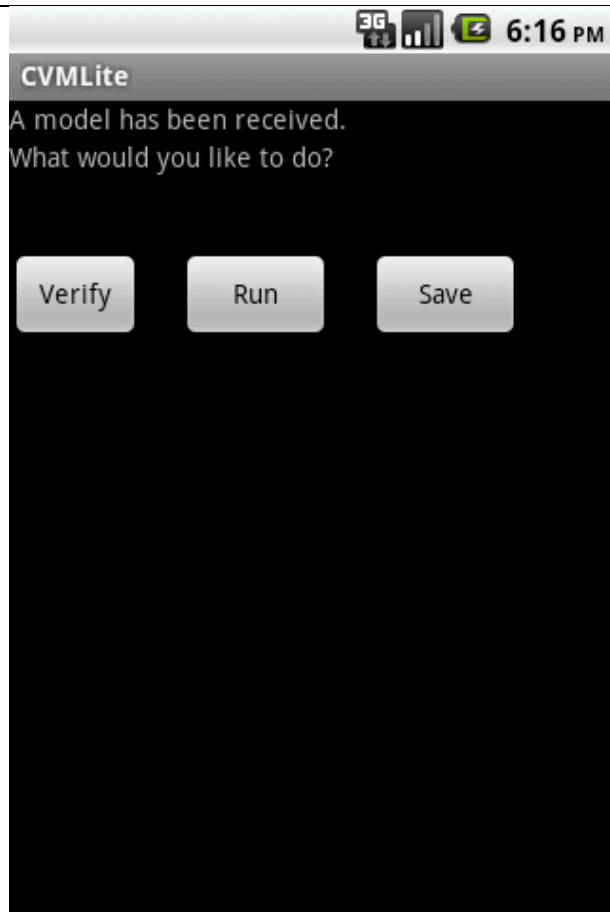


Figure 37



Figure 38

Appendix F – Detailed Class Diagrams

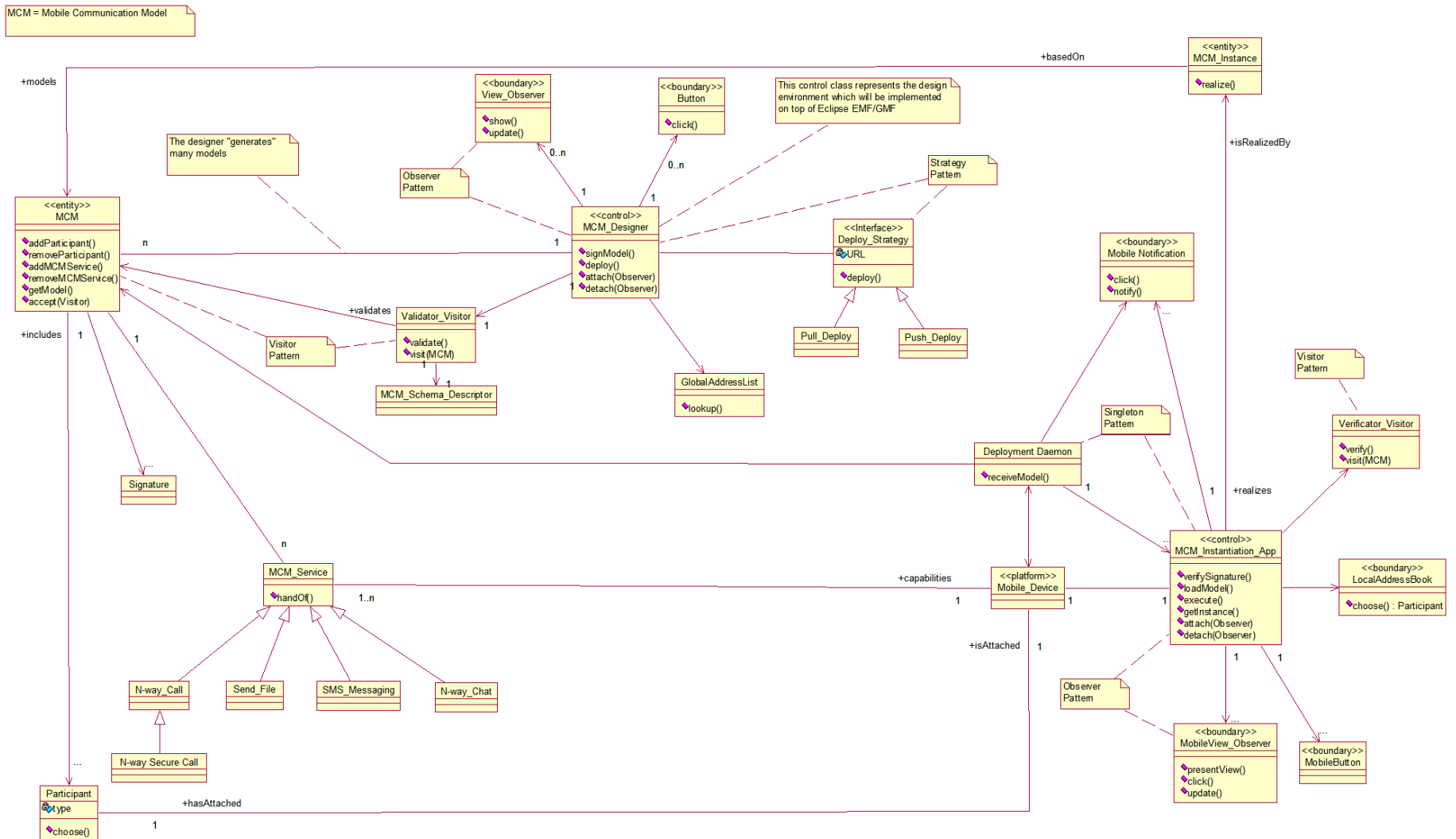


Figure 39

Appendix G – Class interfaces

As we automatically generated most of the code using Eclipse EMF and GMF, here we present the meta-model used for the generation of code in Figure 23, plus the code interfaces for the mobile platform in Figure 24.

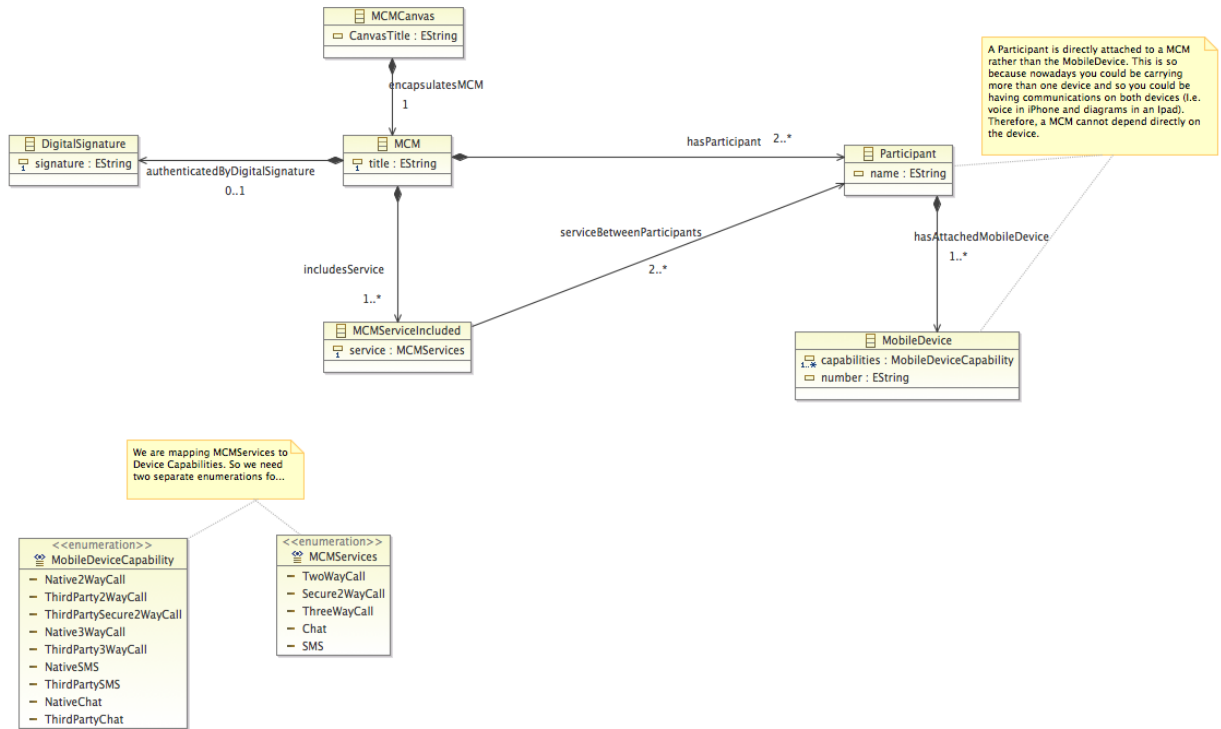


Figure 40

SE.java:

```
public abstract void onCreate(Bundle savedInstanceState);  
public abstract void runModel();  
public abstract int Com_2WayChat(String username);  
public abstract int Com_2WayCall(String number);  
public abstract String getModel();
```

RunModel.java:

```
public abstract void onCreate(Bundle savedInstanceState);  
public abstract int Com_2WayChat(String username);  
public abstract int Com_2WayCall(String number);
```

UserChoice.java:


```
public abstract void runModel();  
public abstract void onClick(View v);
```

Figure 41

Appendix H – Project Schedule

Project Leader Assignment Key

 = Xabriel J. Collazo-Mojica

 = Karl Morris

Start date: 09 / 07 / 2010

Start date: 12 / 04 / 2010

Current Week													X
Tasks	Weeks	09/07 to 09/14	09/14 to 09/21	09/21 to 09/28	09/28 to 10/05	10/05 to 10/12	10/12 to 10/19	10/19 to 10/26	10/26 to 11/02	11/02 to 11/09	11/09 to 11/16	11/16 to 11/23	11/23 to 12/04
	Project definition												
	Req. Elicitation												
	Req. Analysis												
	Software Design												
	Prototype Imp. and V&V												
	Tasks	Weeks	09/07 to 09/14	09/14 to 09/21	09/21 to 09/28	09/28 to 10/05	10/05 to 10/12	10/12 to 10/19	10/19 to 10/26	10/26 to 11/02	11/02 to 11/09	11/09 to 11/16	11/16 to 11/23
Current Week													X

Figure 42

Appendix I – Diary of meeting and tasks

Recurring meetings: Every Tuesday and Thursday.

Meeting notes: Friday, October 8

Present: Karl Morris, Xabriel Collaza-Mojica, Zhimin Yuan

Items Discussed:

- Breakdown of SSD and task assignments
- Proposed Architecture model and UML profile
- UML tools for diagram designs

Items for next meeting:

- UML Profile
- Generative Architecture
- Demonstration of Modeling Environment
- Begin compilation of SSD Draft

Next meeting scheduled for: TBD

Meeting notes: Friday, October 15

Present: Karl Morris, Xabriel Collaza-Mojica, Zhimin Yuan

Items Discussed:

- Mobile Architecture Metamodel Draft
- UML Profile
- CML review

Items for next meeting:

- UML Profile
- Generative Architecture
- Demonstration of Modeling Environment

Next meeting scheduled for: TBD

Meeting notes: Tuesday, October 19

Present: Karl Morris, Xabriel Collaza-Mojica, Zhimin Yuan

Items Discussed:

- Review refinement of Meta-models
- Refinement of tasks for second deliverable

Next meeting scheduled for: Thursday October 21.

Meeting notes: Thursday, October 21

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

Review refinement of Meta-models

Refinement of tasks for second deliverable

Next meeting scheduled for: Tuesday, October 26.

Meeting notes: Tuesday, October 26

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Review refinement of Meta-models
- Refinement of tasks for second deliverable

Next meeting scheduled for: Thursday October 28.

Meeting notes: Thursday, October 28

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- UML tool capabilities
- Updates given on progress with SSD

Next meeting scheduled for: Monday, November 1.

Meeting notes: Monday, November 1

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Review refinement of Meta-models
- Refinement of tasks for second deliverable

Next meeting scheduled for: Tuesday, November 2.

Meeting notes: Thursday, November 4

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Compilation of SSD

Next meeting scheduled for: Monday, November 8.

Meeting notes: Monday, November 8

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Conversion of UML documents from Star UML into Rational Rose
- Compilation of final SSD

Next meeting scheduled for: TBD

Meeting notes: Tuesday, November 16

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Completed format of mobile communication model format

Next meeting scheduled for: Tuesday, November 23

Meeting notes: Tuesday, November 16

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Demo of design environment
- Prototype of mobile instantiation application presented

Next meeting scheduled for: Tuesday, November 30

Meeting notes: Tuesday, November 30

Present: Karl Morris, Xabriel Collaza-Mojica

Items Discussed:

- Demo of mobile instantiation application
- Modeling environment finalized

Next meeting scheduled for: TBD