Laboratory for Advanced Software Systems
University of Luxembourg

iCrash :
A Crisis Management Case Study
Messir Analysis Document
- v 1.4 -

(*Report type: Specification*)

Tuesday 30th May, 2017 - 20:20

Contents

List of Figures

Listings

Chapter 1

Introduction

1.1 Overview

iCrash is a simple system dedicated to any person who wants to inform of a car crash crisis situation in order to allow for crisis handling. At anytime and anywhere, anyone can be the witness or victim of a car crash and might be in a situation allowing for alerting this crisis. The *iCrash* system has for objectives to support crisis declaration and secure administration and crisis handling by the *iCrash* professional users.

1.2 Purpose and recipients of the document

This document is an analysis document complying with the Μεσσιρ methodology [1]. Its intent is to provide an example of a precise specification of the functional properties of the *iCrash* system.

The recipients of this document are:

- the *iCrash* system's buyer company (ABC): this document is used as a contractual document jointly with any other document considered as useful (as requirement elicitation document, ...) in order to have a higher degree of precision in requirement description. It is also used as a basis document for the *iCrash* system validation using specification based testing.
- the *iCrash* system development company (ADC) is expected to use this document as the basis for development (mainly design, implementation, maintenance). It is also used for verification and validation using test plans defined using the analysis models described in this document and according to the Μεσσιρ methodology.

1.3 Application Domain

The *iCrash* system belongs to the Crisis Management Systems Domain. It is a system dedicated to crisis professional and non professional end users. It has to be considered as an autonomous and external service for the society. It is not an institutional system certified and guaranteed by any governmental entity and thus, must be used with caution.

1.4 Definitions, acronyms and abbreviations

N.A.

1.5 Document structure

The document structure is designed to be coherent with the Messir methodology [1]. Section 2 provides a general description of the system purpose, its users, its environment and some general non functional requirements. A more detailed description of the non functional requirements, if any, are provided in section **??**. The system operation triggered by events sent by the external actors belonging to the environment are described in Section 3. The *iCrash* concepts used to represent the any persistent or transient information is given in Section 4. The precise specification of the system operations in term of system's state changes, events sent together with the constraints on the allowed sequences of system operations are described in Section 5.

Chapter 2

General Description

In the context of the Messir method, the information provided in this section is intended to present the system for which the Messir analysis is provided. The content of this section is made accordingly to the requirements elicitation document that might have been done during the project but also adapted coherently in order to be an abstract introduction to the Messir analysis.

2.1 Domain Stakeholders

All stakeholders of the system are detailed in this section. After a brief description of a stakeholder, its objectives are first stated. Thereafter, the responsibilities of the stakeholder are detailed which help to achieve the stakeholder objectives to a certain degree. While the objectives characterize the general problems addressed by the *iCrash* system, the responsibilities describe concrete actions that are expected from a stakeholder. Some of these responsibilities can be traced looking at the use case described in Section B.1,and hence must be supported by the *iCrash* system. All stakeholders listed in this section have an interest in the system or are affected by the system in some way, but only a subset of the stakeholders are directly involved in the use cases described. Let us remind that use case diagrams or descriptions are not Messir analysis phase mandatory outputs. They are proposed as informal means to help understanding the semantics of the system specification made of the mandatory analysis models, which provide a complete executable specification.

2.1.1 Communication Company

A Communication Company is a company that has the capacity to ensure communication of information between its customers and the *iCrash* system. The objectives of a Communication Company are:

- to be able to deliver any SMS sent by any human to the *iCrash* 's phone number.
- to be able to transmit SMS messages from the ABC company that owns the *iCrash* system to any human having an SMS compatible device accessible using a phone number.

In order to achieve these objectives, the responsibilities of a Communication Company are:

- ensure confidentiality and integrity of the information sent by a human to the *iCrash* system or from the system to a human.
- to be always available and reliable.

2.1.2 Humans

A human is any person who considers himself related to a car crash either as a witness, a victim or an anonymous person. The objectives of a human are:

- inform the *iCrash* system about the crisis situation he detected.
- be sure that the ABC company has been informed about the situation.
- to be informed about the situation of the crisis he his related to as a victim or witness.

In order to achieve these objectives, the responsibilities of a human are:

- to provide has much details as possible concerning the crisis to the ABC company.
- to declare a crisis only if the crisis is real.
- to have access to the SMS compatible communication device he used to communication with the *iCrash* system.

2.1.3 Coordinators

A coordinator is a employee of the ABC company being responsible of handling one or several crisis. The objectives of a coordinator are:

- to securely monitor the existing alerts and crisis.
- to securely manage alerts and crisis until their termination.

In order to achieve these objectives, the responsibilities of a coordinator are:

- to be capable to determine how an alert received should be considered.
- to be available to react to requests to handle alerts and crisis.
- to be autonomous in handling crisis and to report on its handling.
- to be able to decide when a crisis or an alert can be closed.
- to know its system identification information for secure usage of the system.

2.1.4 Administrator

An administrator is a employee of the ABC company being responsible of administrating the *iCrash* system. The objectives of an administrator are:

- to add or delete coordinator actors from the system and its environment.

In order to achieve these objectives, the responsibilities of a coordinator are:

- know the company employees that can be coordinators and that have access to the system.
- to know its system identification information for secure usage of the system.
- to know the security policy of the ABC company.
- to communicate the coordinators their identification information for secure system usage.

2.1.5 Creator

Any system has a `Creator` stakeholder which is a technician who is installing the *iCrash* system on the targeted deployment infrastructure.

The objectives of a `Creator` are:

- to install the *iCrash* system
- to define the values for the initial system's state
- to define the values for the initial system's environment
- to ensure the integration of the *iCrash* system with its initial environment

In order to achieve these objectives, the responsibilities of a `Creator` are:

- provide the necessary data to the *iCrash* system for its initialization.

2.1.6 Activator

An `activator` is a logical representation of the active part the *iCrash* system. It represents an implicit stakeholder belonging to the system's environment that interacts with the *iCrash* system autonomously without the need of a external entity. It is usually used for representing time triggered functionalities.

The objectives of a `activator` are:

- to communicate the current time to the system
- to notify the administrator that some crisis are still pending for a too long time.

In order to achieve these objectives, the responsibilities of a `activator` are:

- to know the current universal time
- to send the messages to the system according to the time constraints specifically defined for it.

2.2 System's Actors

The objective of this section is not to provide the full requirement elicitation document in this section but to reuse a part of this document to provide a informal introduction to the Messir specification of the system under development. The use case model is made of a use case diagrams modelling abstractly and informally the actors and their use cases together with a set of use cases descriptions. In addition, those diagrams and description tables are adapted to the Messir specification since actor and messages names together with parameters are partly adapted to be consistent with the specification identifiers (see [1] for more details).

Among all the stakeholders presented in the previous section, we can determine five types of direct actors[1]:

- `actComCompany`: for the Communication Company stakeholder.
- `actAdministrator`: for the Administrator stakeholder.
- `actCoordinator`: for the Coordinators stakeholders.
- `actActivator`: for the Activator stakeholder.
- `actMsrCreator`: for the Creator stakeholder.

In addition to those system actors, we can add five other types of actors related to the system's ones. Those five actors are grouped into two categories:

- *Indirect actors*
  - *Witness*: for any human that is a witness of a car crash
  - *Victim*: for any human that is a victim of a car crash
  - *Anonymous*: for any human that want to inform about a car crash while staying anonymous.
- *Abstract actors*
  - `actHuman`: represent abstractly any kind of human being actor wanting to communicate with the ABC system in the context of a car crash.
  - `actAuthenticated`: for the logical Activator stakeholder.

2.3 Use Cases Model

This section contains the use cases elicited during the requirements elicitation phase. The use cases are textually described as suggested by the Messir method and inspired by the standard Cokburn template [2].

2.3.1 Use Cases

2.3.1.1 summary-suDeployAndRun

The goal is to install the iCrash system on its infrastructure and to exploit its capacities related to the secure administration and efficient handling of car crash situations depending on alerts received.

[1] The naming conventions in Messir propose to start each type name by lowercase letters indicating the meta model type used (i.e. act for actors, ct for class type, ... In addition to ease the reading it makes the translational semantics into Prolog code more straightforward.

Use-Case Description	
Name	suDeployAndRun
Scope	system
Level	summary
Primary actor(s)	
1	actAdministrator[active]
Secondary actor(s)	
1	actMsrCreator[active]
2	actCoordinator[active, multiple]
3	actActivator[proactive]
4	actComCompany[active]
Goal(s) description	
The goal is to install the iCrash system on its infrastructure and to exploit its capacities related to the secure administration and efficient handling of car crash situations depending on alerts received.	
Reuse	
1	oeCreateSystemAndEnvironment [1..1]
2	ugAdministrateTheSystem [1..*]
3	suGlobalCrisisHandling [1..*]
4	oeSetClock [1..*]
5	oeSollicitateCrisisHandling [0..*]
6	oeAlert [1..*]
Protocol condition(s)	
1	the iCrash system has never been deployed and used
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the iCrash system has been created and has handled the crisis situations for which it received alerts through the communication company.
Main Steps	
a	the actor actMsrCreator executes the oeCreateSystemAndEnvironment use case
b	the actor actAdministrator executes the ugAdministrateTheSystem use case
c	the actor actComCompany executes the oeAlert use case
d	the actor actActivator executes the oeSetClock use case
e	the actor actActivator executes the oeSollicitateCrisisHandling use case
f	the actor actCoordinator executes the suGlobalCrisisHandling use case
Steps Ordering Constraints	
1	step (a) must be always the first step.
2	step (f) can be executed by different actCoordinator actors.
3	if (e) then previously (d).

Figure 2.1 shows the use case diagram for the suDeployAndRun summary use case

2.3.1.2 summary-suGlobalCrisisHandling

the actCoordinator's goal is to monitor the alerts received and the corresponding crisis in order to act as necessary to handle the crisis.

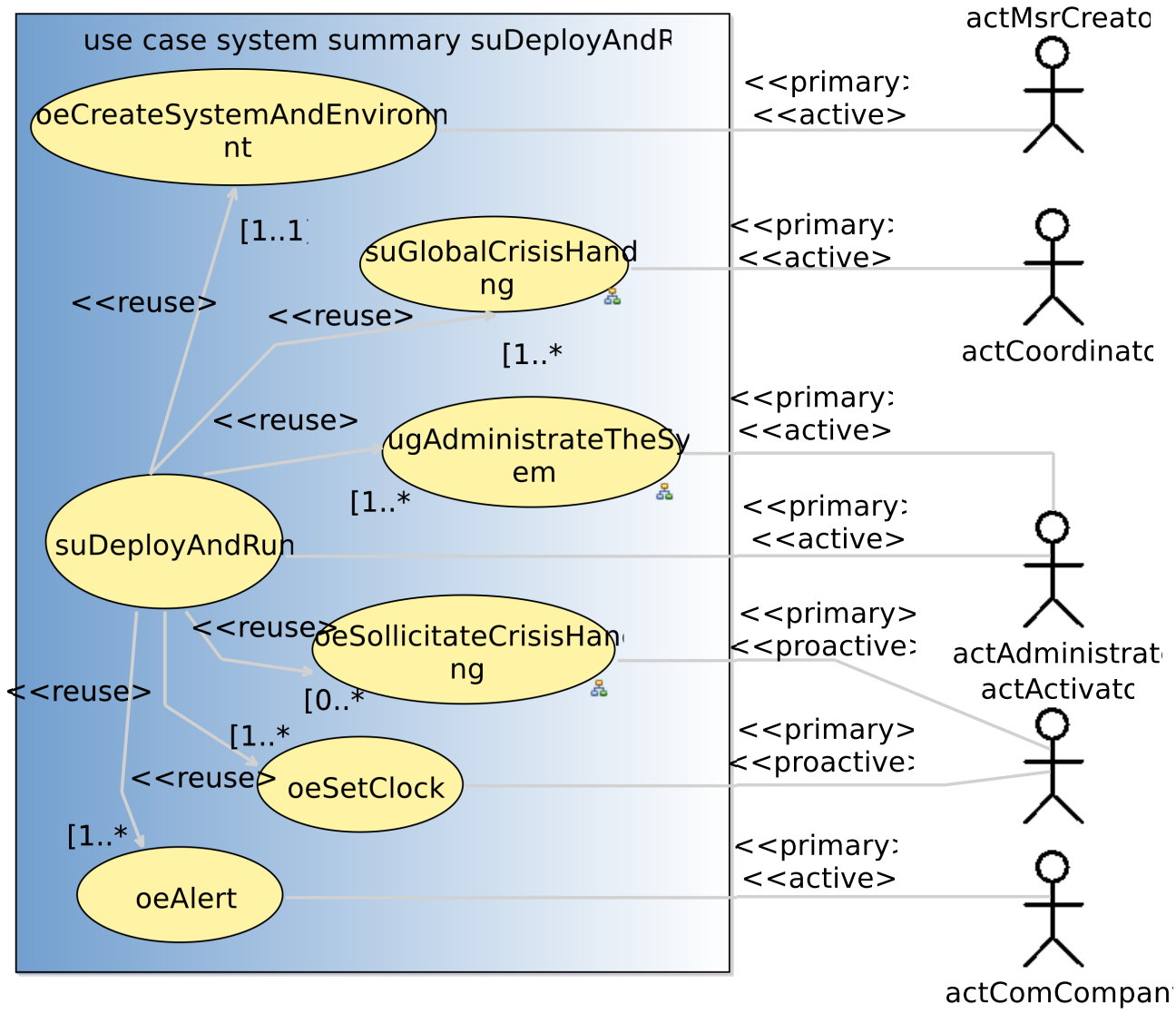

Figure 2.1: suDeployAndRun summary use case

Use-Case Description	
Name	suGlobalCrisisHandling
Scope	system
Level	summary
Primary actor(s)	
1	actCoordinator[active]
Goal(s) description	
the actCoordinator's goal is to monitor the alerts received and the corresponding crisis in order to act as necessary to handle the crisis.	
Reuse	
1	ugSecurelyUseSystem [1..*]
2	ugMonitor [1..*]
3	ugManageCrisis [1..*]
Protocol condition(s)	
1	the iCrash system has been deployed
2	the coordinator actor involded in the use case has been declared by the actor actAdministrator
Pre-condition(s)	
1	none
Main post-condition(s)	
1	modifications have been made by the coordinator on existing alerts or crisis OR the coordinator requested an updated status on existing alerts or crisis.
Main Steps	
a	the actor actCoordinator executes the ugSecurelyUseSystem use case
b	the actor actCoordinator executes the ugMonitor use case
c	the actor actCoordinator executes the ugManageCrisis use case
Steps Ordering Constraints	
1	steps (a) (b) and (c) executions are interleaved (steps (b) and (c) have their protocol constrained by steps of (a)).
2	steps (a) (b) and (c) can be executed multiple times.

Figure 2.2 shows the use case diagram for the suGlobalCrisisHandling user goal use case

2.3.1.3 usergoal-ugAdministrateTheSystem

the actAdministrator's goal is to follow an identification procedure to be allowed to add or delete the necessary crisis coordinators that will be granted the responsibility to handle alerts and crisis.

Use-Case Description	
Name	ugAdministrateTheSystem
Scope	system
Level	usergoal
Primary actor(s)	
1	actAdministrator[active]
Goal(s) description	
the actAdministrator's goal is to follow an identification procedure to be allowed to add or delete the necessary crisis coordinators that will be granted the responsibility to handle alerts and crisis.	

continues in next page . . .

... Use-Case Description table continuation

Reuse	
1	ugSecurelyUseSystem [1..*]
2	oeAddCoordinator [1..*]
3	oeDeleteCoordinator [0..*]
4	oeRankDownCoordinator [0..*]
5	oeAddPointOfInterest [1..*]
6	oeEditPointOfInterest [0..*]
7	oeDeletePointOfInterest [0..*]
Protocol condition(s)	
1	the iCrash system has been deployed
Pre-condition(s)	
1	none
Main post-condition(s)	
1	modifications have been made to the system and its environment concerning existing or new coordinators. modifications have been made to the system concerning existing or new Points of interest.
Main Steps	
a	the actor actAdministrator executes the ugSecurelyUseSystem use case
b	the actor actAdministrator executes the oeAddPointOfInterest use case
c	the actor actAdministrator executes the oeEditPointOfInterest use case
d	the actor actAdministrator executes the oeDeletePointOfInterest use case
e	the actor actAdministrator executes the oeAddCoordinator use case
f	the actor actAdministrator executes the oeDeleteCoordinator use case
g	the actor actAdministrator executes the oeRankDownCoordinator use case
Steps Ordering Constraints	
1	steps (a) (b) (c) (d) (e) (f) and (g) executions are interleaved (steps (b) (c) (d) (e) (f) and (g) have their protocol constrained by steps of (a)).
2	steps (a) (b) (c) (d) (e) (f) (g) can be executed multiple times.

Figure 2.3 shows the use case diagram for the ugAdministrateTheSystem user goal use case

2.3.1.4 usergoal-ugManageCrisis

The goal is to do an action that makes the handling of a crisis or an alert progress.

USE-CASE DESCRIPTION	
Name	ugManageCrisis
Scope	system
Level	usergoal
Primary actor(s)	
1	actCoordinator[active]
Goal(s) description	
The goal is to do an action that makes the handling of a crisis or an alert progress.	
Reuse	
1	oeValidateAlert [0..*]
2	oeSetCrisisStatus [0..*]

continues in next page ...

... Use-Case Description table continuation

3	oeSetCrisisHandler [0..*]
4	oeReportOnCrisis [0..*]
5	oeCloseCrisis [0..*]
6	oeInvalidateAlert [0..*]
Protocol condition(s)	
1	the iCrash system has been deployed
Pre-condition(s)	
1	none
Main post-condition(s)	
1	there exist one alert or one crisis whose related information has been changed.
Main Steps	
a	the actor actCoordinator executes the oeValidateAlert use case
b	the actor actCoordinator executes the oeSetCrisisStatus use case
c	the actor actCoordinator executes the oeSetCrisisHandler use case
d	the actor actCoordinator executes the oeReportOnCrisis use case
e	the actor actCoordinator executes the oeCloseCrisis use case
f	the actor actCoordinator executes the oeInvalidateAlert use case
Steps Ordering Constraints	
1	managing a crisis is doing one of the indicated use cases.

Figure 2.4 shows the use case diagram for the ugManageCrisis user goal use case

2.3.1.5 usergoal-ugMonitor

the actCoordinator's goal is to get the detailed list of existing crisis or alerts to decide on next actions to undertake.

USE-CASE DESCRIPTION	
Name	ugMonitor
Scope	system
Level	usergoal
Primary actor(s)	
1	actCoordinator[active]
Goal(s) description	
the actCoordinator's goal is to get the detailed list of existing crisis or alerts to decide on next actions to undertake.	
Reuse	
1	oeGetCrisisSet [0..*]
2	oeGetAlertsSet [0..*]
Protocol condition(s)	
1	the iCrash system has been deployed
Pre-condition(s)	
1	none
Main post-condition(s)	
1	none
Main Steps	

continues in next page ...

... Use-Case Description table continuation

a	the actor `actCoordinator` executes the `oeGetAlertsSet` use case
b	the actor `actCoordinator` executes the `oeGetCrisisSet` use case

Figure 2.5 shows the use case diagram for the ugMonitor user goal use case

2.3.1.6 usergoal-ugSecurelyUseSystem

the actAuthenticated's goal is to follow an identification procedure to be allowed to act as a coordinator or as an administrator. A password reset is always possible. After the identification procedure fails 3 times in a row a captcha test need to be performed.

Use-Case Description	
Name	ugSecurelyUseSystem
Scope	system
Level	usergoal
Primary actor(s)	
1	`actAuthenticated[active]`
Secondary actor(s)	
1	`actAdministrator[]`
Goal(s) description	
the actAuthenticated's goal is to follow an identification procedure to be allowed to act as a coordinator or as an administrator. A password reset is always possible. After the identification procedure fails 3 times in a row a captcha test need to be performed.	
Reuse	
1	`oeLogin [1..*]`
2	`oeFillCaptcha [0..*]`
3	`oeResetPassword [0..1]`
4	`oeLogout [1..1]`
Protocol condition(s)	
1	the iCrash system has been deployed
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the actAuthenticated is known by the system not to be logged.
Main Steps	
a	the actor `actAuthenticated` executes the `oeLogin` use case
b	the actor `actAuthenticated` executes the `oeFillCaptcha` use case
c	the actor `actAuthenticated` executes the `oeResetPassword` use case
d	the actor `actAuthenticated` executes the `oeLogout` use case
Steps Ordering Constraints	
1	step (a) or step (c) is always the first one. step (a) must always precede step (d). step (b) is only needed when step (a) fails 3 times in a row. step (d) is always the last step

Figure 2.6 shows the use case diagram for the ugSecurelyUseSystem user goal use case

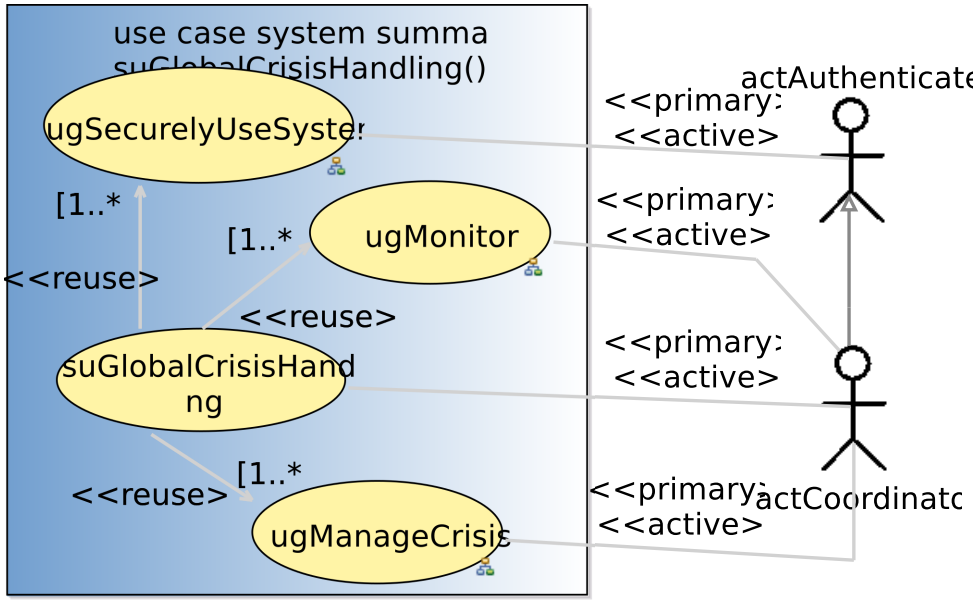

Figure 2.2: suGlobalCrisisHandling user goal use case

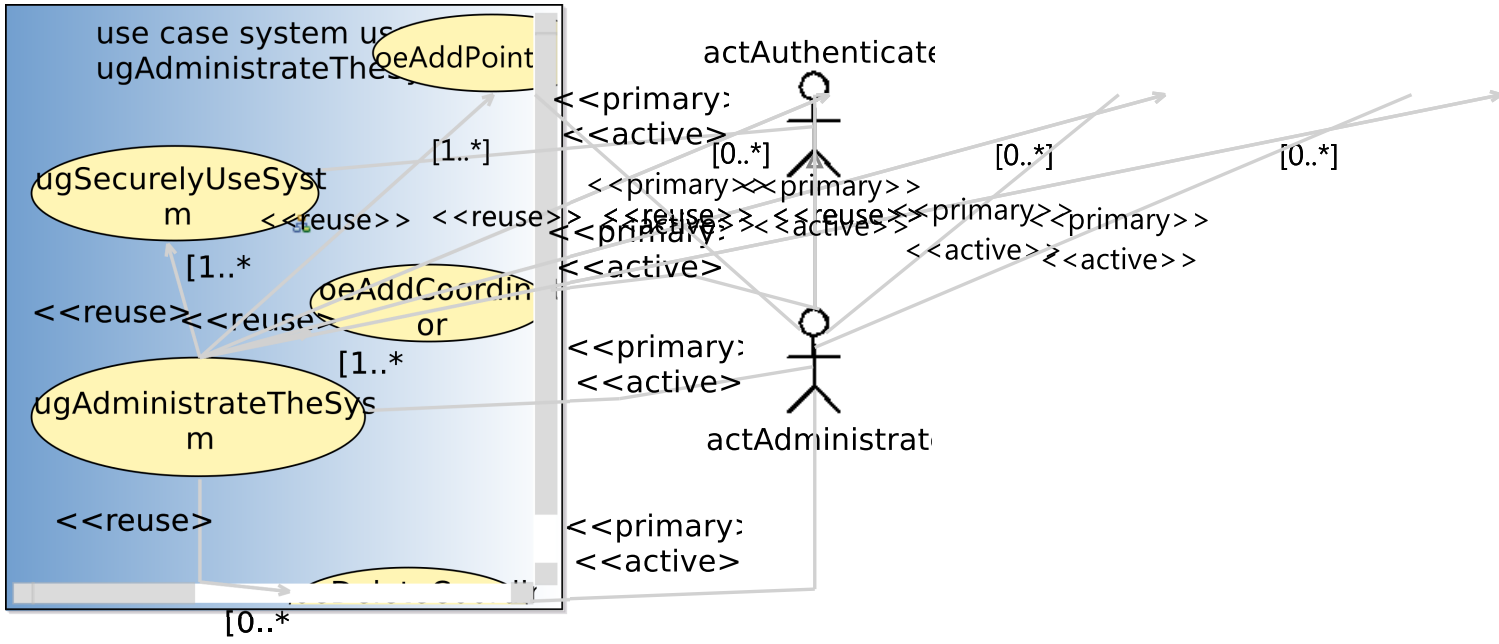

Figure 2.3: ugAdministrateTheSystem user goal use case

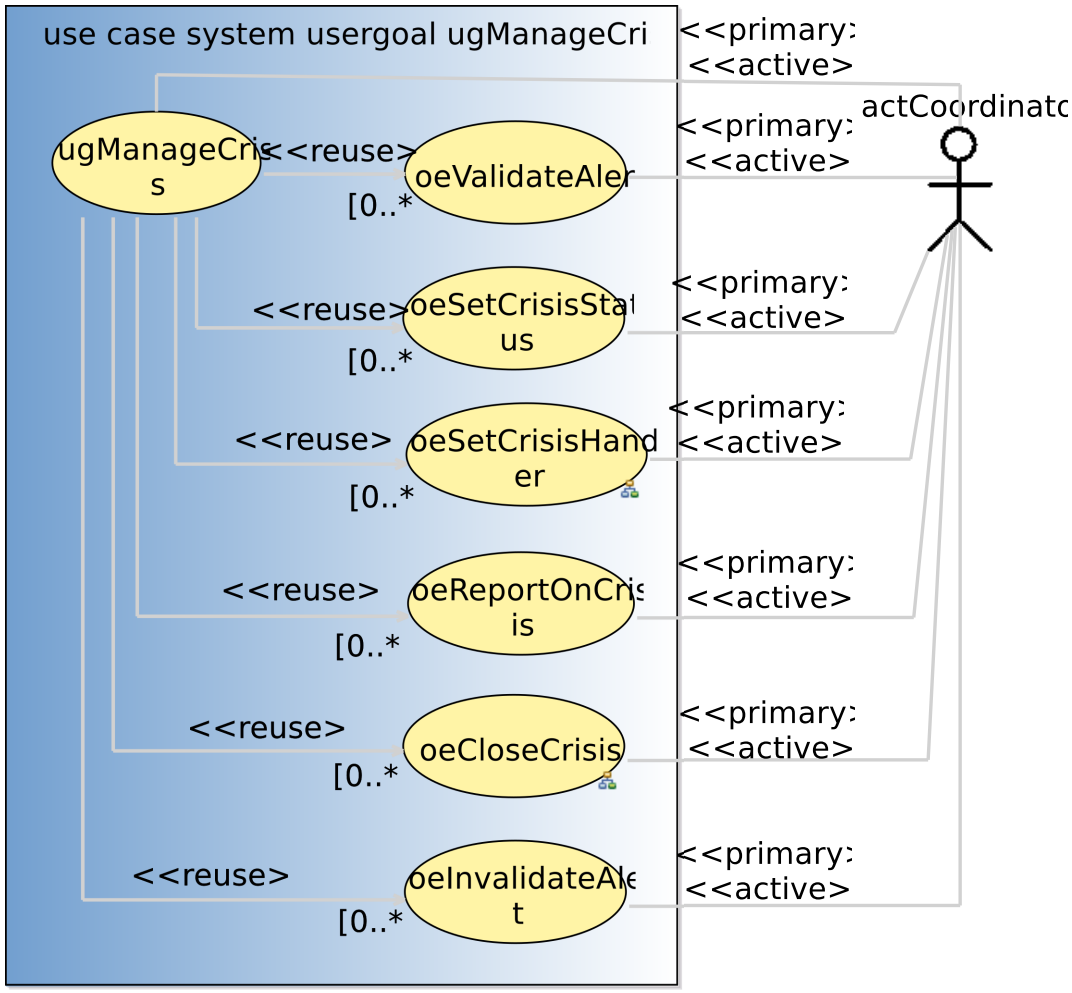

Figure 2.4: ugManageCrisis user goal use case

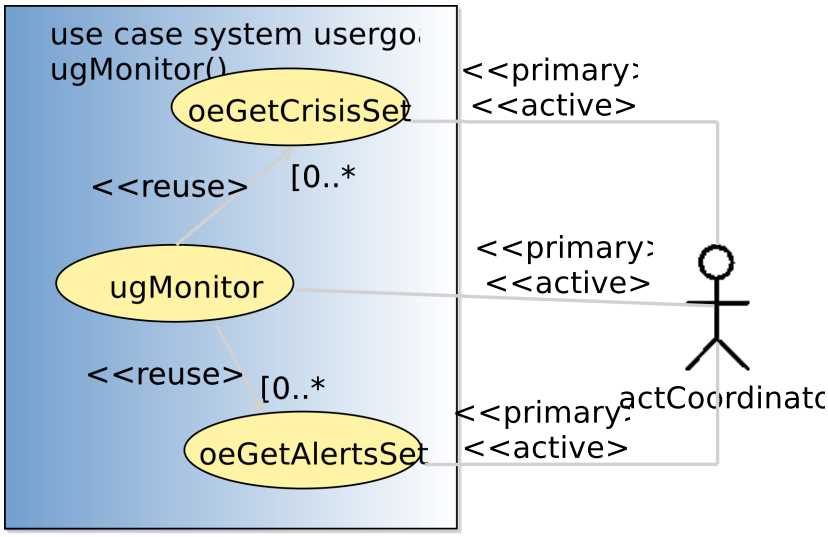

Figure 2.5: ugMonitor user goal use case

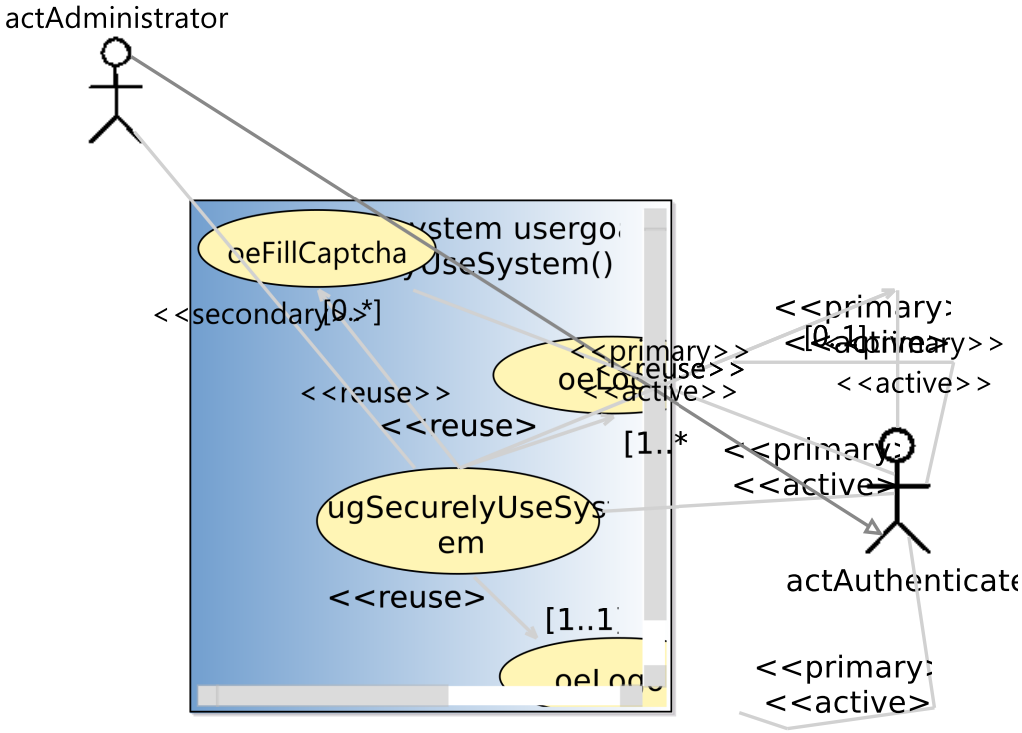

Figure 2.6: ugSecurelyUseSystem user goal use case

2.3.1.7 subfunction-oeClosestToALocation

Gives a sorted list back (From closest until furthest)

Use-Case Description	
Name	oeClosestToALocation
Scope	system
Level	subfunction
Primary actor(s)	
1	`actAdministrator[active]`
Goal(s) description	
Gives a sorted list back (From closest until furthest)	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

2.3.1.8 subfunction-oeFillCaptcha

Use-Case Description	
Name	oeFillCaptcha
Scope	system
Level	subfunction
Parameters	
AdtString: ptString 1	

Use-Case Description	
Name	oeRankDownCoordinator
Scope	system
Level	subfunction
Parameters	
AdtCoordinatorID: dtCoordinatorID 1	
Primary actor(s)	
1	`actAdministrator[active]`
Goal(s) description	
the goal is to demote a coordinator by lowering his experience rank	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

2.3.1.10 subfunction-oeResetPassword

goal is to request a new password in order to be able to request access secured system operations anew.

Use-Case Description	
Name	oeResetPassword
Scope	system
Level	subfunction
Primary actor(s)	
1	`actAuthenticated[active]`
Secondary actor(s)	
1	`actAdministrator[passive]`
Goal(s) description	
goal is to request a new password in order to be able to request access secured system operations anew.	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

2.3.1.11 subfunction-oeSelectCategories

goal is to display/select categories from the system's state

Use-Case Description	
Name	oeSelectCategories
Scope	system
Level	subfunction
Primary actor(s)	
1	`actAdministrator[active]`
Goal(s) description	
goal is to display/select categories from the system's state	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

2.3.1.12 subfunction-oeSetCrisisHandler

goal is to declare himself as been the handler of a crisis having the specified id.

Use-Case Description	
Name	oeSetCrisisHandler
Scope	system
Level	subfunction
Parameters	
AdtCrisisID: dtCrisisID 1	
Primary actor(s)	
1	`actCoordinator[active]`
Secondary actor(s)	
1	`actCoordinator[passive]`
2	`actComCompany[passive, multiple]`
Goal(s) description	
goal is to declare himself as been the handler of a crisis having the specified id.	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

Figure 2.7 shows the use case diagram for the oeSetCrisisHandler subfunction use case

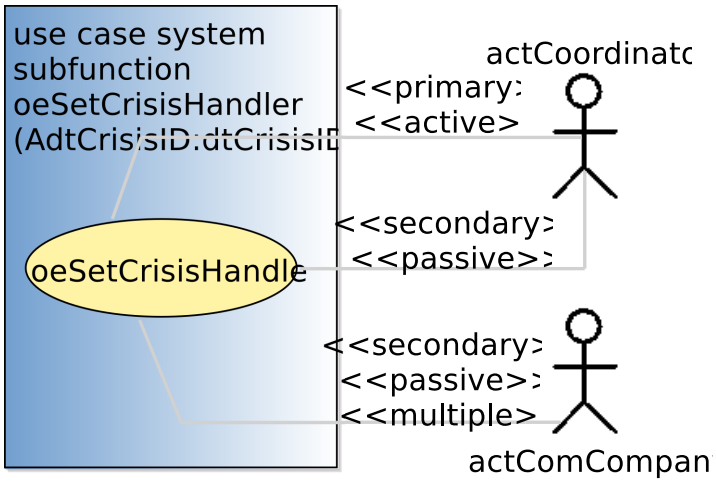

Figure 2.7: oeSetCrisisHandler subfunction use case

2.3.1.13 subfunction-oeSetCrisisType

goal is to define the difficulty level (based on the severity) of a specific crisis.

Use-Case Description	
Name	oeSetCrisisType
Scope	system
Level	subfunction
Primary actor(s)	
1	`actCoordinator[active]`
Goal(s) description	
goal is to define the difficulty level (based on the severity) of a specific crisis.	
Protocol condition(s)	
1	
Pre-condition(s)	
1	
Main post-condition(s)	
1	
Additional Information	
none	

2.3.1.14 subfunction-oeSollicitateCrisisHandling

the actActivator's goal is to decrease the number of unhandled crisis.

Use-Case Description	
Name	oeSollicitateCrisisHandling
Scope	system
Level	subfunction
Primary actor(s)	
1	`actActivator[proactive]`
Secondary actor(s)	
1	`actCoordinator[passive, multiple]`
2	`actAdministrator[passive]`
Goal(s) description	
the actActivator's goal is to decrease the number of unhandled crisis.	
Protocol condition(s)	

. . . Use-Case Description table continuation

2	the reminder period for the concerned crisis is initialized.

Figure 2.8 shows the use case diagram for the oeSollicitateCrisisHandling subfunction use case

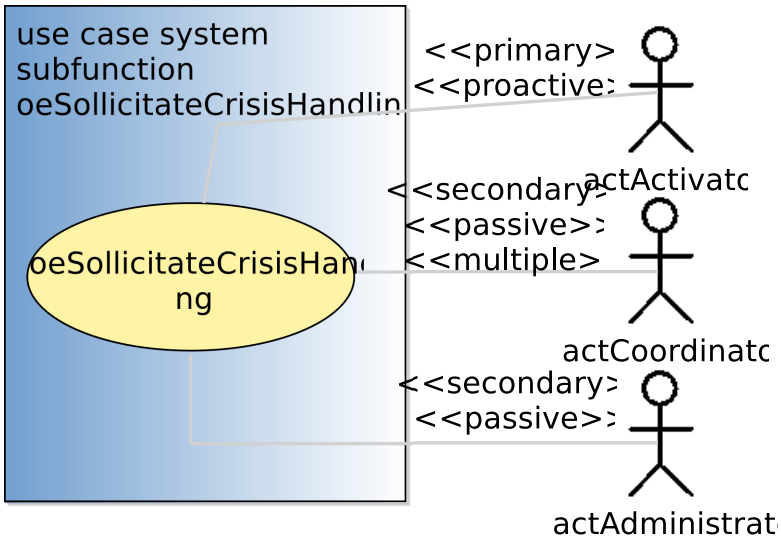

Figure 2.8: oeSollicitateCrisisHandling subfunction use case

2.3.2 Use Case Instance(s)

2.3.2.1 Use-Case Instance - uciSimpleAndCompletePart01:suDeployAndRun

First part of a use case instance for the summary use case `suDeployAndRun` illustrating a simple and complete interaction scenario primarily handled by an administrator in a concrete situation.

SUMMARY USE-CASE INSTANCE	
Instantiated Use Case	
suDeployAndRun	
Instance ID	
uciSimpleAndCompletePart01	
Remarks	
a	shows the system initilization and the first administrative tasks by the administrator.
b	The unique and always existing `actMsrCreator` actor instance (named here `theCreator`) requests the initialization of the system and its environment (made of one administrator identified here by `bill`), one activator actor (identified by `theClock`) and indicating that the number of communication company actor instances for the system's environment is 4 (one of them is identified here by `tango`)
c	the administrator logs in to initialize a coordinator
d	an alert is received. Time is goind one without having the coordinator handling the alert which let's the proactive actor trigger the automatic sollicitation of crisis handling.
e	this first part stops before the coordinator logs in the system.

Figure 2.9 shows the sequence diagram representing the first part of a simple and complete use case instance for the summary use case `suDeployAndRun`.

2.3.2.2 Use-Case Instance - uciSimpleAndCompletePart02:suDeployAndRun

Second part of a simple and complete use case instance for the summary use case `suDeployAndRun` illustrating a simple and complete interaction scenario primarily handled by an administrator in a concrete situation.

SUMMARY USE-CASE INSTANCE	
Instantiated Use Case	
suDeployAndRun	
Instance ID	
uciSimpleAndCompletePart02	
Remarks	
a	starts when the coordinator logs in the system until the full handling of all the existing crisis.
b	shows an instantiated case of handling of a crisis by a coordinator until its closure after reporting.

Figure 2.10 shows the sequence diagram representing the second part of a simple and complete use case instance for the summary use case `suDeployAndRun`.

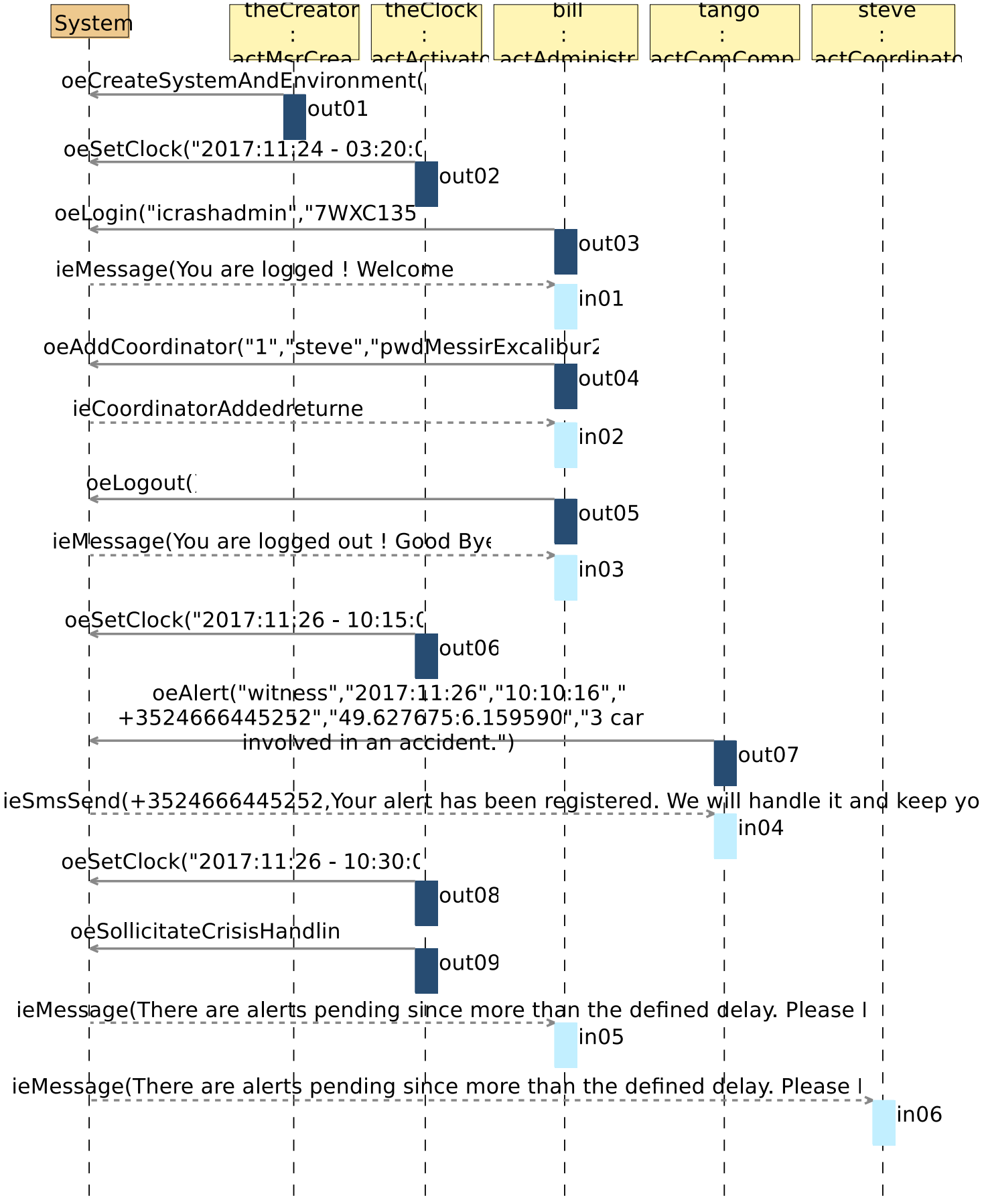

Figure 2.9: uci-suDeployAndRun-uciSimpleAndComplete-Part01

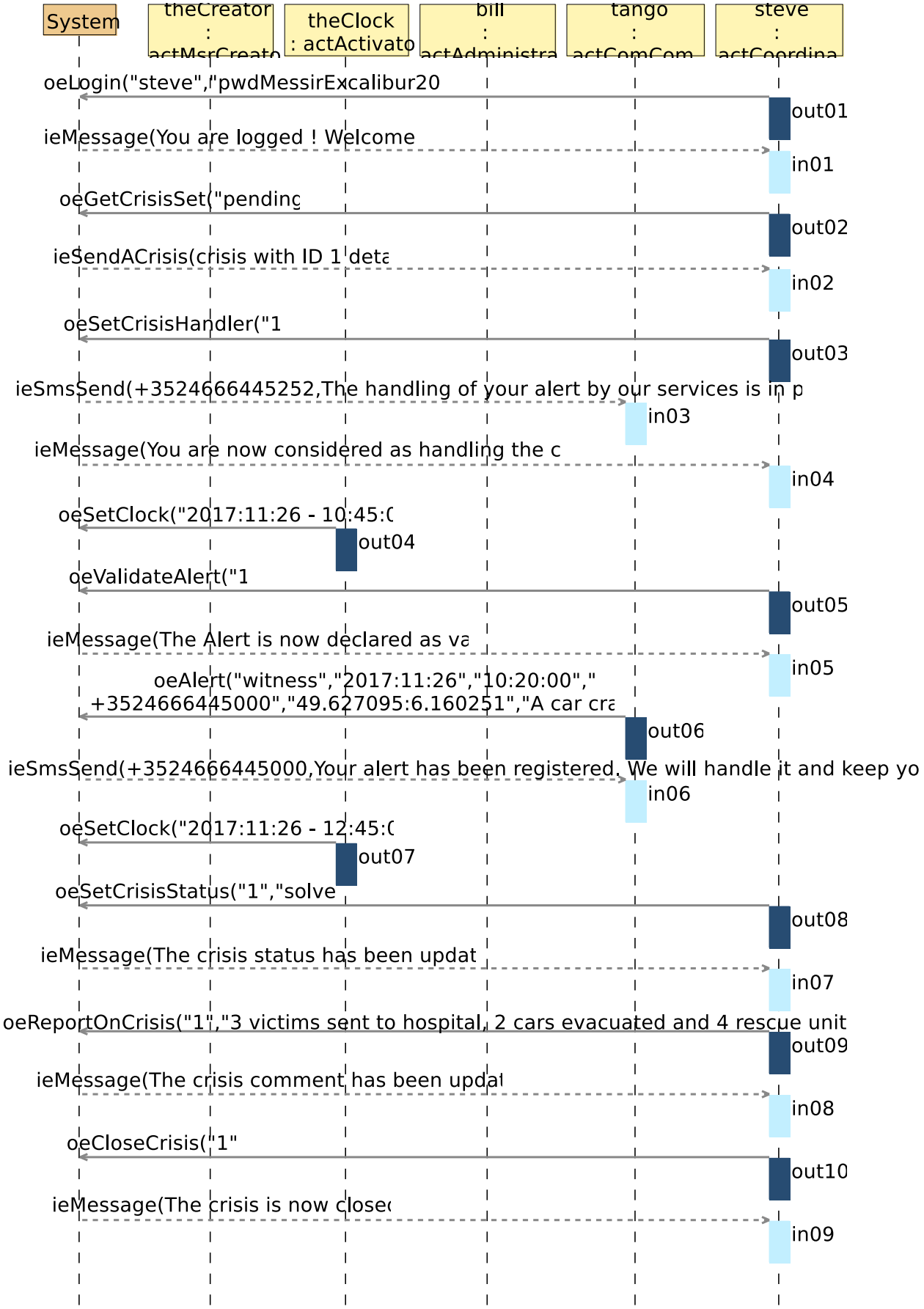

Figure 2.10: uci-suDeployAndRun-uciSimpleAndComplete-Part02 use case instance sequence diagram

2.3.2.3 Use-Case Instance - uciugAdministrateTheSystem:ugAdministrateTheSystem

use case instance for the use case `ugAdministrateTheSystem` showing the addition of a coordinator and his demotion in rank by the administrator

USERGOAL USE-CASE INSTANCE
Instantiated Use Case ugAdministrateTheSystem
Instance ID uciugAdministrateTheSystem

Figure 2.11 shows the sequence diagram representing a use case instance for the user goal use case `ugAdministrateTheSystem`.

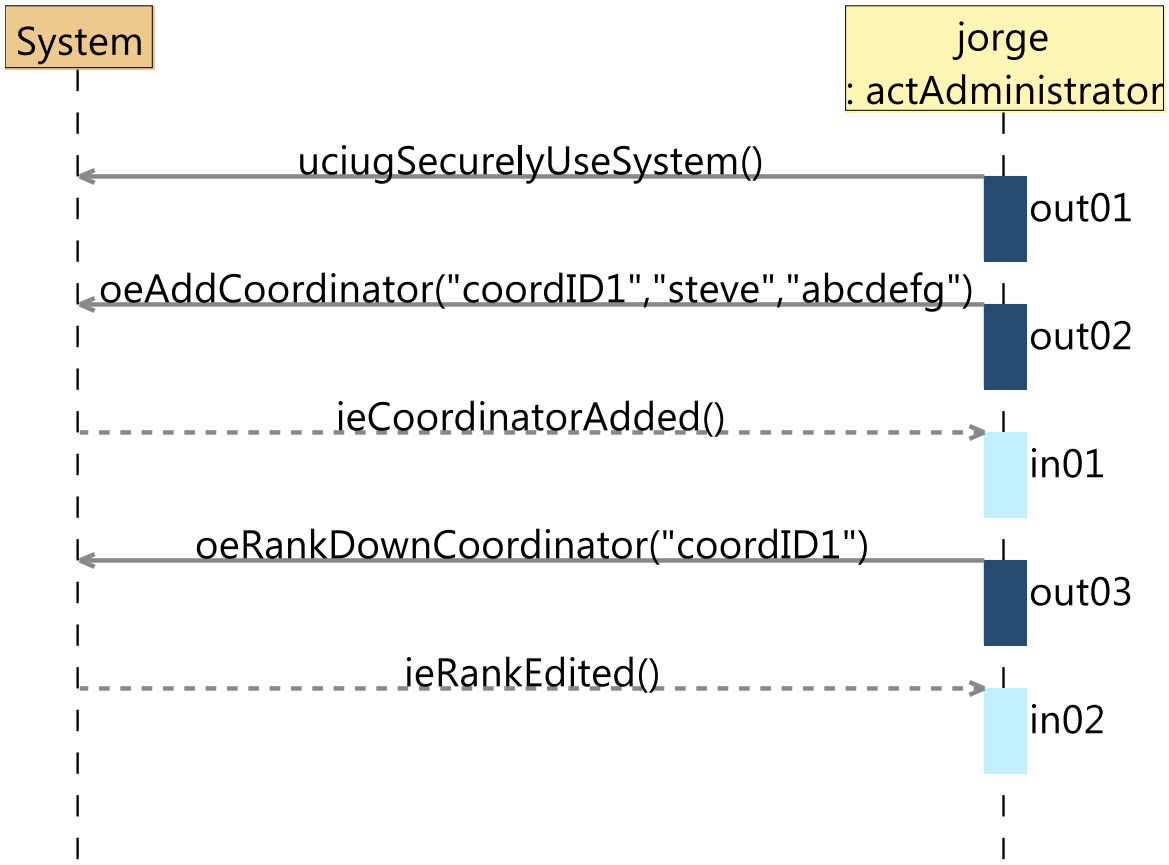

Figure 2.11: uci-ugAdministrateTheSystem

2.3.2.4 Use-Case Instance - uciugAdministrateTheSystemPointOfInterest:ugAdministrateTheSystem

use case instance for the use case `ugAdministrateTheSystem` showing the addition/deletion/edition of a point of interest.

USERGOAL USE-CASE INSTANCE
Instantiated Use Case ugAdministrateTheSystem
Instance ID uciugAdministrateTheSystemPointOfInterest

Figure 2.12 shows the sequence diagram representing a use case instance for the user goal use case `ugAdministrateTheSystem`.

2.3.2.5 Use-Case Instance - uciugSecurelyUseSystem:ugSecurelyUseSystem

use case instance for the user goal use case `ugSecurelyUseSystem` illustrating a simple interaction scenario primarily handled by an administrator in a concrete situation. In this instance there is no password reset asked, neither a captcha test asked.

USERGOAL USE-CASE INSTANCE	
Instantiated Use Case ugSecurelyUseSystem	
Instance ID uciugSecurelyUseSystem	
Remarks	
a	A captcha test is asked after three failed attempts of login.
b	A password reset is always possible. A mail has to be given to the system.

Figure 2.13 view

2.3.2.6 Use-Case Instance - uciugSecurelyUseSystemFailLogin:ugSecurelyUseSystem

use case instance for the user goal use case `ugSecurelyUseSystem` illustrating a simple interaction scenario primarily handled by an administrator in a concrete situation. In this instance a captcha test is asked after 3 failed attempts of login. Once the `actAuthenticated` successfully entered the right captcha, he's able to connect.

USERGOAL USE-CASE INSTANCE	
Instantiated Use Case ugSecurelyUseSystem	
Instance ID uciugSecurelyUseSystemFailLogin	
Remarks	
a	A captcha test is asked after three failed attempts of login.
b	A password reset is always possible. A mail has to be given to the system.

Figure 2.14 view

Figure 2.12: uci-ugAdministrateTheSystemPointOfInterest

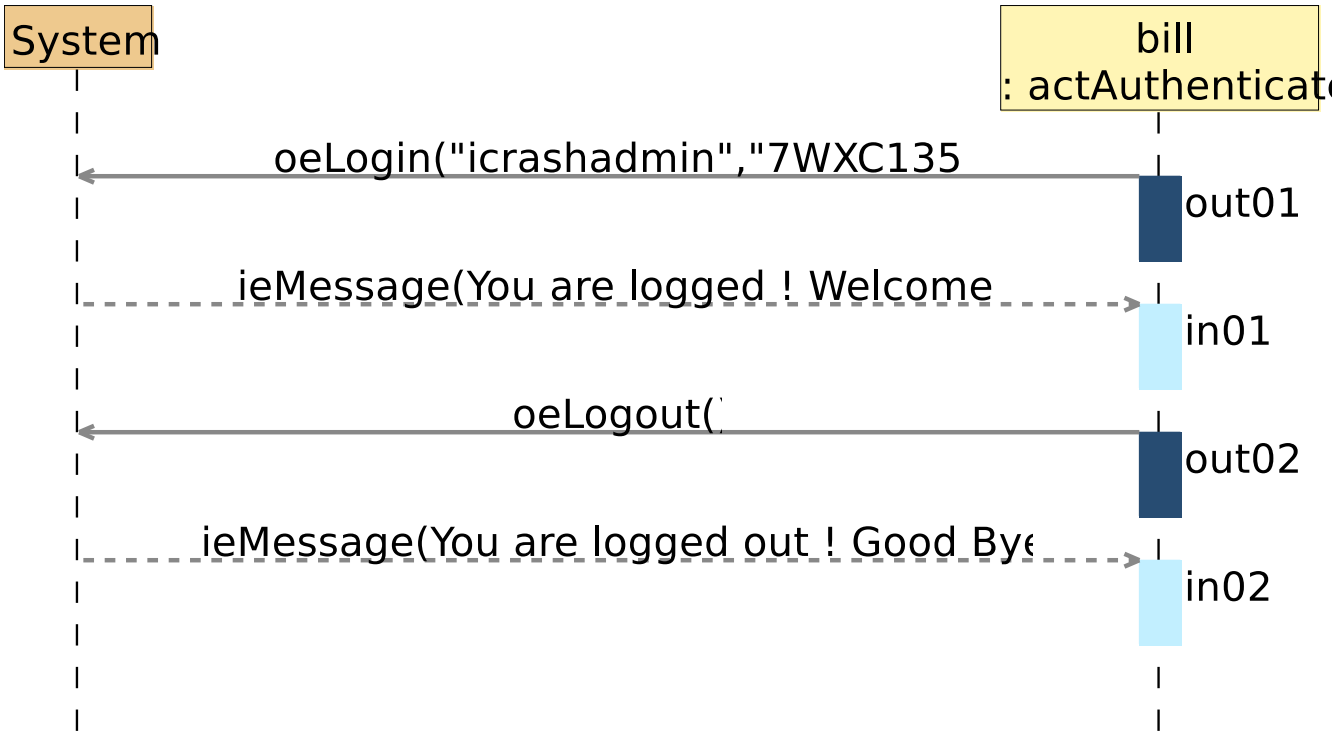

Figure 2.13:

2.3.2.7 Use-Case Instance - uciugSecurelyUseSystemFailLoginResetPassword:ugSecurelyUseSyste

use case instance for the user goal use case `ugSecurelyUseSystem` illustrating a simple interaction scenario primarily handled by an administrator in a concrete situation. In this instance a captcha test is asked. The `actAuthenticated` asks for a password reset and logs in after retrieving it from his mails.

USERGOAL USE-CASE INSTANCE	
Instantiated Use Case	
ugSecurelyUseSystem	
Instance ID	
uciugSecurelyUseSystemFailLoginResetPassword	
Remarks	
a	A captcha test is asked after three failed attempts of login.
b	A password reset is always possible. A mail has to be given to the system.

Figure 2.15

2.3.2.8 Use-Case Instance - uciugSecurelyUseSystemWithDirectResetPassword:ugSecurelyUseSys

use case instance for the user goal use case `ugSecurelyUseSystem` illustrating a simple interaction scenario primarily handled by an administrator in a concrete situation. In this instance a password reset is directly asked. The `actAuthenticated` may logon after retrieving the password from his mails.

USERGOAL USE-CASE INSTANCE
Instantiated Use Case
ugSecurelyUseSystem
Instance ID
uciugSecurelyUseSystemWithDirectResetPassword


System
Theuser
: actAuthenticated
oeLogin("failed","login")
out01
ieMessage(Your password is not correct. Wrong attempts : 1)
in01
oeLogin("failed","login")
out02
ieMessage(Your password is not correct. Wrong attempts : 2)
in02
oeLogin("failed","login")
out03
ieMessage(Your password is not correct. Wrong attempts : 3.)
in03
ieMessage(You need to fill out a captcha to be able to login now. You can also do a
in04
oeFillCaptcha("wrong captcha")
out04
ieMessage(Your captcha is not correct. Try again.)
in05
oeFillCaptcha("successful captcha")
out05
ieMessage(Your captcha is correct. You may try to login again.)
in06
oeLogin("Theuser","7WXC1359")
out06
ieMessage(You are logged ! Welcome ...)
in07
oeLogout()
out07
ieMessage(You are logged out ! Good Bye ...)


System
Theuser
: actAuthenticated
oeLogin("failed","login")
out01
ieMessage(Your password is not correct. Wrong attempts : 1
in01
oeLogin("failed","login")
out02
ieMessage(Your password is not correct. Wrong attempts : 2
in02
oeLogin("failed","login")
out03
ieMessage(Your password is not correct. Wrong attempts : 3
in03
ieMessage(You need to fill out a captcha to be able to login now. You can also c
in04
oeResetPassword("Theuser")
out04
ieMessage(Your password has been reset. Check your mails to be able to
in05
ieMessage2(The password of "Theuser" has be
oeLogin("Theuser","NewPassword")
out05
ieMessage(You are logged ! Welcome ...)
in07
oeLogout()
out06
ieMessage(You are logged out ! Good Bye ...)
in08

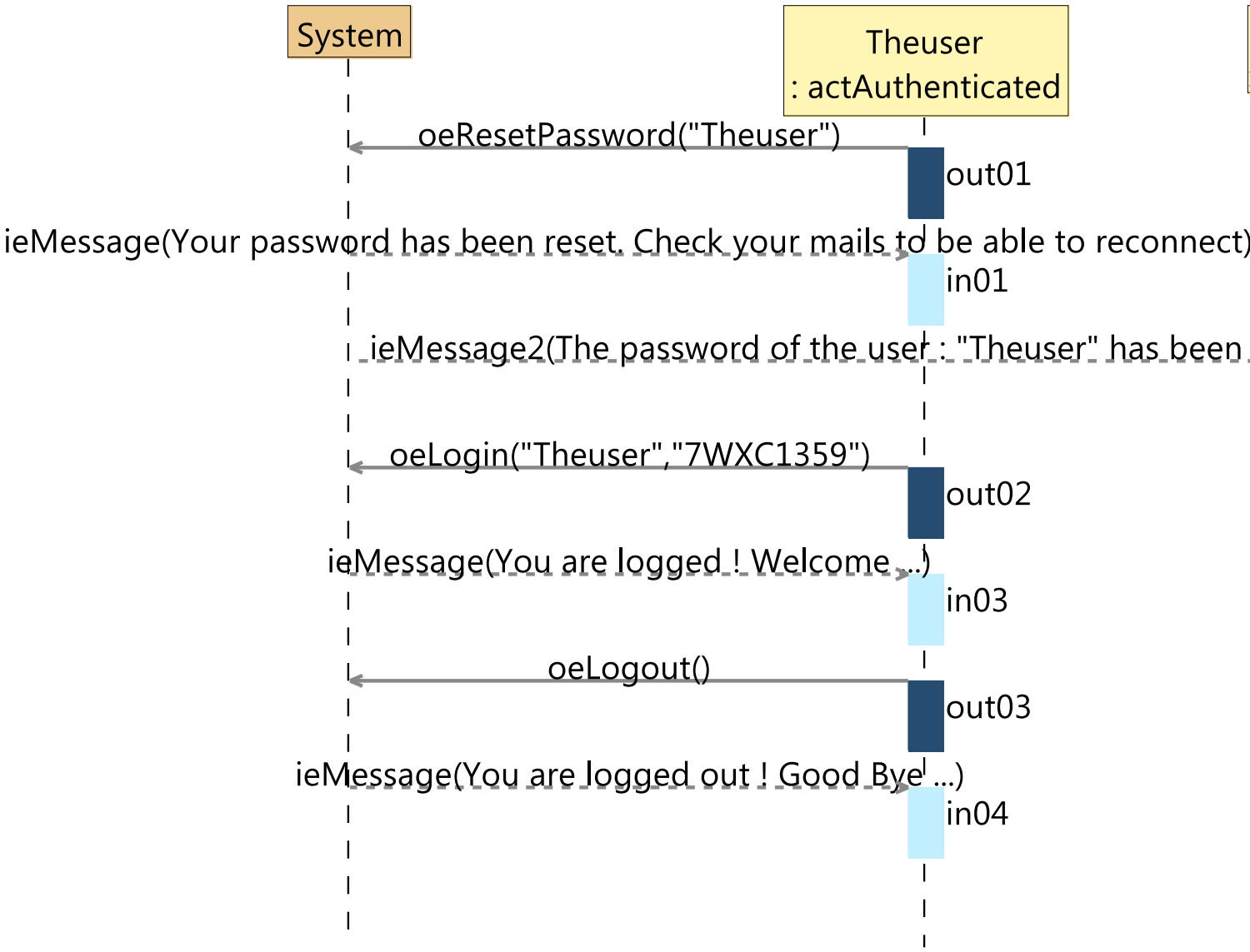
System
Theuser
: actAuthenticated
oeResetPassword("Theuser")
out01
ieMessage(Your password has been reset. Check your mails to be able to reconnect)
in01
ieMessage2(The password of the user : "Theuser" has been
oeLogin("Theuser","7WXC1359")
out02
ieMessage(You are logged ! Welcome ...)
in03
oeLogout()
out03
ieMessage(You are logged out ! Good Bye ...)
in04

Figure 2.16:

Chapter 3

Environment Model

We provide below the view(s) defined for the Messiᴘ environment model (cf. [1]) of the system.

3.1 Local view 01

Figure 3.1 shows the local view giving the second part of the environment model of the system in term of its state class, actors with their input and output interfaces and all related associations.

3.2 Local view 02

Figure 3.2 shows the local view giving the second part the environment model of the system in term of its state class, actors with their input and output interfaces and all related associations.

3.3 Local view 03

Figure 3.3 shows the local view for the administrator actor and interfaces

3.4 Local view 04

Figure 3.4 shows the local view for the coordinator actor and interfaces

3.5 Local view 05

Figure 3.5 shows the local view for the authenticated actor and interfaces

3.6 Global view 01

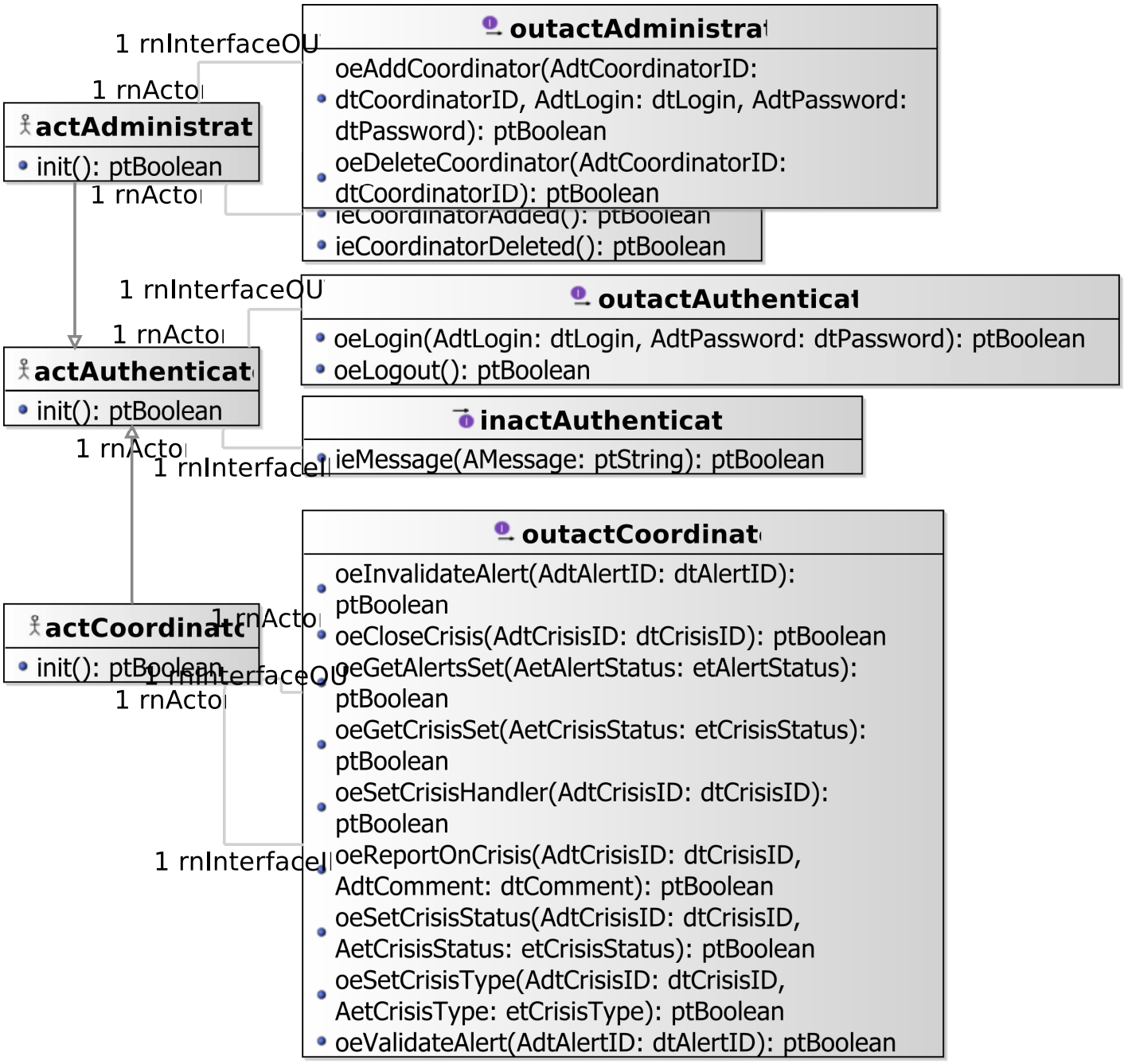

Figure 3.1: Environment Model - Local View 01. environment model local view - Part 1.

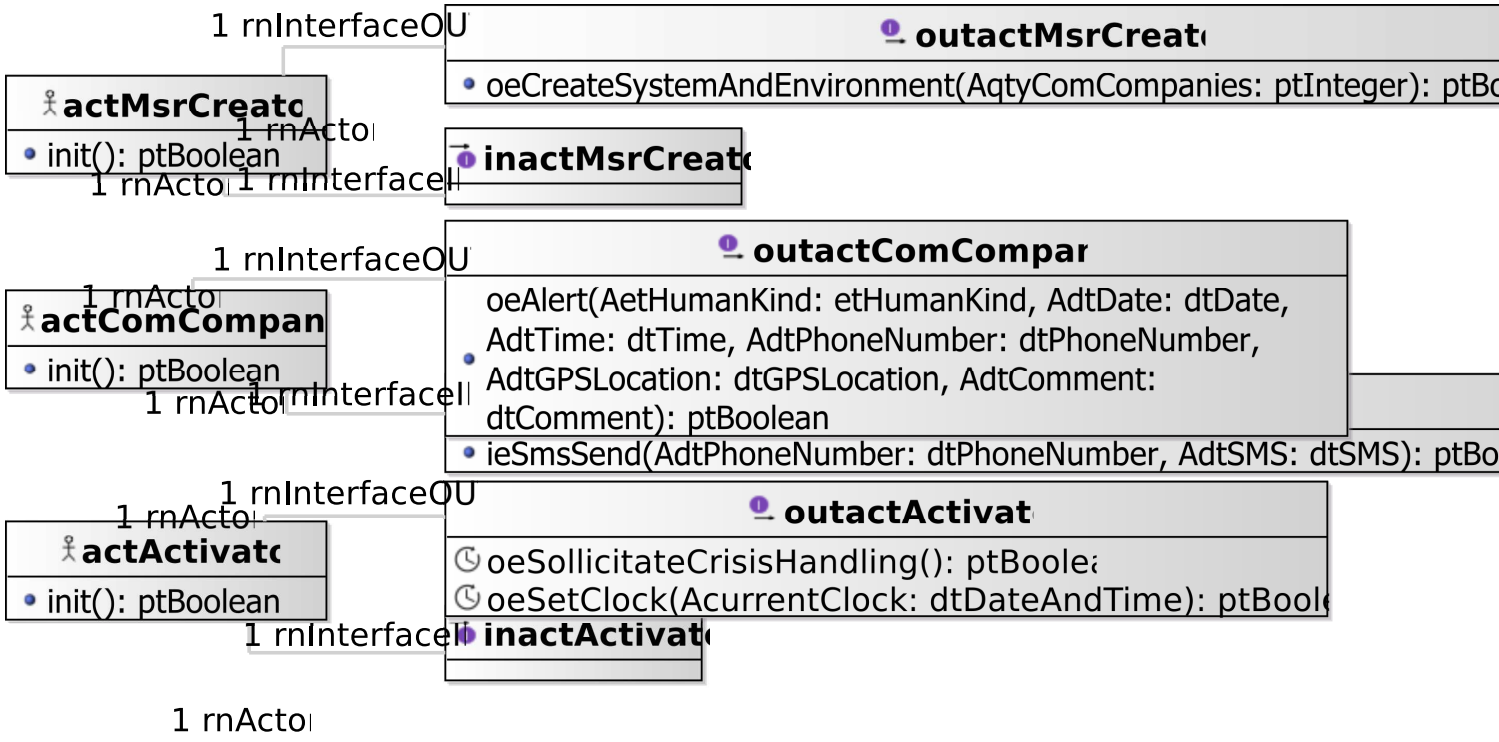

Figure 3.2: Environment Model - Local View 02. environment model local view - Part 2.

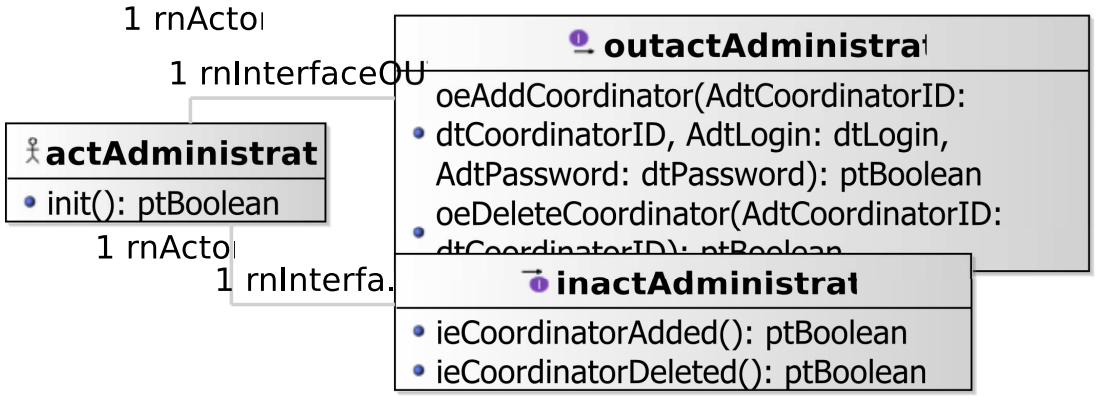

Figure 3.3: Environment Model - Local View 03. administrator actor environment model view.

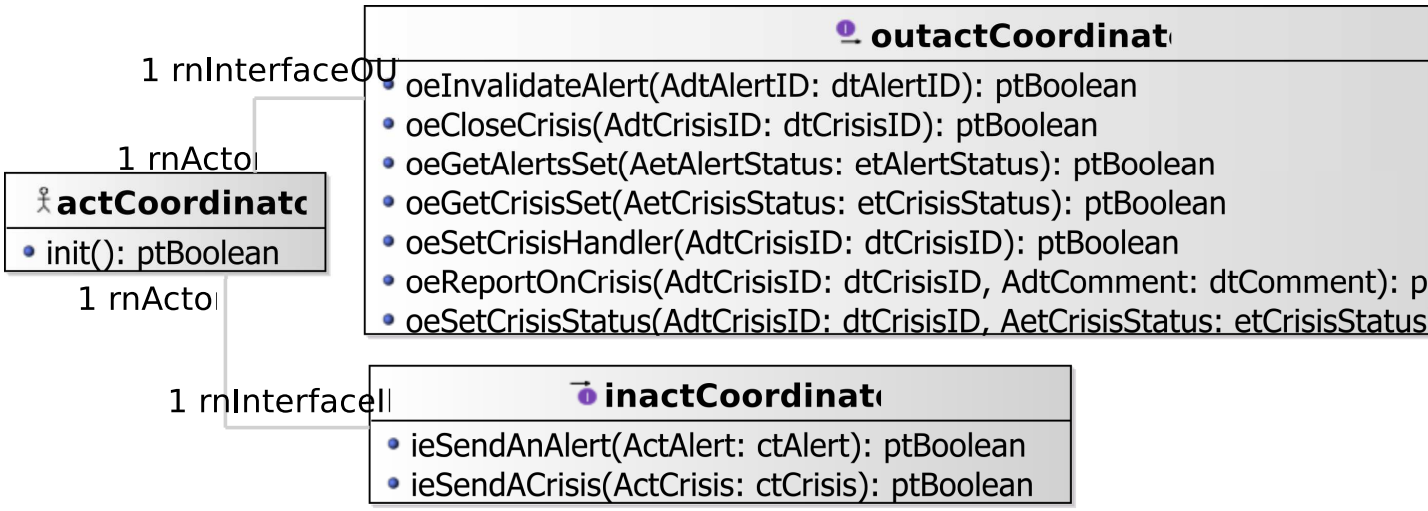

Figure 3.4: Environment Model - Local View 04. coordinator actor environment model view.

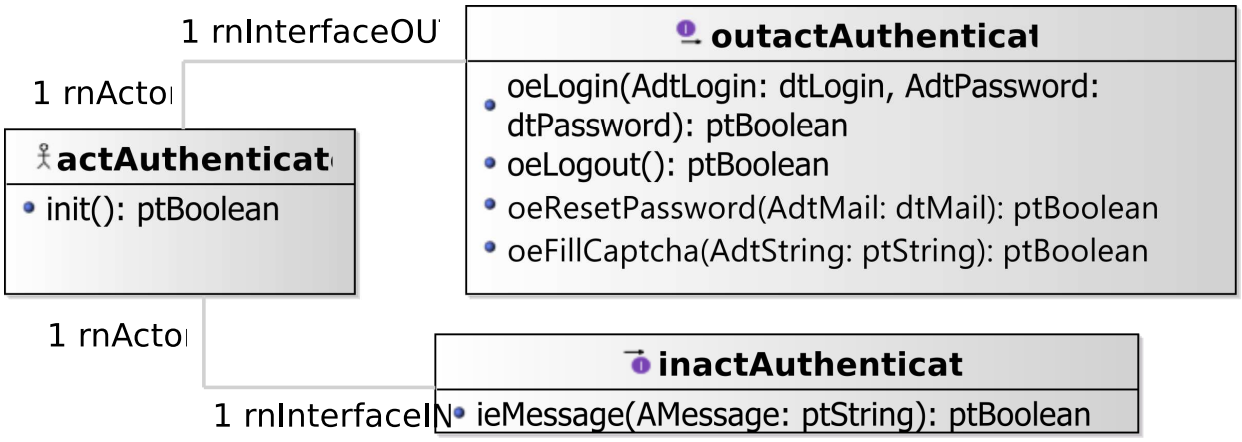

Figure 3.5: Environment Model - Local View 05. authenticated actor environment model local view.

Figure 3.6 shows a global view for all actors with their relationships with ctState

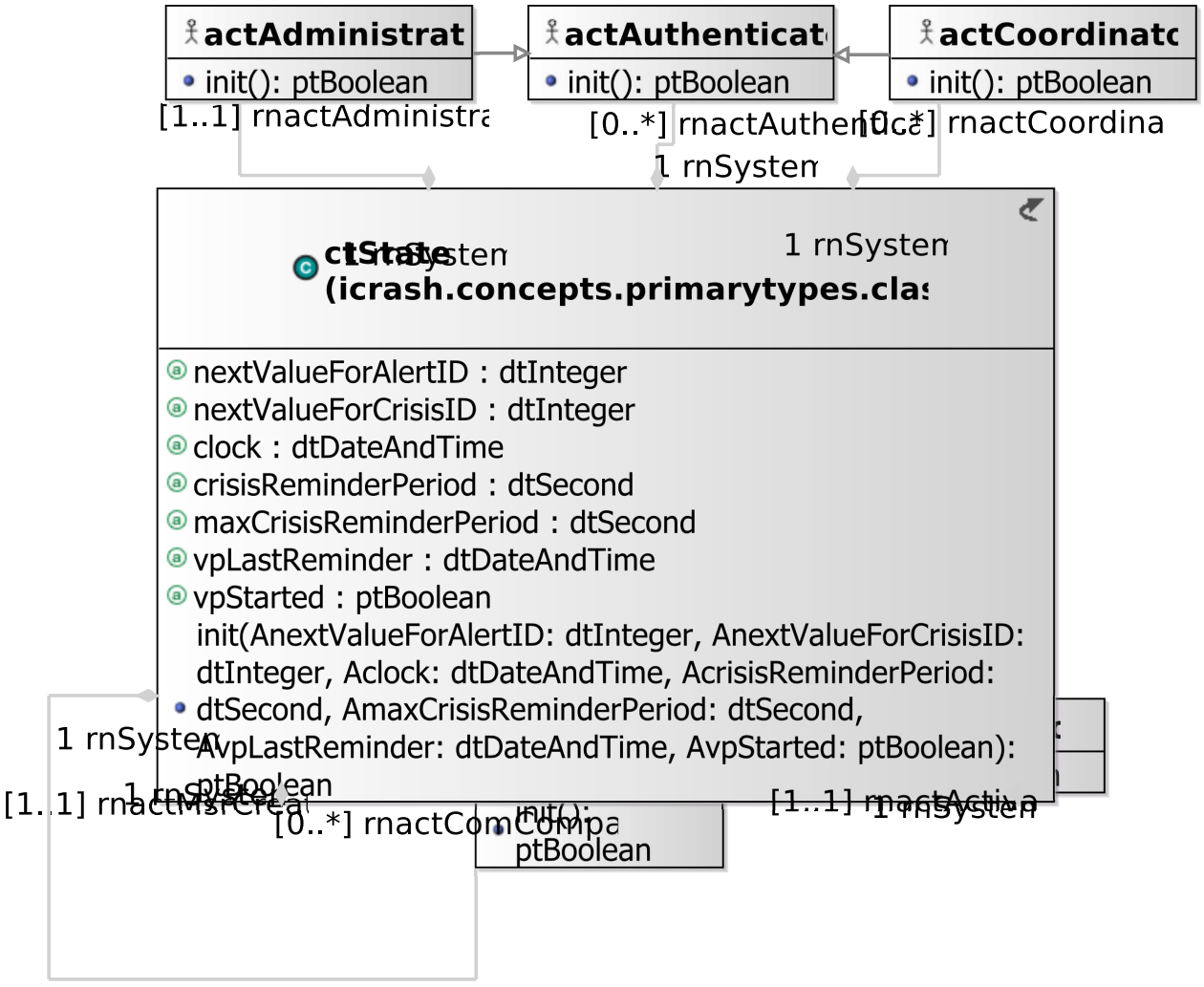

Figure 3.6: Environment Model - Global View 01. em-gv-01 environment model global view.

3.7 Actors and Interfaces Descriptions

We provide for the given views the description of the actors together with their associated input and output interface descriptions.

3.7.1 `actActivator` Actor

Actor
actActivator
represents a logical actor for time automatic message sending based on system's or environment status.
OutputInterfaces

continues in next page . . .

... Actor table continuation

OUT 1	**[proactive] oeSollicitateCrisisHandling():ptBoolean** used to avoid crisis to stay too long in an not handled status.
OUT 2	**[proactive] oeSetClock(AcurrentClock:dtDateAndTime):ptBoolean** used to update the system's time

3.7.2 `actAdministrator` Actor

ACTOR	
actAdministrator represents an actor responsible of administration tasks for the *iCrash* system.	
Extends	
icrash.environment.actAuthenticated	
OutputInterfaces	
OUT 1	**oeAddCoordinator(AdtCoordinatorID:dtCoordinatorID, AdtLogin:dtLogin, AdtPassword:dtPassword):ptBoolean** sent to add a new coordinator in the system's post state and environment's post state.
OUT 2	**oeDeleteCoordinator(AdtCoordinatorID:dtCoordinatorID):ptBoolean** sent to delete an existing coordinator in the system's post state and environment's post state.
OUT 3	**oeAddPointOfInterest(AetCategory:etCategory, AdtGPSLocation:dtGPSLocation, AdtDescription:dtDescription):ptBoolean** sent to add a new point of interest in the system's post state.
OUT 4	**oeDeletePointOfInterest(AdtPointOfInterestID:dtPointOfInterestID):ptBoo** sent to delete a new point of interest in the system's post state.
OUT 5	**oeEditPointOfInterest(AdtPointsOfInterestID:dtPointOfInterestID, AetCategory:etCategory, AdtGPSLocation:dtGPSLocation, AdtDescription:dtDescription):ptBoolean** sent to edit a new point of interest in the system's post state.
OUT 6	**oeSelectCategories(AdtCategory:etCategory):ptBoolean** sent to select categories in the system's post state.
InputInterfaces	
IN 1	**ieCoordinatorAdded():ptBoolean** its reception confirms the creation of the requested coodinator.
IN 2	**ieCoordinatorDeleted():ptBoolean** its reception confirms the deletion of the requested coodinator.
IN 3	**iePointOfInterestAdded():ptBoolean** its reception confirms the creation of the requested point of interest.
IN 4	**iePointOfInterestDeleted():ptBoolean** its reception confirms the deletion of the requested point of interest.
IN 5	**iePointOfInterestEdited():ptBoolean** its reception confirms the edition of the requested point of interest.
IN 6	**oeSelectCategories(AdtCategory:etCategory):ptBoolean** its reception confirms the selection of the requested category.

3.7.3 `actAuthenticated` Actor

Actor	
actAuthenticated	
abstract actor providing reusable input and output interfaces for actors that need to authenticate themselves.	
OutputInterfaces	
OUT 1	**oeLogin(AdtLogin:dtLogin, AdtPassword:dtPassword):ptBoolean** sent to request authorization to request access secured system operations.
OUT 2	**oeLogout():ptBoolean** sent to end the secured access to specific system operations.
InputInterfaces	
IN 1	**ieMessage(AMessage:ptString):ptBoolean** allows for receiving general textual messages.

3.7.4 `actComCompany` Actor

Actor	
actComCompany	
represents the communication company stakeholder ensuring the input/ouput of textual messages with humans having communicaiton devices.	
OutputInterfaces	
OUT 1	**oeAlert(AetHumanKind:etHumanKind, AdtDate:dtDate, AdtTime:dtTime, AdtPhoneNumber:dtPhoneNumber, AdtGPSLocation:dtGPSLocation, AdtComment:dtComment):ptBoolean** sent to alert of a potential crisis situation.
InputInterfaces	
IN 1	**ieSmsSend(AdtPhoneNumber:dtPhoneNumber, AdtSMS:dtSMS):ptBoolean** allows for receiving textual messages to be dispatched to the communication company customers having the provided phone number.

3.7.5 `actCoordinator` Actor

Actor	
actCoordinator	
represents actor responsible of handling one or several crisis for the *iCrash* system.	
Extends	
icrash.environment.actAuthenticated	
OutputInterfaces	
OUT 1	**oeInvalidateAlert(AdtAlertID:dtAlertID):ptBoolean** sent to indicate that an alert should be considered as closed.
OUT 2	**oeCloseCrisis(AdtCrisisID:dtCrisisID):ptBoolean** sent to indicate that a crisis should be considered as closed.
OUT 3	**oeGetAlertsSet(AetAlertStatus:etAlertStatus):ptBoolean** sent to request all the ctAlert instances having a specific status.
OUT 4	**oeGetCrisisSet(AetCrisisStatus:etCrisisStatus):ptBoolean** sent to request all the ctCrisis instances having a specific status.
OUT 5	**oeSetCrisisHandler(AdtCrisisID:dtCrisisID):ptBoolean** sent to declare himself as been the handler of a crisis having the specified id.

continues in next page . . .

... Actor table continuation

OUT 6	**oeReportOnCrisis(AdtCrisisID:dtCrisisID, AdtComment:dtComment):ptBoolea** sent to update the textual information available for a specific handled crisis.
OUT 7	**oeSetCrisisStatus(AdtCrisisID:dtCrisisID, AetCrisisStatus:etCrisisStatu** sent to define the handling status of a specific crisis.
OUT 8	**oeSetCrisisType(AdtCrisisID:dtCrisisID, AetCrisisType:etCrisisType):ptB** sent to define the gravity type of a specific crisis.
OUT 9	**oeValidateAlert(AdtAlertID:dtAlertID):ptBoolean** sent to indicate that a specific alert is not a fake.
InputInterfaces	
IN 1	**ieSendAnAlert(ActAlert:ctAlert):ptBoolean** allows for receiving a requested ctAlert instance.
IN 2	**ieSendACrisis(ActCrisis:ctCrisis):ptBoolean** allows for receiving a requested ctCrisis instance.

3.7.6 `actMsrCreator` Actor

ACTOR	
actMsrCreator Represents the creator stakeholder in charge of state and environment initialization.	
OutputInterfaces	
OUT 1	**oeCreateSystemAndEnvironment(AqtyComCompanies:ptInteger):ptBoolean** sent to request the initialization of the system's class instances and the environment actors instances.

Chapter 4

Concept Model

4.1 PrimaryTypes-Classes

4.1.1 Local view 01

Figure 4.1 shows the local view on all the primary types class types.

4.1.2 Local view 02

Figure 4.2 shows the local view of the ctState primary type class type.

4.1.3 Local view 03

Figure 4.3 shows the local view of the ctAlert primary type class type.

4.1.4 Local view 04

Figure 4.4 shows the local view of the ctCrisis primary type class type.

4.1.5 Local view 08

Figure 4.5 views

4.1.6 Global view 01

Figure 4.6 shows the global view on primary types class types showing the association(s) types with the actor classes of the environment model.

4.2 PrimaryTypes-Datatypes

4.2.1 Local view 06

Figure 4.7 view

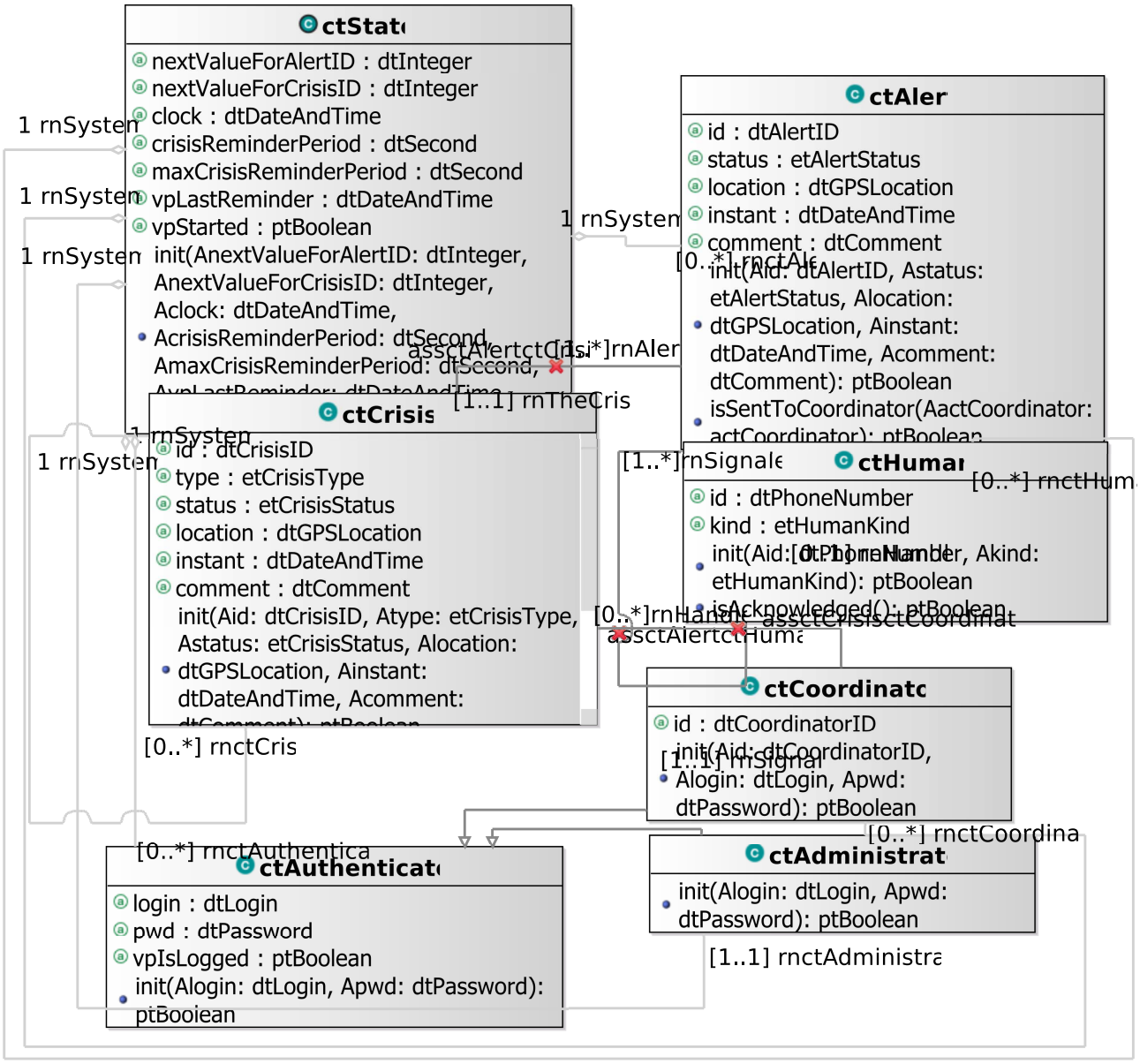

Figure 4.1: Concept Model - PrimaryTypes-Classes local view 01. Local view of all the primary types class types .

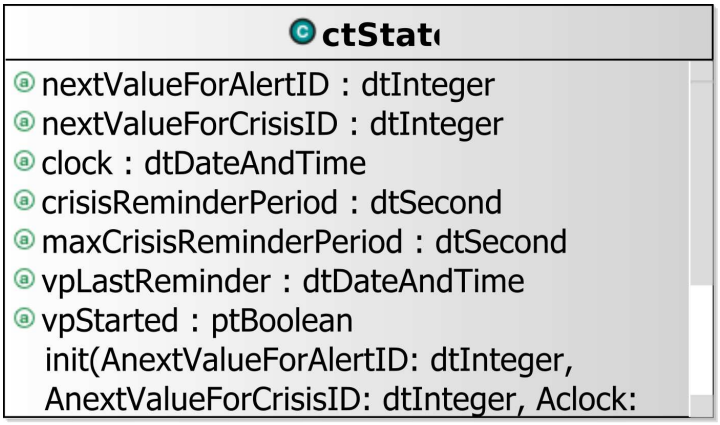

Figure 4.2: Concept Model - PrimaryTypes-Classes local view 02. local view of the ctState primary type.

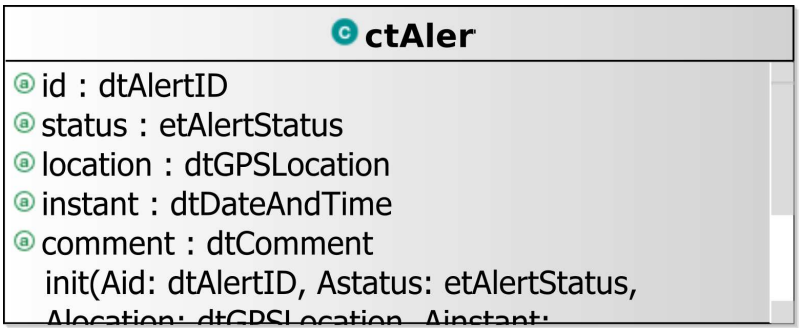

Figure 4.3: Concept Model - PrimaryTypes-Classes local view 03. local view of the ctAlert primary type.

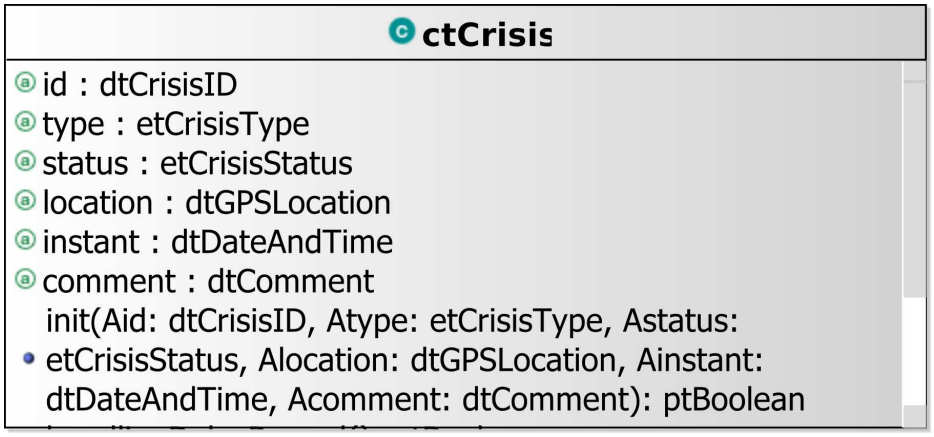

Figure 4.4: Concept Model - PrimaryTypes-Classes local view 04. local view of the ctCrisis primary type.

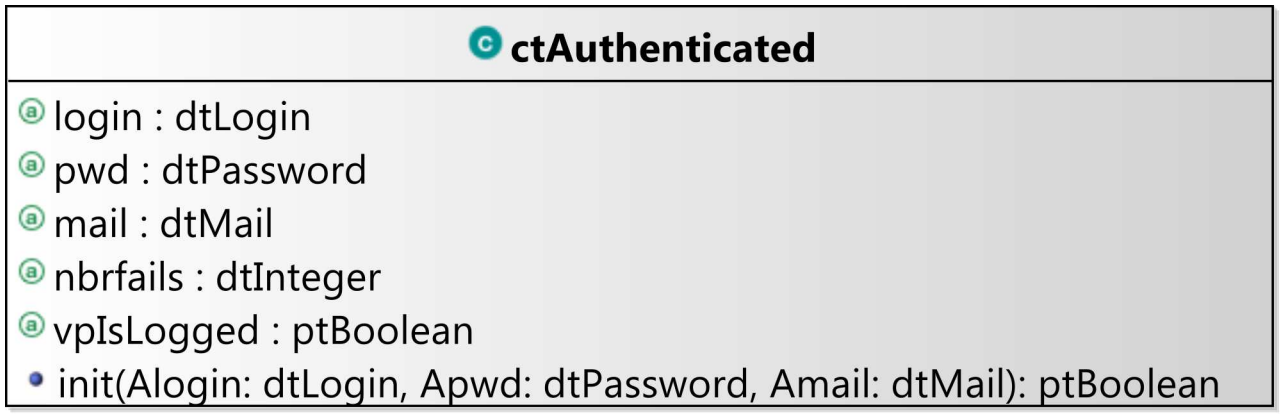

Figure 4.5: Concept Model - PrimaryTypes-Classes local view 08. .

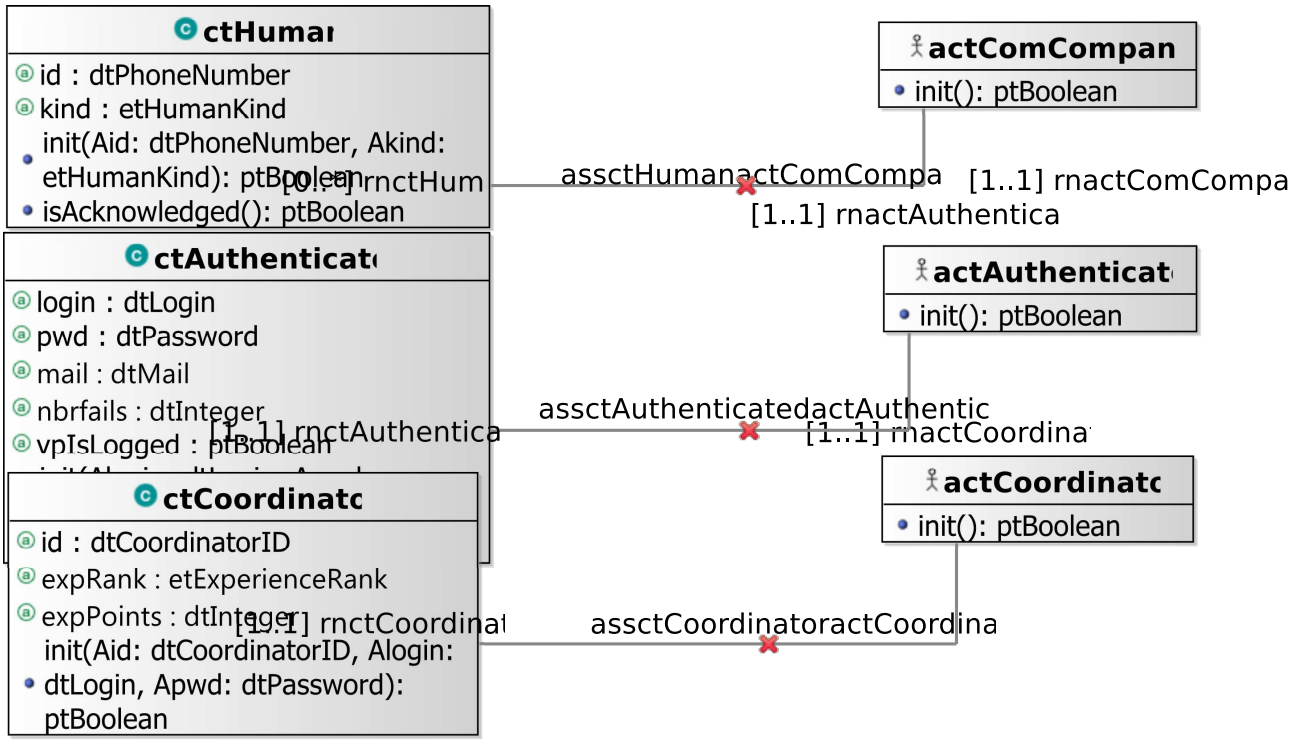

Figure 4.6: Concept Model - PrimaryTypes-Classes global view 01. Primary types class types global view - cm-pt-ct-gv-01 .

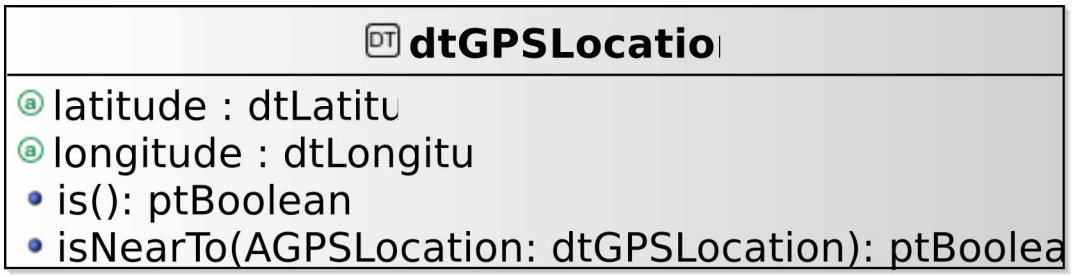

Figure 4.7: Concept Model - PrimaryTypes-Datatypes local view 06. .

4.2.2 Global view 01

Figure 4.8 shows a global view on the *iCrash* primary types datatype types.

4.3 SecondaryTypes-Datatypes

4.3.1 Local view 01

Figure 4.9 shows the local view of the secondary types datatype types.

4.4 Concept Model Types Descriptions

This section provides the textual descriptions of all the types defined in the concept model and that can be part of the graphical views provided.

4.4.1 Primary types - Class types descriptions

The table below is providing comments on the graphical views given for the class types of the primary types. Type logical operations are precisely specified in the operation model.

Classes	
ctAdministrator	
used to caracterize internally the entity that is responsible of administrating the *iCrash* system.	
operation	**init(Alogin:dtLogin, Apwd:dtPassword):ptBoolean**
	used to initialize the current object as a new instance of the ctAdministrator type.
ctAlert	
Used to model crisis alerts sent by any human having communication capability using communication companies belonging to the system's environment	
attribute	**comment: dtComment**
	a textual description providing unstructured information on the alert.
attribute	**id: dtAlertID**
	the alert unique identification information.
attribute	**instant: dtDateAndTime**
	the date and time at which the alert notification has been sent.
attribute	**location: dtGPSLocation**

continues in next page . . .

... Classes table continuation

	the position of the alert provided by the space-based satellite navigation system used by the human using the communication company to inform the *iCrash* system of a crisis.
attribute	**status: etAlertStatus** the alert validation status
operation	**init(Aid:dtAlertID, Astatus:etAlertStatus, Alocation:dtGPSLocation, Ainstant:dtDateAndTime, Acomment:dtComment):ptBoolean** used to initialize the current object as a new instance of the ctAlert type.
operation	**isSentToCoordinator(AactCoordinator:actCoordinator):ptBoolean** used to provide a given coordinator with current alert information.
ctAuthenticated	
used to model system's representation about actors that need to authenticate to access some specific functionalities.	
attribute	**login: dtLogin** an identifier for authentication.
attribute	**pwd: dtPassword** a key for authentication.
attribute	**vpIsLogged: ptBoolean** used to determine the access status.
operation	**init(Alogin:dtLogin, Apwd:dtPassword, Amail:dtMail):ptBoolean** used to initialize the current object as a new instance of the ctAuthenticated type.
ctCoordinator	
used to model system's representation about the actors that have the responsibility to handle alerts and crisis.	
attribute	**id: dtCoordinatorID** a unique identification information.
operation	**init(Aid:dtCoordinatorID, Alogin:dtLogin, Apwd:dtPassword):ptBoolean** used to initialize the current object as a new instance of the ctCoordinator type.
ctCrisis	
Used to model crisis that are infered from the reception of at least one alert message. Crisis aer entities that are handled by the *iCrash* system.	
attribute	**comment: dtComment** a textual description providing unstructured information on the crisis handling.
attribute	**id: dtCrisisID** the crisis unique identification information.
attribute	**instant: dtDateAndTime** the date and time at which the first related alert notification has been sent.
attribute	**location: dtGPSLocation** the position of the crisis equal by the one of the first alert received and associated to the crisis.
attribute	**status: etCrisisStatus** the crisis handling status.
attribute	**type: etCrisisType** an indication of the gravity of the crisis.
operation	**handlingDelayPassed():ptBoolean** used to determine if the crisis stood too longly in a pending status since last reminder.

continues in next page ...

. . . Classes table continuation

operation	**init(Aid:dtCrisisID, Atype:etCrisisType, Astatus:etCrisisStatus, Alocation:dtGPSLocation, Ainstant:dtDateAndTime, Acomment:dtComment):ptBoolean** used to initialize the current object as a new instance of the ctAlert type.
operation	**isAllocatedIfPossible():ptBoolean** used to allocate a crisis to a coordinator if any or to alert the administrator of crisis waiting to be handled.
operation	**isSentToCoordinator(AactCoordinator:actCoordinator):ptBoolean** used to provide a given coordinator with current crisis information.
operation	**maxHandlingDelayPassed():ptBoolean** used to determine if the crisis stood too longly in a pending status since its creation.
ctHuman	
used to model system's representation about the indirect actors that has alerted of potential crisis.	
attribute	**id: dtPhoneNumber** the number of the communication device used to send an alert to *iCrash* system.
attribute	**kind: etHumanKind** role with respect to the alert notified.
operation	**init(Aid:dtPhoneNumber, Akind:etHumanKind):ptBoolean** `init`: used to initialize the current object as a new instance of the ctHuman type.
ctPointOfInterest	
Used to model Points of interest added by the administrator into the system.	
attribute	**description: dtDescription** a textual description providing unstructured information on the Points of interest.
attribute	**id: dtPointOfInterestID** the Point of interest unique identification information.
attribute	**location: dtGPSLocation** the position of the Points of interest provided by the space-based satellite navigation system used by the administrator.
attribute	**type: etCategory** the category type.
operation	**init(Aid:dtPointOfInterestID, Atype:etCategory, Alocation:dtGPSLocation, Adescription:dtDescription):ptBoolean** used to initialize the current object as a new instance of the ctPointOfInterest type.
ctState	
used to model the system. Each system specified using Messiр must include a ctState class for which there is only one instance at any state of the abstract machine after creation.	
attribute	**clock: dtDateAndTime** used to represent the system local time.
attribute	**crisisReminderPeriod: dtSecond** used to define the delay between two reminders after which a reminder must be sent to the administrator and to the known coordinators to encourage them to handle the crisis.
attribute	**maxCrisisReminderPeriod: dtSecond** used to define the maximum delay after which the crisis is ramdomly allocated to a coordinator if any or an alert message is sent to the administrator in order to encourage him to add coordinators.
attribute	**nextValueForAlertID: dtInteger**

continues in next page . . .

... Classes table continuation

	`nextValueForAlertID: dtInteger`: used to associate each alert declared with a unique idenitification value.
attribute	**`nextValueForCrisisID: dtInteger`** used to associate each crisis declared with a unique idenitification value.
attribute	**`vpLastReminder: dtDateAndTime`** date and time of the last reminder.
attribute	**`vpStarted: ptBoolean`** used to avoid reacting to an actor message if the system is not started (i.e. oeCreateSystemAndEnvironment not executed).
operation	**`init(AnextValueForAlertID:dtInteger, AnextValueForCrisisID:dtIntege`** **`AnextValueForPointOfInterestID:dtInteger, Aclock:dtDateAndTime,`** **`AcrisisReminderPeriod:dtSecond, AmaxCrisisReminderPeriod:dtSecond,`** **`AvpLastReminder:dtDateAndTime, AvpStarted:ptBoolean):ptBoolean`** used to initialize the current object as a new instance of the ctState type.

4.4.2 Primary types - Datatypes types descriptions

The table below is providing comments on the graphical views given for the datatype types of the primary types.

DATATYPES	
dtAlertID	
A string used to identify alerts.	
extends	dtString
operation	**`is():ptBoolean`** used to determine which strings are considered as valid alert identifiers.
dtCaptcha	
a datatype made of an alphanumeric value used to test the user's human nature and slow down intrusions as also automatic responses.	
extends	dtString
operation	**`is():ptBoolean`** used to determine which strings are considered as valid captchas.
dtComment	
a datatype made of a string value used to receive,store and send textual information about crisis and alerts.	
extends	dtString
operation	**`is():ptBoolean`** used to determine which strings are considered as valid comments.
dtCoordinatorID	
A string used to identify coordinators.	
extends	dtString
operation	**`is():ptBoolean`** used to determine which strings are considered as valid coordinators identifiers.
dtCrisisID	
A string used to identify crisis.	
extends	dtString
operation	**`is():ptBoolean`** used to determine which strings are considered as valid crisis identifiers.

continues in next page ...

. . . Datatypes table continuation

dtDescription	
a datatype made of a string value used to store information about Points of interest.	
extends	dtString
operation	**is():ptBoolean** used to determine which strings are considered as valid descriptions.
dtExpPoints	
number of experience points accumulated by a coordinator based on the amount of crises handled	
extends	dtInteger
operation	**is():ptBoolean** used to determine which integers are considered to be DtExpPoints (value between 0 and 120)
dtGPSLocation	
used to define coordinates of geograpical positions on earth. It is defined a couple made of a latitude and a longitude.	
attribute	**latitude: dtLatitude** for the latitude part of the coordinate.
attribute	**longitude: dtLongitude** for the longitude part of the coordinate.
operation	**DistanceTo(AGPSLocation:dtGPSLocation):ptReal** used to determine how close two locations are and to give an integer value representing the distance.
operation	**is():ptBoolean** used to determine which couples are considered as valid dtGPSLocation values.
operation	**isNearTo(AGPSLocation:dtGPSLocation):ptBoolean** used to determine if locations are considered enough close to be treated as equivalent in the application domain context.
dtLatitude	
used to define a latitude value of a geograpical positions on earth.	
extends	dtReal
operation	**is():ptBoolean** used to determine which strings are considered as valid dtLatitude.
dtLogin	
a login string used to authentify an *iCrash* user	
extends	dtString
operation	**is():ptBoolean** used to determine which strings are considered as valid dtLogin.
dtLongitude	
used to define a longitude value of a geograpical positions on earth.	
extends	dtReal
operation	**is():ptBoolean** used to determine which strings are considered as valid dtLongitude.
dtMail	
a mail adrress used to reset the password of an *iCrash* user	
extends	dtString
operation	**is():ptBoolean** used to determine which strings are considered as valid dtMail.
dtPassword	

continues in next page . . .

... Datatypes table continuation

a password string used to authentify an *iCrash* user	
extends	dtString
operation	**generatePassword():dtPassword** used to generate a new password which is considered as secure fulfilling set requirements. For example at least 8 char. with at least one captial letter and at least one lower case and 2 digits.
operation	**is():ptBoolean** used to determine which strings are considered as valid dtPassword.
dtPhoneNumber	
a string used to store the phone number from the human declaring the crisis or the alert.	
extends	dtString
operation	**is():ptBoolean** used to determine which strings are considered as valid dtPhoneNumber.
dtPointOfInterestID	
A string used to identify Points of interest.	
extends	dtString
operation	**is():ptBoolean** used to determine which strings are considered as valid Points of interest identifiers.

ENUMERATIONS	
etAlertStatus	
this type is used to indicate the different validation status of an alert.	
operation	**is():ptBoolean** used to determine which litteral belongs to the enumeration.
etCategory	
this type is used to indicate the different types of categories available for Points of interest.	
operation	**is():ptBoolean** used to determine which litteral belongs to the enumeration.
etCrisisStatus	
this type is used to indicate the different handling status of a crisis.	
operation	**is():ptBoolean** used to determine which litteral belongs to the enumeration.
etCrisisType	
this type is used to indicate the different types of a crisis.	
operation	**is():ptBoolean** used to determine which litteral belongs to the enumeration.
etExperienceRank	
the experience rank indicates the crisis level the coordinator is allowed to handle	
etHumanKind	
this type is used to indicate the kind of human that informs about a car crash crisis.	
operation	**is():ptBoolean** used to determine which litteral belongs to the enumeration.

4.4.3 Primary types - Association types descriptions

The table below is providing comments on the association types of the primary types.

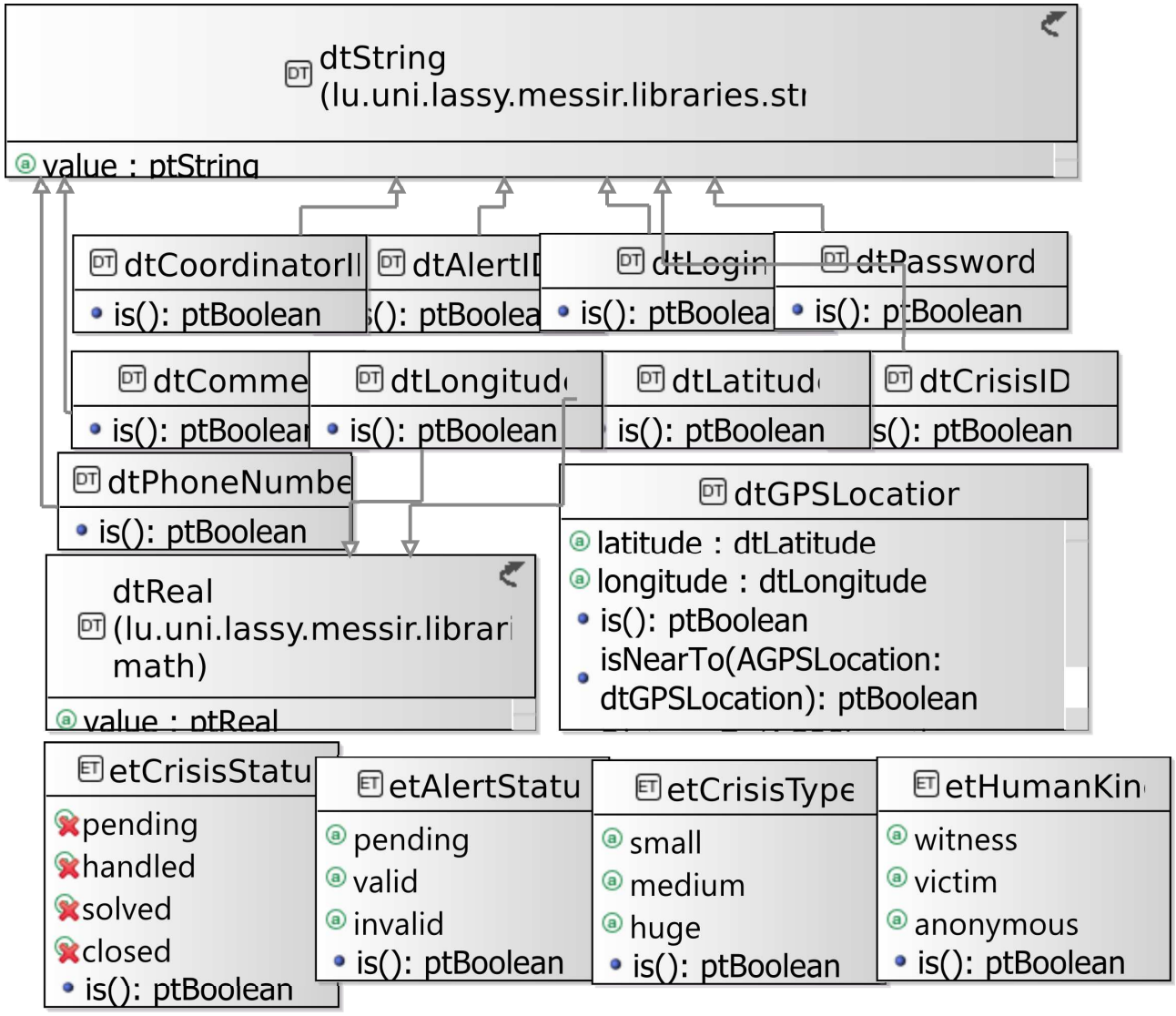

Figure 4.8: Concept Model - PrimaryTypes-Datatypes global view 01. global view of primary types datatype types - cm-pt-dt-gv-01 .

Figure 4.9: Concept Model - SecondaryTypes-Datatypes local view 01. Local view of the secondary types datatype types.

Associations
assctAlertctCrisis a crisis is related to one or more alerts as the alerts judged to concern all the same crisis due to their location. An alert alerts exactly one crisis.
assctAlertctHuman alerts are notified by human through the communication company. We need to keep an internal representation of those human to allow for communication of alert handling.
assctAuthenticatedactAuthenticated mainly used to determine if the login request of an authenticated actor can be granted based on the given credentials and the registered ones.
assctCoordinatoractCoordinator frequent messages must be sent to coordinator especially in relation to crisis they handle.
assctCrisisctCoordinator at any point in time we need to know if a coordinator is handling existing crisis or not.
assctHumanactComCompany in order to communicate with humans who informed about potential crisis, we need to record the communication company to use to send them messages.
assctPointOfInterestactAdministrator The administrator is the only one who can handle Points of interest and there is only one administrator.

4.4.4 Primary types - Aggregation types descriptions

There are no aggregation types for the primary types.

4.4.4.1 Primary types - Composition types descriptions

There are no composition types for the primary types.

4.4.5 Secondary types - Class types descriptions

There are no elements in this category in the system analysed.

4.4.6 Secondary types - Datatypes types descriptions

The table below is providing comments on the graphical views given for the datatype types of the secondary types.

Datatypes	
dtSMS a datatype made of a string value used to send textual information to human mobile devices.	
attribute	**`value: ptString`** the textual information.
operation	**`is():ptBoolean`** used to determine which strings are considered as valid comments.

4.4.7 Secondary types - Association types descriptions

There are no association types for the secondary types.

4.4.8 Secondary types - Aggregation types descriptions

There are no aggregation types for the secondary types.

4.4.9 Secondary types - Composition types descriptions

There are no composition types for the secondary types.

Chapter 5

Operation Model

This section contains the operation schemes of each operation defined in either an actor, its output interface, in a primary or secondary type (class, datatype or enumeration types). The Messir OCL code listing is joined to the comment table.

5.1 Environment - Out Interface Operation Scheme for actActivator

5.1.1 Operation Model for oeSetClock

The `oeSetClock` operation has the following properties:

Operation	
oeSetClock[proactive]	
An active message used to statically set the date and time information in the system's state.	
Parameters	
1	**AcurrentClock: dtDateAndTime** the date and time to be considered as the actual one.
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is supposed to be created and initialized and the provided date and time value is greater than the one known by the system.
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	the ctState instance post-state is updated to have its clock attribute equal to the given date and time.
Post-Condition (protocol)	
PostP 1	none

The listing 5.1 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Pre Protocol:*/
3 preP{let TheSystem: ctState in
```

```
4   let AvpStarted: ptBoolean in
5
6  /* PreP01 */
7   self.rnActor.rnSystem = TheSystem
8   and self.rnActor.rnSystem.vpStarted = AvpStarted
9   and AvpStarted = true
10  and TheSystem.clock.lt(AcurrentClock)}
11
12 /* Pre Functional:*/
13 preF{true}
14
15 /* Post Functional:*/
16 postF{let TheSystem: ctState in
17  self.rnActor.rnSystem = TheSystem
18
19 /* PostF01 */
20  and TheSystem@post.clock = AcurrentClock}
21
22 /* Post Protocol:*/
23 postP{ true}
```

Listing 5.1: Messir (MCL-oriented) specification of the operation *oeSetClock*.

5.1.2 Operation Model for oeSollicitateCrisisHandling

The `oeSollicitateCrisisHandling` operation has the following properties:

Operation	
oeSollicitateCrisisHandling[proactive] A proactive message (message of a pro-active actor with no parameter triggered automatically if the pre protocol condition is true) used to avoid crisis to stay too long in an not handled status.	
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	there exist some crisis that are in pending status and for which the duration between the current ctState clock information and the last reminder is greater than the crisis reminder period duration.
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	if there exist coordinators and crisis who stood in a not handled status more than the maximum allowed time then those crisis are randomly allocated to the existing coordinators.
PostF 2	for all other crisis who stood too longly in a not handled status but not more than the maximum delay allowed then a reminder message is sent to the administrator and all coordinator actors of the environment to sollicitate handling of those crisis.
Post-Condition (protocol)	
PostP 1	the value of the last reminder known by the system at post state is the system's clock value.

The listing 5.2 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Pre Protocol:*/
```

```
preP{let TheSystem: ctState in
 let AvpStarted: ptBoolean in
 let ColctCrisisToHandle:
    Bag(ctCrisis) in

 self.rnActor.rnSystem = TheSystem

/* PreP01 */
 and TheSystem.vpStarted

/* PreP02 */
 and TheSystem.rnctCrisis->select(handlingDelayPassed())
   = ColctCrisisToHandle
 and ColctCrisisToHandle->size().geq(1)}

/* Pre Functional:*/
preF{true}

/* Post Functional:*/
postF{let TheSystem: ctState in
 let AMessageForCrisisHandlers: dtComment in
 let ColctCrisisToAllocateIfPossible:Bag(ctCrisis) in

 self.rnActor.rnSystem = TheSystem
/* PostF01 */
 and TheSystem.rnctCrisis->select(maxHandlingDelayPassed())
   = ColctCrisisToAllocateIfPossible
 and ColctCrisisToAllocateIfPossible->forAll(isAllocatedIfPossible())

 /* PostF02 */
 and TheSystem.rnctCrisis->select(handlingDelayPassed())
 = ColctCrisisToHandle

 and ColctCrisisToHandle->msrColSubtract(ColctCrisisToAllocateIfPossible)
   = ColctCrisisToRemind

 and if (ColctCrisisToRemind->size().geq(1))
   then (AMessageForCrisisHandlers.value
      ='There are alerts pending since more than the defined delay. Please REACT !'
      and TheSystem.rnactAdministrator.
        rnInterfaceIN^ieMessage(AMessageForCrisisHandlers)
        and TheSystem.rnactCoordinator
          ->forAll(rnInterfaceIN^ieMessage(AMessageForCrisisHandlers))
      )
   else true
   endif}

/* Post Protocol:*/
postP{ let TheSystem: ctState in
 let TheClock: dtDateAndTime in

 self.rnActor.rnSystem = TheSystem
 and TheSystem.clock = TheClock
 and TheSystem@post.vpLastReminder = TheClock}
```

Listing 5.2: Messiр (MCL-oriented) specification of the operation *oeSollicitateCrisisHandling*.

Figure 5.1 shows concept model elements in the scope of the oeSollicitateCrisisHandling operation

5.2 Environment - Out Interface Operation Scheme for actAdministrator

5.2.1 Operation Model for oeAddCoordinator

The `oeAddCoordinator` operation has the following properties:

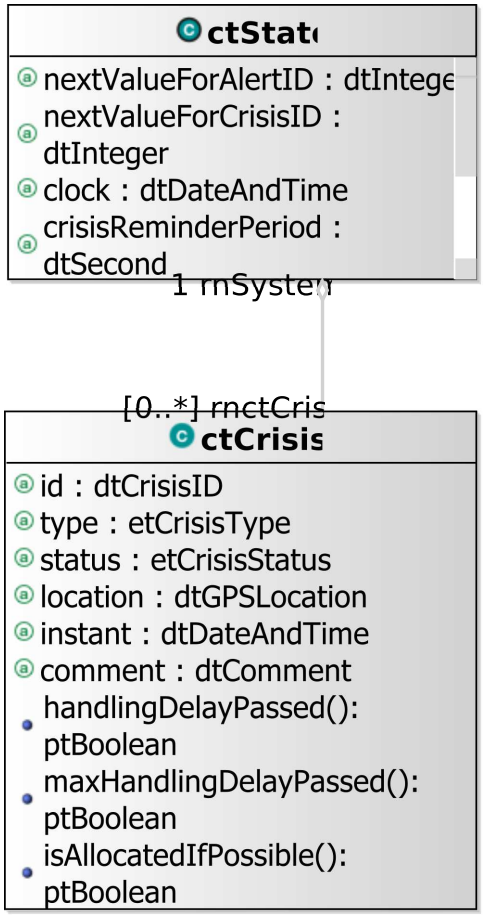

Figure 5.1: oeSollicitateCrisisHandling operation scope

OPERATION	
oeAddCoordinator	
sent to add a new coordinator in the system's post state and environment's post state.	
Parameters	
1	**AdtCoordinatorID: dtCoordinatorID** used to initialize the id field
2	**AdtLogin: dtLogin** used to initialize the login field
3	**AdtPassword: dtPassword** used to initialize the password field
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there cannot exist a ctCoordinator instance with the same `id` attribute as the one the administrator wants to delete.
Post-Condition (functional)	
PostF 1	the environment has a new instance of coordinator actor allowing for input/output message communication with the system.
PostF 2	the system's state has a new instance of ctCoordinator initialized with the given values.
PostF 3	the new actor instance and ctCoordinator instance are related.
PostF 4	the new actor instance and ctCoordinator instance are related according to the authenticated association.
PostF 5	the administrator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

The listing 5.3 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4   let TheActor:actAdministrator in
5
6   self.rnActor.rnSystem = TheSystem
7   and self.rnActor = TheActor
8
9  /* PreP01 */
10  and TheSystem.vpStarted = true
11 /* PreP02 */
12  and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Pre Functional:*/
15 preF{let TheSystem: ctState in
16  let TheActor:actAdministrator in
17  let ColctCoordinators:Bag(ctCoordinator) in
18
19  self.rnActor.rnSystem = TheSystem
20  and self.rnActor = TheActor
```

```
21 /* PreF01 */
22  and TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
23    = ColctCoordinators
24  and ColctCoordinators->isEmpty() = true}
25
26 /* Post Functional:*/
27 postF{let TheSystem: ctState in
28  let TheactCoordinator:actCoordinator in
29  let ThectCoordinator:ctCoordinator in
30  self.rnActor.rnSystem = TheSystem
31  and self.rnActor = TheActor
32 /* PostF01 */
33  TheactCoordinator.init()
34 /* PostF02 */
35  and ThectCoordinator.init(AdtCoordinatorID,AdtLogin,AdtPassword)
36
37 /* PostF03 */
38  and TheactCoordinator@post.rnctCoordinator = ThectCoordinator
39
40 /* PostF04 */
41  and ThectCoordinator@post.rnactAuthenticated = TheactCoordinator
42
43 /* PostF05 */
44  and TheActor.rnInterfaceIN^ieCoordinatorAdded()}
45
46 /* Post Protocol:*/
47 postP{ true}
```

Listing 5.3: Messir (MCL-oriented) specification of the operation *oeAddCoordinator*.

5.2.2 Operation Model for oeAddPointOfInterest

The `oeAddPointOfInterest` operation has the following properties:

OPERATION	
oeAddPointOfInterest	
sent to add a new Point of interest in the system's post state .	
Parameters	
1	**AetCategory: etCategory** used to initialize the category field
2	**AdtGPSLocation: dtGPSLocation** used to initialize the location field
3	**AdtDescription: dtDescription** used to initialize the description field
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there cannot exist a ctPointOfInterest instance with the same `id` attribute.
Post-Condition (functional)	
PostF 1	the ctState attribute for the next value for alert IDs is incremented by one at post.
PostF 2	the system's state has a new instance of ctPointOfInterest initialized with the given values.
PostF 3	the administrator actor is informed about the satisfaction of its request.

continues in next page . . .

. . . Operation table continuation

Post-Condition (protocol)
PostP 1

The listing 5.4 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4     let TheActor:actAdministrator in
5
6     self.rnActor.rnSystem = TheSystem
7     and self.rnActor = TheActor
8
9    /* PreP01 */
10    and TheSystem.vpStarted = true
11   /* PreP02 */
12    and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Post Functional:*/
15 postF{let TheSystem: ctState in
16   let TheactAdminitrator:actAdministrator in
17    let ActPointOfInterest:ctPointOfInterest in
18    let AetPointOfInterestCategory:etCategory in
19    let APointOfInterestDescription:dtDescription in
20    let AdtPointOfInterestID:dtPointOfInterestID in
21    let ColctPointOfInterest:Bag(ctPointOfInterest) in
22
23    self.rnActor.rnSystem = TheSystem
24    and self.rnActor = TheactComCompany
25
26    TheSystem.nextValueForPointOfInterestID=PrenextValueForPointOfInterestID
27    and PrenextValueForPointOfInterestID.add(1) = PostnextValueForPointOfInterestID
28
29   and TheSystem.rnctPointOfInterest->select(id.eq(PostnextValueForPointOfInterestID))
30      = ColctPointOfInterest
31    and ColctPointOfInterest->isEmpty() = true
32
33    and TheSystem@post.nextValueForPointOfInterestID = PostnextValueForPointOfInterestID
34   and TheSystem.nextValueForPointOfInterestID.todtString().eq(AdtPointOfInterestID)
35
36     /* PostF03 */
37
38   and ActPointOfInterest.init(AdtPointOfInterestID,
39          AetCategory,
40          AdtGPSLocation,
41          AdtDescription)
42     and TheactAdministrator.rnInterfaceIN^iePointOfInterestAdded()}
43
44 /* Post Protocol:*/
45 postP{ true}
```

Listing 5.4: Messir (MCL-oriented) specification of the operation *oeAddPointOfInterest*.

5.2.3 Operation Model for oeClosestToALocation

The `oeClosestToALocation` operation has the following properties:

OPERATION

continues in next page . . .

... Operation table continuation

oeClosestToALocation Sorts a list with the closest location to a given location to the furthest location to a given location.	
Parameters	
1	**AdtGPSLocation: dtGPSLocation**
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged).
Pre-Condition (functional)	
PreF 1	
Post-Condition (functional)	
PostF 1	returns a sorted list so that the closest location to a given location is on top to the furthest location to a given location at the end.
Post-Condition (protocol)	
PostP 1	

The listing 5.5 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4     let TheActor:actAdministrator in
5
6      self.rnActor.rnSystem = TheSystem
7      and self.rnActor = TheActor
8
9    /* PreP01 */
10    and TheSystem.vpStarted = true
11   /* PreP02 */
12    and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Post Functional:*/
15 postF{let TheSystem: ctState in
16    let TheActor:actAdministrator in
17    let CollctPointOfInterest:ctPointOfInterest in
18    let temp:ctPointOfInterest in
19    let j:dtInteger in
20    let flag:ptBoolean in
21    self.rnActor.rnSystem = TheSystem
22    and self.rnActor = TheActor
23   /* PostF01 */
24   and j = 0
25    TheSystem.rnctPointOfInterest->select(AllctPointOfInterest)
26    = CollctPointOfInterest
27    SortCtPointOfInterest = CollctPointOfInterest.sort()
28    and while (j.lt(SortctPointOfInterest.length-1)
29    if (SortCtPointOfInterset.get(j).location.distanceTo(AdtGPSLocation)< SortCtPointOfInterest.get(
          j+1).location.distanceTO(AdtGPSLocation) )
30     then (true and j+1)
31     else (false and break)
32     endif
33     )}
```

```
34
35 /* Post Protocol:*/
36 postP{ true}
```

Listing 5.5: Messir (MCL-oriented) specification of the operation *oeClosestToALocation*.

5.2.4 Operation Model for oeDeleteCoordinator

The `oeDeleteCoordinator` operation has the following properties:

OPERATION	
oeDeleteCoordinator	
sent to delete an existing coordinator in the system's post state and environment's post state.	
Parameters	
1	**AdtCoordinatorID: dtCoordinatorID** used for ctCoordinator instance retrieval
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one ctCoordinator instance with the same `id` attribute than the one the administrator wants to create.
Post-Condition (functional)	
PostF 1	the ctCoordinator class instance having the required id do not belong anymore to the post state as well as is related actCoordinator actor instance.
PostF 2	the administrator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

The listing 5.6 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Pre Protocol:*/
3 preP{let TheSystem: ctState in
4  let TheActor:actAdministrator in
5
6  self.rnActor.rnSystem = TheSystem
7  and self.rnActor = TheActor
8
9 /* PreP01 */
10  and TheSystem.vpStarted = true
11 /* PreP02 */
12  and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Pre Functional:*/
15 preF{let TheSystem: ctState in
16  let TheActor:actAdministrator in
17
18  self.rnActor.rnSystem = TheSystem
```

```
19  and self.rnActor = TheActor
20 /* PreF01 */
21  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
22  = ColctCoordinators
23  and ColctCoordinators->size().eq(1)}
24
25 /* Post Functional:*/
26 postF{let TheSystem: ctState in
27  let TheActor:actAdministrator in
28  let ThectCoordinator:ctCoordinator in
29  self.rnActor.rnSystem = TheSystem
30  and self.rnActor = TheActor
31 /* PostF01 */
32  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
33  = ThectCoordinator
34  and ThectCoordinator.rnactCoordinator->forAll(msrIsKilled)
35  and ThectCoordinator.msrIsKilled
36
37  /* PostF02 */
38  and TheActor.rnInterfaceIN^ieCoordinatorDeleted()
39
40  /* Post Protocol:*/
41 /* PostP01 */
42  and true}
43
44 /* Post Protocol:*/
45 postP{ true}
```

Listing 5.6: Messir (MCL-oriented) specification of the operation *oeDeleteCoordinator*.

5.2.5 Operation Model for oeDeletePointOfInterest

The `oeDeletePointOfInterest` operation has the following properties:

OPERATION	
oeDeletePointOfInterest	
sent to delete an existing Point of interest in the system's post state.	
Parameters	
1	**AdtPointOfInterestID: dtPointOfInterestID** used for ctPointOfInterest instance retrieval
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist only one ctPointOfInterest instance with the same `id` attribute than the one the administrator wants to delete.
Post-Condition (functional)	
PostF 1	the ctPointOfInterest class instance having the required id does not belong anymore to the post state.
PostF 2	the administrator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

The listing 5.7 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4     let TheActor:actAdministrator in
5
6      self.rnActor.rnSystem = TheSystem
7      and self.rnActor = TheActor
8
9    /* PreP01 */
10    and TheSystem.vpStarted = true
11   /* PreP02 */
12    and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Pre Functional:*/
15 preF{let TheSystem: ctState in
16    let TheActor:actAdministrator in
17
18    self.rnActor.rnSystem = TheSystem
19    and self.rnActor = TheActor
20   /* PreF01 */
21    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
22    = ColctPointOfInterest
23    and ColctPointOfInterest->size().eq(1)}
24
25 /* Post Functional:*/
26 postF{let TheSystem: ctState in
27    let TheActor:actAdministrator in
28    let ThectPointOfInterest:ctPointOfInterest in
29    self.rnActor.rnSystem = TheSystem
30    and self.rnActor = TheActor
31   /* PostF01 */
32    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
33    = ThectPointOfInterest
34    and ThectPointOfInterest.msrIsKilled
35
36    /* PostF02 */
37    and TheActor.rnInterfaceIN^iePointOfInterestDeleted()
38    and true}
39
40 /* Post Protocol:*/
41 postP{ true}
```

Listing 5.7: Messir (MCL-oriented) specification of the operation *oeDeletePointOfInterest*.

5.2.6 Operation Model for oeEditPointOfInterest

The `oeEditPointOfInterest` operation has the following properties:

Operation	
oeEditPointOfInterest	
sent to modify an existing Point of interest in the system's post state .	
Parameters	
1	**AdtPointsOfInterestID: dtPointOfInterestID**
	used to initialize the id field
2	**AetCategory: etCategory**
	used to initialize the category field
3	**AdtGPSLocation: dtGPSLocation**
	used to initialize the location field
4	**AdtDescription: dtDescription**

continues in next page . . .

... Operation table continuation

	used to initialize the description field
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exists a ctPointOfInterest instance with the same `id` attribute.
Post-Condition (functional)	
PostF 1	The values for the attributes selected by the system's actor actAdministrator are modified and replaced by the given values.
PostF 2	the administrator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

The listing 5.8 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4     let TheActor:actAdministrator in
5
6     self.rnActor.rnSystem = TheSystem
7     and self.rnActor = TheActor
8
9    /* PreP01 */
10    and TheSystem.vpStarted = true
11   /* PreP02 */
12    and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Pre Functional:*/
15 preF{let TheSystem: ctState in
16    let TheActor:actAdministrator in
17
18    self.rnActor.rnSystem = TheSystem
19    and self.rnActor = TheActor
20   /* PreF01 */
21    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
22    = PointOfInterest
23    and PointOfInterest->size().eq(1)
24    /*PostF01*/
25    and PointOfInterest@post.Category = AetCategory
26    and PointOfInterest@post.Location = AdtGPSLocation
27    and PointOfInterest@post.Description = AdtDescription
28
29    /*PostF02*/
30    and TheActor.rnInterfaceIN^iePointOfInterestEdited()
31    and true}
```

Listing 5.8: Messir (MCL-oriented) specification of the operation *oeEditPointOfInterest*.

5.2.7 Operation Model for oeRankDownCoordinator

The `oeRankDownCoordinator` operation has the following properties:

OPERATION	
oeRankDownCoordinator	
sent to demote a coordinator to the rank below his current rank	
Parameters	
1	**AdtCoordinatorID: dtCoordinatorID**
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged).
Pre-Condition (functional)	
PreF 1	there exists one coordinator having the given id in the pre-state.
PreF 2	the coordinator specified by the id has a higher rank than Novice (i.e. either Intermediate or Expert).
Post-Condition (functional)	
PostF 1	a message is sent to the ctAdministrator, confirming the operation has been successful, else the administrator is notified that the operation was unsuccessful.
PostF 2	the ctCoordinator attribute expRank is decremented by one (i.e. expert to intermediate, intermediate to novice) and his expPoints are adjusted accordingly at post (i.e. expert to intermediate -> 20 ExpPoints, intermediate to novice -> 0 ExpPoints).
Post-Condition (protocol)	
PostP 1	

The listing 5.9 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem:ctState in
4    let Jack:actAdministrator in
5    self.rnActor.rnSystem = TheSystem
6    and self.rnActor = Jack
7
8    /* PreP01 */
9     and TheSystem.vpStarted = true
10   /* PreP02 */
11     and Jack.rnctAuthenticated.vpIsLogged = true}
12
13 /* Pre Functional:*/
14 preF{let TheSystem:ctState in
15   let Jack:actAdministrator in
16   self.rnActor.rnSystem = TheSystem
17   and self.rnActor = Jack
18
19   /* PreF01*/
20   TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
21    = theCoordinator
22    and theCoordinator->size().eq(1)
23
24    /*PreF02*/
25    and (theCoordinator.expRank.eq("Intermediate") or theCoordinator.expRank.eq("Expert"))}
26
27 /* Post Functional:*/
```

```
28 postF{let TheSystem:ctState in
29   let Jack:actAdministrator in
30   self.rnActor.rnSystem = TheSystem
31   and self.rnActor = Jack
32
33   TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
34    = theCoordinator
35
36   /*PostF01*/
37   and Jack.rnInterfaceIN^ieRankEdited()
38
39   /*PostF02*/
40
41   if(theCoordinator.expRank.eq("Expert") and theCoordinator.expPoints >= 60)then(theCoordinator.
       @post.expRank.eq("Intermediate") and theCoordinator.@post.expPoints = 20)
42   else(theCoordinator.@post.expRank.eq("Novice") and theCoordinator.@post.expPoints = 0)endif}
43
44 /* Post Protocol:*/
45 postP{ true}
```

Listing 5.9: Messir (MCL-oriented) specification of the operation *oeRankDownCoordinator*.

5.2.8 Operation Model for oeSelectCategories

The `oeSelectCategories` operation has the following properties:

OPERATION	
oeSelectCategories	
Used to select all the rows that contain the category selected by the Administrator.	
Parameters	
1	**AdtCategory: etCategory** used to initialize the category field
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctAdministrator instance is considered logged).
Pre-Condition (functional)	
PreF 1	
Post-Condition (functional)	
PostF 1	all the points of interest with the given category are selected
Post-Condition (protocol)	
PostP 1	

The listing 5.10 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Pre Protocol:*/
3 preP{let TheSystem: ctState in
4    let TheActor:actAdministrator in
5
6     self.rnActor.rnSystem = TheSystem
7     and self.rnActor = TheActor
```

```
8
9    /* PreP01 */
10    and TheSystem.vpStarted = true
11   /* PreP02 */
12    and TheActor.rnctAuthenticated.vpIsLogged = true}
13
14 /* Post Functional:*/
15 postF{let TheSystem: ctState in
16    let TheActor:actAdministrator in
17    let CollctPointOfInterest : ctPointOfInterest in
18    let SortedCollctPointOfInterest:ctPointOfInterest in
19
20    self.rnActor.rnSystem = TheSystem
21    and self.rnActor = TheActor
22   /* PostF01 */
23   // need iterator
24   and SortedCollctPointOfInterest = CollctPointOfInterest.sort(AetCategory)
25    and (while i.lt(SortCollctPointOfInterest)
26    if (CollctPointofInterest.get(i).category.equals(AetCatgory))
27     then (result =true and i++)
28     else (result = false and break)
29    endif
30    )
31
32   return result}
33
34 /* Post Protocol:*/
35 postP{ true}
```

Listing 5.10: Messir (MCL-oriented) specification of the operation *oeSelectCategories*.

5.3 Environment - Out Interface Operation Scheme for actAuthenticated

5.3.1 Operation Model for oeFillCaptcha

The `oeFillCaptcha` operation has the following properties:

Operation	
oeFillCaptcha	
sent to verify correctness of the captcha test to be able to continue the login procedure.	
Parameters	
1	**AdtCaptcha: dtCaptcha** first information used to determine the result of the captcha test, it should correspond to the given captcha
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor is not already logged in ! (i.e. the associated ctAuthenticated instance is not considered logged)
PreP 3	the nbrfails attribute of the ctAuthenticated instance should be higher or equal 3 for a captcha test to be asked (therefore also to be filled).
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	

continues in next page . . .

... Operation table continuation

PostF 1	if the captcha provided by the actor corresponds to the one he should provide then a success message (captcha was filled correctly) is sent to the actor; else the actor is notified that he gave incorrect data and needs to give the test another try.
Post-Condition (protocol)	
PostP 1	if the captcha test is successful the nbrfails attribute goes down to zero again.

The listing 5.11 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem:ctState in
4   let Bob:actAuthenticated in
5   self.rnActor.rnSystem = TheSystem
6   and self.rnActor = Bob
7
8  /* PreP01 */
9   and TheSystem.vpStarted = true
10 /* PreP02 */
11  and TheActor.rnctAuthenticated.vpIsLogged = false
12  /* PreP03 */
13  and TheActor.rnctAuthenticated.nbrfails >= 3}
14
15 /* Pre Functional:*/
16 preF{true}
17
18 /* Post Functional:*/
19 postF{let TheSystem: ctState in
20  let Bob:actAuthenticated in
21
22  let AptStringMessageForTheactAuthenticated:ptString in
23  let AdtNewCaptcha:dtCaptcha in
24
25  self.rnActor.rnSystem = TheSystem
26  and self.rnActor = TheactAuthenticated
27
28  and /* PostF01 */
29    if (TheActor.rnctAuthenticated.nextValueForCapthca
30      = AdtCaptcha)
31    then
32      (TheSystem.nextValueForCaptcha = AdtNewCaptcha.generateCaptcha()
33
34      and AptStringMessageForTheactAuthenticated.eq('The captcha test was successful, you may now
       try to login again.')
35       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
36       )
37    else (AptStringMessageForTheactAuthenticated
38       .eq('Wrong captcha, you need to succeed the captcha tets in order to be able to login. A new
       captcha will be send.')
39       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
40       )
41    endif}
42
43 /* Post Protocol:*/
44 postP{ let TheSystem: ctState in
45  let Bob:actAuthenticated in
46
47  self.rnActor.rnSystem = TheSystem
48  and self.rnActor = TheactAuthenticated
49 /* PostP01 */
50  if (TheActor.rnctAuthenticated.nextValueForCapthca = AdtCaptcha
51  )
52  then (Bob.rnctAuthenticated@post.nbrfails = 0)
```

```
53  else true
54  endif}
```

Listing 5.11: Messir (MCL-oriented) specification of the operation *oeFillCaptcha*.

5.3.2 Operation Model for oeResetPassword

The `oeResetPassword` operation has the following properties:

Operation	
oeResetPassword sent to request a password reset to be able to request access secured system operations again.	
Parameters	
1	**AdtLogin: dtLogin** This information is used to reset a specific password which will be sent on the user's e-mail
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor is not already logged in ! (i.e. the associated ctAuthenticated instance is not considered logged)
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	if the login provided by the actor corresponds to the one that belongs to the ctAuthenticated instance he is related to then a successful password reset message is sent to the actor and a message is sent to the actor Administrator that tells which user resets his password; else the actor is notified that he gave incorrect data and all the administrator actors existing in the environment are notified of an intrusion attempt.
Post-Condition (protocol)	
PostP 1	if the reset information is correct then the actor is known to have a new password ! (i.e. the associated ctAuthenticated instance with given login will get a new password.)

The listing 5.12 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem:ctState in
4   let Bob:actAuthenticated in
5   self.rnActor.rnSystem = TheSystem
6   and self.rnActor = Bob
7
8  /* PreP01 */
9   and TheSystem.vpStarted = true}
10
11 /* Pre Functional:*/
12 preF{true}
13
14 /* Post Functional:*/
15 postF{let TheSystem: ctState in
16  let Bob:actAuthenticated in
17
```

```
18  let AptStringMessageForTheactAuthenticated: ptString in
19  let AptStringMessageForTheactAdministrator:ptString in
20  let AdtNewPassword:dtPassword in
21
22  self.rnActor.rnSystem = TheSystem
23  and self.rnActor = Bob
24
25  and /* PostF01 */
26    if (Bob.rnctAuthenticated.login
27      = AdtLogin
28      )
29    then (
30
31      Bob.rnctAuthenticated.pwd = AdtNewPassword.generatePassword()
32
33       and AptStringMessageForTheactAuthenticated.eq('Your password has been successfully reseted,
      please check your mails.')
34       and Bob.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
35
36       and AptStringMessageForTheactAdministrator.eq(AdtLogin)
37       and AptStringMessageForTheactAdministrator.dtStringConcat('has reset his password.')
38       and TheSystem.rnactAdministrator
39         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
40
41       )
42    else (AptStringMessageForTheactAuthenticated
43       .eq('Wrong identification information ! Please try again ...')
44       and Bob.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
45       and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
46       and TheSystem.rnactAdministrator
47         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
48       )
49    endif}
50
51 /* Post Protocol:*/
52 postP{ if(TheactAuthenticated.rnctAuthenticated.login = AdtLogin
53  )
54  then ()
55
56  else true
57  endif}
```

Listing 5.12: Messir (MCL-oriented) specification of the operation *oeResetPassword*.

5.3.3 Operation Model for oeLogin

The `oeLogin` operation has the following properties:

OPERATION	
oeLogin	
sent to request authorization to request access secured system operations.	
Parameters	
1	**AdtLogin: dtLogin** first information used to determine accessibility rights for the actual actor.
2	**AdtPassword: dtPassword** second information used to determine accessibility rights for the actual actor.
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor is not already logged in ! (i.e. the associated ctAuthenticated instance is not considered logged)

continues in next page . . .

... Operation table continuation

PreP 3	the nbrfails attribute is less than 3 (which means the actor attempted less than three times in a row a login without captcha test)
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	if the login and password provided by the actor correspond to the ones that belong to the ctAuthenticated instance he is related to then a welcome message is sent to the actor (n.b. the logged status is changed as a post-protocol condition); else the actor is notified that he gave incorrect data and all the administrator actors existing in the environment are notified of an intrusion attempt.
Post-Condition (protocol)	
PostP 1	if the authentication information is correct then the actor is known to be logged in ! (i.e. the associated ctAuthenticated instance with given login and password is considered logged)
PostP 2	if the authentication information is correct then the actor number of failed attempts(nbrfails) goes down to zero to ensure a correct initialization.

The listing 5.13 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4   let TheActor:actAuthenticated in
5   self.rnActor.rnSystem = TheSystem
6   and self.rnActor = TheActor
7
8  /* PreP01 */
9   and TheSystem.vpStarted = true
10 /* PreP02 */
11  and TheActor.rnctAuthenticated.vpIsLogged = false
12  /* PreP03 */
13  and TheActor.rnctAuthenticated.nbrfails < 3}
14
15 /* Pre Functional:*/
16 preF{/* PreF01 */
17 true}
18
19 /* Post Functional:*/
20 postF{let TheSystem: ctState in
21  let TheactAuthenticated:actAuthenticated in
22
23  let AptStringMessageForTheactAuthenticated: ptString in
24  let AptStringMessageForTheactAdministrator:ptString in
25
26  self.rnActor.rnSystem = TheSystem
27  and self.rnActor = TheactAuthenticated
28
29  and /* PostF01 */
30    if (TheactAuthenticated.rnctAuthenticated.pwd
31      = AdtPassword
32      and TheactAuthenticated.rnctAuthenticated.login
33        = AdtLogin
34      )
35    then (AptStringMessageForTheactAuthenticated.eq('You are logged ! Welcome ...')
36       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
37       )
38    else (AptStringMessageForTheactAuthenticated
39       .eq('Wrong identification information ! Please try again ...')
```

```
40        and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
41        and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
42        and TheSystem.rnactAdministrator
43          .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
44        )
45     endif}
46
47  /* Post Protocol:*/
48  postP{ let TheSystem: ctState in
49   let TheactAuthenticated:actAuthenticated in
50
51   self.rnActor.rnSystem = TheSystem
52   and self.rnActor = TheactAuthenticated
53  /* PostP01 */
54   if (TheactAuthenticated.rnctAuthenticated.pwd = AdtPassword
55     and TheactAuthenticated.rnctAuthenticated.login = AdtLogin
56     )
57   then (TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = true
58     and TheactAuthenticated.rnctAuthenticated@post.nbrfails = 0
59   )
60   else (TheactAuthenticated.rnctAuthenticated@post.nbrfails = TheactAuthenticated.
        rnctAuthenticated@post.nbrfails@pre + 1)
61   endif}
```

Listing 5.13: Messir (MCL-oriented) specification of the operation *oeLogin*.

5.3.4 Operation Model for oeLogout

The `oeLogout` operation has the following properties:

OPERATION	
oeLogout	
sent to end the secured access to specific system operations.	
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor is currently logged in ! (i.e. the associated ctAuthenticated instance is considered logged)
Pre-Condition (functional)	
PreF 1	
Post-Condition (functional)	
PostF 1	a logout confirmation message is sent to the actor (n.b. the logged status is changed as a post-protocol condition)
Post-Condition (protocol)	
PostP 1	the actor is known to be logged out ! (i.e. the associated ctAuthenticated instance with given login and password is considered logged out)

The listing 5.14 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4   let TheActor:actAdministrator in
```

```
5   self.rnActor.rnSystem = TheSystem
6   and self.rnActor = TheActor
7
8  /* PreP01 */
9   and TheSystem.vpStarted = true
10 /* PreP02 */
11  and TheActor.rnctAuthenticated.vpIsLogged = true}
12
13 /* Pre Functional:*/
14 preF{/* PreF01 */
15 true}
16
17 /* Post Functional:*/
18 postF{let TheSystem: ctState in
19  let TheactAuthenticated:actAuthenticated in
20  let AptStringMessageForTheactAuthenticated: ptString in
21
22  self.rnActor.rnSystem = TheSystem
23  and self.rnActor = TheactAuthenticated
24
25  /* PostF01 */
26  AptStringMessageForTheactAuthenticated.eq('You are logged out ! Good Bye ...')
27  and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)}
28
29 /* Post Protocol:*/
30 postP{ let TheSystem: ctState in
31  let TheactAuthenticated:actAuthenticated in
32
33  self.rnActor.rnSystem = TheSystem
34  and self.rnActor = TheactAuthenticated.asSet
35 /* PostP01 */
36  TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = false}
```

Listing 5.14: Messir (MCL-oriented) specification of the operation *oeLogout*.

5.4 Environment - Out Interface Operation Scheme for actComCompany

5.4.1 Operation Model for oeAlert

The `oeAlert` operation has the following properties:

OPERATION	
oeAlert	
Any human having a phone able to connect to the communication companies using the *iCrash* system can send his company an sms message with structured information in order to declare an alert.	
Parameters	
1	**AetHumanKind: etHumanKind** the kind of human informing of an alert.
2	**AdtDate: dtDate** the date of the alert
3	**AdtTime: dtTime** the time of the alert
4	**AdtPhoneNumber: dtPhoneNumber** the phone number of the human sending the alert SMS message
5	**AdtGPSLocation: dtGPSLocation** the GPS position of the phone at the date and time the message was sent.
6	**AdtComment: dtComment**

continues in next page . . .

... Operation table continuation

	a free text message sent by the human providing information on the alert that he wants to declare
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is supposed to be created and initialized.
Pre-Condition (functional)	
PreF 1	the date and time the alert is declared is supposed to be in the past with respect to the current time known by the system.
Post-Condition (functional)	
PostF 1	the ctState attribute for the next value for alert IDs is incremented by one at post.
PostF 2	a new alert instance exists in the post state with status pending, instant information (resp. GPS location and comment) based on date and time provided (resp. position and comment); and with alert ID being a string conversion of the dtInteger value available in the pre state in the ctState instance.
PostF 3	if there exist no already registered alert near to the alert currently declared then a new crisis is added in the post state and initialized with: its ID being the one provided by the ctState instance (which is incremented by one in the post state), its type considered as small, its status being pending, its declared time being the same than the alert and a default comment indicating that a report will come later on. else the crisis to which the new alert must be related to is the one related to any alert nearby in the pre-state.
PostF 4	the post state relates the new alert to the previously characterized crisis.
PostF 5	if there is no ctHuman instance having same phone number and same kind in the pre-state then a new one is added in the post-state with given phone number and kind and is associated to the communication company actor used to declare the alert. else the pre-state one is chosen
PostF 6	and this specified ctHuman is related to the new alert thus indicating he has signled the alert.
Post-Condition (protocol)	
PostP 1	none

The listing 5.15 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem: ctState in
4   self.rnActor.rnSystem = TheSystem
5
6  /* PreP01 */
7   and TheSystem.vpStarted = true}
8
9  /* Pre Functional:*/
10 preF{let TheSystem: ctState in
11  self.rnActor.rnSystem = TheSystem
12
13 /* PreF01 */
14  and (TheSystem.clock.date.gt(AdtDate)
15     or (TheSystem.clock.date.eq(AdtDate)
16       and TheSystem.clock.time.gt(AdtTime)
17      )
```

```
18     )}
19
20  /* Post Functional:*/
21  postF{let TheSystem: ctState in
22
23   let ActHuman:ctHuman in
24   let TheactComCompany:actComCompany in
25   let ActAlert:ctAlert in
26   let AAlertInstant:dtDateAndTime in
27   let AetAlertStatus:etAlertStatus in
28   let ActAlertNearBy:ctAlert in
29   let ActCrisis:ctCrisis in
30   let AdtCrisisID:dtCrisisID in
31   let AetCrisisType:etCrisisType in
32   let AetCrisisStatus:etCrisisStatus in
33   let ACrisisInstant:dtDateAndTime in
34   let ACrisisdtComment:dtComment in
35   let AptStringMessage:ptString in
36   let AdtSMS:dtSMS in
37   let AdtAlertID:dtAlertID in
38
39   self.rnActor.rnSystem = TheSystem
40   and self.rnActor = TheactComCompany
41  /* PostF01 */
42   TheSystem.nextValueForAlertID=PrenextValueForAlertID
43   and PrenextValueForAlertID.add(1) = PostnextValueForAlertID
44   and TheSystem@post.nextValueForAlertID = PostnextValueForAlertID
45
46   /* PostF02 */
47  and AAlertInstant.date=AdtDate
48  and AAlertInstant.time=AdtTime
49
50  and AetAlertStatus=pending
51
52  and TheSystem.nextValueForAlertID.todtString().eq(AdtAlertID)
53
54  and ActAlert.init(AdtAlertID,
55           AetAlertStatus,
56           AdtGPSLocation,
57           AAlertInstant,
58           AdtComment)
59
60   /* PostF03 */
61  and TheSystem.rnctAlert.select(location.isNearTo(AdtGPSLocation)) = ColctAlertsNearBy
62  and if (ColctAlertsNearBy->size()=0)
63    then (TheSystem.nextValueForCrisisID = PrenextValueForCrisisID
64       and PrenextValueForCrisisID.add(1) = PostnextValueForCrisisID
65       and TheSystem@post.nextValueForCrisisID = PostnextValueForCrisisID
66       and TheSystem.nextValueForCrisisID.todtString().eq(AdtCrisisID)
67       and AdtCrisisType = small
68       and AetCrisisStatus = pending
69       and ACrisisInstant= AAlertInstant
70       and ACrisisdtComment = 'no reporting yet defined'
71       and ActCrisis.init( AdtCrisisID,
72                 AdtCrisisType,
73                 AetCrisisStatus,
74                 AdtGPSLocation,
75                 ACrisisInstant,
76                 ACrisisdtComment)
77       )
78   else (ColctAlertsNearBy.rnTheCrisis->msrAny(true) = ActCrisis)
79   endif
80
81   /* PostF04 */
82  and ActAlert@post.rnTheCrisis = ActCrisis
83
84  /* PostF05 */
85  and TheSystem.rnctHuman->select(id.eq(AdtPhoneNumber)) = HumanCol1
86
87  and HumanCol1->select(kind.etEq(AetHumanKind)) = HumanCol2
```

```
88  and if (HumanCol2->msrIsEmpty)
89    then (ActHuman.init(AdtPhoneNumber,AetHumanKind)
90       and ActHuman@post.rnactComCompany = TheactComCompany
91       )
92    else (HumanCol2->any(true) = ActHuman)
93    endif
94
95   and ActHuman.rnSignaled->msrIncluding(ActAlert) = ColAlerts
96
97   and ActHuman@post.rnSignaled = ColAlerts
98
99  /* PostF06 */
100 AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
101 and TheactComCompany.rnInterfaceIN^ieSmsSend(AdtPhoneNumber,AdtSMS)}
102
103 /* Post Protocol:*/
104 postP{ true}
```

Listing 5.15: Messir (MCL-oriented) specification of the operation *oeAlert*.

Figure 5.2 shows concept model elements in the scope of the oeAlert operation

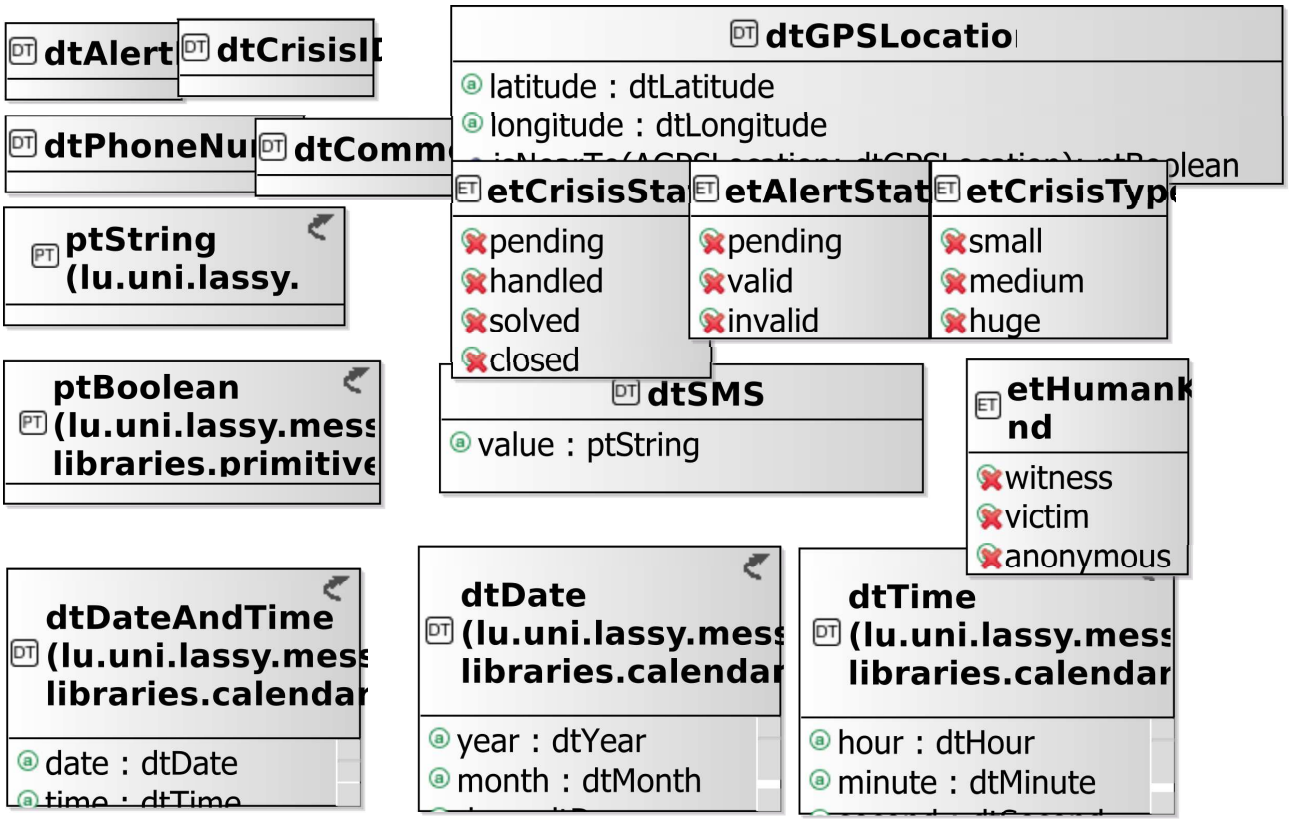

Figure 5.2: oeAlert operation scope

Figure 5.3 shows concept model elements in the scope of the oeAlert operation

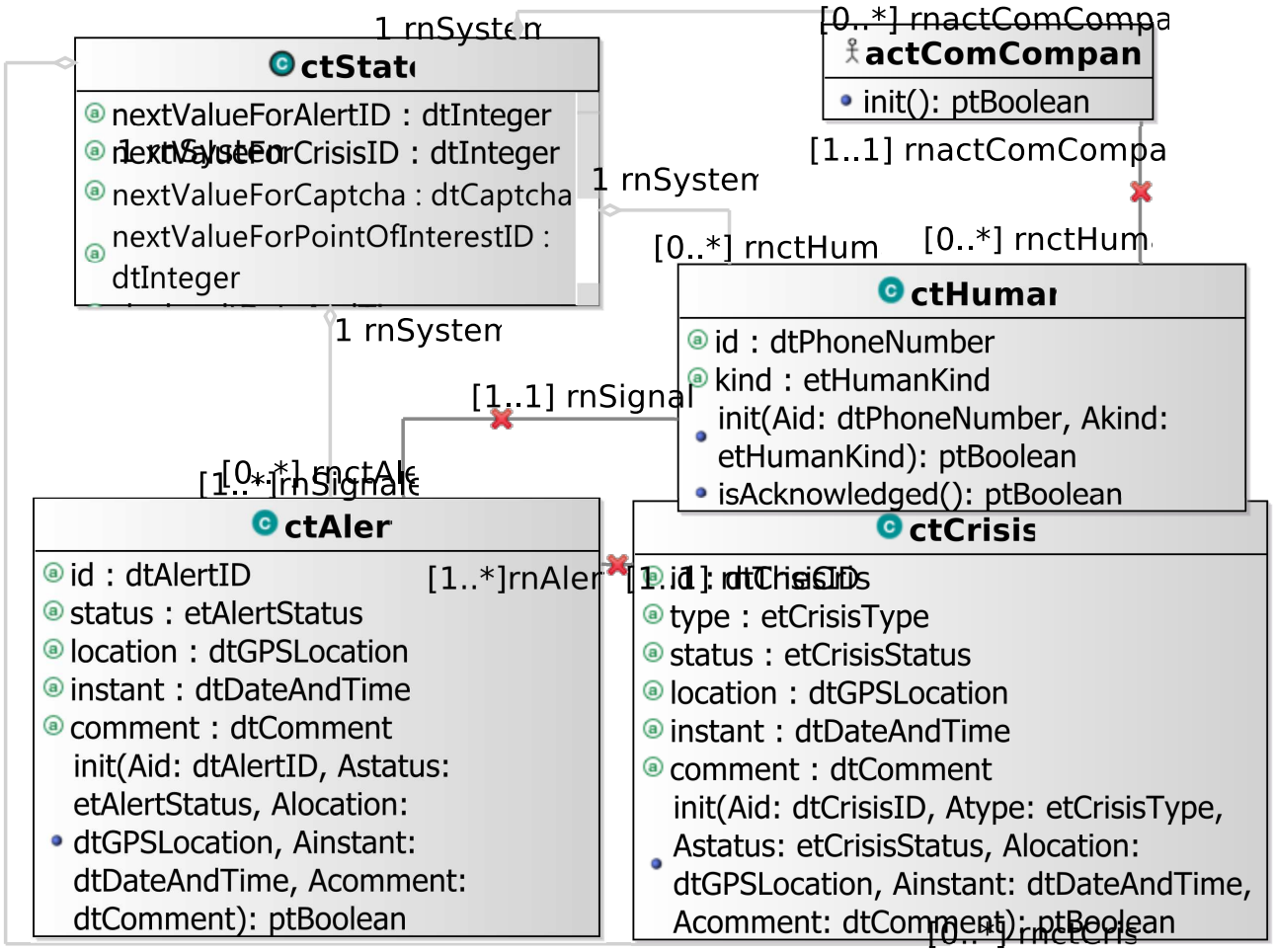

Figure 5.3: oeAlert operation scope

5.5 Environment - Out Interface Operation Scheme for actCoordinator

5.5.1 Operation Model for oeCloseCrisis

The `oeCloseCrisis` operation has the following properties:

Operation	
oeCloseCrisis	
sent to indicate that a crisis should be considered as closed.	
Parameters	
1	**AdtCrisisID: dtCrisisID** the identification information used to determine the crisis to close
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one ctCrisis instance with the same `id` attribute value as the one provided by the coordinator actor who wants to close.
Post-Condition (functional)	
PostF 1	the ctCrisis class instance having the provided id is considered closed in the post state.
PostF 2	There is no handler declared in the system as associated to the crisis.
PostF 3	all the alert instances associated to this crisis do not belong any more to the system's post state.
PostF 4	The coordinator's experience points increase by 1 in the post state and he is promoted if his points reach the threshold needed (Intermediate -> 20, Expert -> 60).
PostF 5	the coordinator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

The listing 5.16 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem:ctState in
4    let Jack:actCoordinator in
5    self.rnActor.rnSystem = TheSystem
6    and self.rnActor = Jack
7
8    /* PreP01 */
9     and TheSystem.vpStarted = true
10   /* PreP02 */
11     and Jack.rnctAuthenticated.vpIsLogged = true}
12
13 /* Pre Functional:*/
14 preF{let TheSystem:ctState in
15   let Jack:actCoordinator in
16   self.rnActor.rnSystem = TheSystem
```

```
17    and self.rnActor = Jack
18
19    /* PreF01*/
20    TheSystem.rnctCrisis->select(id.eq(AdtCrisisID))
21     = theCrisis
22     and theCrisis->size().eq(1)}
23
24  /* Post Functional:*/
25  postF{let TheSystem:ctState in
26    let Jack:actCoordinator in
27    self.rnActor.rnSystem = TheSystem
28    and self.rnActor = Jack
29
30    TheSystem.rnctCrisis->select(id.eq(AdtCrisisID))
31     = theCrisis
32
33    /*PostF01*/
34    theCrisis.@post.status="closed"
35
36    /*PostF04*/
37    if(Jack.expPoints == 59)then(theCoordinator.@post.expRank.eq("Expert"))
38    else if(Jack.expPoints == 19) then (theCoordinator.@post.expRank.eq("Intermediate")) else ()
          endif endif
39    if(Jack.expPoints<120)then(Jack.@post.expPoints=Jack.expPoints+1)else() endif}
40
41  /* Post Protocol:*/
42  postP{ true}
```

Listing 5.16: Messir (MCL-oriented) specification of the operation *oeCloseCrisis*.

5.5.2 Operation Model for oeGetAlertsSet

The `oeGetAlertsSet` operation has the following properties:

OPERATION	
oeGetAlertsSet	
sent to request all the ctAlert instances having a specific status.	
Parameters	
1	**AetAlertStatus: etAlertStatus** the criteria used to select the alerts to send back to the actor
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	the post state is the one obtained by satisfying the `isSentToCoordinator` predicate for each alert having the provided status and for the actor sending the message. (cf. specification of `isSentToCoordinator` predicate given for the `ctAlert` type.
Post-Condition (protocol)	
PostP 1	none

5.5.3 Operation Model for oeGetCrisisSet

The `oeGetCrisisSet` operation has the following properties:

OPERATION	
oeGetCrisisSet	
sent to request all the ctCrisis instances having a specific status.	
Parameters	
1	**AetCrisisStatus: etCrisisStatus** the status information used to determine the crisis to send back to the actor
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	the post state is the one obtained by satisfying the `isSentToCoordinator` predicate for each crisis having the provided status and type for the actor sending the message `ieSendACrisis`. (cf. specification of `isSentToCoordinator` predicate given for the `ctCrisis` type. The crisis is only sent to the coordinator if it matches his rank (small crisis-> novice rank, medium crisis -> intermediate rank, huge crisis -> expert rank).
Post-Condition (protocol)	
PostP 1	none

The listing 5.17 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{let TheSystem:ctState in
4    let Jack:actCoordinator in
5    self.rnActor.rnSystem = TheSystem
6    and self.rnActor = Jack
7
8    /* PreP01 */
9     and TheSystem.vpStarted = true
10   /* PreP02 */
11     and Jack.rnctAuthenticated.vpIsLogged = true}
12
13 /* Pre Functional:*/
14 preF{true}
15
16 /* Post Functional:*/
17 postF{let TheSystem:ctState in
18  let Jack:actCoordinator in
19  self.rnActor.rnSystem = TheSystem
20  and self.rnActor = Jack
21
22  let requiredType:etCrisisType in
23  let j:dtInteger in
24  if(Jack.expRank=="Novice") then(requiredType="small")
25  else if(Jack.expRank=="Intermediate")then(requiredType="medium")
```

```
26  else (requiredType="huge") endif endif
27
28  TheSystem.rnctCrisis->select(status.eq(AetCrisisStatus) and type.eq(requiredType))
29   = CollCtCrises
30   and j=0
31
32   //PostF01
33    while (j.lt(CollCtCrises.length-1)
34    CollCtCrises.get(j).isSentToCoordinator()
35    and j+1
36
37     )}
```

Listing 5.17: Messir (MCL-oriented) specification of the operation *oeGetCrisisSet*.

5.5.4 Operation Model for oeInvalidateAlert

The `oeInvalidateAlert` operation has the following properties:

OPERATION	
oeInvalidateAlert	
sent to indicate that an alert should be considered as closed.	
Parameters	
1	**AdtAlertID: dtAlertID** the identification information used to determine the alert to close
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one ctAlert instance with the same `id` attribute value as the one provided by the coordinator actor who wants to close.
Post-Condition (functional)	
PostF 1	the ctAlert class instance having the provided id is considered closed in the post state.
PostF 2	the coordinator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

5.5.5 Operation Model for oeReportOnCrisis

The `oeReportOnCrisis` operation has the following properties:

OPERATION	
oeReportOnCrisis	
sent to update the textual information available for a specific handled crisis.	
Parameters	
1	**AdtCrisisID: dtCrisisID** the identification information used to determine the crisis to report on
2	**AdtComment: dtComment** the textual information commenting the crisis

continues in next page . . .

... Operation table continuation

Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one crisis in the pre state having the given id.
Post-Condition (functional)	
PostF 1	the comment attribute of the crisis instance having the given id is replaced by the given one and the requesting actor is notified of this update.
Post-Condition (protocol)	
PostP 1	none

5.5.6 Operation Model for oeSetCrisisHandler

The `oeSetCrisisHandler` operation has the following properties:

OPERATION	
oeSetCrisisHandler sent to declare himself as been the handler of a crisis having the specified id.	
Parameters	
1	**AdtCrisisID: dtCrisisID** the identification information used to determine the crisis
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	there exist one crisis having the given id in the pre-state.
Post-Condition (functional)	
PostF 1	the ctCrisis instance having the provided id is in handled status at poststate and is associated to the actor that sends the message (which himself is notified with a textual message as confirmation).
PostF 2	All the alerts related to this crisis are sent to the actor such that he can decide how to handle them.
PostF 3	if the crisis was already handled at pre-sate then the associated handler actor is notified about the change of handler for one of his crisis (n.b. it might be the same even if not relevant).
PostF 4	a message is sent to the communication company for any human related to an alert associated to the crisis. A human will receive as many messages as alerts he sent despite the fact that they might relate to the same crisis (i.e. one alert, one acknoledgement).
Post-Condition (protocol)	
PostP 1	none

5.5.7 Operation Model for oeSetCrisisStatus

The `oeSetCrisisStatus` operation has the following properties:

OPERATION	
oeSetCrisisStatus	
sent to define the handling status of a specific crisis.	
Parameters	
1	**AdtCrisisID: dtCrisisID** the identification information used to determine the crisis
2	**AetCrisisStatus: etCrisisStatus** the new status value
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one crisis in the pre state having the given id.
Post-Condition (functional)	
PostF 1	the crisis status attribute of the crisis instance having the given id is replaced by the given one and the requesting actor is notified of this update.
Post-Condition (protocol)	
PostP 1	none

5.5.8 Operation Model for oeSetCrisisType

The `oeSetCrisisType` operation has the following properties:

OPERATION	
oeSetCrisisType	
sent to define the gravity type of a specific crisis.	
Parameters	
1	**AdtCrisisID: dtCrisisID** the identification information used to determine the crisis
2	**AetCrisisType: etCrisisType** the new type value
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one crisis in the pre state having the given id.
Post-Condition (functional)	

continues in next page . . .

. . . Operation table continuation

PostF 1	the crisis type attribute of the crisis instance having the given id is replaced by the given one and the requesting actor is notified of this update.
Post-Condition (protocol)	
PostP 1	none

5.5.9 Operation Model for oeValidateAlert

The `oeValidateAlert` operation has the following properties:

OPERATION	
oeValidateAlert	
sent to indicate that a specific alert is not a fake.	
Parameters	
1	**AdtAlertID: dtAlertID** the identification information used to determine the alert instance
Return type	
ptBoolean	
Pre-Condition (protocol)	
PreP 1	the system is started
PreP 2	the actor logged previously and did not log out ! (i.e. the associated ctCoordinator instance is considered logged)
Pre-Condition (functional)	
PreF 1	it is supposed that there exist one ctAlert instance with the same `id` attribute value as the one provided by the coordinator actor who wants to validate.
Post-Condition (functional)	
PostF 1	the ctAlert class instance having the provided id is considered as valid in the post state and the coordinator actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

5.6 Environment - Out Interface Operation Scheme for actMsrCreator

5.6.1 Operation Model for oeCreateSystemAndEnvironment

The `oeCreateSystemAndEnvironment` operation has the following properties:

OPERATION	
oeCreateSystemAndEnvironment	
sent to request the initialization of the system's class instances and the environment actors instances.	
Parameters	
1	**AqtyComCompanies: ptInteger** the quantity of communication companies to create in the environment
Return type	
ptBoolean	
Pre-Condition (protocol)	

continues in next page . . .

. . . Operation table continuation

PreP 1	none
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	the ctState instance is initialized with the integer 1 for the crisis and alert counters used for their identifications, a value for the clock corresponding to a default inital time (i.e. January 1st, 1970) the crisis reminder period is set to 300 seconds, the maximum crisis reminder period is fixed to 1200 seconds (i.e. 20 minutes), an initial value for the automatic reminder period equal to the current date and time and the system is considered in a started state. **Those predicates must be satisfied first since all the other depend on the existence of a ctState instance !**
PostF 2	the `actMsrCreator` actor instance is initiated (remember that since the `oeCreateSystemAndEnvironment` is a special event it role is to make consistent the post state thus creating the actor and its interfaces is required even though the sending of this message logically would need the actor and its interfaces to already exist ...).
PostF 3	the environment for communication company actors, in the post state, is made of `AqtyComCompanies` instances allowing for receiving and sending messages to humans.
PostF 4	the environment for administrator actors, in the post state, is made of one instance.
PostF 5	the environment for activator actors, in the post state, is made of one instance allowing for automatic message sending based on current system's and environment state'.
PostF 6	the set of ctAdministrator instances at post is made of one instance initialized with 'icrashadmin' (resp. '7WXC1359') for login (resp. password) values.
PostF 7	the association between ctAdministrator and actAdministrator is made of one couple made of the conjointly specified instances.
Post-Condition (protocol)	
PostP 1	none is given since the only protocol variable to be modified in the post state is the one initialized with the ctState instance (i.e. vpStarted).

The listing 5.18 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Pre Protocol:*/
3  preP{true}
4
5  /* Pre Functional:*/
6  preF{true}
7
8  /* Post Functional:*/
9  postF{let TheSystem: ctState in
10  let AactMsrCreator: actMsrCreator in
11  let AactAdministrator: actAdministrator in
12  let AnextValueForAlertID: dtInteger in
13  let AnextValueForCrisisID: dtInteger in
14  let Aclock: dtDateAndTime in
15  let AcrisisReminderPeriod: dtSecond in
16  let AmaxCrisisReminderPeriod: dtSecond in
17  let AvpStarted: ptBoolean in
18
19  /* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
20  AnextValueForAlertID.value.eq(1)
21  and AnextValueForCrisisID.value.eq(1)
22  and Aclock.date.year.value = 1970
```

```
23  and Aclock.date.month.value = 01
24  and Aclock.date.day.value = 01
25  and Aclock.time.hour.value = 00
26  and Aclock.time.minute.value = 00
27  and Aclock.time.second.value = 00
28
29  and AcrisisReminderPeriod.value.eq(300)
30  and AmaxCrisisReminderPeriod.value.eq(1200)
31  and AvpStarted = true
32  and TheSystem.init(AnextValueForAlertID,
33            AnextValueForCrisisID,
34            Aclock,
35            AcrisisReminderPeriod,
36            AmaxCrisisReminderPeriod,
37            Aclock,
38            AvpStarted
39           )
40  /* PostF02*/
41  and AactMsrCreator.init()
42  /* PostF03 */
43  and let AactComCompanyCol: Bag(actComCompany) in
44  AactComCompanyCol->size() = AqtyComCompanies
45  AactComCompanyCol-> forAll(init())
46  /* PostF04*/
47  and AactAdministrator.init()
48  /* PostF05*/
49  and let AactActivator:actActivator in
50  AactActivator.init()
51 /* PostF06 */
52  and let ActAdministrator:ctAdministrator in
53    let AdtLogin:dtLogin in
54    let AdtPassword:dtPassword in
55    AdtLogin.value.eq('icrashadmin')
56    and AdtPassword.value.eq('7WXC1359')
57    and ActAdministrator.init(AdtLogin,AdtPassword)
58  /* PostF07*/
59  and ActAdministrator@post.rnactAuthenticated = AactAdministrator}
60
61 /* Post Protocol:*/
62 postP{ true}
```

Listing 5.18: Messir (MCL-oriented) specification of the operation *oeCreateSystemAndEnvironment*.

Figure 5.4 shows all the concept model elements in the scope of the oeCreateSystemAndEnvironment operation

5.7 Environment - Actor Operation Scheme for actMsrCreator

5.7.1 Operation Model for init

The `init` operation has the following properties:

Operation
init   used to create an instance of the actor together with its interface instances and update the assocations with the `ctState` instance.
Return type
ptBoolean

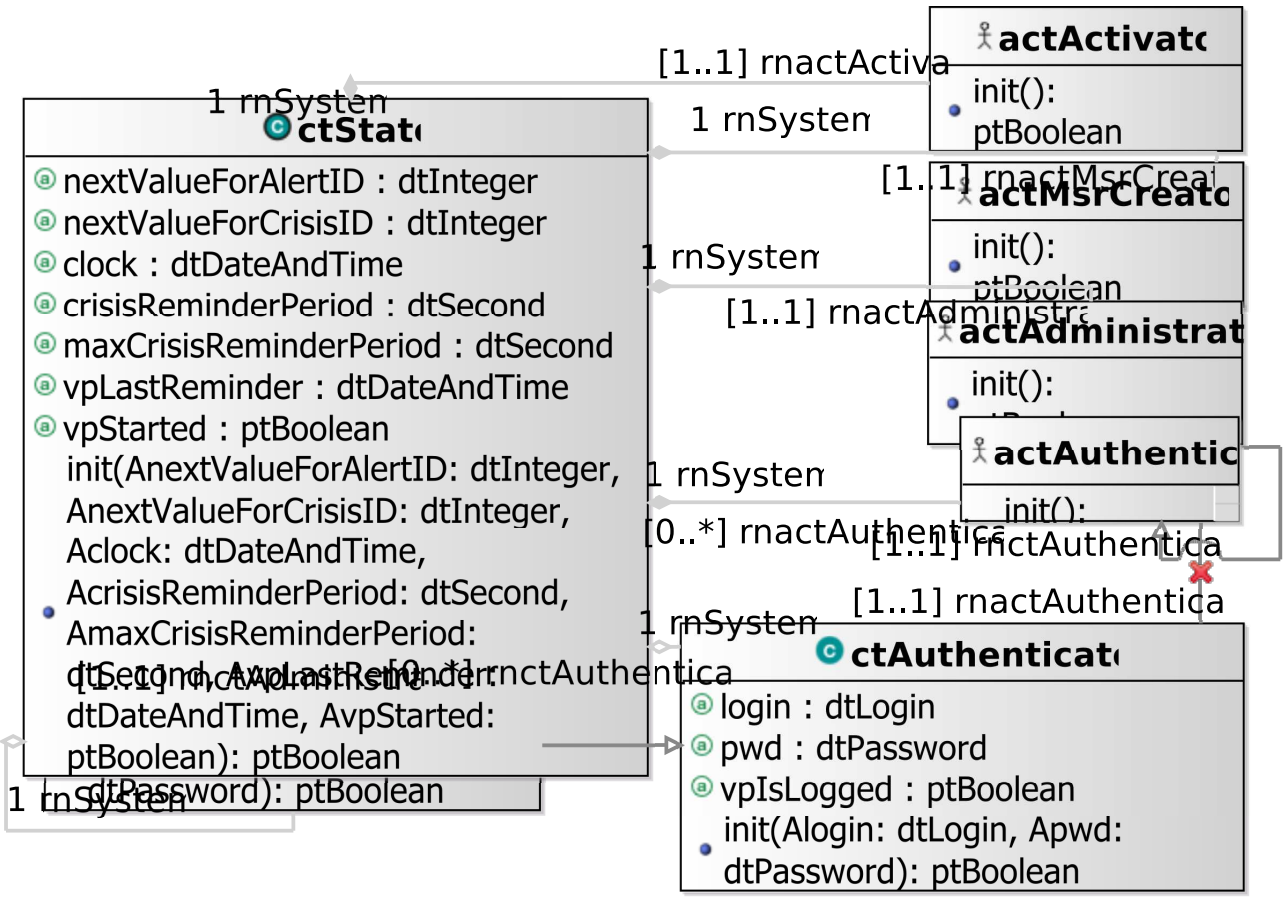

Figure 5.4: oeCreateSystemAndEnvironment operation scope

5.8 Primary Types - Operation Schemes for Class ctAdministrator

5.8.1 Operation Model for init

The `init` operation has the following properties:

OPERATION	
init	
used to initialize the current object as a new instance of the ctAdministrator type.	
Parameters	
1	**Alogin: dtLogin** used to initialize the login field
2	**Apwd: dtPassword** used to initialize the password field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctAdministrator instance having its login and password attributes equal to the one provided as parameters and its vpIsLogged attribute equal to false.

The listing 5.19 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  let Self:ctAdministrator in
6  /* Post F01 */
7  Self.login(Alogin)
8  and Self.pwd = Apwd
9  and Self.vpIsLogged = false
10
11 /* Post F02 */
12 and (Self.oclIsNew and self = Self)
13 )
14 then (result = true)
15 else (result = false)
16 endif}
```

Listing 5.19: Messir (MCL-oriented) specification of the operation *init*.

5.9 Primary Types - Operation Schemes for Class ctAlert

5.9.1 Operation Model for init

The `init` operation has the following properties:

OPERATION
init
used to initialize the current object as a new instance of the ctAlert type.

continues in next page . . .

... Operation table continuation

Parameters	
1	**Aid: dtAlertID** used to initialize the id field
2	**Astatus: etAlertStatus** used to initialize the status field
3	**Alocation: dtGPSLocation** used to initialize the location field
4	**Ainstant: dtDateAndTime** used to initialize the instant field
5	**Acomment: dtComment** used to initialize the comment field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctAlert instance having its attributes equal to the ones provided as parameters.

The listing 5.20 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  let Self:ctAlert in
7  Self.id = Aid
8  and Self.status = Astatus
9  and Self.location = Alocation
10 and Self.instant = Ainstant
11 and Self.comment = Acomment
12 /* Post F02 */
13 and (Self.oclIsNew and self = Self)
14 )
15 then (result = true)
16 else (result = false)
17 endif}
```

Listing 5.20: Messir (MCL-oriented) specification of the operation *init*.

5.9.2 Operation Model for isSentToCoordinator

The `isSentToCoordinator` operation has the following properties:

Operation	
isSentToCoordinator used to provide a given coordinator with current alert information.	
Parameters	
1	**AactCoordinator: actCoordinator** the message destination
Return type	

continues in next page ...

. . . Operation table continuation

ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the message ieSendAnAlert is sent to the input interface of the given coordinator actor with the current alert as parameter value.

The listing 5.21 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  AactCoordinator.rnInterfaceIN.ieSendAnAlert(self)
7  )
8  then (result = true)
9  else (result = false)
10 endif}
```

Listing 5.21: Messir (MCL-oriented) specification of the operation *isSentToCoordinator*.

5.10 Primary Types - Operation Schemes for Class ctAuthenticated

5.10.1 Operation Model for init

The `init` operation has the following properties:

OPERATION	
init	
used to initialize the current object as a new instance of the ctAuthenticated type.	
Parameters	
1	**Alogin: dtLogin** used to initialize the login field
2	**Apwd: dtPassword** used to initialize the password field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctAuthenticated instance having its attributes equal to the ones provided as parameters.

5.11 Primary Types - Operation Schemes for Class ctCoordinator

5.11.1 Operation Model for init

The `init` operation has the following properties:

OPERATION

continues in next page . . .

... Operation table continuation

init	
used to initialize the current object as a new instance of the ctCoordinator type.	
Parameters	
1	**Aid: dtCoordinatorID** used to initialize the id field
2	**Alogin: dtLogin** used to initialize the login field
3	**Apwd: dtPassword** used to initialize the password field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctCoordinator instance having its attributes equal to the ones provided as parameters.

The listing 5.22 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  let Self:ctCoordinator in
7  Self.id = Aid
8  and Self.login = Alogin
9  and Self.pwd = Apwd
10 and Self.vpIsLogged = false
11 /* Post F02 */
12 and (Self.oclIsNew and self = Self)
13 )
14 then (result = true)
15 else (result = false)
16 endif}
```

Listing 5.22: Messir (MCL-oriented) specification of the operation *init*.

5.12 Primary Types - Operation Schemes for Class ctCrisis

5.12.1 Operation Model for init

The `init` operation has the following properties:

OPERATION	
init	
used to initialize the current object as a new instance of the ctCrisis type.	
Parameters	
1	**Aid: dtCrisisID** used to initialize the id field
2	**Atype: etCrisisType** used to initialize the type field

continues in next page . . .

... Operation table continuation

3	**Astatus: etCrisisStatus** used to initialize the status field
4	**Alocation: dtGPSLocation** used to initialize the location field
5	**Ainstant: dtDateAndTime** used to initialize the instant field
6	**Acomment: dtComment** used to initialize the comment field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctCrisis instance having its attributes equal to the ones provided as parameters.

The listing 5.23 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  let Self:ctCrisis in
7  Self.id = Aid
8  and Self.type = Atype
9  and Self.status = Astatus
10 and Self.location = Alocation
11 and Self.instant = Ainstant
12 and Self.comment = Acomment
13 /* Post F02 */
14 and (Self.oclIsNew and self = Self)
15 )
16 then (result = true)
17 else (result = false)
18 endif}
```

Listing 5.23: Messir (MCL-oriented) specification of the operation *init*.

5.12.2 Operation Model for handlingDelayPassed

The `handlingDelayPassed` operation has the following properties:

OPERATION	
handlingDelayPassed used to determine if the crisis stood too longly in a pending status since last reminder.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the crisis is in pending status and if the duration between the current ctState clock information and the last reminder is greater than the crisis reminder period duration.

The listing 5.24 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheSystem:ctState in
4  let CurrentClockSecondsQty:dtInteger in
5  let vpLastReminderSecondsQty:dtInteger in
6  let CrisisReminderPeriod:dtSecond in
7  if
8  ( /* Post F01 */
9   self.rnSystem = TheSystem
10  and self.status = pending
11  and TheSystem.clock.toSecondsQty() = CurrentClockSecondsQty
12  and TheSystem.vpLastReminder.toSecondsQty() = vpLastReminderSecondsQty
13  and TheSystem.crisisReminderPeriod = CrisisReminderPeriod
14  and CurrentClockSecondsQty.sub(vpLastReminderSecondsQty).gt(CrisisReminderPeriod) = true
15 )
16 then (result = true)
17 else (result = false)
18 endif}
```

Listing 5.24: Messir (MCL-oriented) specification of the operation *handlingDelayPassed*.

5.12.3 Operation Model for maxHandlingDelayPassed

The `maxHandlingDelayPassed` operation has the following properties:

OPERATION	
maxHandlingDelayPassed used to determine if the crisis stood too longly in a pending status since its creation.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the crisis is in pending status and if the duration between the current ctState clock information and the crisis instant is greater than the maximum reminder period duration.

The listing 5.25 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheSystem:ctState in
4  let CurrentClockSecondsQty:dtInteger in
5  let CrisisInstantSecondsQty:dtInteger in
6  let MaxCrisisReminderPeriod:dtSecond in
7  if
8  ( /* Post F01 */
9   self.rnSystem = TheSystem
10  and self.status = pending
11  and TheSystem.clock.toSecondsQty() = CurrentClockSecondsQty
12  and Self.instant.toSecondsQty() = CrisisInstantSecondsQty
13  and TheSystem.maxCrisisReminderPeriod = MaxCrisisReminderPeriod
14  and CurrentClockSecondsQty.sub(CrisisInstantSecondsQty)
15                .gt(MaxCrisisReminderPeriod)
16 )
17 then (result = true)
18 else (result = false)
```

```
19 endif}
```

Listing 5.25: Messir (MCL-oriented) specification of the operation *maxHandlingDelayPassed*.

5.12.4 Operation Model for isSentToCoordinator

The `isSentToCoordinator` operation has the following properties:

OPERATION	
isSentToCoordinator	
used to provide a given coordinator with current crisis information.	
Parameters	
1	**AactCoordinator: actCoordinator**
	the message destination actor
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the message ieSendACrisis is sent by the simulator to the input interface of the given coordinator actor with the current crisis as parameter value.

The listing 5.26 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Post Functional:*/
3 postF{if
4 (
5 /* Post F01 */
6  AactCoordinator.rnInterfaceIN.ieSendACrisis(self)
7 )
8 then (result = true)
9 else (result = false)
10 endif}
```

Listing 5.26: Messir (MCL-oriented) specification of the operation *isSentToCoordinator*.

5.12.5 Operation Model for isAllocatedIfPossible

The `isAllocatedIfPossible` operation has the following properties:

OPERATION	
isAllocatedIfPossible	
used to allocate a crisis to a coordinator if any or to alert the administrator of crisis waiting to be handled.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the duration between the crisis creation and the system's clock is greater than the maximum delay defined and

continues in next page . . .

... Operation table continuation

PostF 2	if there exist at least one coordinator then (a) the post state associates to the crisis any of the existing coordinators and (b) the coordinator is informed that he is now the handlers of the crisis whose ID is communicated
PostF 3	else a message is sent to all known administrators to request creation of new coordinators.

The listing 5.27 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if (
4  /* Post F01 */
5  self.maxHandlingDelayPassed()
6  and
7   if (TheSystem.rnactCoordinator->msrIsEmpty = false)
8   then (
9     /* Post F02 */
10    TheSystem.rnactCoordinator->msrAny(true) = TheCoordinatorActor
11    and TheCoordinatorActor.rnctCoordinator = TheCoordinator
12    and self@post.rnHandler = TheCoordinator
13    and self@post.status = handled
14    and self.id.value = TheCrisisIDptString
15    and 'You are now considered as handling the crisis having ID: '
16      .ptStringConcat(TheCrisisIDptString) = TheMessage
17     and TheCoordinatorActor.rnInterfaceIN^ieMessage(TheMessage)
18   )
19   else ( /* Post F03 */
20      TheSystem.rnactAdministrator
21      ->forAll(rnInterfaceIN.ieMessage('Please add new coordinators to handle pending crisis !'))
22   )
23   endif
24   )
25 then (result = true)
26 else (result = false)
27 endif}
```

Listing 5.27: Messir (MCL-oriented) specification of the operation *isAllocatedIfPossible*.

5.13 Primary Types - Operation Schemes for Class ctHuman

5.13.1 Operation Model for init

The `init` operation has the following properties:

OPERATION	
init	
used to initialize the current object as a new instance of the ctHuman type.	
Parameters	
1	**Aid: dtPhoneNumber** used to initialize the id field
2	**Akind: etHumanKind** used to initialize the kind field
Return type	
ptBoolean	

continues in next page ...

... Operation table continuation

Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctHuman instance having its attributes equal to the ones provided as parameters.

The listing 5.28 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  let Self:ctHuman in
7
8  Self.id = Aid
9  and Self.kind = Akind
10
11 /* Post F02 */
12 and (Self.oclIsNew and self = Self)
13 )
14 then (result = true)
15 else (result = false)
16 endif}
```

Listing 5.28: Messir (MCL-oriented) specification of the operation *init*.

5.13.2 Operation Model for isAcknowledged

The `isAcknowledged` operation has the following properties:

OPERATION	
isAcknowledged	
used to specify the property of having sent an alert acknowledge message to the human having declared the alert through its own communication company.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the message ieSmsSend is sent to the related input interface of the related communication company actor with the human phone number and the generic message 'The handling of your alert by our services is in progress !'

5.14 Primary Types - Operation Schemes for Class ctPointOfInterest

5.14.1 Operation Model for init

The `init` operation has the following properties:

OPERATION
init
used to initialize the current object as a new instance of the ctPointOfInterest type.

continues in next page ...

... Operation table continuation

Parameters	
1	**Aid: dtPointOfInterestID** used to initialize the id field
2	**Atype: etCategory** used to initialize the type field
3	**Alocation: dtGPSLocation** used to initialize the location field
4	**Adescription: dtDescription** used to initialize the description field
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctPointOfInterest instance having its attributes equal to the ones provided as parameters.

The listing 5.29 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4     (
5      let Self:ctPointOfInterest in
6      Self.id = Aid
7      and Self.type = Atype
8      and Self.location = Alocation
9      and Self.comment = Acomment
10
11     and (Self.oclIsNew and self = Self)
12    )
13     then (result = true)
14     else (result = false)
15     endif}
```

Listing 5.29: Messir (MCL-oriented) specification of the operation *init*.

5.15 Primary Types - Operation Schemes for Class ctState

5.15.1 Operation Model for init

The `init` operation has the following properties:

OPERATION	
init used to initialize the current object as a new instance of the ctState type.	
Parameters	
1	**AnextValueForAlertID: dtInteger** used to initialize the nextValueForAlertID field
2	**AnextValueForCrisisID: dtInteger** used to initialize the nextValueForCrisisID field

continues in next page ...

... Operation table continuation

3	**Aclock: dtDateAndTime** used to initialize the clock field
4	**AcrisisReminderPeriod: dtSecond** used to initialize the crisisReminderPeriod field
5	**AmaxCrisisReminderPeriod: dtSecond** used to initialize the maxCrisisReminderPeriod field
6	**AvpLastReminder: dtDateAndTime** used to initialize the vpLastReminder field
7	**AvpStarted: ptBoolean** used to initialize the vpStarted field

Return type
ptBoolean

Post-Condition (functional)	
PostF 1	true iff the system poststate includes the current object as a new ctState instance having its attributes equal to the ones provided as parameters.

The listing 5.30 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  /* Post F01 */
6  let Self:ctState in
7
8  Self.nextValueForAlertID = AnextValueForAlertID
9  and Self.nextValueForPointOfInterestID = AnextValueForPointOfInterestID
10 and Self.nextValueForCrisisID = AnextValueForCrisisID
11 and Self.clock = Aclock
12 and Self.crisisReminderPeriod = AcrisisReminderPeriod
13 and Self.maxCrisisReminderPeriod = AmaxCrisisReminderPeriod
14 and Self.vpLastReminder = AvpLastReminder
15 and Self.vpStarted = AvpStarted
16
17 and (Self.oclIsNew and self = Self)
18 )
19 then (result = true)
20 else (result = false)
21 endif}
```

Listing 5.30: Messir (MCL-oriented) specification of the operation *init*.

5.16 Primary Types - Operation Schemes for Datatype dtAlertID

5.16.1 Operation Model for is

The `is` operation has the following properties:

Operation
is used to determine which strings are considered as valid alert identifiers.

continues in next page ...

. . . Operation table continuation

Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	if the length of the value attribute of a dtAlertID is a ptInteger greater than zero and lower or equal to 20 then the operation returns the ptBoolean true, else the ptBoolean false.

The listing 5.31 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4    ( if
5     ( AdtValue.value.length().gt(0)
6      and AdtValue.value.length().leq(20)
7     )
8     then (TheResult = true)
9     else (TheResult = false)
10    endif
11    result = TheResult
12   )}
```

Listing 5.31: Messir (MCL-oriented) specification of the operation *is*.

5.17 Primary Types - Operation Schemes for Datatype dtComment

5.17.1 Operation Model for is

The `is` operation has the following properties:

OPERATION	
is used to determine which strings are considered as valid comments.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the length of the string value is not more than 160 characters.

The listing 5.32 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( MaxLength = 160
6        and AdtValue.value.length().leq(MaxLength)
7       )
8       then (TheResult = true)
9       else (TheResult = false)
10      endif
```

```
11      result = TheResult
12     )}
```

Listing 5.32: Messir (MCL-oriented) specification of the operation *is*.

5.18 Primary Types - Operation Schemes for Datatype dtCoordinatorID

5.18.1 Operation Model for is

The `is` operation has the following properties:

Operation	
is	
used to determine which string are considered as valid alert identifiers.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	if the length of the value attribute of a dtCoordinatorID is a ptInteger greater than zero and lower or equal to 5 than the operation returns the ptBoolean true, else the ptBoolean false.

The listing 5.33 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( AdtValue.value.length().gt(0)
6        and AdtValue.value.length().leq(5)
7       )
8       then (TheResult = true)
9       else (TheResult = false)
10      endif
11      result = TheResult
12     )}
```

Listing 5.33: Messir (MCL-oriented) specification of the operation *is*.

5.19 Primary Types - Operation Schemes for Datatype dtCrisisID

5.19.1 Operation Model for is

The `is` operation has the following properties:

Operation
is
used to determine which strings are considered as valid crisis identifiers.
Return type
ptBoolean

continues in next page . . .

... Operation table continuation

Post-Condition (functional)	
PostF 1	if the length of the value attribute of a dtCrisisID is a ptInteger greater than zero and lower or equal to 10 than the operation returns the ptBoolean true, else the ptBoolean false.

The listing 5.34 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let TheResult: ptBoolean in
    ( if
     ( AdtValue.value.length().gt(0)
      and AdtValue.value.length().leq(10)
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )}
```

Listing 5.34: Messir (MCL-oriented) specification of the operation *is*.

5.20 Primary Types - Operation Schemes for Datatype dtDescription

5.20.1 Operation Model for is

The `is` operation has the following properties:

OPERATION	
is	
used to determine which strings are considered as valid description.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the length of the string value is not more than 280 characters.

The listing 5.35 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let TheResult: ptBoolean in
    ( if
     ( MaxLength = 280
      and AdtValue.value.length().leq(MaxLength)
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
```

```
12      )}
```

Listing 5.35: Messir (MCL-oriented) specification of the operation *is*.

5.21 Primary Types - Operation Schemes for Datatype dtExpPoints

5.21.1 Operation Model for is

The `is` operation has the following properties:

Operation
is
used to determine which integers are considered as valid dtExpPoints
Return type
ptBoolean
Post-Condition (functional)
PostF 1 returns true if the integer has a value between 0 and 120 (inclusive)

The listing 5.36 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4       ( if
5        ( self.value.geq(0)
6         and self.value.leq(+120)
7        )
8        then (TheResult = true)
9        else (TheResult = false)
10       endif
11       result = TheResult
12      )}
```

Listing 5.36: Messir (MCL-oriented) specification of the operation *is*.

5.22 Primary Types - Operation Schemes for Datatype dtGPSLocation

5.22.1 Operation Model for DistanceTo

The `DistanceTo` operation has the following properties:

Operation
DistanceTo
used to determine how close two locations are. In the context of the iCrash system, we compute the distance between two GPS locations using the following Haversine formula. (more details can be found at: http://www.movable-type.co.uk/scripts/latlong.html and http://www.gpsvisualizer.com/calculators#distance)
Return type

continues in next page . . .

... Operation table continuation

ptReal	
Post-Condition (functional)	
PostF 1	The Haversine formula (ACOS(SIN(lat1)*SIN(lat2)+COS(lat1)*COS(lat2)*COS(lon2-lon1))*6371, in which latitudes and longitudes are in radiant applied to the two dtGPS coordinates has to be greater than zero.

The listing 5.37 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let Result :ptReal in
    let EarthRadius : dtReal = 6371000 in
    let Rlat1 : dtReal in let Rlat1a : dtReal in
    let Rlong1 : dtReal in let Rlong1a : dtReal in
    let Deltalat : dtReal in let Deltalong : dtReal in
    let a1: dtReal in let a2:dtReal in
    let a3: dtReal in let a : dtReal in
    let c1: dtReal in let c: dtReal in

    and Rlat1 = self.latitude.toRad()
    and Rlat1a = adtValue.latitude.toRad()
    and Rlong1 = self.longitude.toRad()
    and Rlong1a = adtValue.latitude.toRad()
    and Deltalat = Rlat1a.sub(Rlat1)
    and Deltalong = Rlong1a.sub(Rlong1)
    and a1 = Deltalat.div(2).sin().power(2)
    and a2 = Rlat1.cos().mult(Rlat1a.cos())
    and a3 = Deltalong.div(2).sin().power(2)
    and a4 = a1.add(a2.mult(a3))
    and c1 = a4.sub(1).power(2)
    and c = atan(a4.power(2),c1).mult(2)
    and Result = EarthRadius.mult(c).asptReal()}
```

Listing 5.37: Messir (MCL-oriented) specification of the operation *DistanceTo*.

5.22.2 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which couples are considered as valid dtGPSLocation values.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 true if both latitude and longitude are valid values according to their is operation.

The listing 5.38 provides the Messir (MCL-oriented) specification of the operation.

```

```

```
/* Post Functional:*/
postF{let TheResult: ptBoolean in
    ( if
     ( AdtValue.latitude.is()
      and AdtValue.longitude.is
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )}
```

Listing 5.38: Messir (MCL-oriented) specification of the operation *is*.

5.22.3 Operation Model for isNearTo

The `isNearTo` operation has the following properties:

Operation	
isNearTo	
used to determine if locations are considered enough close to be treated as equivalent in the application domain context. In the context of the iCrash system, we compute the distance between two GPS locations using the following Haversine formula. (more details can be found at: http://www.movable-type.co.uk/scripts/latlong.html and http://www.gpsvisualizer.com/calculators#distance)	
Parameters	
1	**AGPSLocation: dtGPSLocation** the GPS location to be compared to.
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	if the Haversine formula (ACOS(SIN(lat1)*SIN(lat2)+COS(lat1)*COS(lat2)*COS(lon2-lon1))*6371, in which latitudes and longitudes are in radiansapplied to the two dtGPS coordinates is lower to 100 meters) then the predicate is true and false otherwise.

The listing 5.39 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let TheResult: ptBoolean in true
    let EarthRadius: dtReal in
    let MaxDistance: dtReal in
    let ComparedLatitude: dtLatitude in
    let ComparedLongitude: dtLongitude in
    let R1: dtReal in let R1a: dtReal in
    let R2: dtReal in let R2a: dtReal in

    ( if
     ( EarthRadius.value = 6371
      and MaxDistance.value = 100

      and AdtValue.latitude = ComparedLatitude
      and AdtValue.longitude = ComparedLongitude
      and Self.latitude.sin() = R1a
      and AdtValue.latitude.sin().mul(R1a) = R1
      and Self.latitude.cos() = R2a
      and AdtValue.latitude.cos().mul(R2a) = R2
```

```
21
22       and AdtValue.longitude = ComparedLongitude
23       and Self.longitude.sub(ComparedLongitude).cos().mul(R2)
24           .add(R1).acos().mul(EarthRadius).sub(MaxDistance)
25           .value.leq(0)
26      )
27      then (TheResult = true)
28      else (TheResult = false)
29      endif
30      result = TheResult
31     )}
```

Listing 5.39: Messir (MCL-oriented) specification of the operation *isNearTo*.

5.23 Primary Types - Operation Schemes for Datatype dtLatitude

5.23.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which strings are considered as valid dtLatitude.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 is true if the value is a real in the interval [-90.0 , +90.0].

The listing 5.40 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( AdtValue.value.geq(-90.0)
6        and AdtValue.value.leq(+90.0)
7       )
8       then (TheResult = true)
9       else (TheResult = false)
10      endif
11      result = TheResult
12     )}
```

Listing 5.40: Messir (MCL-oriented) specification of the operation *is*.

5.24 Primary Types - Operation Schemes for Datatype dtLogin

5.24.1 Operation Model for is

The `is` operation has the following properties:

OPERATION

continues in next page . . .

. . . Operation table continuation

is
used to determine which strings are considered as valid dtLogin.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 is true of the length of the string value is not more than 20 characters.

The listing 5.41 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      let MaxLength: ptInteger in
5      ( if
6       ( MaxLength = 20
7        and AdtValue.value.length().leq(MaxLength)
8       )
9       then (TheResult = true)
10      else (TheResult = false)
11      endif
12      result = TheResult
13     )}
```

Listing 5.41: Messir (MCL-oriented) specification of the operation *is*.

5.25 Primary Types - Operation Schemes for Datatype dtLongitude

5.25.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which strings are considered as valid dtLongitude.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 is true if the value is a real in the interval [-180.0 , +180.0].

The listing 5.42 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( AdtValue.value.geq(-180.0)
6        and AdtValue.value.leq(+180.0)
7       )
```

```
8       then (TheResult = true)
9       else (TheResult = false)
10      endif
11      result = TheResult
12     )}
```

Listing 5.42: Messir (MCL-oriented) specification of the operation *is*.

5.26 Primary Types - Operation Schemes for Datatype dtPassword

5.26.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which strings are considered as valid dtPassword.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 is true of the length of the string value is at least 6 characters long.

The listing 5.43 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      let MinLength: ptInteger in
5      ( if
6       ( MinLength = 6
7        and AdtValue.value.length().geq(MinLength)
8       )
9       then (TheResult = true)
10      else (TheResult = false)
11      endif
12      result = TheResult
13     )}
```

Listing 5.43: Messir (MCL-oriented) specification of the operation *is*.

5.27 Primary Types - Operation Schemes for Datatype dtPhoneNumber

5.27.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which strings are considered as valid dtPhoneNumber.
Return type

continues in next page . . .

... Operation table continuation

ptBoolean	
Post-Condition (functional)	
PostF 1	is true of the length of the string value is from 4 to 30 characters. No standard is applied !

The listing 5.44 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4    ( if
5     ( AdtValue.value.length().gt(4)
6      and AdtValue.value.length().leq(30)
7     )
8     then (TheResult = true)
9     else (TheResult = false)
10    endif
11    result = TheResult
12   )}
```

Listing 5.44: Messir (MCL-oriented) specification of the operation *is*.

5.28 Primary Types - Operation Schemes for Datatype dtPointOfInterestID

5.28.1 Operation Model for is

The `is` operation has the following properties:

OPERATION	
is	
used to determine which strings are considered as valid alert identifiers.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	if the length of the value attribute of a dtAlertID is a ptInteger greater than zero and lower or equal to 30 then the operation returns the ptBoolean true, else the ptBoolean false.

The listing 5.45 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4       ( if
5       ( AdtValue.value.length().gt(0)
6      and AdtValue.value.length().leq(30)
7       )
8       then (TheResult = true)
9      else (TheResult = false)
10     endif
```

```
11      result = TheResult
12      )}
```

Listing 5.45: Messir (MCL-oriented) specification of the operation *is*.

5.29 Primary Types - Operation Schemes for Enumeration etAlertStatus

5.29.1 Operation Model for is

The `is` operation has the following properties:

Operation
is used to determine which litteral belongs to the enumeration.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 true iff the value is equal to one of the following values: `pending, valid, invalid`

The listing 5.46 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( self = pending
6        or self = valid
7        or self = invalid
8       )
9       then (TheResult = true)
10      else (TheResult = false)
11      endif
12      result = TheResult
13     )}
```

Listing 5.46: Messir (MCL-oriented) specification of the operation *is*.

5.30 Primary Types - Operation Schemes for Enumeration etCategory

5.30.1 Operation Model for is

The `is` operation has the following properties:

Operation
is used to determine which litteral belongs to the enumeration.
Return type

continues in next page . . .

... Operation table continuation

ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the value is equal to one of the following values: `Hospital,Police station,Garage,Parking,Insurance Office,Fire station`

The listing 5.47 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( self = Hospital
6        or self = Police station
7        or self = Garage
8        or self = Parking
9        or self = Insurance Office
10       or self = Fire station
11      )
12      then (TheResult = true)
13      else (TheResult = false)
14      endif
15      result = TheResult
16     )}
```

Listing 5.47: Messir (MCL-oriented) specification of the operation *is*.

5.31 Primary Types - Operation Schemes for Enumeration etCrisisStatus

5.31.1 Operation Model for is

The `is` operation has the following properties:

OPERATION	
is	
used to determine which litteral belongs to the enumeration.	
Return type	
ptBoolean	
Post-Condition (functional)	
PostF 1	true iff the value is equal to one of the following values: `pending, handled, solved, closed`.

The listing 5.48 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
```

```
5       ( self = pending
6        or self = handled
7        or self = solved
8        or self = closed
9       )
10      then (TheResult = true)
11      else (TheResult = false)
12      endif
13      result = TheResult
14     )}
```

Listing 5.48: Messir (MCL-oriented) specification of the operation *is*.

5.32 Primary Types - Operation Schemes for Enumeration etCrisisType

5.32.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is used to determine which litteral belongs to the enumeration.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 true iff the value is equal to one of the following values: `small, medium, huge`

The listing 5.49 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( self = small
6        or self = medium
7        or self = huge
8       )
9       then (TheResult = true)
10      else (TheResult = false)
11      endif
12      result = TheResult
13     )}
```

Listing 5.49: Messir (MCL-oriented) specification of the operation *is*.

5.33 Primary Types - Operation Schemes for Enumeration etExperienceRank

5.33.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which strings are considered as valid etExperienceRank
Return type
ptBoolean
Post-Condition (functional)
PostF 1 returns true if the string is either 'Novice', 'Intermediate' or 'Expert'.

The listing 5.50 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Post Functional:*/
3 postF{let TheResult: ptBoolean in
4     ( if
5      ( self = Novice
6       or self = Intermediate
7       or self = Expert
8      )
9      then (TheResult = true)
10     else (TheResult = false)
11     endif
12     result = TheResult
13    )}
```

Listing 5.50: Messir (MCL-oriented) specification of the operation *is*.

5.34 Primary Types - Operation Schemes for Enumeration etHumanKind

5.34.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is
used to determine which litteral belongs to the enumeration.
Return type
ptBoolean
Post-Condition (functional)
PostF 1 true iff the value is equal to one of the following values: `witness,victim,anonym`

The listing 5.51 provides the Messir (MCL-oriented) specification of the operation.

```
1
2 /* Post Functional:*/
3 postF{let TheResult: ptBoolean in
4     ( if
5      ( self = witness
```

```
6       or self = victim
7       or self = anonymous
8      )
9      then (TheResult = true)
10     else (TheResult = false)
11     endif
12     result = TheResult
13    )}
```

Listing 5.51: Messir (MCL-oriented) specification of the operation *is*.

5.35 Secondary Types - Operation Schemes for Classes

There are no elements in this category in the system analysed.

5.36 Secondary Types - Operation Schemes for Datatype dtSMS

5.36.1 Operation Model for is

The `is` operation has the following properties:

OPERATION
is   used to determine which strings are considered as valid comments
Return type
ptBoolean
Post-Condition (functional)
PostF 1 true iff the length of the string value is not more than 160 characters.

The listing 5.52 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      let MaxLength: ptInteger in
5      ( if
6       ( MaxLength = 160
7        and AdtValue.value.length().leq(MaxLength)
8       )
9       then (TheResult = true)
10      else (TheResult = false)
11      endif
12      result = TheResult
13     )}
```

Listing 5.52: Messir (MCL-oriented) specification of the operation *is*.

5.37 Secondary Types - Operation Schemes for Enumerations

There are no elements in this category in the system analysed.

Chapter 6

Test Model(s)

6.1 Test Model for testcase01

this positive test case intends to verify the correctness of the execution of a simple instance of the `suDeployAndRun` use case.

6.1.1 Test Steps Specification

6.1.1.1 testcase01-ts01oeCreateSystemAndEnvironment-actMsrCreator.outactMsrCreator.oeCreateSy

The `testcase01-ts01oeCreateSystemAndEnvironment-actMsrCreator.outactMsrCreator.oeCreat` has the following properties:

Test Step	
ts01oeCreateSystemAndEnvironment This test step initializes the system state and environment.	
Test Sent Message	
TSM 1	**out:Creator** **sends to system** **actMsrCreator.outactMsrCreator.oeCreateSystemAndEnvironment (AqtyComCompanies)**
Variables	
V 1	**Creator:icrash.environment.actMsrCreator** only actMsrtCreator actors can trigger the system and environment creation and initialization.
Constraints	
C 1	the number of communication company actor instances present in the environment is equal to four to represent all the communication companies available in Luxembourg.
Oracle Constraints	
OC 1	true for testing only the executability (is available and can be triggered) of the operation.

The listing 6.1 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   Creator:actMsrCreator
4   AqtyComCompanies: ptInteger
5  }
6
7  constraints{
8   AqtyComCompanies = 4
9  }
10
11 oracle{
12  constraints{
13   true
14  }
15 }
```

Listing 6.1: Messir (MCL-oriented) specification of the test step *testcase01-ts01oeCreateSystemAndEnvironment*.

6.1.1.2 testcase01-ts02oeSetClock-actActivator.outactActivator.oeSetClock

The `testcase01-ts02oeSetClock-actActivator.outactActivator.oeSetClock` has the following properties:

Test Step	
ts02oeSetClock test the update of the current time.	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSetClock (ACurrentClock)**
Variables	
V 1	**TheActor:actActivator** proactive actor responsible of requesting the update of the system's clock.
Constraints	
C 1 C 2	TheActor is any instance existing in the current environment status. ACurrentClock is a fixed date equal to the 24th November 2017 at 15:20:00 using a 24-hours notation [1].
Oracle Constraints	
OC 1	true for testing only the executability (is available and can be triggered) of the operation.

The listing 6.2 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor:actActivator
4   ACurrentClock:dtDateAndTime
```

[1] for more details see the ISO 8601 Data elements and interchange formats - Information interchange - Representation of dates and times - http://www.iso.org/iso/home/standards/iso8601.htm

```
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactActivator->any2(true)
9   ACurrentClock.date.year.value = 2017
10  ACurrentClock.date.month.value = 11
11  ACurrentClock.date.day.value = 24
12  ACurrentClock.time.hour.value = 15
13  ACurrentClock.time.minute.value = 20
14  ACurrentClock.time.second.value = 00
15 }
16
17 oracle{
18  constraints{
19   true
20  }
21 }
```

Listing 6.2: Messir (MCL-oriented) specification of the test step *testcase01-ts02oeSetClock*.

6.1.1.3 testcase01-ts03oeLogin-actAdministrator.outactAdministrator.oeLogin

The `testcase01-ts03oeLogin-actAdministrator.outactAdministrator.oeLogin` has the following properties:

Test Step	
ts03oeLogin	
test the authentified access of the administrator	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actAdministrator.outactAdministrator.oeLogin (AdtLogin, AdtPassword)**
Variables	
V 1	**TheActor:actAdministrator** an actAdministor actor as subtype of actAuthenticated can send oeLogin messages to the system.
Constraints	
C 1	TheActor is any `actAdministrator` instance existing in the environment. It is thus expected that there exist at least one.
C 2	AdtLogin has its value attribute equal to the primitive string 'icrashadmin' (which is the correct administrator login known by the system after the step one.)
C 3	AdtPassword has its value attribute equal to the primitive string '7WXC1359' (which is the correct administrator password known by the system after the step one.)
Oracle Constraints	
OC 1	the `AMessage` value is expected to be equal to the primitive string 'You are logged ! Welcome ...'
OC 2	TheActor receives from system ieMessage(AMessage)

The listing 6.3 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actAdministrator
4   AdtLogin:dtLogin
5   AdtPassword:dtPassword
6  }
7
8  constraints{
9   TheActor=TheSystem.rnactAdministrator->any2(true)
10  AdtLogin.value.eq('icrashadmin')
11  AdtPassword.value.eq('7WXC1359')
12 }
13
14 oracle{
15  variables{
16   AMessage:ptString
17  }
18  constraints{
19   AMessage = 'You are logged ! Welcome ...'
20   TheActor.inactAdministrator.ieMessage(AMessage)
21  }
22 }
```

Listing 6.3: Messir (MCL-oriented) specification of the test step *testcase01-ts03oeLogin*.

6.1.1.4 testcase01-ts04oeAddCoordinator-actAdministrator.outactAdministrator.oeAddCoordina

The `testcase01-ts04oeAddCoordinator-actAdministrator.outactAdministrator.oeAddCoor` has the following properties:

Test Step	
ts04oeAddCoordinator to test the add of a new coordinator by an administrator.	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actAdministrator.outactAdministrator.oeAddCoordinator (AdtCoordinatorID, AdtLogin, AdtPassword)**
Variables	
V 1	**TheActor:actAdministrator** actAdministor actors as being the only one allowed to add coordinators.
Constraints	
C 1	TheActor is any `actAdministrator` instance existing in the environment. It is expected that there exists at least one which is the same during all the test case.
C 2	AdtCoordinatorID is equal to 1 to set the new coordinator ID
C 3	AdtLogin has its value attribute equal to the primitive string 'steve' which is the ID defined for the new coordinator.
C 4	AdtPassword has its value attribute equal to the primitive string 'pwdMessirExcalibur2017' which is the password to be set for steve.
Oracle Constraints	
OC 1	the administrator should have been acknowledged for the adding of the new coordinator.

The listing 6.4 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
 TheActor : actAdministrator
 AdtCoordinatorID : dtCoordinatorID
 AdtLogin:dtLogin
 AdtPassword:dtPassword
}

constraints{
 TheActor = TheSystem.rnactAdministrator->any2(true)
 AdtCoordinatorID.value.eq('1')
 AdtLogin.value.eq('steve')
 AdtPassword.value.eq('pwdMessirExcalibur2017')
}

oracle{
 constraints{
  TheActor.inactAdministrator.ieCoordinatorAdded()
 }
}
```

Listing 6.4: Messir (MCL-oriented) specification of the test step *testcase01-ts04oeAddCoordinator*.

6.1.1.5 testcase01-ts05oeLogout-actAdministrator.outactAdministrator.oeLogout

The `testcase01-ts05oeLogout-actAdministrator.outactAdministrator.oeLogout` has the following properties:

Test Step	
ts05oeLogout to test the loggout of a connected administrator.	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actAdministrator.outactAdministrator.oeLogout ()**
Variables	
V 1	**TheActor:actAdministrator** an actAdministor actor as subtype of actAuthenticated can send oeLogout messages to the system.
Constraints	
C 1	TheActor is any `actAdministrator` instance existing in the environment. It is expected that there exists at least one which is the same during all the test case.
Oracle Constraints	
OC 1	the `AMessage` value is expected to be equal to the primitive string 'You are logged out ! Good Bye ...'
OC 2	the administrator should have received the messahe AMessage.

The listing 6.5 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
 TheActor : actAdministrator
}

constraints{
 TheActor = TheSystem.rnactAdministrator->any2(true)
}

oracle{
 variables{
  AMessage:ptString
 }
 constraints{
  AMessage = 'You are logged out ! Good Bye ...'
  TheActor.inactAdministrator.ieMessage(AMessage)
 }
}
```

Listing 6.5: Messir (MCL-oriented) specification of the test step *testcase01-ts05oeLogout*.

6.1.1.6 testcase01-ts06oeSetClock02-actActivator.outactActivator.oeSetClock

The `testcase01-ts06oeSetClock02-actActivator.outactActivator.oeSetClock` has the following properties:

Test Step	
ts06oeSetClock02 test the update of the current time.	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSetClock (ACurrentClock)**
Variables	
V 1	**TheActor:icrash.environment.actActivator** proactive actors responsible of requesting the update of the system's clock.
Constraints	
C 1 C 2	TheActor is any instance existing in the current environment status. ACurrentClock is a fixed date equal to the 26th November 2017 at 10:15:00 using a 24-hours notation.
Oracle Constraints	
OC 1	true for testing only the executability (is available and can be triggered) of the operation.

The listing 6.6 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
```

```
3   TheActor:actActivator
4   ACurrentClock:dtDateAndTime
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactActivator->any2(true)
9   ACurrentClock.date.year.value = 2017
10  ACurrentClock.date.month.value = 11
11  ACurrentClock.date.day.value = 26
12  ACurrentClock.time.hour.value = 10
13  ACurrentClock.time.minute.value = 15
14  ACurrentClock.time.second.value = 00
15 }
16
17 oracle{
18  constraints{
19   true
20  }
21 }
```

Listing 6.6: Messir (MCL-oriented) specification of the test step *testcase01-ts06oeSetClock02*.

6.1.1.7 testcase01-ts07oeAlert1-actComCompany.outactComCompany.oeAlert

The `testcase01-ts07oeAlert1-actComCompany.outactComCompany.oeAlert` has the following properties:

Test Step	
ts07oeAlert1 tests the declaration of a new alert functionality.	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actComCompany.outactComCompany.oeAlert (AetHumanKind, AdtDate, AdtTime, AdtPhoneNumber, AdtGPSLocation, AdtComment)**
Variables	
V 1	**TheActor:actComCompany** actComCompany actors transfer alert declaration messages.
Constraints	
C 1	TheActor is any instance existing in the current environment status. It is expected to exist at least one.
C 2	AetHumanKind is equal to witness
C 3	AdtDate is equal to the 26th of November 2017
C 4	AdtTime is equal to 10:10:16 using a 24-hours.
C 5	AdtPhoneNumber is equal to the ptString value '+3524666445252'.
C 6	AdtGPSLocation is equal to (49.627675 , 6.159590).
C 7	AdtComment is equal to '3 cars involved in an accident.'
Oracle Constraints	
OC 1	AdtSMS is equal to the ptString 'Your alert has been registered. We will handle it and keep you informed'.

continues in next page . . .

... Test Step table continuation

OC 2	AdtSMS is sent to the phone number AdtPhoneNumber using the communication company having sent the alert using its ieSmsSend input message.

The listing 6.7 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actComCompany
4   AetHumanKind:etHumanKind
5   AdtDate:dtDate
6   AdtTime:dtTime
7   AdtPhoneNumber:dtPhoneNumber
8   AdtGPSLocation:dtGPSLocation
9   AdtComment:dtComment
10  }
11
12 constraints{
13  TheActor = TheSystem.rnactComCompany->any2(true)
14  AetHumanKind = witness
15  AdtDate.year.value = 2017
16  AdtDate.month.value = 11
17  AdtDate.day.value = 26
18  AdtTime.hour.value = 10
19  AdtTime.minute.value = 10
20  AdtTime.second.value = 16
21  AdtPhoneNumber.value = '+3524666445252'
22  AdtGPSLocation.latitude.value = 49.627675
23  AdtGPSLocation.longitude.value = 6.159590
24  AdtComment.value = '3 cars involved in an accident.'
25 }
26
27 oracle{
28  variables{
29   AdtSMS:dtSMS
30  }
31  constraints{
32   AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
33   TheActor.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
34  }
35 }
```

Listing 6.7: Messir (MCL-oriented) specification of the test step *testcase01-ts07oeAlert1*.

6.1.1.8 testcase01-ts08oeSetClock03-actActivator.outactActivator.oeSetClock

The `testcase01-ts08oeSetClock03-actActivator.outactActivator.oeSetClock` has the following properties:

Test Step
ts08oeSetClock03 test the update of the current time.
Test Sent Message

continues in next page ...

... Test Step table continuation

TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSetClock (ACurrentClock)**
Variables	
V 1	**TheActor:actActivator** proactive actor responsible of requesting the update of the system's clock.
Constraints	
C 1 C 2	TheActor is any instance existing in the current environment status. ACurrentClock is a fixed date equal to the 26th November 2017 at 10:30:00 using a 24-hours notation.
Oracle Constraints	
OC 1	true for testing only the executability (is available and can be triggered) of the operation.

The listing 6.8 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor:actActivator
4   ACurrentClock:dtDateAndTime
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactActivator->any2(true)
9   ACurrentClock.date.year.value = 2017
10  ACurrentClock.date.month.value = 11
11  ACurrentClock.date.day.value = 26
12  ACurrentClock.time.hour.value = 10
13  ACurrentClock.time.minute.value = 30
14  ACurrentClock.time.second.value = 00
15 }
16
17 oracle{
18  constraints{
19   true
20  }
21 }
```

Listing 6.8: Messir (MCL-oriented) specification of the test step *testcase01-ts08oeSetClock03*.

6.1.1.9 testcase01-ts09oeSollicitateCrisisHandling-actActivator.outactActivator.oeSollicitateCrisisHan

The `testcase01-ts09oeSollicitateCrisisHandling-actActivator.outactActivator.oeSollicit` has the following properties:

Test Step
ts09oeSollicitateCrisisHandling test the proactive sollication to handle an alert.
Test Sent Message

continues in next page ...

... Test Step table continuation

TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSollicitateCrisisHandling** ()
Variables	
V 1	**TheActor:icrash.environment.actActivator** proactive actor responsible of triggering sollicitation functionality.
Constraints	
C 1	TheActor is any instance existing in the current environment status. It is expected to exist at least one.
Oracle Variables	
OV 1	**TheAdministrator:actAdministrator** actAdministrator actors can be sollicitated to handle alerts.
OV 2	**TheCoordinator:actCoordinator** actCoordinator actors can be sollicitated to handle alerts.
OV 3	**AMessageForCrisisHandlers:ptString** messages sent to sollicitated actors are of type ptString.
Oracle Constraints	
OC 1	TheAdministrator is any instance existing in the current environment status. It is expected to exist at least one.
OC 2	TheCoordinator is any instance existing in the current environment status. It is expected to exist at least one.
OC 3	AMessageForCrisisHandlers is equal to the ptString 'There are alerts pending since more than the defined delay. Please REACT !'
OC 4	TheCoordinator and TheAdministrator have received the message AMessageForCrisisHandlers.

The listing 6.9 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actActivator
4  }
5
6  constraints{
7   TheActor = TheSystem.rnactActivator->any2(true)
8  }
9
10 oracle{
11  variables{
12   TheAdministrator:actAdministrator
13   TheCoordinator:actCoordinator
14   AMessageForCrisisHandlers:ptString
15  }
16  constraints{
17   TheAdministrator = TheSystem.rnactAdministrator->any2(true)
18   TheCoordinator = TheSystem.rnactCoordinator->any2(true)
19   AMessageForCrisisHandlers = 'There are alerts pending since more than the defined delay. Please
         REACT !'
20   TheAdministrator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
```

```
21    TheCoordinator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
22   }
23  }
```

Listing 6.9: Messir (MCL-oriented) specification of the test step *testcase01-ts09oeSollicitateCrisisHandling*.

6.1.1.10 testcase01-ts10oeLogin02-actAuthenticated.outactAuthenticated.oeLogin

The `testcase01-ts10oeLogin02-actAuthenticated.outactAuthenticated.oeLogin` has the following properties:

Test Step	
ts10oeLogin02 test the authentified access of the coordinator	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actAuthenticated.outactAuthenticated.oeLogin (AdtLogin, AdtPassword)**
Variables	
V 1	**TheActor:actCoordinator** an actCoordinator actor as subtype of actAuthenticated can send oeLogin messages to the system.
Constraints	
C 1	TheActor is any `actAdministrator` instance existing in the environment. It is thus expected that there exist at least one.
C 2	AdtLogin has its value attribute equal to the primitive string 'icrashadmin' (which is the correct administrator login known by the system after the step one.)
C 3	AdtPassword has its value attribute equal to the primitive string '7WXC1359' (which is the correct administrator password known by the system after the step one.)
Oracle Constraints	
OC 1	the `AMessage` value is expected to be equal to the primitive string 'You are logged ! Welcome ...'

The listing 6.10 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
 TheActor : actCoordinator
 AdtLogin:dtLogin
 AdtPassword:dtPassword
}

constraints{
 TheActor = TheSystem.rnactCoordinator->select(a | a.rnctCoordinator.login.value.eq('steve'))->any2
     (true)
 AdtLogin.value.eq('steve')
 AdtPassword.value.eq('pwdMessirExcalibur2017')
```

```
12 }
13
14 oracle{
15  variables{
16   AMessage:ptString
17  }
18  constraints{
19   AMessage = 'You are logged ! Welcome ...'
20   TheActor.inactAuthenticated.ieMessage(AMessage)
21  }
22 }
```

Listing 6.10: Messir (MCL-oriented) specification of the test step *testcase01-ts10oeLogin02*.

6.1.1.11 testcase01-ts11oeGetCrisisSet-actCoordinator.outactCoordinator.oeGetCrisisSet

The `testcase01-ts11oeGetCrisisSet-actCoordinator.outactCoordinator.oeGetCrisisSet` has the following properties:

Test Step	
ts11oeGetCrisisSet cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeGetCrisisSet (AetCrisisStatus)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**AetCrisisStatus:icrash.concepts.primarytypes.datatypes.etCrisisStatus** cf. actor documentation
V 3	**ActCrisis:icrash.concepts.primarytypes.classes.ctCrisis** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AetCrisisStatus value is pending
Oracle Constraints	
OC 1	ActCrisis is any ctCrisis instance that has been sent to TheActor.

The listing 6.11 provides the Messir (MCL-oriented) specification of the test step.

```
1
2 variables{
3  TheActor : actCoordinator
4  AetCrisisStatus : etCrisisStatus
5 }
6
7 constraints{
```

```
 TheActor=TheSystem.rnactCoordinator
       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
       ->any2(true)
 AetCrisisStatus = pending
}

oracle{
 variables{
  ActCrisis:ctCrisis
 }
 constraints{
  TheActor.inactCoordinator.ieSendACrisis(ActCrisis)
 }
}
```

Listing 6.11: Messir (MCL-oriented) specification of the test step *testcase01-ts11oeGetCrisisSet*.

6.1.1.12 testcase01-ts12oeSetCrisisHandler-actCoordinator.outactCoordinator.oeSetCrisisHandler

The `testcase01-ts12oeSetCrisisHandler-actCoordinator.outactCoordinator.oeSetCrisisHand` has the following properties:

Test Step	
ts12oeSetCrisisHandler cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeSetCrisisHandler (AdtCrisisID)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**TheComCompany:icrash.environment.actComCompany** cf. actor documentation
V 3	**TheCoordinator:icrash.environment.actCoordinator** cf. actor documentation
V 4	**AdtCrisisID:icrash.concepts.primarytypes.datatypes.dtCrisisID** cf. actor documentation
V 5	**AMessage:lu.uni.lassy.messir.libraries.primitives.ptString** cf. actor documentation
V 6	**AdtPhoneNumber:icrash.concepts.primarytypes.datatypes.dtPhoneNumber** cf. actor documentation
V 7	**AdtSMS:icrash.concepts.secondarytypes.datatypes.dtSMS** cf. actor documentation
V 8	**ActAlert:icrash.concepts.primarytypes.classes.ctAlert** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AdtCrisisID as a value of 1

continues in next page . . .

... Test Step table continuation

C 3	AMessage is the string 'You are now considered as handling the crisis !'
C 4	AdtPhoneNumber
C 5	AdtSMS has for value the string 'The handling of your alert by our services is in progress !'
Oracle Constraints	
OC 1	there is a communication company actor that received the message ieSmsSend(AdtPhoneNumber,AdtSMS)
OC 2	there is a coordinator actor that received an alert using the message ieSendAnAlert(ActAlert)

The listing 6.12 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actCoordinator
4   AdtCrisisID : dtCrisisID
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactCoordinator
9         ->select(a | a.rnctCoordinator.login.value.eq('steve'))
10        ->any2(true)
11 }
12
13 oracle{
14  variables{
15   AMessage:ptString
16   AdtPhoneNumber:dtPhoneNumber
17   AdtSMS:dtSMS
18   ActAlert:ctAlert
19   TheComCompany: actComCompany
20   TheCoordinator:actCoordinator
21  }
22  constraints{
23   AMessage = 'You are now considered as handling the crisis !'
24   AdtSMS.value = 'The handling of your alert by our services is in progress !'
25   TheComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
26   TheCoordinator.inactCoordinator.ieSendAnAlert(ActAlert)
27   TheActor.inactAuthenticated.ieMessage(AMessage)
28  }
29 }
```

Listing 6.12: Messir (MCL-oriented) specification of the test step *testcase01-ts12oeSetCrisisHandler*.

6.1.1.13 testcase01-ts13oeSetClock04-actActivator.outactActivator.oeSetClock

The `testcase01-ts13oeSetClock04-actActivator.outactActivator.oeSetClock` has the following properties:

TEST STEP
ts13oeSetClock04 cf. actor documentation
Test Sent Message

continues in next page ...

... Test Step table continuation

TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSetClock (ACurrentClock)**
Variables	
V 1	**TheActor:icrash.environment.actActivator** cf. actor documentation
V 2	**ACurrentClock:lu.uni.lassy.messir.libraries.calendar.dtDateAndTime** cf. actor documentation
Constraints	
C 1	TheActor
C 2	ACurrentClock

The listing 6.13 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
 TheActor:actActivator
 ACurrentClock:dtDateAndTime
}

constraints{
 TheActor=TheSystem.rnactActivator->any2(true)
 ACurrentClock.date.year.value = 2017
 ACurrentClock.date.month.value = 11
 ACurrentClock.date.day.value = 26
 ACurrentClock.time.hour.value = 10
 ACurrentClock.time.minute.value = 45
 ACurrentClock.time.second.value = 00
}

oracle{
 constraints{
  true
 }
}
```

Listing 6.13: Messir (MCL-oriented) specification of the test step *testcase01-ts13oeSetClock04*.

6.1.1.14 testcase01-ts14oeValidateAlert-actCoordinator.outactCoordinator.oeValidateAlert

The `testcase01-ts14oeValidateAlert-actCoordinator.outactCoordinator.oeValidateAlert` has the following properties:

Test Step
ts14oeValidateAlert cf. actor documentation
Test Sent Message

continues in next page ...

... Test Step table continuation

TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeValidateAlert (AdtAlertID)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**AdtAlertID:icrash.concepts.primarytypes.datatypes.dtAlertID** cf. actor documentation
V 3	**AMessage:lu.uni.lassy.messir.libraries.primitives.ptString** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AdtAlertID
C 3	AMessage
Oracle Constraints	
OC 1	

The listing 6.14 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actCoordinator
4   AdtAlertID : dtAlertID
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactCoordinator
9         ->select(a | a.rnctCoordinator.login.value.eq('steve'))
10        ->any2(true)
11 }
12
13 oracle{
14  variables{
15   AMessage:ptString
16  }
17  constraints{
18   AMessage = 'The Alert is now declared as valid !'
19   TheActor.actAuthenticated.inactAuthenticated.ieMessage(AMessage)
20  }
21 }
```

Listing 6.14: Messir (MCL-oriented) specification of the test step *testcase01-ts14oeValidateAlert*.

6.1.1.15 testcase01-ts15oeAlert2-actComCompany.outactComCompany.oeAlert

The `testcase01-ts15oeAlert2-actComCompany.outactComCompany.oeAlert` has the following properties:

Test Step	
ts15oeAlert2 cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actComCompany.outactComCompany.oeAlert (AetHumanKind, AdtDate, AdtTime, AdtPhoneNumber, AdtGPSLocation, AdtComment)**
Variables	
V 1	**TheActor:icrash.environment.actComCompany** cf. actor documentation
V 2	**AetHumanKind:icrash.concepts.primarytypes.datatypes.etHumanKind** cf. actor documentation
V 3	**AdtDate:lu.uni.lassy.messir.libraries.calendar.dtDate** cf. actor documentation
V 4	**AdtTime:lu.uni.lassy.messir.libraries.calendar.dtTime** cf. actor documentation
V 5	**AdtPhoneNumber:icrash.concepts.primarytypes.datatypes.dtPhoneNumber** cf. actor documentation
V 6	**AdtGPSLocation:icrash.concepts.primarytypes.datatypes.dtGPSLocation** cf. actor documentation
V 7	**AdtComment:icrash.concepts.primarytypes.datatypes.dtComment** cf. actor documentation
V 8	**AdtSMS:icrash.concepts.secondarytypes.datatypes.dtSMS** cf. actor documentation
Constraints	
C 1	TheActor
C 2	AetHumanKind
C 3	AdtDate
C 4	AdtTime
C 5	AdtPhoneNumber
C 6	AdtGPSLocation
C 7	AdtComment
C 8	AdtSMS
Oracle Constraints	
OC 1	

The listing 6.15 provides the Messir (MCL-oriented) specification of the test step.

```
1
2 variables{
3  TheActor : actComCompany
4  AetHumanKind:etHumanKind
5  AdtDate:dtDate
6  AdtTime:dtTime
```

```
 7   AdtPhoneNumber:dtPhoneNumber
 8   AdtGPSLocation:dtGPSLocation
 9   AdtComment:dtComment
10  }
11
12  constraints{
13   TheActor = TheSystem.rnactComCompany->any2(true)
14   AetHumanKind = witness
15   AdtDate.year.value = 2017
16   AdtDate.month.value = 11
17   AdtDate.day.value = 26
18   AdtTime.hour.value = 10
19   AdtTime.minute.value = 20
20   AdtTime.second.value = 00
21   AdtPhoneNumber.value = '+3524666445000'
22   AdtGPSLocation.latitude.value = 49.627095
23   AdtGPSLocation.longitude.value = 6.160251
24   AdtComment.value = 'A car crash just happened.'
25  }
26
27  oracle{
28   variables{
29    AdtSMS:dtSMS
30   }
31   constraints{
32    AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
33    TheActor.actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
34   }
35  }
```

Listing 6.15: Messir (MCL-oriented) specification of the test step *testcase01-ts15oeAlert2*.

6.1.1.16 testcase01-ts16oeSetClock05-actActivator.outactActivator.oeSetClock

The `testcase01-ts16oeSetClock05-actActivator.outactActivator.oeSetClock` has the following properties:

Test Step	
ts16oeSetClock05 cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actActivator.outactActivator.oeSetClock (ACurrentClock)**
Variables	
V 1 V 2	**TheActor:icrash.environment.actActivator** cf. actor documentation **ACurrentClock:lu.uni.lassy.messir.libraries.calendar.dtDateAndTime** cf. actor documentation
Constraints	
C 1 C 2	TheActor ACurrentClock

The listing 6.16 provides the Messir (MCL-oriented) specification of the test step.

```

variables{
 TheActor:actActivator
 ACurrentClock:dtDateAndTime
}

constraints{
 TheActor=TheSystem.rnactActivator->any2(true)
 ACurrentClock.date.year.value = 2017
 ACurrentClock.date.month.value = 11
 ACurrentClock.date.day.value = 26
 ACurrentClock.time.hour.value = 12
 ACurrentClock.time.minute.value = 45
 ACurrentClock.time.second.value = 00
}

oracle{
 constraints{
  true
 }
}
```

Listing 6.16: Messir (MCL-oriented) specification of the test step *testcase01-ts16oeSetClock05*.

6.1.1.17 testcase01-ts17oeSetCrisisStatus-actCoordinator.outactCoordinator.oeSetCrisisStatus

The `testcase01-ts17oeSetCrisisStatus-actCoordinator.outactCoordinator.oeSetCrisisStatu` has the following properties:

Test Step	
ts17oeSetCrisisStatus cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeSetCrisisStatus (AdtCrisisID, AetCrisisStatus)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**AdtCrisisID:icrash.concepts.primarytypes.datatypes.dtCrisisID** cf. actor documentation
V 3	**AetCrisisStatus:icrash.concepts.primarytypes.datatypes.etCrisisStatus** cf. actor documentation
V 4	**AMessage:lu.uni.lassy.messir.libraries.primitives.ptString** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AdtCrisisID

continues in next page . . .

... Test Step table continuation

C 3	AetCrisisStatus
C 4	AMessage
Oracle Constraints	
OC 1	

The listing 6.17 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actCoordinator
4   AdtCrisisID : dtCrisisID
5   AetCrisisStatus : etCrisisStatus
6  }
7
8  constraints{
9   TheActor=TheSystem.rnactCoordinator
10        ->select(a | a.rnctCoordinator.login.value.eq('steve'))
11        ->any2(true)
12 }
13
14 oracle{
15  variables{
16   AMessage:ptString
17  }
18  constraints{
19   AMessage = 'The crisis status has been updated !'
20   TheActor.inactAuthenticated.ieMessage(AMessage)
21  }
22 }
```

Listing 6.17: Messir (MCL-oriented) specification of the test step *testcase01-ts17oeSetCrisisStatus*.

6.1.1.18 testcase01-ts18oeReportOnCrisis-actCoordinator.outactCoordinator.oeReportOnCrisis

The `testcase01-ts18oeReportOnCrisis-actCoordinator.outactCoordinator.oeReportOnCri` has the following properties:

Test Step	
ts18oeReportOnCrisis cf. actor documentation	
Test Sent Message	
TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeReportOnCrisis (AdtCrisisID, AdtComment)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**AdtCrisisID:icrash.concepts.primarytypes.datatypes.dtCrisisID**

continues in next page ...

... Test Step table continuation

	cf. actor documentation
V 3	**AdtComment:icrash.concepts.primarytypes.datatypes.dtComment** cf. actor documentation
V 4	**AMessage:lu.uni.lassy.messir.libraries.primitives.ptString** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AdtCrisisID
C 3	AdtComment
C 4	AMessage
Oracle Constraints	
OC 1	

The listing 6.18 provides the MESSIR (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actCoordinator
4   AdtCrisisID : dtCrisisID
5   AdtComment : dtComment
6  }
7
8  constraints{
9   TheActor=TheSystem.rnactCoordinator
10        ->select(a | a.rnctCoordinator.login.value.eq('steve'))
11        ->any2(true)
12 }
13
14 oracle{
15  variables{
16   AMessage:ptString
17  }
18  constraints{
19   AMessage = 'The crisis comment has been updated !'
20   TheActor.inactAuthenticated.ieMessage(AMessage)
21  }
22 }
```

Listing 6.18: MESSIR (MCL-oriented) specification of the test step *testcase01-ts18oeReportOnCrisis*.

6.1.1.19 testcase01-ts19oeCloseCrisis-actCoordinator.outactCoordinator.oeCloseCrisis

The `testcase01-ts19oeCloseCrisis-actCoordinator.outactCoordinator.oeCloseCrisis` has the following properties:

TEST STEP
ts19oeCloseCrisis cf. actor documentation
Test Sent Message

continues in next page ...

... Test Step table continuation

TSM 1	**out:TheActor** **sends to system** **actCoordinator.outactCoordinator.oeCloseCrisis (AdtCrisisID)**
Variables	
V 1	**TheActor:icrash.environment.actCoordinator** cf. actor documentation
V 2	**AdtCrisisID:icrash.concepts.primarytypes.datatypes.dtCrisisID** cf. actor documentation
V 3	**AMessage:lu.uni.lassy.messir.libraries.primitives.ptString** cf. actor documentation
Constraints	
C 1	TheActor is the coordinator actor related to a coordinator in the system's state having steve as login value
C 2	AdtCrisisID
C 3	AMessage
Oracle Constraints	
OC 1	

The listing 6.19 provides the Messir (MCL-oriented) specification of the test step.

```
1
2  variables{
3   TheActor : actCoordinator
4   AdtCrisisID : dtCrisisID
5  }
6
7  constraints{
8   TheActor=TheSystem.rnactCoordinator
9         ->select(a | a.rnctCoordinator.login.value.eq('steve'))
10        ->any2(true)
11 }
12
13 oracle{
14  variables{
15   AMessage:ptString
16  }
17  constraints{
18   AMessage = 'The crisis is now closed !'
19   TheActor.inactAuthenticated.ieMessage(AMessage)
20  }
21 }
```

Listing 6.19: Messir (MCL-oriented) specification of the test step *testcase01-ts19oeCloseCrisis*.

6.1.2 Test Case Instance - instance01

6.1.3 Test Case Instance - instance01Part01

Figure 6.1 Sequence diagram representing the first part of a simple and complete testcase instance for *iCrash* .

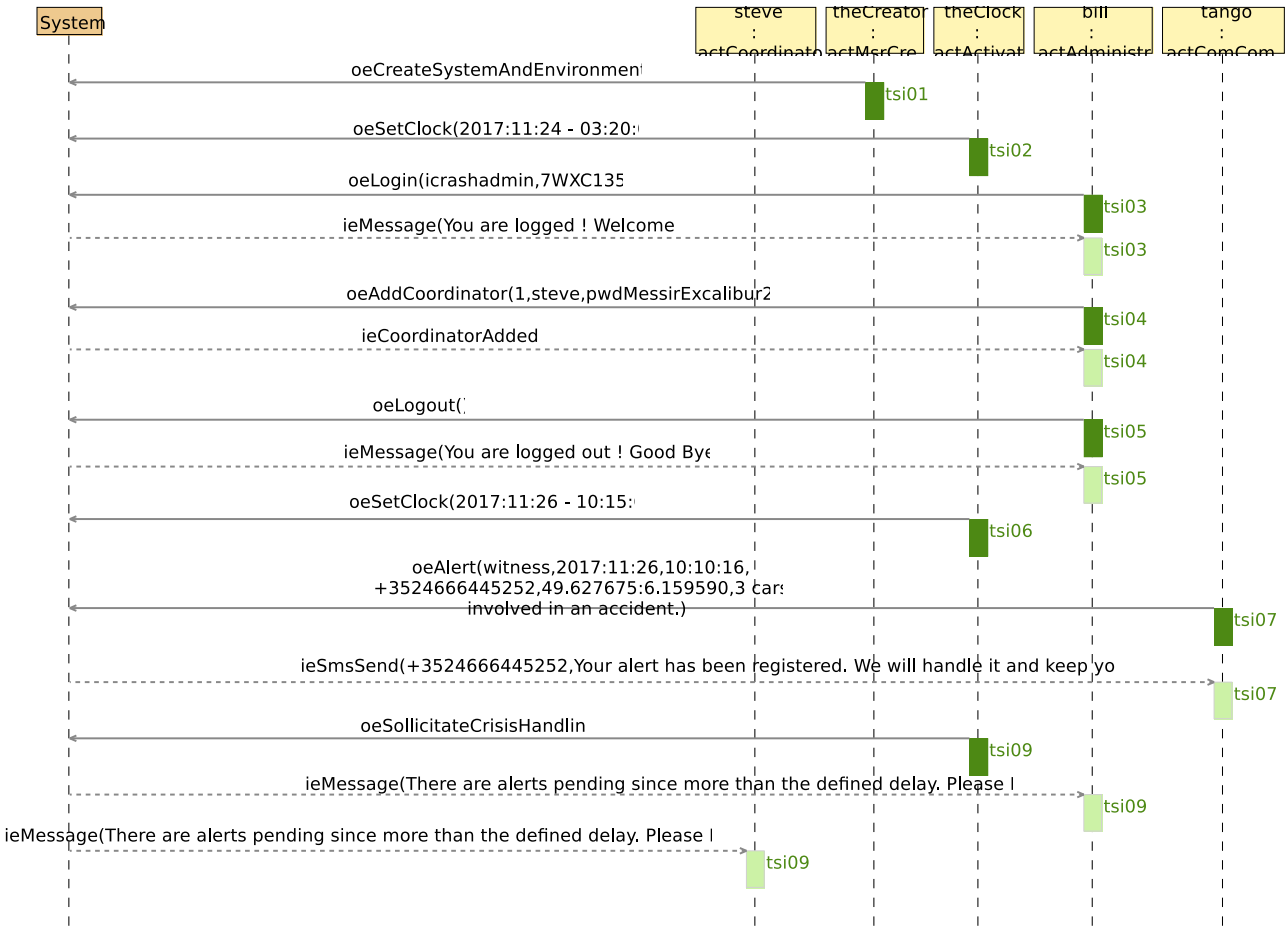

Figure 6.1: tci-testcase01-instance01-Part01 testcase instance sequence diagram

6.1.4 Test Case Instance - instance01Part02

Figure 6.2 Sequence diagram representing the second part of a simple and complete testcase instance for *iCrash* .

Figure 6.2: tci-testcase01-instance01-Part02 testcase instance sequence diagram

Chapter 7

Additional Constraints

7.1 Quality Constraints

Description of all the constraints that concern the required quality criteria according to their ISO definition [3].

7.1.1 Functional suitability

Constraints on the degree to which the product provides functions that meet stated and implied needs when the product is used under specified conditions.

7.1.1.1 Functional completeness

List of requirements on the degree to which the set of functions covers all the specified tasks and user objectives.

1. (to be filled)

7.1.1.2 Functional correctness

List of requirements on the degree to which the set of functions covers all the specified tasks and user objectives.

1. (to be filled)

7.1.1.3 Functional appropriateness

List of requirements on the degree to which the functions facilitate the accomplishment of specified tasks and objectives.

1. (to be filled)

7.1.2 Performance efficiency

Constraints on the performance relative to the amount of resources used under stated conditions

7.1.2.1 Time behaviour

List of requirements on the degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.

1. (to be filled)

7.1.2.2 Resource utilization

List of requirements on the degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.

1. (to be filled)

7.1.2.3 Capacity

List of requirements on the degree to which the maximum limits of a product or system parameter meet requirements.

1. (to be filled)

7.1.3 Compatibility

Constraints on the degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

7.1.3.1 Co-existence

List of requirements on the degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.

1. (to be filled)

7.1.3.2 Interoperability

List of requirements on the degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.

1. (to be filled)

7.1.4 Usability

Constraints on the usability degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

7.1.4.1 Appropriateness recognizability

List of requirements on the degree to which users can recognize whether a product or system is appropriate for their needs.

1. (to be filled)

7.1.4.2 Learnability

List of requirements on the degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use.

1. (to be filled)

7.1.4.3 Operability

List of requirements on the degree to which a product or system has attributes that make it easy to operate and control.

1. (to be filled)

7.1.4.4 User error protection

List of requirements on the degree to which a system protects users against making errors.

1. (to be filled)

7.1.4.5 User interface aesthetics

List of requirements on the degree to which a user interface enables pleasing and satisfying interaction for the user.

1. (to be filled)

7.1.4.6 Accessibility

List of requirements on the degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

1. (to be filled)

7.1.5 Reliability

Contraints on the degree to which a system, product or component performs specified functions under specified conditions for a specified period of time.

7.1.5.1 Maturity

List of requirements on the degree to which a system, product or component meets needs for reliability under normal operation.

1. (to be filled)

7.1.5.2 Availability

List of requirements on the degree to which a system, product or component is operational and accessible when required for use.

1. (to be filled)

7.1.5.3 Fault tolerance

List of requirements on the degree to which a system, product or component operates as intended despite the presence of hardware or software faults.

1. (to be filled)

7.1.5.4 Recoverability

List of requirements on the degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.

1. (to be filled)

7.1.6 Security

Contraints on the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.

7.1.6.1 Confidentiality

List of requirements on the degree to which a product or system ensures that data are accessible only to those authorized to have access.

1. (to be filled)

7.1.6.2 Integrity

List of requirements on the degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.

1. (to be filled)

7.1.6.3 Non-repudiation

List of requirements on the degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.

1. (to be filled)

7.1.6.4 Accountability

List of requirements on the degree to which the actions of an entity can be traced uniquely to the entity.

1. (to be filled)

7.1.6.5 Authenticity

List of requirements on the degree to which the identity of a subject or resource can be proved to be the one claimed.

1. (to be filled)

7.1.7 Maintainability

Contraints on the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers.

7.1.7.1 Modularity

List of requirements on the degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.

1. (to be filled)

7.1.7.2 Reusability

List of requirements on the degree to which an asset can be used in more than one system, or in building other assets.

1. (to be filled)

7.1.7.3 Analysability

List of requirements on the degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.

1. (to be filled)

7.1.7.4 Modifiability

List of requirements on the degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.

1. (to be filled)

7.1.7.5 Testability

List of requirements on the degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.

1. (to be filled)

7.1.8 Portability

Contraints on the degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another.

7.1.8.1 Adaptability

List of requirements on the degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.

1. (to be filled)

7.1.8.2 Installability

List of requirements on the degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.

1. (to be filled)

7.1.8.3 Replaceability

List of requirements on the degree to which a product can replace another specified software product for the same purpose in the same environment.

1. (to be filled)

7.2 Other Constraints

Any other unclassified constraints judged as required for the product under development.

Appendix A

Undocumented Messir Specification Elements

A.1 Undocumented Use Cases

A.1.1 Undocumented Subfunction Level Use Cases

- icrash.usecases.subfunctions.oeFillCaptcha

A.2 Undocumented Use Case Instances

A.2.1 Undocumented Use Case Instance Views

- uci-uciugSecurelyUseSystemFailLoginResetPassword
- uci-uciugSecurelyUseSystemWithDirectResetPassword

A.3 Undocumented Operation Specifications

- icrash.concepts.primarytypes.datatypes.dtCaptcha.is
- icrash.concepts.primarytypes.datatypes.dtMail.is

A.4 Undocumented Test-Case Instance Specifications

- lu.uni.lassy.excalibur.examples.icrash.tests.testcase01.instance01.instance01
- lu.uni.lassy.excalibur.examples.icrash.tests.testcase01.instance01.instance01Part01
- lu.uni.lassy.excalibur.examples.icrash.tests.testcase01.instance01.instance01Part02

Appendix B

Specification project lu.uni.lassy.excalibur.examples.icrash

B.1 Use Cases Model

This section contains the use cases elicited during the requirements elicitation phase. The use cases are textually described as suggested by the Messir method and inspired by the standard Cokburn template [2].

B.1.1 Use Cases

B.1.1.1 subfunction-oeCloseCrisis

the `actCoordinator`'s goal is to declare a crisis as closed.

Use-Case Description	
Name	oeCloseCrisis
Scope	system
Level	subfunction
Primary actor(s)	
1	`actCoordinator[active]`
Goal(s) description	
the `actCoordinator`'s goal is to declare a crisis as closed.	
Protocol condition(s)	
1	the iCrash system has been deployed.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the crisis is know by the system to be closed.
2	a message `ieMessage(AMessage)` is sent to the `actCoordinator` to inform him that his crisis is now considered as closed.

Figure B.1 shows the use case diagram for the oeCloseCrisis subfunction use case

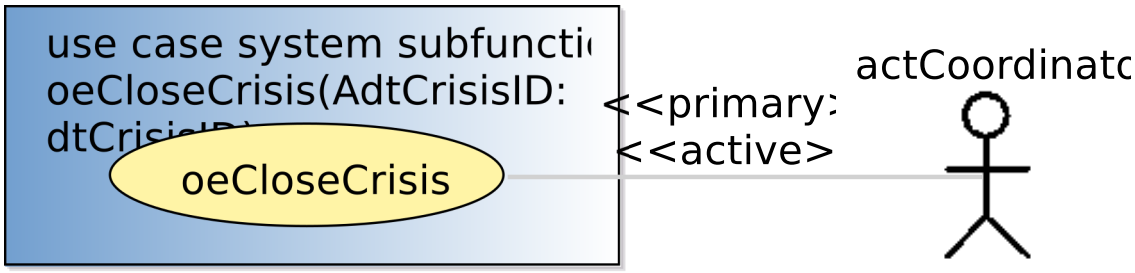

Figure B.1: oeCloseCrisis subfunction use case

Appendix C

Messir Specification Files Listing

C.1 File ./src-gen/messir-spec/.views.msr

```
//
//DON'T TOUCH THIS FILE !!!
//
package uuid7e0d382938204f3c9036c123484468fb {
 Concept Model {}
}
```

Listing C.1: Messir Spec. file .views.msr.

C.2 File ./src-gen/messir-spec/operations/concepts/secondarytypes-datatypes/dtSMS.msr

```
package icrash.operations.concepts.secondarytypes.datatypes.dtSMS{

import lu.uni.lassy.messir.libraries.primitives
import lu.uni.lassy.messir.libraries.calendar
import lu.uni.lassy.messir.libraries.math

import icrash.concepts.primarytypes.datatypes
import icrash.concepts.primarytypes.classes
import icrash.concepts.secondarytypes.datatypes
import icrash.concepts.secondarytypes.classes

Operation Model {
operation: icrash.concepts.secondarytypes.datatypes.dtSMS.is():ptBoolean{
  postF{
    let TheResult: ptBoolean in
    let MaxLength: ptInteger in
    ( if
     ( MaxLength = 160
      and AdtValue.value.length().leq(MaxLength)
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )}
prolog{ "src/Operations/Concepts/SecondaryTypesDatatypes/SecondaryTypesDatatypes-dtSMS-is.pl"}
}
}
}
```

Listing C.2: Messir Spec. file dtSMS.msr.

C.3 File ./src-gen/messir-spec/operations/environment/environment-actActivator-oeSetClock.msr

```
package icrash.operations.environment.actActivator.oeSetClock {

import icrash.environment

import lu.uni.lassy.messir.libraries.primitives
import lu.uni.lassy.messir.libraries.calendar
import lu.uni.lassy.messir.libraries.math

import icrash.concepts.primarytypes.datatypes
import icrash.concepts.primarytypes.classes

 Operation Model {

operation: actActivator.outactActivator.oeSetClock(AcurrentClock:dtDateAndTime):ptBoolean
{
preP{
 let TheSystem: ctState in
 let AvpStarted: ptBoolean in

/* PreP01 */
 self.rnActor.rnSystem = TheSystem
 and self.rnActor.rnSystem.vpStarted = AvpStarted
 and AvpStarted = true
 and TheSystem.clock.lt(AcurrentClock)
}
preF{true}

postF{
 let TheSystem: ctState in
 self.rnActor.rnSystem = TheSystem

/* PostF01 */
 and TheSystem@post.clock = AcurrentClock
}
postP{true}

prolog{"src/Operations/Environment/OUT/outactActivator-oeSetClock.pl"}

}
}
}
```

Listing C.3: Messir Spec. file environment-actActivator-oeSetClock.msr.

C.4 File ./src-gen/messir-spec/operations/environment/environment-actActivator-oeSollicitateCrisisHandling.msr

```
package icrash.operations.environment.actActivator.oeSollicitateCrisisHandling {

import lu.uni.lassy.messir.libraries.primitives
import lu.uni.lassy.messir.libraries.math
import lu.uni.lassy.messir.libraries.string
import lu.uni.lassy.messir.libraries.calendar

import icrash.concepts.primarytypes.datatypes
import icrash.concepts.primarytypes.classes
import icrash.environment

Operation Model {

operation: actActivator.outactActivator.oeSollicitateCrisisHandling():ptBoolean
{
preP{
 let TheSystem: ctState in
```

```
18  let AvpStarted: ptBoolean in
19  let ColctCrisisToHandle:
20     Bag(ctCrisis) in
21
22  self.rnActor.rnSystem = TheSystem
23
24 /* PreP01 */
25  and TheSystem.vpStarted
26
27 /* PreP02 */
28  and TheSystem.rnctCrisis->select(handlingDelayPassed())
29    = ColctCrisisToHandle
30  and ColctCrisisToHandle->size().geq(1)
31 }
32 preF{true}
33
34 postF{
35  let TheSystem: ctState in
36  let AMessageForCrisisHandlers: dtComment in
37  let ColctCrisisToAllocateIfPossible:Bag(ctCrisis) in
38
39  self.rnActor.rnSystem = TheSystem
40 /* PostF01 */
41  and TheSystem.rnctCrisis->select(maxHandlingDelayPassed())
42    = ColctCrisisToAllocateIfPossible
43  and ColctCrisisToAllocateIfPossible->forAll(isAllocatedIfPossible())
44
45  /* PostF02 */
46  and TheSystem.rnctCrisis->select(handlingDelayPassed())
47  = ColctCrisisToHandle
48
49  and ColctCrisisToHandle->msrColSubtract(ColctCrisisToAllocateIfPossible)
50    = ColctCrisisToRemind
51
52  and if (ColctCrisisToRemind->size().geq(1))
53    then (AMessageForCrisisHandlers.value
54       ='There are alerts pending since more than the defined delay. Please REACT !'
55       and TheSystem.rnactAdministrator.
56         rnInterfaceIN^ieMessage(AMessageForCrisisHandlers)
57         and TheSystem.rnactCoordinator
58           ->forAll(rnInterfaceIN^ieMessage(AMessageForCrisisHandlers))
59       )
60    else true
61    endif
62 }
63 postP{
64  let TheSystem: ctState in
65  let TheClock: dtDateAndTime in
66
67  self.rnActor.rnSystem = TheSystem
68  and TheSystem.clock = TheClock
69  and TheSystem@post.vpLastReminder = TheClock
70 }
71
72 prolog{"src/Operations/Environment/OUT/outactActivator-oeSollicitateCrisisHandling.pl"}
73 }
74 }
75 }
```

Listing C.4: Messir Spec. file environment-actActivator-oeSollicitateCrisisHandling.msr.

C.5 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeAddCoordinator.msr

```
1 package icrash.operations.environment.actAdministrator.oeAddCoordinator {
2
3 import lu.uni.lassy.messir.libraries.primitives
4
```

```
5 import icrash.concepts.primarytypes.datatypes
6 import icrash.concepts.primarytypes.classes
7 import icrash.environment
8
9 Operation Model {
10
11 operation: actAdministrator.outactAdministrator.oeAddCoordinator(AdtCoordinatorID:dtCoordinatorID,
      AdtLogin:dtLogin, AdtPassword:dtPassword):ptBoolean
12 {
13 preP{
14  let TheSystem: ctState in
15  let TheActor:actAdministrator in
16
17  self.rnActor.rnSystem = TheSystem
18  and self.rnActor = TheActor
19
20 /* PreP01 */
21  and TheSystem.vpStarted = true
22 /* PreP02 */
23  and TheActor.rnctAuthenticated.vpIsLogged = true
24  }
25 preF{
26  let TheSystem: ctState in
27  let TheActor:actAdministrator in
28  let ColctCoordinators:Bag(ctCoordinator) in
29
30  self.rnActor.rnSystem = TheSystem
31  and self.rnActor = TheActor
32 /* PreF01 */
33  and TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
34    = ColctCoordinators
35  and ColctCoordinators->isEmpty() = true
36   }
37 postF{
38  let TheSystem: ctState in
39  let TheactCoordinator:actCoordinator in
40  let ThectCoordinator:ctCoordinator in
41  self.rnActor.rnSystem = TheSystem
42  and self.rnActor = TheActor
43 /* PostF01 */
44  TheactCoordinator.init()
45 /* PostF02 */
46  and ThectCoordinator.init(AdtCoordinatorID,AdtLogin,AdtPassword)
47
48 /* PostF03 */
49  and TheactCoordinator@post.rnctCoordinator = ThectCoordinator
50
51 /* PostF04 */
52  and ThectCoordinator@post.rnactAuthenticated = TheactCoordinator
53
54 /* PostF05 */
55  and TheActor.rnInterfaceIN^ieCoordinatorAdded()
56 }
57 postP{true}
58
59 prolog{"src/Operations/Environment/OUT/outactAdministrator-oeAddCoordinator.pl"}
60 }
61 }
62 }
```

Listing C.5: Messir Spec. file environment-actAdministrator-oeAddCoordinator.msr.

C.6 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeAddPointOfInterest.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeAddPointOfInterest {
2
3 import lu.uni.lassy.messir.libraries.primitives
```

```
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
11 Operation Model {
12
13  operation: icrash.environment.actAdministrator.outactAdministrator.oeAddPointOfInterest(
        AetCategory:etCategory, AdtGPSLocation:dtGPSLocation, AdtDescription:dtDescription):ptBoolean{
14  // include below the specification information (pre,post or ocl or prolog)
15  preP{
16   let TheSystem: ctState in
17   let TheActor:actAdministrator in
18
19   self.rnActor.rnSystem = TheSystem
20   and self.rnActor = TheActor
21
22  /* PreP01 */
23   and TheSystem.vpStarted = true
24  /* PreP02 */
25   and TheActor.rnctAuthenticated.vpIsLogged = true
26   }
27
28  postF{
29   let TheSystem: ctState in
30  let TheactAdminitrator:actAdministrator in
31   let ActPointOfInterest:ctPointOfInterest in
32   let AetPointOfInterestCategory:etCategory in
33   let APointOfInterestDescription:dtDescription in
34   let AdtPointOfInterestID:dtPointOfInterestID in
35   let ColctPointOfInterest:Bag(ctPointOfInterest) in
36
37   self.rnActor.rnSystem = TheSystem
38   and self.rnActor = TheactComCompany
39
40   TheSystem.nextValueForPointOfInterestID=PrenextValueForPointOfInterestID
41   and PrenextValueForPointOfInterestID.add(1) = PostnextValueForPointOfInterestID
42
43  and TheSystem.rnctPointOfInterest->select(id.eq(PostnextValueForPointOfInterestID))
44     = ColctPointOfInterest
45   and ColctPointOfInterest->isEmpty() = true
46
47   and TheSystem@post.nextValueForPointOfInterestID = PostnextValueForPointOfInterestID
48  and TheSystem.nextValueForPointOfInterestID.todtString().eq(AdtPointOfInterestID)
49
50    /* PostF03 */
51
52  and ActPointOfInterest.init(AdtPointOfInterestID,
53         AetCategory,
54         AdtGPSLocation,
55         AdtDescription)
56    and TheactAdministrator.rnInterfaceIN^iePointOfInterestAdded()
57   }
58    postP{true}
59
60 }
61 }
62 }
```

Listing C.6: Messir Spec. file environment-actAdministrator-oeAddPointOfInterest.msr.

C.7 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeClosestToALocation.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeClosestToALocation {
2
```

```
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9
10 import icrash.environment
11
12  Operation Model {
13
14   operation: icrash.environment.actAdministrator.outactAdministrator.oeClosestToALocation(
        AdtGPSLocation:dtGPSLocation):ptBoolean /* List*/{
15   // include below the specification information (pre,post or ocl or prolog)
16
17   preP{
18    let TheSystem: ctState in
19    let TheActor:actAdministrator in
20
21     self.rnActor.rnSystem = TheSystem
22     and self.rnActor = TheActor
23
24   /* PreP01 */
25    and TheSystem.vpStarted = true
26   /* PreP02 */
27    and TheActor.rnctAuthenticated.vpIsLogged = true
28   }
29   postF{
30    let TheSystem: ctState in
31    let TheActor:actAdministrator in
32    let CollctPointOfInterest:ctPointOfInterest in
33    let temp:ctPointOfInterest in
34    let j:dtInteger in
35    let flag:ptBoolean in
36    self.rnActor.rnSystem = TheSystem
37    and self.rnActor = TheActor
38   /* PostF01 */
39   and j = 0
40    TheSystem.rnctPointOfInterest->select(AllctPointOfInterest)
41    = CollctPointOfInterest
42    SortCtPointOfInterest = CollctPointOfInterest.sort()
43    and while (j.lt(SortctPointOfInterest.length-1)
44    if (SortCtPointOfInterset.get(j).location.distanceTo(AdtGPSLocation)< SortCtPointOfInterest.get(j
          +1).location.distanceTO(AdtGPSLocation) )
45     then (true and j+1)
46     else (false and break)
47     endif
48     )
49
50    }
51    postP{true}
52   }
53  }
54 }
```

Listing C.7: Messir Spec. file environment-actAdministrator-oeClosestToALocation.msr.

C.8 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeDeleteCoordinator.msr

```
1 package icrash.operations.environment.actAdministrator.oeDeleteCoordinator {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.calendar
6
7 import icrash.environment
8
```

```
9 import icrash.concepts.primarytypes.datatypes
10 import icrash.concepts.primarytypes.classes
11
12 Operation Model {
13
14 operation: actAdministrator.outactAdministrator.oeDeleteCoordinator(AdtCoordinatorID:dtCoordinatorID
       ):ptBoolean
15 {
16 preP{
17  let TheSystem: ctState in
18  let TheActor:actAdministrator in
19
20  self.rnActor.rnSystem = TheSystem
21  and self.rnActor = TheActor
22
23 /* PreP01 */
24  and TheSystem.vpStarted = true
25 /* PreP02 */
26  and TheActor.rnctAuthenticated.vpIsLogged = true
27 }
28 preF{
29  let TheSystem: ctState in
30  let TheActor:actAdministrator in
31
32  self.rnActor.rnSystem = TheSystem
33  and self.rnActor = TheActor
34 /* PreF01 */
35  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
36  = ColctCoordinators
37  and ColctCoordinators->size().eq(1)
38 }
39 postF{
40  let TheSystem: ctState in
41  let TheActor:actAdministrator in
42  let ThectCoordinator:ctCoordinator in
43  self.rnActor.rnSystem = TheSystem
44  and self.rnActor = TheActor
45 /* PostF01 */
46  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
47  = ThectCoordinator
48  and ThectCoordinator.rnactCoordinator->forAll(msrIsKilled)
49  and ThectCoordinator.msrIsKilled
50
51  /* PostF02 */
52  and TheActor.rnInterfaceIN^ieCoordinatorDeleted()
53
54  /* Post Protocol:*/
55 /* PostP01 */
56  and true
57  }
58  postP{true}
59
60  prolog{"src/Operations/Environment/OUT/outactAdministrator-oeDeleteCoordinator.pl"}
61  }
62      }
63     }
```

Listing C.8: Messir Spec. file environment-actAdministrator-oeDeleteCoordinator.msr.

C.9 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeDeletePointOfInterest.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeDeletePointOfInterest {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
```

```
7 import icrash.environment
8
9 import icrash.concepts.primarytypes.datatypes
10 import icrash.concepts.primarytypes.classes
11
12  Operation Model {
13
14   operation: icrash.environment.actAdministrator.outactAdministrator.oeDeletePointOfInterest(
         AdtPointOfInterestID:dtPointOfInterestID):ptBoolean{
15   // include below the specification information (pre,post or ocl or prolog)
16   preP{
17    let TheSystem: ctState in
18    let TheActor:actAdministrator in
19
20     self.rnActor.rnSystem = TheSystem
21     and self.rnActor = TheActor
22
23   /* PreP01 */
24    and TheSystem.vpStarted = true
25   /* PreP02 */
26    and TheActor.rnctAuthenticated.vpIsLogged = true
27   }
28   preF{
29    let TheSystem: ctState in
30    let TheActor:actAdministrator in
31
32    self.rnActor.rnSystem = TheSystem
33    and self.rnActor = TheActor
34   /* PreF01 */
35    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
36    = ColctPointOfInterest
37    and ColctPointOfInterest->size().eq(1)
38   }
39   postF{
40    let TheSystem: ctState in
41    let TheActor:actAdministrator in
42    let ThectPointOfInterest:ctPointOfInterest in
43    self.rnActor.rnSystem = TheSystem
44    and self.rnActor = TheActor
45   /* PostF01 */
46    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
47    = ThectPointOfInterest
48    and ThectPointOfInterest.msrIsKilled
49
50    /* PostF02 */
51    and TheActor.rnInterfaceIN^iePointOfInterestDeleted()
52    and true
53  }
54  postP{true}
55   }
56  }
57 }
```

Listing C.9: Messir Spec. file environment-actAdministrator-oeDeletePointOfInterest.msr.

C.10 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeEditPointOfInterest.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeEditPointOfInterest {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
```

```
11 Operation Model {
12
13  operation: icrash.environment.actAdministrator.outactAdministrator.oeEditPointOfInterest(
        AdtPointsOfInterestID:dtPointOfInterestID, AetCategory:etCategory, AdtGPSLocation:
        dtGPSLocation, AdtDescription:dtDescription):ptBoolean{
14
15   preP{
16    let TheSystem: ctState in
17    let TheActor:actAdministrator in
18
19    self.rnActor.rnSystem = TheSystem
20    and self.rnActor = TheActor
21
22   /* PreP01 */
23    and TheSystem.vpStarted = true
24   /* PreP02 */
25    and TheActor.rnctAuthenticated.vpIsLogged = true
26    }
27
28    preF{
29    let TheSystem: ctState in
30    let TheActor:actAdministrator in
31
32    self.rnActor.rnSystem = TheSystem
33    and self.rnActor = TheActor
34   /* PreF01 */
35    TheSystem.rnctPointOfInterest->select(id.eq(AdtPointOfInterestID))
36    = PointOfInterest
37    and PointOfInterest->size().eq(1)
38    /*PostF01*/
39    and PointOfInterest@post.Category = AetCategory
40    and PointOfInterest@post.Location = AdtGPSLocation
41    and PointOfInterest@post.Description = AdtDescription
42
43    /*PostF02*/
44    and TheActor.rnInterfaceIN^iePointOfInterestEdited()
45    and true
46   }
47
48  }
49 }
50 }
```

Listing C.10: Messir Spec. file environment-actAdministrator-oeEditPointOfInterest.msr.

C.11 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeRankDownCoordinator.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeRankDownCoordinator {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
11 Operation Model {
12
13  operation: icrash.environment.actAdministrator.outactAdministrator.oeRankDownCoordinator(
        AdtCoordinatorID:dtCoordinatorID):ptBoolean{
14  // include below the specification information (pre,post or ocl or prolog)
15
16  preP{
17
18  let TheSystem:ctState in
19  let Jack:actAdministrator in
```

```
20  self.rnActor.rnSystem = TheSystem
21  and self.rnActor = Jack
22
23  /* PreP01 */
24   and TheSystem.vpStarted = true
25  /* PreP02 */
26    and Jack.rnctAuthenticated.vpIsLogged = true
27
28  }
29
30  preF{
31
32  let TheSystem:ctState in
33  let Jack:actAdministrator in
34  self.rnActor.rnSystem = TheSystem
35  and self.rnActor = Jack
36
37  /* PreF01*/
38  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
39   = theCoordinator
40   and theCoordinator->size().eq(1)
41
42   /*PreF02*/
43   and (theCoordinator.expRank.eq("Intermediate") or theCoordinator.expRank.eq("Expert"))
44
45  }
46
47  postF{
48
49  let TheSystem:ctState in
50  let Jack:actAdministrator in
51  self.rnActor.rnSystem = TheSystem
52  and self.rnActor = Jack
53
54  TheSystem.rnctCoordinator->select(id.eq(AdtCoordinatorID))
55   = theCoordinator
56
57  /*PostF01*/
58  and Jack.rnInterfaceIN^ieRankEdited()
59
60  /*PostF02*/
61
62  if(theCoordinator.expRank.eq("Expert") and theCoordinator.expPoints >= 60)then(theCoordinator.
        @post.expRank.eq("Intermediate") and theCoordinator.@post.expPoints = 20)
63  else(theCoordinator.@post.expRank.eq("Novice") and theCoordinator.@post.expPoints = 0)endif
64  }
65
66  postP{
67   true
68  }
69
70  }
71 }
72 }
```

Listing C.11: Messir Spec. file environment-actAdministrator-oeRankDownCoordinator.msr.

C.12 File ./src-gen/messir-spec/operations/environment/environment-actAdministrator-oeSelectCategories.msr

```
1 package icrash.environment.operations.actAdministrator.outactAdministrator.oeSelectCategories {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8 import icrash.environment
```

```
 9
10 import icrash.concepts.primarytypes.datatypes
11 import icrash.concepts.primarytypes.classes
12  Operation Model {
13
14   operation: icrash.environment.actAdministrator.outactAdministrator.oeSelectCategories(AetCategory:
         etCategory):ptBoolean{
15   // include below the specification information (pre,post or ocl or prolog)
16   preP{
17    let TheSystem: ctState in
18    let TheActor:actAdministrator in
19
20     self.rnActor.rnSystem = TheSystem
21     and self.rnActor = TheActor
22
23   /* PreP01 */
24    and TheSystem.vpStarted = true
25   /* PreP02 */
26    and TheActor.rnctAuthenticated.vpIsLogged = true
27   }
28   postF{
29    let TheSystem: ctState in
30    let TheActor:actAdministrator in
31    let CollctPointOfInterest : ctPointOfInterest in
32    let SortedCollctPointOfInterest:ctPointOfInterest in
33
34    self.rnActor.rnSystem = TheSystem
35    and self.rnActor = TheActor
36   /* PostF01 */
37   // need iterator
38   and SortedCollctPointOfInterest = CollctPointOfInterest.sort(AetCategory)
39    and (while i.lt(SortCollctPointOfInterest)
40    if (CollctPointofInterest.get(i).category.equals(AetCatgory))
41     then (result =true and i++)
42     else (result = false and break)
43    endif
44    )
45
46   return result
47
48    }
49    postP{true}
50   }
51  }
52 }
```

Listing C.12: Messir Spec. file environment-actAdministrator-oeSelectCategories.msr.

C.13 File ./src-gen/messir-spec/operations/environment/environment-actAuthenticated-oeFillCaptcha.msr

```
1 package icrash.environment.operations.actAuthenticated.outactAuthenticated.oeFillCaptcha {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
11  Operation Model {
12
13   operation: icrash.environment.actAuthenticated.outactAuthenticated.oeFillCaptcha(AdtCaptcha:
         dtCaptcha):ptBoolean{
14 preP{
15
16  let TheSystem:ctState in
```

```
17 let Bob:actAuthenticated in
18 self.rnActor.rnSystem = TheSystem
19 and self.rnActor = Bob
20
21 /* PreP01 */
22 and TheSystem.vpStarted = true
23 /* PreP02 */
24 and TheActor.rnctAuthenticated.vpIsLogged = false
25 /* PreP03 */
26 and TheActor.rnctAuthenticated.nbrfails >= 3
27 }
28
29 preF{
30
31 true
32
33 }
34
35 postF{
36
37 let TheSystem: ctState in
38 let Bob:actAuthenticated in
39
40 let AptStringMessageForTheactAuthenticated:ptString in
41 let AdtNewCaptcha:dtCaptcha in
42
43 self.rnActor.rnSystem = TheSystem
44 and self.rnActor = TheactAuthenticated
45
46 and /* PostF01 */
47   if (TheActor.rnctAuthenticated.nextValueForCapthca
48     = AdtCaptcha)
49   then
50     (TheSystem.nextValueForCaptcha = AdtNewCaptcha.generateCaptcha()
51
52     and AptStringMessageForTheactAuthenticated.eq('The captcha test was successful, you may now try
       to login again.')
53      and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
54      )
55   else (AptStringMessageForTheactAuthenticated
56      .eq('Wrong captcha, you need to succeed the captcha tets in order to be able to login. A new
      captcha will be send.')
57      and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
58      )
59   endif
60
61 }
62
63 postP{
64
65 let TheSystem: ctState in
66 let Bob:actAuthenticated in
67
68 self.rnActor.rnSystem = TheSystem
69 and self.rnActor = TheactAuthenticated
70 /* PostP01 */
71 if (TheActor.rnctAuthenticated.nextValueForCapthca = AdtCaptcha
72 )
73 then (Bob.rnctAuthenticated@post.nbrfails = 0)
74 else true
75 endif
76
77 }
78
79   }
80  }
81 }
```

Listing C.13: Messir Spec. file environment-actAuthenticated-oeFillCaptcha.msr.

C.14 File ./src-gen/messir-spec/operations/environment/environment-actAuthenticated-oeResetPassword.msr

```
1 package icrash.environment.operations.actAuthenticated.outactAuthenticated.oeResetPassword {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
11  Operation Model {
12
13   operation: icrash.environment.actAuthenticated.outactAuthenticated.oeResetPassword(AdtLogin:
        dtLogin):ptBoolean{
14
15 preP{
16
17  let TheSystem:ctState in
18  let Bob:actAuthenticated in
19  self.rnActor.rnSystem = TheSystem
20  and self.rnActor = Bob
21
22 /* PreP01 */
23  and TheSystem.vpStarted = true
24
25 }
26
27 preF {
28
29 true
30 }
31
32 postF{
33
34  let TheSystem: ctState in
35  let Bob:actAuthenticated in
36
37  let AptStringMessageForTheactAuthenticated: ptString in
38  let AptStringMessageForTheactAdministrator:ptString in
39  let AdtNewPassword:dtPassword in
40
41  self.rnActor.rnSystem = TheSystem
42  and self.rnActor = Bob
43
44  and /* PostF01 */
45    if (Bob.rnctAuthenticated.login
46      = AdtLogin
47      )
48    then (
49
50      Bob.rnctAuthenticated.pwd = AdtNewPassword.generatePassword()
51
52       and AptStringMessageForTheactAuthenticated.eq('Your password has been successfully reseted,
      please check your mails.')
53       and Bob.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
54
55       and AptStringMessageForTheactAdministrator.eq(AdtLogin)
56       and AptStringMessageForTheactAdministrator.dtStringConcat('has reset his password.')
57       and TheSystem.rnactAdministrator
58         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
59
60       )
61    else (AptStringMessageForTheactAuthenticated
62       .eq('Wrong identification information ! Please try again ...')
63       and Bob.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
64       and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
```

```
65       and TheSystem.rnactAdministrator
66         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
67       )
68    endif
69
70 }
71
72 postP{
73
74  if(TheactAuthenticated.rnctAuthenticated.login = AdtLogin
75  )
76  then ()
77
78  else true
79  endif
80
81 }
82
83   }
84  }
85 }
```

Listing C.14: Messir Spec. file environment-actAuthenticated-oeResetPassword.msr.

C.15 File ./src-gen/messir-spec/operations/environment/environment-actAuthenticated.msr

```
1 package icrash.operations.environment.actAuthenticated{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 import icrash.concepts.primarytypes.datatypes
6 import icrash.concepts.primarytypes.classes
7 import icrash.concepts.secondarytypes.datatypes
8 import icrash.concepts.secondarytypes.classes
9 import icrash.environment
10
11 Operation Model {
12
13 operation: actAuthenticated.outactAuthenticated.oeLogin(AdtLogin:dtLogin, AdtPassword:dtPassword):
       ptBoolean
14 {
15 preP{
16  let TheSystem: ctState in
17  let TheActor:actAuthenticated in
18  self.rnActor.rnSystem = TheSystem
19  and self.rnActor = TheActor
20
21 /* PreP01 */
22  and TheSystem.vpStarted = true
23 /* PreP02 */
24  and TheActor.rnctAuthenticated.vpIsLogged = false
25  /* PreP03 */
26  and TheActor.rnctAuthenticated.nbrfails < 3
27 }
28 preF{
29 /* PreF01 */
30 true
31 }
32 postF{
33  let TheSystem: ctState in
34  let TheactAuthenticated:actAuthenticated in
35
36  let AptStringMessageForTheactAuthenticated: ptString in
37  let AptStringMessageForTheactAdministrator:ptString in
38
39  self.rnActor.rnSystem = TheSystem
40  and self.rnActor = TheactAuthenticated
```

```
41
42  and /* PostF01 */
43    if (TheactAuthenticated.rnctAuthenticated.pwd
44      = AdtPassword
45      and TheactAuthenticated.rnctAuthenticated.login
46        = AdtLogin
47      )
48    then (AptStringMessageForTheactAuthenticated.eq('You are logged ! Welcome ...')
49       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
50       )
51    else (AptStringMessageForTheactAuthenticated
52       .eq('Wrong identification information ! Please try again ...')
53       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
54       and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
55       and TheSystem.rnactAdministrator
56         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
57       )
58    endif
59 }
60 postP{
61  let TheSystem: ctState in
62  let TheactAuthenticated:actAuthenticated in
63
64  self.rnActor.rnSystem = TheSystem
65  and self.rnActor = TheactAuthenticated
66 /* PostP01 */
67  if (TheactAuthenticated.rnctAuthenticated.pwd = AdtPassword
68    and TheactAuthenticated.rnctAuthenticated.login = AdtLogin
69    )
70  then (TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = true
71    and TheactAuthenticated.rnctAuthenticated@post.nbrfails = 0
72  )
73  else (TheactAuthenticated.rnctAuthenticated@post.nbrfails = TheactAuthenticated.
       rnctAuthenticated@post.nbrfails@pre + 1)
74  endif
75 }
76 prolog{"src/Operations/Environment/OUT/outactAuthenticated-oeLogin.pl"}
77 }
78 /*-------------------------------------------------------------------------------*/
79
80 operation: actAuthenticated.outactAuthenticated.oeLogout():ptBoolean{
81
82 preP{
83  let TheSystem: ctState in
84  let TheActor:actAdministrator in
85  self.rnActor.rnSystem = TheSystem
86  and self.rnActor = TheActor
87
88 /* PreP01 */
89  and TheSystem.vpStarted = true
90 /* PreP02 */
91  and TheActor.rnctAuthenticated.vpIsLogged = true
92 }
93 preF{
94 /* PreF01 */
95 true
96 }
97 postF{
98  let TheSystem: ctState in
99  let TheactAuthenticated:actAuthenticated in
100  let AptStringMessageForTheactAuthenticated: ptString in
101
102  self.rnActor.rnSystem = TheSystem
103  and self.rnActor = TheactAuthenticated
104
105  /* PostF01 */
106  AptStringMessageForTheactAuthenticated.eq('You are logged out ! Good Bye ...')
107  and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
108 }
109 postP{
```

```
110  let TheSystem: ctState in
111  let TheactAuthenticated:actAuthenticated in
112
113  self.rnActor.rnSystem = TheSystem
114  and self.rnActor = TheactAuthenticated.asSet
115 /* PostP01 */
116  TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = false
117 }
118 prolog{"src/Operations/Environment/OUT/outactAuthenticated-oeLogout.pl"}
119 }
120 }
121 }
```

Listing C.15: Messir Spec. file environment-actAuthenticated.msr.

C.16 File ./src-gen/messir-spec/operations/environment/environment-actComCompany.msr

```
 1 // Do not add/remove lines because code is inserted in slides
 2
 3 package icrash.operations.environment.actComCompany{
 4
 5 import lu.uni.lassy.messir.libraries.primitives
 6 import lu.uni.lassy.messir.libraries.calendar
 7 import lu.uni.lassy.messir.libraries.math
 8
 9 import icrash.concepts.primarytypes.datatypes
10 import icrash.concepts.primarytypes.classes
11 import icrash.concepts.secondarytypes.datatypes
12
13 import icrash.environment
14
15 Operation Model {
16
17 operation: actComCompany.outactComCompany.oeAlert(
18  AetKind:etHumanKind,
19  AdtMyDate:dtDate,
20  AdtTime:dtTime,
21  AdtPhoneNumber:dtPhoneNumber,
22  AdtGPSLocation:dtGPSLocation,
23  AdtComment:dtComment
24 ):ptBoolean{
25
26 preP{
27  let TheSystem: ctState in
28  self.rnActor.rnSystem = TheSystem
29
30 /* PreP01 */
31  and TheSystem.vpStarted = true
32 }
33 preF{
34  let TheSystem: ctState in
35  self.rnActor.rnSystem = TheSystem
36
37 /* PreF01 */
38  and (TheSystem.clock.date.gt(AdtDate)
39     or (TheSystem.clock.date.eq(AdtDate)
40       and TheSystem.clock.time.gt(AdtTime)
41      )
42    )
43 }
44 postF{
45  let TheSystem: ctState in
46
47  let ActHuman:ctHuman in
48  let TheactComCompany:actComCompany in
49  let ActAlert:ctAlert in
50  let AAlertInstant:dtDateAndTime in
```

```
51 let AetAlertStatus:etAlertStatus in
52 let ActAlertNearBy:ctAlert in
53 let ActCrisis:ctCrisis in
54 let AdtCrisisID:dtCrisisID in
55 let AetCrisisType:etCrisisType in
56 let AetCrisisStatus:etCrisisStatus in
57 let ACrisisInstant:dtDateAndTime in
58 let ACrisisdtComment:dtComment in
59 let AptStringMessage:ptString in
60 let AdtSMS:dtSMS in
61 let AdtAlertID:dtAlertID in
62
63 self.rnActor.rnSystem = TheSystem
64 and self.rnActor = TheactComCompany
65 /* PostF01 */
66 TheSystem.nextValueForAlertID=PrenextValueForAlertID
67 and PrenextValueForAlertID.add(1) = PostnextValueForAlertID
68 and TheSystem@post.nextValueForAlertID = PostnextValueForAlertID
69
70 /* PostF02 */
71 and AAlertInstant.date=AdtDate
72 and AAlertInstant.time=AdtTime
73
74 and AetAlertStatus=pending
75
76 and TheSystem.nextValueForAlertID.todtString().eq(AdtAlertID)
77
78 and ActAlert.init(AdtAlertID,
79          AetAlertStatus,
80          AdtGPSLocation,
81          AAlertInstant,
82          AdtComment)
83
84 /* PostF03 */
85 and TheSystem.rnctAlert.select(location.isNearTo(AdtGPSLocation)) = ColctAlertsNearBy
86 and if (ColctAlertsNearBy->size()=0)
87  then (TheSystem.nextValueForCrisisID = PrenextValueForCrisisID
88     and PrenextValueForCrisisID.add(1) = PostnextValueForCrisisID
89     and TheSystem@post.nextValueForCrisisID = PostnextValueForCrisisID
90     and TheSystem.nextValueForCrisisID.todtString().eq(AdtCrisisID)
91     and AdtCrisisType = small
92     and AetCrisisStatus = pending
93     and ACrisisInstant= AAlertInstant
94     and ACrisisdtComment = 'no reporting yet defined'
95     and ActCrisis.init( AdtCrisisID,
96               AdtCrisisType,
97               AetCrisisStatus,
98               AdtGPSLocation,
99               ACrisisInstant,
100              ACrisisdtComment)
101    )
102 else (ColctAlertsNearBy.rnTheCrisis->msrAny(true) = ActCrisis)
103 endif
104
105 /* PostF04 */
106 and ActAlert@post.rnTheCrisis = ActCrisis
107
108 /* PostF05 */
109 and TheSystem.rnctHuman->select(id.eq(AdtPhoneNumber)) = HumanCol1
110
111 and HumanCol1->select(kind.etEq(AetHumanKind)) = HumanCol2
112 and if (HumanCol2->msrIsEmpty)
113  then (ActHuman.init(AdtPhoneNumber,AetHumanKind)
114     and ActHuman@post.rnactComCompany = TheactComCompany
115     )
116  else (HumanCol2->any(true) = ActHuman)
117  endif
118
119 and ActHuman.rnSignaled->msrIncluding(ActAlert) = ColAlerts
120
```

```
121  and ActHuman@post.rnSignaled = ColAlerts
122
123 /* PostF06 */
124 AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
125 and TheactComCompany.rnInterfaceIN^ieSmsSend(AdtPhoneNumber,AdtSMS)
126 }
127  /* Post Protocol:*/
128 /* PostP01 */
129 postP{true}
130
131 prolog{"src/Operations/Environment/OUT/outactComCompany-oeAlert.pl"}
132 }
133 }
134 }
```

Listing C.16: Messir Spec. file environment-actComCompany.msr.

C.17 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeCloseCrisis.msr

```
 1 package icrash.operations.environment.actCoordinator.oeCloseCrisis {
 2
 3 import lu.uni.lassy.messir.libraries.primitives
 4 import lu.uni.lassy.messir.libraries.math
 5 import lu.uni.lassy.messir.libraries.string
 6 import lu.uni.lassy.messir.libraries.calendar
 7 import icrash.concepts.primarytypes.datatypes
 8 import icrash.concepts.primarytypes.classes
 9 import icrash.environment
10
11 Operation Model {
12
13 operation: actCoordinator.outactCoordinator.oeCloseCrisis(AdtCrisisID:dtCrisisID):ptBoolean{
14
15  preP{
16
17   let TheSystem:ctState in
18   let Jack:actCoordinator in
19   self.rnActor.rnSystem = TheSystem
20   and self.rnActor = Jack
21
22   /* PreP01 */
23    and TheSystem.vpStarted = true
24   /* PreP02 */
25     and Jack.rnctAuthenticated.vpIsLogged = true
26
27   }
28
29  preF{
30
31   let TheSystem:ctState in
32   let Jack:actCoordinator in
33   self.rnActor.rnSystem = TheSystem
34   and self.rnActor = Jack
35
36   /* PreF01*/
37   TheSystem.rnctCrisis->select(id.eq(AdtCrisisID))
38    = theCrisis
39    and theCrisis->size().eq(1)
40
41   }
42
43  postF{
44
45   let TheSystem:ctState in
46   let Jack:actCoordinator in
47   self.rnActor.rnSystem = TheSystem
48   and self.rnActor = Jack
```

```
49
50   TheSystem.rnctCrisis->select(id.eq(AdtCrisisID))
51    = theCrisis
52
53   /*PostF01*/
54   theCrisis.@post.status="closed"
55
56   /*PostF04*/
57   if(Jack.expPoints == 59)then(theCoordinator.@post.expRank.eq("Expert"))
58   else if(Jack.expPoints == 19) then (theCoordinator.@post.expRank.eq("Intermediate")) else () endif
         endif
59   if(Jack.expPoints<120)then(Jack.@post.expPoints=Jack.expPoints+1)else() endif
60
61   }
62
63  postP{
64    true
65   }
66 }
67 }
68 }
```

Listing C.17: Messir Spec. file environment-actCoordinator-oeCloseCrisis.msr.

C.18 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeGetAlertsSet.msr

```
1 package icrash.operations.environment.actCoordinator.oeGetAlertsSet {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8 import icrash.concepts.primarytypes.datatypes
9 import icrash.environment
10
11 Operation Model {
12
13 operation: actCoordinator.outactCoordinator.oeGetAlertsSet(AetAlertStatus:etAlertStatus):ptBoolean{
14 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeGetAlertsSet.pl"}
15 }
16 }
17 }
```

Listing C.18: Messir Spec. file environment-actCoordinator-oeGetAlertsSet.msr.

C.19 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeGetCrisisSet.msr

```
1 package icrash.operations.environment.actCoordinator.oeGetCrisisSet {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.environment
10
11 Operation Model {
12
13 operation: actCoordinator.outactCoordinator.oeGetCrisisSet(AetCrisisStatus:etCrisisStatus):ptBoolean
       {
14
15  preP{
```

```
16
17  let TheSystem:ctState in
18  let Jack:actCoordinator in
19  self.rnActor.rnSystem = TheSystem
20  and self.rnActor = Jack
21
22  /* PreP01 */
23   and TheSystem.vpStarted = true
24  /* PreP02 */
25    and Jack.rnctAuthenticated.vpIsLogged = true
26
27  }
28
29 preF{
30  true
31 }
32
33 postF{
34
35 let TheSystem:ctState in
36 let Jack:actCoordinator in
37 self.rnActor.rnSystem = TheSystem
38 and self.rnActor = Jack
39
40 let requiredType:etCrisisType in
41 let j:dtInteger in
42 if(Jack.expRank=="Novice") then(requiredType="small")
43 else if(Jack.expRank=="Intermediate")then(requiredType="medium")
44 else (requiredType="huge") endif endif
45
46 TheSystem.rnctCrisis->select(status.eq(AetCrisisStatus) and type.eq(requiredType))
47  = CollCtCrises
48  and j=0
49
50  //PostF01
51   while (j.lt(CollCtCrises.length-1)
52   CollCtCrises.get(j).isSentToCoordinator()
53   and j+1
54
55    )
56
57  }
58 }
59 }
60 }
```

Listing C.19: Messir Spec. file environment-actCoordinator-oeGetCrisisSet.msr.

C.20 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeInvalidateAlert.msr

```
1 package icrash.operations.environment.actCoordinator.oeInvalidateAlert {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.environment
9
10 Operation Model {
11
12 operation: actCoordinator.outactCoordinator.oeInvalidateAlert(AdtAlertID:dtAlertID):ptBoolean{
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeInvalidateAlert.pl"}
14 }
15 }
16 }
```

Listing C.20: Messir Spec. file environment-actCoordinator-oeInvalidateAlert.msr.

C.21 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeReportOnCrisis.msr

```
1 package icrash.operations.environment.actCoordinator.oeReportOnCrisis {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.environment
9
10 Operation Model {
11
12 operation: actCoordinator.outactCoordinator.oeReportOnCrisis(AdtCrisisID:dtCrisisID, AdtComment:
      dtComment):ptBoolean{
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeReportOnCrisis.pl"}
14 }
15
16 }
17 }
```

Listing C.21: Messir Spec. file environment-actCoordinator-oeReportOnCrisis.msr.

C.22 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeSetCrisisHandler.msr

```
1 package icrash.operations.environment.actCoordinator.oeSetCrisisHandler {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8 import icrash.concepts.primarytypes.datatypes
9 import icrash.concepts.primarytypes.classes
10 import icrash.concepts.secondarytypes.datatypes
11 import icrash.environment
12
13 Operation Model {
14
15 operation: actCoordinator.outactCoordinator.oeSetCrisisHandler(AdtCrisisID:dtCrisisID):ptBoolean{
16 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeSetCrisisHandler.pl"}
17 }
18
19 }
20 }
```

Listing C.22: Messir Spec. file environment-actCoordinator-oeSetCrisisHandler.msr.

C.23 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeSetCrisisStatus.msr

```
1 package icrash.operations.environment.actCoordinator.oeSetCrisisStatus {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.environment
9
10 Operation Model {
11
```

```
12 operation: actCoordinator.outactCoordinator.oeSetCrisisStatus(AdtCrisisID:dtCrisisID,
       AetCrisisStatus:etCrisisStatus):ptBoolean{
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeSetCrisisStatus.pl"}
14 }
15
16 }
17 }
```

Listing C.23: Messir Spec. file environment-actCoordinator-oeSetCrisisStatus.msr.

C.24 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeSetCrisisType.msr

```
1 package icrash.operations.environment.actCoordinator.oeSetCrisisType {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.environment
9
10 Operation Model {
11
12 operation: actCoordinator.outactCoordinator.oeSetCrisisType(AdtCrisisID:dtCrisisID, AetCrisisType:
       etCrisisType):ptBoolean{
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeSetCrisisType.pl"}
14 }
15
16 }
17 }
```

Listing C.24: Messir Spec. file environment-actCoordinator-oeSetCrisisType.msr.

C.25 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeValidateAlert.msr

```
1 package icrash.operations.environment.actCoordinator.oeValidateAlert {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.environment
9
10 Operation Model {
11
12 operation: actCoordinator.outactCoordinator.oeValidateAlert(AdtAlertID:dtAlertID):ptBoolean{
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeValidateAlert.pl"}
14 }
15
16 }
17 }
```

Listing C.25: Messir Spec. file environment-actCoordinator-oeValidateAlert.msr.

C.26 File ./src-gen/messir-spec/operations/environment/environment-actMsrCreator-init.msr

```
1 package icrash.operations.icrash.environment.actMsrCreator.init {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import icrash.environment
```

```
5
6 Operation Model {
7
8 operation: actMsrCreator.init():ptBoolean{}
9 // generic operation provided by the simulator
10 }
11 }
```

Listing C.26: Messir Spec. file environment-actMsrCreator-init.msr.

C.27 File ./src-gen/messir-spec/operations/environment/environment-actMsrCreator-oeCreateSystemAndEnvironment.msr

```
1 package icrash.operations.environment.actMsrCreator.oeCreateSystemAndEnvironment{
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.calendar
6
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.concepts.secondarytypes.datatypes
10 import icrash.concepts.secondarytypes.classes
11 import icrash.environment
12
13 Operation Model {
14
15 operation: actMsrCreator.outactMsrCreator.oeCreateSystemAndEnvironment(AqtyComCompanies:ptInteger):
       ptBoolean
16 {preP{true}
17 preF{true}
18 postF{
19  let TheSystem: ctState in
20  let AactMsrCreator: actMsrCreator in
21  let AactAdministrator: actAdministrator in
22  let AnextValueForAlertID: dtInteger in
23  let AnextValueForCrisisID: dtInteger in
24  let Aclock: dtDateAndTime in
25  let AcrisisReminderPeriod: dtSecond in
26  let AmaxCrisisReminderPeriod: dtSecond in
27  let AvpStarted: ptBoolean in
28
29  /* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
30  AnextValueForAlertID.value.eq(1)
31  and AnextValueForCrisisID.value.eq(1)
32  and Aclock.date.year.value = 1970
33  and Aclock.date.month.value = 01
34  and Aclock.date.day.value = 01
35  and Aclock.time.hour.value = 00
36  and Aclock.time.minute.value = 00
37  and Aclock.time.second.value = 00
38
39  and AcrisisReminderPeriod.value.eq(300)
40  and AmaxCrisisReminderPeriod.value.eq(1200)
41  and AvpStarted = true
42  and TheSystem.init(AnextValueForAlertID,
43            AnextValueForCrisisID,
44            Aclock,
45            AcrisisReminderPeriod,
46            AmaxCrisisReminderPeriod,
47            Aclock,
48            AvpStarted
49           )
50  /* PostF02*/
51  and AactMsrCreator.init()
52  /* PostF03 */
53  and let AactComCompanyCol: Bag(actComCompany) in
54  AactComCompanyCol->size() = AqtyComCompanies
```

```
55 AactComCompanyCol-> forAll(init())
56 /* PostF04*/
57 and AactAdministrator.init()
58 /* PostF05*/
59 and let AactActivator:actActivator in
60 AactActivator.init()
61/* PostF06 */
62 and let ActAdministrator:ctAdministrator in
63   let AdtLogin:dtLogin in
64   let AdtPassword:dtPassword in
65   AdtLogin.value.eq('icrashadmin')
66   and AdtPassword.value.eq('7WXC1359')
67   and ActAdministrator.init(AdtLogin,AdtPassword)
68 /* PostF07*/
69 and ActAdministrator@post.rnactAuthenticated = AactAdministrator}
70postP{true}
71
72prolog{ "src/Operations/Environment/OUT/outactMsrCreator-oeCreateSystemAndEnvironment.pl"}
73
74 }
75 }
76
77 }
```

Listing C.27: Messir Spec. file environment-actMsrCreator-oeCreateSystemAndEnvironment.msr.

C.28 File ./src-gen/messir-spec/environment/environment.msr

```
1package icrash.environment{
2
3import icrash.concepts.primarytypes.datatypes
4import icrash.concepts.primarytypes.classes
5import icrash.concepts.secondarytypes.datatypes
6import lu.uni.lassy.messir.libraries.primitives
7import lu.uni.lassy.messir.libraries.math
8import lu.uni.lassy.messir.libraries.calendar
9
10 Environment Model {
11
12  actor actMsrCreator role rnactMsrCreator cardinality [1..1] {
13
14   operation init():ptBoolean
15
16   input interface inactMsrCreator {
17   }
18   output interface outactMsrCreator {
19    operation oeCreateSystemAndEnvironment(AqtyComCompanies:ptInteger ):ptBoolean
20   }
21  }
22
23  actor actAdministrator role rnactAdministrator cardinality [1..1]
24      extends actAuthenticated {
25
26   operation init():ptBoolean
27
28   output interface outactAdministrator{
29
30    operation oeAddCoordinator(
31           AdtCoordinatorID:dtCoordinatorID ,
32           AdtLogin:dtLogin ,
33           AdtPassword:dtPassword ):ptBoolean
34
35    operation oeDeleteCoordinator(
36           AdtCoordinatorID:dtCoordinatorID ):ptBoolean
37
38      operation oeAddPointOfInterest(
39        AetCategory:etCategory,
40        AdtGPSLocation:dtGPSLocation,
41         AdtDescription:dtDescription) :ptBoolean
```

```
42
43      operation oeDeletePointOfInterest(
44          AdtPointOfInterestID:dtPointOfInterestID) : ptBoolean
45
46      operation oeEditPointOfInterest(
47                AdtPointsOfInterestID:dtPointOfInterestID,
48                AetCategory:etCategory,
49                AdtGPSLocation:dtGPSLocation,
50                AdtDescription:dtDescription) :ptBoolean
51
52      operation oeRankDownCoordinator(AdtCoordinatorID:dtCoordinatorID):ptBoolean
53
54      operation oeSelectCategories(AdtCategory:etCategory): ptBoolean //List
55
56      operation oeClosestToALocation(AdtGPSLocation:dtGPSLocation) : ptBoolean //List
57    }
58
59    input interface inactAdministrator{
60
61     operation ieCoordinatorAdded():ptBoolean
62     operation ieCoordinatorDeleted():ptBoolean
63     operation iePointOfInterestAdded():ptBoolean
64     operation iePointOfInterestDeleted():ptBoolean
65     operation iePointOfInterestEdited():ptBoolean
66     operation ieRankEdited():ptBoolean
67     operation ieMessage2(AMessage:ptString):ptBoolean
68     operation ieCategorySelected(ACategory:etCategory):ptBoolean
69      }
70   }
71
72   actor actCoordinator
73       role rnactCoordinator
74       cardinality [0..*]
75       extends actAuthenticated{
76
77    operation init():ptBoolean
78
79    output interface outactCoordinator{
80     operation oeInvalidateAlert(AdtAlertID:dtAlertID ):ptBoolean
81     operation oeCloseCrisis(AdtCrisisID:dtCrisisID ):ptBoolean
82     operation oeGetAlertsSet(AetAlertStatus:etAlertStatus ):ptBoolean
83     operation oeGetCrisisSet(AetCrisisStatus:etCrisisStatus ):ptBoolean
84     operation oeSetCrisisHandler(AdtCrisisID:dtCrisisID ):ptBoolean
85     operation oeReportOnCrisis(
86            AdtCrisisID:dtCrisisID ,
87            AdtComment:dtComment
88            ):ptBoolean
89     operation oeSetCrisisStatus(
90            AdtCrisisID:dtCrisisID ,
91            AetCrisisStatus:etCrisisStatus
92            ):ptBoolean
93     operation oeSetCrisisType(
94                     AdtCrisisID:dtCrisisID ,
95                     AetCrisisType:etCrisisType
96                     ):ptBoolean
97     operation oeValidateAlert(AdtAlertID:dtAlertID ):ptBoolean
98    }
99
100   input interface inactCoordinator{
101    operation ieSendAnAlert(ActAlert:ctAlert ):ptBoolean
102    operation ieSendACrisis(ActCrisis:ctCrisis ):ptBoolean
103     }
104  }
105
106  actor actComCompany role rnactComCompany cardinality [0..*]{
107
108   operation init():ptBoolean
109
110   output interface outactComCompany{
111    operation oeAlert(
```

```
112          AetHumanKind:etHumanKind ,
113          AdtDate:dtDate ,
114          AdtTime:dtTime ,
115          AdtPhoneNumber:dtPhoneNumber ,
116          AdtGPSLocation:dtGPSLocation ,
117          AdtComment:dtComment
118          ):ptBoolean
119    }
120
121    input interface inactComCompany{
122     operation ieSmsSend(AdtPhoneNumber:dtPhoneNumber ,
123               AdtSMS:dtSMS
124               ):ptBoolean
125     }
126   }
127
128   actor actAuthenticated role rnactAuthenticated cardinality [0..*]{
129
130    operation init():ptBoolean
131
132    output interface outactAuthenticated{
133     operation oeLogin(AdtLogin:dtLogin , AdtPassword:dtPassword ):ptBoolean
134     operation oeLogout():ptBoolean
135     operation oeResetPassword(AdtLogin:dtLogin):ptBoolean
136     operation oeFillCaptcha(AdtCaptcha:dtCaptcha):ptBoolean
137    }
138
139    input interface inactAuthenticated{
140     operation ieMessage(AMessage:ptString):ptBoolean
141     }
142   }
143
144   actor actActivator[proactive] role rnactActivator cardinality [1..1]{
145
146    operation init():ptBoolean
147
148    output interface outactActivator{
149     proactive operation oeSollicitateCrisisHandling():ptBoolean
150     proactive operation oeSetClock(AcurrentClock:dtDateAndTime ):ptBoolean
151    }
152
153    input interface inactActivator{
154     }
155   }
156  }}
```

Listing C.28: Messir Spec. file environment.msr.

C.29 File ./src-gen/messir-spec/concepts/primarytypes-associations.msr

```
1 package icrash.concepts.primarytypes.associations {
2
3 import icrash.concepts.primarytypes.datatypes
4 import icrash.concepts.primarytypes.classes
5 import icrash.environment
6 import lu.uni.lassy.messir.libraries.primitives
7
8  Concept Model {
9
10   Primary Types{
11
12 // Internal
13
14 association assctAlertctCrisis
15  ctAlert(rnAlerts)[1..*]
16  ctCrisis (rnTheCrisis)[1..1]
17
```

```
18 association assctAlertctHuman
19  ctAlert(rnSignaled)[1..*]
20  ctHuman (rnSignaler)[1..1]
21
22 association assctCrisisctCoordinator
23  ctCrisis(rnHandled)[0..*]
24  ctCoordinator(rnHandler)[0..1]
25
26 // With Actors
27
28    association assctHumanactComCompany
29       ctHuman(rnctHuman)[0..*]
30       actComCompany(rnactComCompany)[1..1]
31
32    association assctCoordinatoractCoordinator
33       ctCoordinator(rnctCoordinator)[1..1]
34       actCoordinator(rnactCoordinator)[1..1]
35
36    association assctAuthenticatedactAuthenticated
37       ctAuthenticated(rnctAuthenticated)[1..1]
38       actAuthenticated(rnactAuthenticated)[1..1]
39
40    association assctPointOfInterestactAdministrator
41       actAdministrator(rnactAdministrator)[1..1]
42       ctPointOfInterest(rnctPointOfInterest)[0..*]
43          }
44  }
45 }
```

Listing C.29: Messir Spec. file primarytypes-associations.msr.

C.30 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctAdministrator.msr

```
 1 package icrash.operations.concepts.primarytypes.classes.ctAdministrator{
 2
 3 import lu.uni.lassy.messir.libraries.primitives
 4
 5 import icrash.concepts.primarytypes.datatypes
 6 import icrash.concepts.primarytypes.classes
 7
 8 Operation Model {
 9
10 operation: icrash.concepts.primarytypes.classes.ctAdministrator.init(
11  Alogin:dtLogin ,
12  Apwd:dtPassword
13  ):ptBoolean{
14 postF{
15 if
16 (
17 let Self:ctAdministrator in
18 /* Post F01 */
19 Self.login(Alogin)
20 and Self.pwd = Apwd
21 and Self.vpIsLogged = false
22
23 /* Post F02 */
24 and (Self.oclIsNew and self = Self)
25 )
26 then (result = true)
27 else (result = false)
28 endif
29 }
30 prolog{ "src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctAdministrator-init.pl"}
31  }
32 }
```

```
33 }
```

Listing C.30: Messir Spec. file primarytypes-classes-ctAdministrator.msr.

C.31 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctAlert.msr

```
1 package icrash.operations.concepts.primarytypes.classes.ctAlert{
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.calendar
5
6 import icrash.concepts.primarytypes.datatypes
7 import icrash.concepts.primarytypes.classes
8
9 import icrash.environment
10
11 Operation Model {
12
13 operation: icrash.concepts.primarytypes.classes.ctAlert.init(Aid:dtAlertID , Astatus:etAlertStatus ,
       Alocation:dtGPSLocation , Ainstant:dtDateAndTime , Acomment:dtComment
14 ):ptBoolean{
15 postF{
16 if
17 (
18 /* Post F01 */
19 let Self:ctAlert in
20 Self.id = Aid
21 and Self.status = Astatus
22 and Self.location = Alocation
23 and Self.instant = Ainstant
24 and Self.comment = Acomment
25 /* Post F02 */
26 and (Self.oclIsNew and self = Self)
27 )
28 then (result = true)
29 else (result = false)
30 endif
31 }
32 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctAlert-init.pl"}
33 }
34
35 operation: icrash.concepts.primarytypes.classes.ctAlert.isSentToCoordinator(AactCoordinator:
      actCoordinator ):ptBoolean
36 {
37 postF{
38 if
39 (
40 /* Post F01 */
41 AactCoordinator.rnInterfaceIN.ieSendAnAlert(self)
42 )
43 then (result = true)
44 else (result = false)
45 endif
46 }
47 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctAlert-isSentToCoordinator.
      pl"}
48
49 }
50 }
51 }
```

Listing C.31: Messir Spec. file primarytypes-classes-ctAlert.msr.

C.32 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctAuthenticated.msr

```
1 package icrash.operations.concepts.primarytypes.classes.ctAuthenticated {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import icrash.concepts.primarytypes.datatypes
5 import icrash.concepts.primarytypes.classes
6
7 Operation Model {
8
9 operation: icrash.concepts.primarytypes.classes.ctAuthenticated.init(Alogin:dtLogin, Apwd:dtPassword
      , Amail:dtMail):ptBoolean{
10 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctAuthenticated-init.pl"}
11 }
12 }
13
14 }
```

Listing C.32: Messir Spec. file primarytypes-classes-ctAuthenticated.msr.

C.33 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctCoordinator.msr

```
1 package icrash.operations.concepts.primarytypes.classes.ctCoordinator.init {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import icrash.concepts.primarytypes.datatypes
5 import icrash.concepts.primarytypes.classes
6
7 Operation Model {
8
9 operation: icrash.concepts.primarytypes.classes.ctCoordinator.init(Aid:dtCoordinatorID, Alogin:
      dtLogin, Apwd:dtPassword):ptBoolean
10 {
11 postF{
12 if
13 (
14 /* Post F01 */
15 let Self:ctCoordinator in
16 Self.id = Aid
17 and Self.login = Alogin
18 and Self.pwd = Apwd
19 and Self.vpIsLogged = false
20 /* Post F02 */
21 and (Self.oclIsNew and self = Self)
22 )
23 then (result = true)
24 else (result = false)
25 endif}
26 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCoordinator-init.pl"}
27 }
28 }
29 }
```

Listing C.33: Messir Spec. file primarytypes-classes-ctCoordinator.msr.

C.34 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctCrisis.msr

```
1 package icrash.operations.concepts.primarytypes.classes.ctCrisis {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.calendar
6
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9 import icrash.concepts.secondarytypes.datatypes
```

```
10 import icrash.concepts.secondarytypes.classes
11 import lu.uni.lassy.messir.libraries.primitives
12
13 import icrash.environment
14
15 Operation Model {
16 //-------------------------------------------------------------
17 operation: icrash.concepts.primarytypes.classes.ctCrisis.init(
18         Aid:dtCrisisID,
19         Atype:etCrisisType,
20         Astatus:etCrisisStatus,
21         Alocation:dtGPSLocation,
22         Ainstant:dtDateAndTime,
23         Acomment:dtComment
24        ):ptBoolean{
25 postF{
26 if
27 (
28 /* Post F01 */
29 let Self:ctCrisis in
30 Self.id = Aid
31 and Self.type = Atype
32 and Self.status = Astatus
33 and Self.location = Alocation
34 and Self.instant = Ainstant
35 and Self.comment = Acomment
36 /* Post F02 */
37 and (Self.oclIsNew and self = Self)
38 )
39 then (result = true)
40 else (result = false)
41 endif}
42 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-init.pl"}}
43 //-------------------------------------------------------------
44 operation: icrash.concepts.primarytypes.classes.ctCrisis.handlingDelayPassed():ptBoolean
45 {
46 postF{
47 let TheSystem:ctState in
48 let CurrentClockSecondsQty:dtInteger in
49 let vpLastReminderSecondsQty:dtInteger in
50 let CrisisReminderPeriod:dtSecond in
51 if
52 ( /* Post F01 */
53  self.rnSystem = TheSystem
54  and self.status = pending
55  and TheSystem.clock.toSecondsQty() = CurrentClockSecondsQty
56  and TheSystem.vpLastReminder.toSecondsQty() = vpLastReminderSecondsQty
57  and TheSystem.crisisReminderPeriod = CrisisReminderPeriod
58  and CurrentClockSecondsQty.sub(vpLastReminderSecondsQty).gt(CrisisReminderPeriod) = true
59 )
60 then (result = true)
61 else (result = false)
62 endif
63 }
64 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-handlingDelayPassed
       .pl"}}
65 //-------------------------------------------------------------
66 operation: icrash.concepts.primarytypes.classes.ctCrisis.maxHandlingDelayPassed():ptBoolean
67 {
68 postF{
69 let TheSystem:ctState in
70 let CurrentClockSecondsQty:dtInteger in
71 let CrisisInstantSecondsQty:dtInteger in
72 let MaxCrisisReminderPeriod:dtSecond in
73 if
74 ( /* Post F01 */
75  self.rnSystem = TheSystem
76  and self.status = pending
77  and TheSystem.clock.toSecondsQty() = CurrentClockSecondsQty
78  and Self.instant.toSecondsQty() = CrisisInstantSecondsQty
```

```
 and TheSystem.maxCrisisReminderPeriod = MaxCrisisReminderPeriod
 and CurrentClockSecondsQty.sub(CrisisInstantSecondsQty)
               .gt(MaxCrisisReminderPeriod)
)
then (result = true)
else (result = false)
endif
}
prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-
    maxHandlingDelayPassed.pl"}}
//-------------------------------------------------------------
operation: icrash.concepts.primarytypes.classes.ctCrisis.isSentToCoordinator(AactCoordinator:
    actCoordinator):ptBoolean
{
postF{
if
(
/* Post F01 */
 AactCoordinator.rnInterfaceIN.ieSendACrisis(self)
)
then (result = true)
else (result = false)
endif}
prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-isSentToCoordinator
    .pl" }}
//-------------------------------------------------------------
operation: icrash.concepts.primarytypes.classes.ctCrisis.isAllocatedIfPossible():ptBoolean
{
postF{
if (
/* Post F01 */
self.maxHandlingDelayPassed()
and
 if (TheSystem.rnactCoordinator->msrIsEmpty = false)
 then (
   /* Post F02 */
   TheSystem.rnactCoordinator->msrAny(true) = TheCoordinatorActor
   and TheCoordinatorActor.rnctCoordinator = TheCoordinator
   and self@post.rnHandler = TheCoordinator
   and self@post.status = handled
   and self.id.value = TheCrisisIDptString
   and 'You are now considered as handling the crisis having ID: '
     .ptStringConcat(TheCrisisIDptString) = TheMessage
    and TheCoordinatorActor.rnInterfaceIN^ieMessage(TheMessage)
  )
 else ( /* Post F03 */
    TheSystem.rnactAdministrator
    ->forAll(rnInterfaceIN.ieMessage('Please add new coordinators to handle pending crisis !'))
  )
 endif
 )
then (result = true)
else (result = false)
endif
}
prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-
    isAllocatedIfPossible.pl"}
}
}
}
```

Listing C.34: Messir Spec. file primarytypes-classes-ctCrisis.msr.

C.35 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctHuman.msr

```
package icrash.operations.concepts.primarytypes.classes.ctHuman.init {

```

```
3 import lu.uni.lassy.messir.libraries.primitives
4 import icrash.concepts.primarytypes.datatypes
5
6 import icrash.concepts.primarytypes.classes
7
8 Operation Model {
9
10 operation: icrash.concepts.primarytypes.classes.ctHuman.init(Aid:dtPhoneNumber, Akind:etHumanKind):
      ptBoolean
11 {
12 postF{
13 if
14 (
15 /* Post F01 */
16 let Self:ctHuman in
17
18 Self.id = Aid
19 and Self.kind = Akind
20
21 /* Post F02 */
22 and (Self.oclIsNew and self = Self)
23 )
24 then (result = true)
25 else (result = false)
26 endif
27 }
28 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctHuman-init.pl"}
29 }
30 operation: icrash.concepts.primarytypes.classes.ctHuman.isAcknowledged():ptBoolean{
31 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctHuman-isAcknowledged.pl"}
32 }
33 }
34 }
```

Listing C.35: Messir Spec. file primarytypes-classes-ctHuman.msr.

C.36 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctPointOfInterest-init.msr

```
1 package icrash.concepts.primarytypes.classes.operations.classes.ctPointOfInterest.init {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9
10 Operation Model {
11
12  operation: icrash.concepts.primarytypes.classes.ctPointOfInterest.init(Aid:dtPointOfInterestID,
13                   Atype:etCategory,
14                   Alocation:dtGPSLocation,
15                   Adescription:dtDescription
16  ):ptBoolean{
17  // include below the specification information (pre,post or ocl or prolog)
18  postF{
19  if
20  (
21   let Self:ctPointOfInterest in
22   Self.id = Aid
23   and Self.type = Atype
24   and Self.location = Alocation
25   and Self.comment = Acomment
26
27   and (Self.oclIsNew and self = Self)
28  )
29   then (result = true)
```

```
30    else (result = false)
31    endif}
32   }
33  }
34 }
```

Listing C.36: Messir Spec. file primarytypes-classes-ctPointOfInterest-init.msr.

C.37 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctState.msr

```
1 package icrash.operations.concepts.primarytypes.classes.ctState{
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.calendar
5 import lu.uni.lassy.messir.libraries.math
6
7 import icrash.concepts.primarytypes.classes
8
9 Operation Model {
10
11  operation: icrash.concepts.primarytypes.classes.ctState.init(
12   AnextValueForPointOfInterestID: dtInteger,
13   AnextValueForAlertID: dtInteger,
14   AnextValueForCrisisID: dtInteger ,
15   dtAclock:dtDateAndTime,
16   AcrisisReminderPeriod: dtSecond,
17   AmaxCrisisReminderPeriod: dtSecond ,
18   AvpLastReminder: dtDateAndTime ,
19   AvpStarted:ptBoolean ):ptBoolean{
20 postF{
21 if
22 (
23 /* Post F01 */
24 let Self:ctState in
25
26 Self.nextValueForAlertID = AnextValueForAlertID
27 and Self.nextValueForPointOfInterestID = AnextValueForPointOfInterestID
28 and Self.nextValueForCrisisID = AnextValueForCrisisID
29 and Self.clock = Aclock
30 and Self.crisisReminderPeriod = AcrisisReminderPeriod
31 and Self.maxCrisisReminderPeriod = AmaxCrisisReminderPeriod
32 and Self.vpLastReminder = AvpLastReminder
33 and Self.vpStarted = AvpStarted
34
35 and (Self.oclIsNew and self = Self)
36 )
37 then (result = true)
38 else (result = false)
39 endif
40 }
41 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctState-init.pl" }
42  }
43  }
44 }
```

Listing C.37: Messir Spec. file primarytypes-classes-ctState.msr.

C.38 File ./src-gen/messir-spec/concepts/primarytypes-classes.msr

```
1 package icrash.concepts.primarytypes.classes {
2
3 import icrash.concepts.primarytypes.datatypes
4 import icrash.environment
5 import lu.uni.lassy.messir.libraries.primitives
6 import lu.uni.lassy.messir.libraries.math
7 import lu.uni.lassy.messir.libraries.calendar
```

```
8
9  Concept Model {
10
11   Primary Types{
12
13    state class ctState {
14     attribute nextValueForAlertID:dtInteger
15     attribute nextValueForCrisisID:dtInteger
16     attribute nextValueForCaptcha: dtCaptcha
17     attribute nextValueForPointOfInterestID:dtInteger
18     attribute clock:dtDateAndTime
19     attribute crisisReminderPeriod:dtSecond
20     attribute maxCrisisReminderPeriod:dtSecond
21     attribute vpLastReminder:dtDateAndTime
22     attribute vpStarted:ptBoolean
23
24     operation init( AnextValueForAlertID:dtInteger,
25             AnextValueForCrisisID:dtInteger,
26             AnextValueForPointOfInterestID: dtInteger,
27              Aclock:dtDateAndTime,
28             AcrisisReminderPeriod:dtSecond ,
29             AmaxCrisisReminderPeriod:dtSecond ,
30             AvpLastReminder:dtDateAndTime ,
31             AvpStarted:ptBoolean ): ptBoolean
32    }
33
34    class ctAlert role rnctAlert cardinality [0..*]{
35     attribute id:dtAlertID
36     attribute status: etAlertStatus
37     attribute location:dtGPSLocation
38     attribute instant:dtDateAndTime
39     attribute comment:dtComment
40
41     operation init(    Aid:dtAlertID ,
42           Astatus:etAlertStatus ,
43           Alocation:dtGPSLocation ,
44           Ainstant:dtDateAndTime ,
45           Acomment:dtComment ):ptBoolean
46     operation isSentToCoordinator(AactCoordinator:actCoordinator ):ptBoolean
47
48    }
49
50    class ctCrisis role rnctCrisis cardinality [0..*]{
51     attribute id:dtCrisisID
52     attribute type:etCrisisType
53     attribute status: etCrisisStatus
54     attribute location:dtGPSLocation
55     attribute instant:dtDateAndTime
56     attribute comment:dtComment
57
58     operation init(
59           Aid:dtCrisisID ,
60           Atype:etCrisisType ,
61           Astatus:etCrisisStatus ,
62           Alocation:dtGPSLocation ,
63           Ainstant:dtDateAndTime ,
64           Acomment:dtComment ):ptBoolean
65
66     operation handlingDelayPassed():ptBoolean
67     operation maxHandlingDelayPassed():ptBoolean
68     operation isSentToCoordinator(AactCoordinator:actCoordinator ):ptBoolean
69     operation isAllocatedIfPossible():ptBoolean
70    }
71
72    class ctHuman role rnctHuman cardinality [0..*]{
73     attribute id:dtPhoneNumber
74     attribute kind:etHumanKind
75
76     operation init(
77           Aid:dtPhoneNumber ,
```

```
78          Akind:etHumanKind ):ptBoolean
79     operation isAcknowledged():ptBoolean
80    }
81
82     class ctPointOfInterest role rnctPointOfInterest cardinality [0..*] {
83      attribute id:dtPointOfInterestID
84      attribute type:etCategory
85      attribute location:dtGPSLocation
86      attribute description:dtDescription
87
88      operation init(
89            Aid:dtPointOfInterestID,
90            Atype:etCategory,
91            Alocation:dtGPSLocation,
92            Adescription:dtDescription) : ptBoolean
93     }
94
95    class ctAuthenticated
96      role rnctAuthenticated
97      cardinality [0..*]{
98
99     attribute login:dtLogin
100    attribute pwd: dtPassword
101    attribute mail: dtMail
102    attribute nbrfails: dtInteger
103    attribute vpIsLogged:ptBoolean
104
105    operation init(
106          Alogin:dtLogin ,
107          Apwd:dtPassword,
108          Amail:dtMail ):ptBoolean
109   }
110
111   class ctCoordinator
112     role rnctCoordinator
113     cardinality [0..*]
114     {
115
116    attribute id:dtCoordinatorID
117    attribute expRank:etExperienceRank
118    attribute expPoints:dtInteger
119
120    operation init(
121          Aid:dtCoordinatorID ,
122          Alogin:dtLogin ,
123          Apwd:dtPassword ):ptBoolean
124   }
125
126   class ctAdministrator
127     role rnctAdministrator
128     cardinality [1..1]
129     {
130
131    operation init(
132          Alogin:dtLogin ,
133          Apwd:dtPassword ):ptBoolean
134   }
135  }
136 }
137 }
```

Listing C.38: Messir Spec. file primarytypes-classes.msr.

C.39 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtAlertID.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtAlertID{
2
```

```
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtAlertID.is():ptBoolean{
8
9   postF{
10  let TheResult: ptBoolean in
11  ( if
12   ( AdtValue.value.length().gt(0)
13    and AdtValue.value.length().leq(20)
14   )
15   then (TheResult = true)
16   else (TheResult = false)
17   endif
18   result = TheResult
19  )}
20  prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtAlertID-is.pl"}
21 }
22 }
23 }
```

Listing C.39: Messir Spec. file primarytypes-datatypes-dtAlertID.msr.

C.40 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtComment.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtComment{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtComment.is():ptBoolean{
8
9     postF{
10    let TheResult: ptBoolean in
11    ( if
12     ( MaxLength = 160
13      and AdtValue.value.length().leq(MaxLength)
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )
20   }
21  prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtComment-is.pl"}
22 }
23 }
24 }
```

Listing C.40: Messir Spec. file primarytypes-datatypes-dtComment.msr.

C.41 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtCoordinatorID.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtCoordinatorID{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6   operation: icrash.concepts.primarytypes.datatypes.dtCoordinatorID.is():ptBoolean{
7
8   postF{
9     let TheResult: ptBoolean in
```

```
10     ( if
11      ( AdtValue.value.length().gt(0)
12       and AdtValue.value.length().leq(5)
13      )
14      then (TheResult = true)
15      else (TheResult = false)
16      endif
17      result = TheResult
18     )
19   }
20   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtCoordinatorID-is.pl"
       }
21  }
22 }
23 }
```

Listing C.41: Messir Spec. file primarytypes-datatypes-dtCoordinatorID.msr.

C.42 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtCrisisID.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtCrisisID{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtCrisisID.is():ptBoolean{
8
9   postF{
10    let TheResult: ptBoolean in
11    ( if
12     ( AdtValue.value.length().gt(0)
13      and AdtValue.value.length().leq(10)
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )
20   }
21   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtCrisisID-is.pl"}
22  }
23 }
24 }
```

Listing C.42: Messir Spec. file primarytypes-datatypes-dtCrisisID.msr.

C.43 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtDescription-is.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.dtDescription.is {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Operation Model {
9
10   operation: icrash.concepts.primarytypes.datatypes.dtDescription.is():ptBoolean{
11   // include below the specification information (pre,post or ocl or prolog)
12   postF{
13
14    let TheResult: ptBoolean in
15    ( if
```

```
16      ( MaxLength = 280
17       and AdtValue.value.length().leq(MaxLength)
18      )
19      then (TheResult = true)
20      else (TheResult = false)
21      endif
22      result = TheResult
23     )
24
25   }
26
27  }
28 }
29 }
```

Listing C.43: Messir Spec. file primarytypes-datatypes-dtDescription-is.msr.

C.44 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtExpPoints-is.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.dtExpPoints.is {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Operation Model {
9
10  operation: icrash.concepts.primarytypes.datatypes.dtExpPoints.is():ptBoolean{
11  // include below the specification information (pre,post or ocl or prolog)
12
13  postF{
14    let TheResult: ptBoolean in
15    ( if
16     ( self.value.geq(0)
17      and self.value.leq(+120)
18     )
19     then (TheResult = true)
20     else (TheResult = false)
21     endif
22     result = TheResult
23    )
24   }
25
26  }
27 }
28 }
```

Listing C.44: Messir Spec. file primarytypes-datatypes-dtExpPoints-is.msr.

C.45 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtGPSLocation-DistanceTo.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.dtGPSLocation.DistanceTo {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7 import icrash.concepts.primarytypes.datatypes
8 import icrash.concepts.primarytypes.classes
9  Operation Model {
10
```

```
11  operation: icrash.concepts.primarytypes.datatypes.dtGPSLocation.DistanceTo(AGPSLocation:
        dtGPSLocation):ptReal{
12  // include below the specification information (pre,post or ocl or prolog)
13  postF{
14    let Result :ptReal in
15    let EarthRadius : dtReal = 6371000 in
16    let Rlat1 : dtReal in let Rlat1a : dtReal in
17    let Rlong1 : dtReal in let Rlong1a : dtReal in
18    let Deltalat : dtReal in let Deltalong : dtReal in
19    let a1: dtReal in let a2:dtReal in
20    let a3: dtReal in let a : dtReal in
21    let c1: dtReal in let c: dtReal in
22
23    and Rlat1 = self.latitude.toRad()
24    and Rlat1a = adtValue.latitude.toRad()
25    and Rlong1 = self.longitude.toRad()
26    and Rlong1a = adtValue.latitude.toRad()
27    and Deltalat = Rlat1a.sub(Rlat1)
28    and Deltalong = Rlong1a.sub(Rlong1)
29    and a1 = Deltalat.div(2).sin().power(2)
30    and a2 = Rlat1.cos().mult(Rlat1a.cos())
31    and a3 = Deltalong.div(2).sin().power(2)
32    and a4 = a1.add(a2.mult(a3))
33    and c1 = a4.sub(1).power(2)
34    and c = atan(a4.power(2),c1).mult(2)
35    and Result = EarthRadius.mult(c).asptReal()
36        }
37  }
38 }
39 }
```

Listing C.45: Messir Spec. file primarytypes-datatypes-dtGPSLocation-DistanceTo.msr.

C.46 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtGPSLocation.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtGPSLocation{
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5
6 import icrash.concepts.primarytypes.datatypes
7 import icrash.concepts.primarytypes.classes
8 import icrash.concepts.secondarytypes.datatypes
9 import icrash.concepts.secondarytypes.classes
10
11 Operation Model {
12
13  operation: icrash.concepts.primarytypes.datatypes.dtGPSLocation.is():ptBoolean{
14  postF{
15    let TheResult: ptBoolean in
16    ( if
17     ( AdtValue.latitude.is()
18      and AdtValue.longitude.is
19     )
20     then (TheResult = true)
21     else (TheResult = false)
22     endif
23     result = TheResult
24    )
25  }
26  prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtGPSLocation-is.pl"}
27  }
28
29  operation: icrash.concepts.primarytypes.datatypes.dtGPSLocation.isNearTo(aGPSLocation:
        dtGPSLocation):ptBoolean{
30  postF{
31    let TheResult: ptBoolean in true
```

```
32     let EarthRadius: dtReal in
33     let MaxDistance: dtReal in
34     let ComparedLatitude: dtLatitude in
35     let ComparedLongitude: dtLongitude in
36     let R1: dtReal in let R1a: dtReal in
37     let R2: dtReal in let R2a: dtReal in
38
39     ( if
40      ( EarthRadius.value = 6371
41       and MaxDistance.value = 100
42
43       and AdtValue.latitude = ComparedLatitude
44       and AdtValue.longitude = ComparedLongitude
45       and Self.latitude.sin() = R1a
46       and AdtValue.latitude.sin().mul(R1a) = R1
47       and Self.latitude.cos() = R2a
48       and AdtValue.latitude.cos().mul(R2a) = R2
49
50       and AdtValue.longitude = ComparedLongitude
51       and Self.longitude.sub(ComparedLongitude).cos().mul(R2)
52           .add(R1).acos().mul(EarthRadius).sub(MaxDistance)
53           .value.leq(0)
54      )
55      then (TheResult = true)
56      else (TheResult = false)
57      endif
58      result = TheResult
59     )
60   }
61   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtGPSLocation-isNearTo
         .pl"}
62   }
63   operation: icrash.concepts.primarytypes.datatypes.dtLatitude.is():ptBoolean{
64   postF{
65     let TheResult: ptBoolean in
66     ( if
67      ( AdtValue.value.geq(-90.0)
68       and AdtValue.value.leq(+90.0)
69      )
70      then (TheResult = true)
71      else (TheResult = false)
72      endif
73      result = TheResult
74     )}
75   prolog{ "src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtLatitude-is.pl"}
76   }
77   operation: icrash.concepts.primarytypes.datatypes.dtLongitude.is():ptBoolean{
78   postF{
79     let TheResult: ptBoolean in
80     ( if
81      ( AdtValue.value.geq(-180.0)
82       and AdtValue.value.leq(+180.0)
83      )
84      then (TheResult = true)
85      else (TheResult = false)
86      endif
87      result = TheResult
88     )}
89   prolog{ "src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtLongitude-is.pl"}
90  }
91 }
92 }
```

Listing C.46: Messir Spec. file primarytypes-datatypes-dtGPSLocation.msr.

C.47 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtLogin.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtLogin{
```

```
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtLogin.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    let MaxLength: ptInteger in
11    ( if
12     ( MaxLength = 20
13      and AdtValue.value.length().leq(MaxLength)
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )
20    }
21   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtLogin-is.pl"}
22  }
23 }
24 }
```

Listing C.47: Messir Spec. file primarytypes-datatypes-dtLogin.msr.

C.48 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPassword.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtPassword{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtPassword.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    let MinLength: ptInteger in
11    ( if
12     ( MinLength = 6
13      and AdtValue.value.length().geq(MinLength)
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )
20 }
21   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtPassword-is.pl"}
22  }
23 }
24 }
```

Listing C.48: Messir Spec. file primarytypes-datatypes-dtPassword.msr.

C.49 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPhoneNumber.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.dtPhoneNumber{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.dtPhoneNumber.is():ptBoolean{
```

```
 8
 9   postF{
10   let TheResult: ptBoolean in
11   ( if
12    ( AdtValue.value.length().gt(4)
13     and AdtValue.value.length().leq(30)
14    )
15    then (TheResult = true)
16    else (TheResult = false)
17    endif
18    result = TheResult
19   )
20 }
21   prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtPhoneNumber-is.pl"}
22  }
23 }
24 }
```

Listing C.49: Messir Spec. file primarytypes-datatypes-dtPhoneNumber.msr.

C.50 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPointOfInterestID-is.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.dtPointOfInterestID.is {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Operation Model {
9
10   operation: icrash.concepts.primarytypes.datatypes.dtPointOfInterestID.is():ptBoolean{
11   // include below the specification information (pre,post or ocl or prolog)
12   postF{
13   let TheResult: ptBoolean in
14      ( if
15      ( AdtValue.value.length().gt(0)
16     and AdtValue.value.length().leq(30)
17      )
18      then (TheResult = true)
19     else (TheResult = false)
20     endif
21     result = TheResult
22     )}
23   }
24  }
25 }
```

Listing C.50: Messir Spec. file primarytypes-datatypes-dtPointOfInterestID-is.msr.

C.51 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etAlertStatus.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.etAlertStatus{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.etAlertStatus.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    ( if
11     ( self = pending
12      or self = valid
```

```
13       or self = invalid
14      )
15      then (TheResult = true)
16      else (TheResult = false)
17      endif
18      result = TheResult
19     )
20   }
21 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesDatatypes-etAlertStatus-is.pl"}
22 }
23 }
24 }
```

Listing C.51: Messir Spec. file primarytypes-datatypes-etAlertStatus.msr.

C.52 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etCategory-is.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.etCategory.is {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Operation Model {
9
10   operation: icrash.concepts.primarytypes.datatypes.etCategory.is():ptBoolean{
11   // include below the specification information (pre,post or ocl or prolog)
12   postF{
13     let TheResult: ptBoolean in
14     ( if
15      ( self = Hospital
16       or self = Police station
17       or self = Garage
18       or self = Parking
19       or self = Insurance Office
20       or self = Fire station
21      )
22      then (TheResult = true)
23      else (TheResult = false)
24      endif
25      result = TheResult
26     )
27     }
28   }
29  }
30 }
```

Listing C.52: Messir Spec. file primarytypes-datatypes-etCategory-is.msr.

C.53 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etCrisisStatus.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.etCrisisStatus{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.etCrisisStatus.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    ( if
11     ( self = pending
12      or self = handled
```

```
13       or self = solved
14       or self = closed
15      )
16      then (TheResult = true)
17      else (TheResult = false)
18      endif
19      result = TheResult
20     )
21   }
22   prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesDatatypes-etCrisisStatus-is.pl"}
23 }
24 }
25 }
```

Listing C.53: Messir Spec. file primarytypes-datatypes-etCrisisStatus.msr.

C.54 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etCrisisType.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.etCrisisType{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.etCrisisType.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    ( if
11     ( self = small
12      or self = medium
13      or self = huge
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )
20    }
21 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesDatatypes-etCrisisType-is.pl"}
22 }
23 }
24 }
```

Listing C.54: Messir Spec. file primarytypes-datatypes-etCrisisType.msr.

C.55 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etExperienceRank-is.msr

```
1 package icrash.concepts.primarytypes.datatypes.operations.datatypes.etExperienceRank.is {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.math
5 import lu.uni.lassy.messir.libraries.string
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Operation Model {
9
10   operation: icrash.concepts.primarytypes.datatypes.etExperienceRank.is():ptBoolean{
11   // include below the specification information (pre,post or ocl or prolog)
12
13   postF{
14     let TheResult: ptBoolean in
15     ( if
16      ( self = Novice
17       or self = Intermediate
```

```
18       or self = Expert
19      )
20      then (TheResult = true)
21      else (TheResult = false)
22      endif
23      result = TheResult
24     )
25    }
26
27   }
28  }
29 }
```

Listing C.55: Messir Spec. file primarytypes-datatypes-etExperienceRank-is.msr.

C.56 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-etHumanKind.msr

```
1 package icrash.operations.concepts.primarytypes.datatypes.etHumanKind{
2
3 import lu.uni.lassy.messir.libraries.primitives
4
5 Operation Model {
6
7   operation: icrash.concepts.primarytypes.datatypes.etHumanKind.is():ptBoolean{
8   postF{
9     let TheResult: ptBoolean in
10    ( if
11     ( self = witness
12      or self = victim
13      or self = anonymous
14     )
15     then (TheResult = true)
16     else (TheResult = false)
17     endif
18     result = TheResult
19    )}
20 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesDatatypes-etHumanKind-is.pl"}
21 }
22 }
23 }
```

Listing C.56: Messir Spec. file primarytypes-datatypes-etHumanKind.msr.

C.57 File ./src-gen/messir-spec/concepts/primarytypes-datatypes.msr

```
1 package icrash.concepts.primarytypes.datatypes {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.string
5 import lu.uni.lassy.messir.libraries.math
6 import lu.uni.lassy.messir.libraries.calendar
7
8  Concept Model {
9
10   Primary Types {
11     datatype dtPointOfInterestID extends dtString {
12      operation is():ptBoolean
13     }
14    datatype dtAlertID extends dtString {
15    operation is():ptBoolean
16   }
17   datatype dtCrisisID extends dtString {
18    operation is():ptBoolean
19   }
```

```
20   datatype dtLogin extends dtString {
21    operation is():ptBoolean
22   }
23   datatype dtPassword extends dtString {
24    operation is():ptBoolean
25    external operation generatePassword():dtPassword
26   }
27   datatype dtMail extends dtString {
28    operation is():ptBoolean
29   }
30   datatype dtCaptcha extends dtString {
31    operation is():ptBoolean
32    external operation generateCaptcha():dtCaptcha
33   }
34   datatype dtCoordinatorID extends dtString {
35    operation is():ptBoolean
36   }
37   datatype dtPhoneNumber extends dtString {
38    operation is():ptBoolean
39   }
40   datatype dtComment extends dtString {
41    operation is():ptBoolean
42   }
43   datatype dtDescription extends dtString {
44    operation is():ptBoolean
45   }
46   datatype dtLatitude extends dtReal {
47    operation is():ptBoolean
48   }
49   datatype dtLongitude extends dtReal {
50    operation is():ptBoolean
51   }
52   datatype dtGPSLocation {
53    attribute latitude: dtLatitude
54    attribute longitude: dtLongitude
55    operation is():ptBoolean
56    operation isNearTo(AGPSLocation:dtGPSLocation ):ptBoolean
57    operation DistanceTo(AGPSLocation:dtGPSLocation) : ptReal
58   }
59   datatype dtExpPoints extends dtInteger{
60    operation is():ptBoolean
61   }
62
63   enum etCrisisStatus {
64    constants["pending", "handled", "solved","closed"]
65    operation is():ptBoolean
66   }
67   enum etAlertStatus {
68    constants["pending", "valid", "invalid"]
69    operation is():ptBoolean
70   }
71   enum etCrisisType {
72    constants["small", "medium", "huge"]
73    operation is():ptBoolean
74   }
75   enum etHumanKind {
76    constants["witness", "victim", "anonymous"]
77    operation is():ptBoolean
78   }
79   enum etCategory {
80    constants["Hospital","Police station","Garage","Parking","Insurance Office","Fire station" ]
81    operation is():ptBoolean
82   }
83   enum etExperienceRank {
84    constants["novice", "intermediate", "expert"]
85    operation is():ptBoolean
86   }
87  }
88 }
```

```
89 }
```

Listing C.57: Messir Spec. file primarytypes-datatypes.msr.

C.58 File ./src-gen/messir-spec/concepts/secondarytypes-associations.msr

```
1 package icrash.concepts.secondarytypes.associations {
2
3  Concept Model {
4
5   Secondary Types{
6
7   }
8  }
9 }
```

Listing C.58: Messir Spec. file secondarytypes-associations.msr.

C.59 File ./src-gen/messir-spec/concepts/secondarytypes-classes.msr

```
1 package icrash.concepts.secondarytypes.classes {
2
3  Concept Model {
4
5   Secondary Types{
6
7   }
8  }
9 }
```

Listing C.59: Messir Spec. file secondarytypes-classes.msr.

C.60 File ./src-gen/messir-spec/concepts/secondarytypes-datatypes.msr

```
1 package icrash.concepts.secondarytypes.datatypes {
2
3 import lu.uni.lassy.messir.libraries.primitives
4 import lu.uni.lassy.messir.libraries.string
5
6 import icrash.concepts.primarytypes.datatypes
7
8  Concept Model {
9
10  Secondary Types {
11
12   datatype dtSMS {
13    attribute value: ptString
14    operation is():ptBoolean
15   }
16  }
17 }
18 }
```

Listing C.60: Messir Spec. file secondarytypes-datatypes.msr.

C.61 File ./src-gen/messir-spec/usecases/subfunctions-usecases.msr

```
1 package icrash.usecases.subfunctions {
2
```

```
3 import lu.uni.lassy.messir.libraries.primitives
4
5 import icrash.concepts.primarytypes.datatypes
6 import icrash.concepts.primarytypes.classes
7 import icrash.concepts.secondarytypes.datatypes
8 import lu.uni.lassy.messir.libraries.primitives
9 import lu.uni.lassy.messir.libraries.math
10 import lu.uni.lassy.messir.libraries.calendar
11
12 import icrash.environment
13
14 Use Case Model {
15
16 //-----------------------------------------------------
17 use case system subfunction oeAddCoordinator(AdtCoordinatorID:dtCoordinatorID, AdtLogin:dtLogin,
       AdtPassword:dtPassword){
18  actor actAdministrator[primary,active]
19  returned messages {
20   ieCoordinatorAdded() returned to actAdministrator
21  }
22 }
23 //-----------------------------------------------------
24 use case system subfunction oeAlert(
25                  AetKind:etHumanKind,
26                  AdtMyDate:dtDate,
27                  AdtTime:dtTime,
28                  AdtPhoneNumber:dtPhoneNumber,
29                  AdtGPSLocation:dtGPSLocation,
30                  AdtComment:dtComment) {
31  actor actComCompany[primary,active]
32  returned messages {
33   ieSmsSend(AdtPhoneNumber,AdtSMS) returned to actComCompany
34   }
35 }
36 //-----------------------------------------------------
37 use case system subfunction oeInvalidateAlert(AdtAlertID:dtAlertID) {
38  actor actCoordinator[primary,active]
39  actor actComCompany[secondary,passive]
40  returned messages {
41   ieMessage(AMessage) returned to actCoordinator
42  }
43 }
44 //-----------------------------------------------------
45 use case system subfunction oeCloseCrisis(AdtCrisisID:dtCrisisID) {
46  actor actCoordinator[primary,active]
47  returned messages {
48   ieMessage(AMessage) returned to actCoordinator
49  }  }
50 //-----------------------------------------------------
51 use case system subfunction oeCreateSystemAndEnvironment(AqtyComCompanies:ptInteger) {
52  actor actMsrCreator[primary,active]
53 }
54 //-----------------------------------------------------
55 use case system subfunction oeDeleteCoordinator(AdtCoordinatorID:dtCoordinatorID) {
56  actor actAdministrator[primary,active]
57  returned messages {
58   ieCoordinatorDeleted() returned to actAdministrator
59  }
60 }
61 //-----------------------------------------------------
62 use case system subfunction oeGetAlertsSet(AetAlertStatus:etAlertStatus) {
63  actor actCoordinator[primary,active]
64  returned messages {
65   ieSendAnAlert(ActAlert) returned to actCoordinator
66  }
67 }
68 //-----------------------------------------------------
69 use case system subfunction oeGetCrisisSet(AetCrisisStatus:etCrisisStatus){
70  actor actCoordinator[primary,active]
71  returned messages {
```

```
72    ieSendACrisis(ActCrisis) returned to actCoordinator
73   }
74  }
75 //------------------------------------------------------
76  use case system subfunction oeSetCrisisHandler(AdtCrisisID:dtCrisisID) {
77   actor actCoordinator[primary,active]
78   actor actCoordinator[secondary,passive]
79   actor actComCompany[secondary,passive,multiple]
80   returned messages {
81    ieMessage(AMessage)
82     returned to actCoordinator
83    ieSendAnAlert(ActAlert)
84     returned to actCoordinator
85    ieSmsSend(AdtPhoneNumber,AdtSMS)
86     returned to actComCompany
87   }
88  }
89 //------------------------------------------------------
90  use case system subfunction oeLogin(AdtLogin:dtLogin , AdtPassword:dtPassword) {
91   actor actAuthenticated[primary,active]
92   returned messages {
93    ieMessage(AMessage) returned to actAuthenticated
94   }
95  }
96 //------------------------------------------------------
97  use case system subfunction oeLogout() {
98   actor actAuthenticated[primary,active]
99   returned messages {
100   ieMessage(AMessage) returned to actAuthenticated
101  }
102 }
103 //------------------------------------------------------
104 use case system subfunction oeReportOnCrisis(AdtCrisisID:dtCrisisID,AdtComment:dtComment) {
105  actor actCoordinator[primary,active]
106  returned messages {
107   ieMessage(AMessage) returned to actCoordinator
108  }
109 }
110 //------------------------------------------------------
111 use case system subfunction oeSetClock(AcurrentClock:dtDateAndTime) {
112  actor actActivator[primary,proactive]
113 }
114 //------------------------------------------------------
115 use case system subfunction oeSetCrisisStatus(AdtCrisisID:dtCrisisID ,AetCrisisStatus:
      etCrisisStatus) {
116  actor actCoordinator[primary,active]
117  returned messages {
118   ieMessage(AMessage) returned to actCoordinator
119  }
120 }
121 //------------------------------------------------------
122 use case system subfunction oeSollicitateCrisisHandling() {
123  actor actActivator[primary,proactive]
124  actor actCoordinator[secondary,passive,multiple]
125  actor actAdministrator[secondary,passive]
126  returned messages {
127   ieMessage(AMessage) returned to actCoordinator
128   //ieMessage(AMessage) returned to actAdministrator
129  }
130 }
131 //------------------------------------------------------
132 use case system subfunction oeValidateAlert(AdtAlertID:dtAlertID) {
133  actor actCoordinator[primary,active]
134  returned messages {
135   ieMessage(AMessage) returned to actCoordinator
136  }
137 }
138 //------------------------------------------------------
139 use case system subfunction oeResetPassword(AdtMail:dtMail) {
140  actor actAuthenticated[primary,active]
```

```
141   actor actAdministrator[secondary,passive]
142   returned messages {
143    ieMessage(AMessage) returned to actAuthenticated
144     ieMessage2(BMessage) returned to actAdministrator
145   }
146  }
147  use case system subfunction oeFillCaptcha(AdtString:ptString) {
148   actor actAuthenticated[primary,active]
149   returned messages {
150    ieMessage(AMessage) returned to actAuthenticated
151   }
152  }
153  //-------------------------------------------------------
154  use case system subfunction oeAddPointOfInterest(
155                AdtCategory:etCategory,
156                AdtGPSLocation:dtGPSLocation,
157                AdtDescription:dtDescription)
158                {
159    actor actAdministrator[primary, active]
160    returned messages {
161     iePointOfInterestAdded() returned to actAdministrator
162
163    }
164
165   }
166 //-------------------------------------------------------
167  use case system subfunction oeDeletePointOfInterest(AdtPointOfInterestID:dtPointOfInterestID)
168   {
169    actor actAdministrator[primary, active]
170    returned messages {
171     iePointOfInterestDeleted() returned to actAdministrator
172    }
173   }
174 //-------------------------------------------------------
175  use case system subfunction oeEditPointOfInterest(AdtPointOfInterestID:dtPointOfInterestID,
176                 AdtCategory:etCategory,
177                 AdtGPSLocation:dtGPSLocation,
178                 AdtDescription:dtDescription) {
179    actor actAdministrator[primary, active]
180    returned messages {
181     iePointOfInterestEdited() returned to actAdministrator
182    }
183  }
184 //-------------------------------------------------------
185  use case system subfunction oeRankDownCoordinator(AdtCoordinatorID:dtCoordinatorID) {
186
187    actor actAdministrator[primary,active]
188    returned messages {
189     ieRankEdited() returned to actAdministrator
190    }
191  }
192 //-------------------------------------------------------
193 use case system subfunction oeSelectCategories(AdtCategory:etCategory) {
194    actor actAdministrator[primary, active]
195    returned messages{
196     ieCategorySelected(AdtCategory) returned to actAdministrator
197    }
198   }
199 use case system subfunction oeClosestToALocation(aDtGPSLocation:dtGPSLocation) {
200    actor actAdministrator[primary, active]
201
202  }
203
204 use case system subfunction oeSetCrisisType() {
205    actor actCoordinator[primary, active]
206    returned messages {
207       ieMessage(AMessage) returned to actCoordinator
208     }
209   }
210  }
```

```
211 }
```

Listing C.61: Messir Spec. file subfunctions-usecases.msr.

C.62 File ./src-gen/messir-spec/test/tc-testcase01.msr

```
 1 package lu.uni.lassy.excalibur.examples.icrash.tests.testcase01 {
 2
 3 import lu.uni.lassy.messir.libraries.string
 4 import lu.uni.lassy.messir.libraries.primitives
 5 import lu.uni.lassy.messir.libraries.math
 6 import lu.uni.lassy.messir.libraries.calendar
 7
 8 import icrash.concepts.primarytypes.associations
 9 import icrash.concepts.primarytypes.classes
10 import icrash.concepts.primarytypes.datatypes
11 import icrash.concepts.secondarytypes.datatypes
12 import icrash.environment
13
14  Test Model{
15   test case testcase01 order 01 {
16 //-------------------------------------------------------------------
17    test step ts01oeCreateSystemAndEnvironment order 01 {
18     variables{
19      Creator:actMsrCreator
20      AqtyComCompanies: ptInteger
21      }
22     constraints{
23       AqtyComCompanies = 4
24      }
25     test message{
26      out:Creator sends to system actMsrCreator.outactMsrCreator.oeCreateSystemAndEnvironment(
       AqtyComCompanies)
27      }
28     oracle{
29      constraints{
30       true
31      }
32     }
33     prolog{"src/Tests/system/01/system-sim-01-01-oeCreateSystemAndEnvironment.pl"}
34    }
35 //-------------------------------------------------------------------
36  test step ts02oeSetClock order 02{
37      variables{
38       TheActor:actActivator
39       ACurrentClock:dtDateAndTime
40       }
41      constraints{
42       TheActor=TheSystem.rnactActivator->any2(true)
43
44       ACurrentClock.date.year.value = 2017
45       ACurrentClock.date.month.value = 11
46       ACurrentClock.date.day.value = 24
47       ACurrentClock.time.hour.value = 15
48       ACurrentClock.time.minute.value = 20
49       ACurrentClock.time.second.value = 00
50      }
51  test message{
52      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
53     }
54   oracle{
55     constraints{
56      true
57     }
58    }
59    }
60 //-------------------------------------------------------------------
61
62 test step ts03oeLogin order 03{
```

```
63     variables{
64       TheActor : actAdministrator
65       AdtLogin:dtLogin
66       AdtPassword:dtPassword
67      }
68     constraints{
69       TheActor=TheSystem.rnactAdministrator->any2(true)
70       AdtLogin.value.eq('icrashadmin')
71       AdtPassword.value.eq('7WXC1359')
72      }
73     test message{
74      out:TheActor sends to system actAdministrator.outactAdministrator.oeLogin(AdtLogin,AdtPassword)
75     }
76    oracle{
77     variables{
78      AMessage:ptString
79     }
80     constraints{
81      AMessage = 'You are logged ! Welcome ...'
82      TheActor.inactAdministrator.ieMessage(AMessage)
83     }
84    }
85    }
86 //-------------------------------------------------------------------------
87  test step ts04oeAddCoordinator order 04{
88     variables{
89       TheActor : actAdministrator
90       AdtCoordinatorID : dtCoordinatorID
91       AdtLogin:dtLogin
92       AdtPassword:dtPassword
93      }
94     constraints{
95       TheActor = TheSystem.rnactAdministrator->any2(true)
96       AdtCoordinatorID.value.eq('1')
97       AdtLogin.value.eq('steve')
98       AdtPassword.value.eq('pwdMessirExcalibur2017')
99      }
100    test message{
101     out:TheActor
102     sends to system actAdministrator.outactAdministrator.oeAddCoordinator
103                                 (AdtCoordinatorID,
104                                  AdtLogin,
105                                  AdtPassword)
106    }
107    oracle{
108     constraints{
109       TheActor.inactAdministrator.ieCoordinatorAdded()
110      }
111    }
112    }
113 //-------------------------------------------------------------------------
114  test step ts05oeLogout order 05{
115     variables{
116       TheActor : actAdministrator
117      }
118     constraints{
119       TheActor = TheSystem.rnactAdministrator->any2(true)
120      }
121     test message{
122      out:TheActor sends to system actAdministrator.outactAdministrator.oeLogout()
123     }
124     oracle{
125      variables{
126       AMessage:ptString
127      }
128      constraints{
129      AMessage = 'You are logged out ! Good Bye ...'
130      TheActor.inactAdministrator.ieMessage(AMessage)
131     }
132    }
```

```
133   }
134 //-------------------------------------------------------------------
135 test step ts06oeSetClock02 order 06{
136    variables{
137      TheActor:actActivator
138      ACurrentClock:dtDateAndTime
139     }
140    constraints{
141      TheActor=TheSystem.rnactActivator->any2(true)
142      ACurrentClock.date.year.value = 2017
143      ACurrentClock.date.month.value = 11
144      ACurrentClock.date.day.value = 26
145      ACurrentClock.time.hour.value = 10
146      ACurrentClock.time.minute.value = 15
147      ACurrentClock.time.second.value = 00
148     }
149    test message{
150     out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
151    }
152    oracle{
153     constraints{
154     true
155    }
156   }
157   }
158 //-------------------------------------------------------------------
159 test step ts07oeAlert1 order 07{
160    variables{
161      TheActor : actComCompany
162      AetHumanKind:etHumanKind
163      AdtDate:dtDate
164      AdtTime:dtTime
165      AdtPhoneNumber:dtPhoneNumber
166      AdtGPSLocation:dtGPSLocation
167      AdtComment:dtComment
168     }
169    constraints{
170      TheActor = TheSystem.rnactComCompany->any2(true)
171      AetHumanKind = witness
172      AdtDate.year.value = 2017
173      AdtDate.month.value = 11
174      AdtDate.day.value = 26
175      AdtTime.hour.value = 10
176      AdtTime.minute.value = 10
177      AdtTime.second.value = 16
178      AdtPhoneNumber.value = '+3524666445252'
179      AdtGPSLocation.latitude.value = 49.627675
180      AdtGPSLocation.longitude.value = 6.159590
181      AdtComment.value = '3 cars involved in an accident.'
182     }
183    test message{
184     out:TheActor
185     sends to system actComCompany.outactComCompany.oeAlert( AetHumanKind,
186                                 AdtDate,
187                                 AdtTime,
188                                 AdtPhoneNumber,
189                                 AdtGPSLocation,
190                                 AdtComment)
191    }
192    oracle{
193     variables{
194       AdtSMS:dtSMS
195     }
196     constraints{
197     AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
198     TheActor.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
199    }
200   }
201   }
202 //-------------------------------------------------------------------
```

```
203 test step ts08oeSetClock03 order 08{
204    variables{
205      TheActor:actActivator
206      ACurrentClock:dtDateAndTime
207     }
208   constraints{
209      TheActor=TheSystem.rnactActivator->any2(true)
210      ACurrentClock.date.year.value = 2017
211      ACurrentClock.date.month.value = 11
212      ACurrentClock.date.day.value = 26
213      ACurrentClock.time.hour.value = 10
214      ACurrentClock.time.minute.value = 30
215      ACurrentClock.time.second.value = 00
216     }
217    test message{
218     out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
219    }
220    oracle{
221     constraints{
222      true
223     }
224   }
225   }
226 //-------------------------------------------------------------------------
227 test step ts09oeSollicitateCrisisHandling order 09{
228    variables{
229      TheActor : actActivator
230     }
231    constraints{
232      TheActor = TheSystem.rnactActivator->any2(true)
233     }
234    test message{
235     out:TheActor sends to system actActivator.outactActivator.oeSollicitateCrisisHandling()
236    }
237    oracle{
238    variables{
239      TheAdministrator:actAdministrator
240      TheCoordinator:actCoordinator
241      AMessageForCrisisHandlers:ptString
242     }
243     constraints{
244     TheAdministrator = TheSystem.rnactAdministrator->any2(true)
245     TheCoordinator = TheSystem.rnactCoordinator->any2(true)
246     AMessageForCrisisHandlers = 'There are alerts pending since more than the defined delay. Please
        REACT !'
247
248     TheAdministrator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
249     TheCoordinator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
250
251 /* this oracle should be written like this (not currently possible due to grammar limitations:
252
253    oracle{
254    variables{
255      TheAdministrator:actAdministrator
256      AMessageForCrisisHandlers:ptString
257     }
258     constraints{
259     AMessageForCrisisHandlers = 'There are alerts pending since more than the defined delay. Please
        REACT !'
260     TheAdministrator = TheSystem.rnactAdministrator->any2(true)
261
262     TheSystem.rnactCoordinator->forAll(TheCoordinator:actCoordinator | TheCoordinator.
      actAuthenticated.inactAuthenticated.ieMessage(AMessage))
263
264 */
265    }
266   }
267   }
268 //-------------------------------------------------------------------------
269 test step ts10oeLogin02 order 10{
```

```
270     variables{
271       TheActor : actCoordinator
272       AdtLogin:dtLogin
273       AdtPassword:dtPassword
274      }
275     constraints{
276       TheActor = TheSystem.rnactCoordinator->select(a | a.rnctCoordinator.login.value.eq('steve'))->
       any2(true)
277       AdtLogin.value.eq('steve')
278       AdtPassword.value.eq('pwdMessirExcalibur2017')
279      }
280     test message{
281      out:TheActor sends to system actAuthenticated.outactAuthenticated.oeLogin(AdtLogin,AdtPassword)
282     }
283     oracle{
284      variables{
285        AMessage:ptString
286      }
287      constraints{
288      AMessage = 'You are logged ! Welcome ...'
289      TheActor.inactAuthenticated.ieMessage(AMessage)
290     }
291    }
292    }
293 //-------------------------------------------------------------------------
294  test step ts11oeGetCrisisSet order 11{
295     variables{
296       TheActor : actCoordinator
297       AetCrisisStatus : etCrisisStatus
298      }
299     constraints{
300       TheActor=TheSystem.rnactCoordinator
301       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
302       ->any2(true)
303       AetCrisisStatus = pending
304      }
305     test message{
306      out:TheActor sends to system actCoordinator.outactCoordinator.oeGetCrisisSet(AetCrisisStatus)
307     }
308     oracle{
309 //TODO - make consistent with test step implementation by adding Prolog code for input messages
310     variables{
311       ActCrisis:ctCrisis
312     }
313     constraints{
314      TheActor.inactCoordinator.ieSendACrisis(ActCrisis)
315     }
316    }
317    }
318 //-------------------------------------------------------------------------
319  test step ts12oeSetCrisisHandler order 12{
320     variables{
321       TheActor : actCoordinator
322       AdtCrisisID : dtCrisisID
323      }
324     constraints{
325       TheActor=TheSystem.rnactCoordinator
326       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
327       ->any2(true)
328       //and AdtCrisisID.value= '1'
329      }
330     test message{
331      out:TheActor sends to system actCoordinator.outactCoordinator.oeSetCrisisHandler(AdtCrisisID)
332     }
333     oracle{
334      variables{
335       AMessage:ptString
336       AdtPhoneNumber:dtPhoneNumber
337       AdtSMS:dtSMS
338       ActAlert:ctAlert
```

```
339
340       TheComCompany: actComCompany
341       TheCoordinator:actCoordinator
342      }
343      constraints{
344       AMessage = 'You are now considered as handling the crisis !'
345       AdtSMS.value = 'The handling of your alert by our services is in progress !'
346       TheComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
347       TheCoordinator.inactCoordinator.ieSendAnAlert(ActAlert)
348       TheActor.inactAuthenticated.ieMessage(AMessage)
349     }
350    }
351    }
352 //-------------------------------------------------------------------------
353  test step ts13oeSetClock04 order 13{
354     variables{
355       TheActor:actActivator
356       ACurrentClock:dtDateAndTime
357      }
358     constraints{
359       TheActor=TheSystem.rnactActivator->any2(true)
360       ACurrentClock.date.year.value = 2017
361       ACurrentClock.date.month.value = 11
362       ACurrentClock.date.day.value = 26
363       ACurrentClock.time.hour.value = 10
364       ACurrentClock.time.minute.value = 45
365       ACurrentClock.time.second.value = 00
366      }
367     test message{
368      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
369     }
370     oracle{
371      constraints{
372      true
373     }
374    }
375    }
376 //-------------------------------------------------------------------------
377  test step ts14oeValidateAlert order 14{
378     variables{
379       TheActor : actCoordinator
380       AdtAlertID : dtAlertID
381      }
382     constraints{
383       TheActor=TheSystem.rnactCoordinator
384       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
385       ->any2(true)
386       //and AdtAlertID.value= '1'
387      }
388     test message{
389      out:TheActor sends to system actCoordinator.outactCoordinator.oeValidateAlert(AdtAlertID)
390     }
391     oracle{
392      variables{
393        AMessage:ptString
394      }
395      constraints{
396      AMessage = 'The Alert is now declared as valid !'
397      TheActor.actAuthenticated.inactAuthenticated.ieMessage(AMessage)
398     }
399    }
400    }
401 //-------------------------------------------------------------------------
402  test step ts15oeAlert2 order 15{
403     variables{
404       TheActor : actComCompany
405       AetHumanKind:etHumanKind
406       AdtDate:dtDate
407       AdtTime:dtTime
408       AdtPhoneNumber:dtPhoneNumber
```

```
409       AdtGPSLocation:dtGPSLocation
410       AdtComment:dtComment
411      }
412     constraints{
413       TheActor = TheSystem.rnactComCompany->any2(true)
414       AetHumanKind = witness
415       AdtDate.year.value = 2017
416       AdtDate.month.value = 11
417       AdtDate.day.value = 26
418       AdtTime.hour.value = 10
419       AdtTime.minute.value = 20
420       AdtTime.second.value = 00
421       AdtPhoneNumber.value = '+3524666445000'
422       AdtGPSLocation.latitude.value = 49.627095
423       AdtGPSLocation.longitude.value = 6.160251
424       AdtComment.value = 'A car crash just happened.'
425      }
426     test message{
427      out:TheActor
428      sends to system actComCompany.outactComCompany.oeAlert( AetHumanKind,
429                                  AdtDate,
430                                  AdtTime,
431                                  AdtPhoneNumber,
432                                  AdtGPSLocation,
433                                  AdtComment)
434     }
435     oracle{
436      variables{
437       AdtSMS:dtSMS
438      }
439      constraints{
440      AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
441      TheActor.actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
442     }
443     }
444    }
445 //---------------------------------------------------------------------
446  test step ts16oeSetClock05 order 16{
447     variables{
448       TheActor:actActivator
449       ACurrentClock:dtDateAndTime
450      }
451     constraints{
452       TheActor=TheSystem.rnactActivator->any2(true)
453       ACurrentClock.date.year.value = 2017
454       ACurrentClock.date.month.value = 11
455       ACurrentClock.date.day.value = 26
456       ACurrentClock.time.hour.value = 12
457       ACurrentClock.time.minute.value = 45
458       ACurrentClock.time.second.value = 00
459      }
460     test message{
461      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
462     }
463     oracle{
464      constraints{
465      true
466     }
467    }
468    }
469 //---------------------------------------------------------------------
470  test step ts17oeSetCrisisStatus order 17{
471     variables{
472       TheActor : actCoordinator
473       AdtCrisisID : dtCrisisID
474       AetCrisisStatus : etCrisisStatus
475      }
476     constraints{
477       TheActor=TheSystem.rnactCoordinator
478       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
```

```
479       ->any2(true)
480       //and AdtCrisisID.value= '1'
481       //and AetCrisisStatus = solved
482      }
483     test message{
484      out:TheActor sends to system actCoordinator.outactCoordinator.oeSetCrisisStatus(AdtCrisisID,
        AetCrisisStatus)
485     }
486     oracle{
487      variables{
488       AMessage:ptString
489      }
490      constraints{
491      AMessage = 'The crisis status has been updated !'
492      TheActor.inactAuthenticated.ieMessage(AMessage)
493     }
494    }
495    }
496 //-------------------------------------------------------------------------
497  test step ts18oeReportOnCrisis order 18{
498     variables{
499       TheActor : actCoordinator
500       AdtCrisisID : dtCrisisID
501       AdtComment : dtComment
502      }
503     constraints{
504       TheActor=TheSystem.rnactCoordinator
505       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
506       ->any2(true)
507       //and AdtCrisisID.value= '1'
508       //and AdtComment.value = '3 victims sent to hospital, 2 cars evacuated and 4 rescue unit
        mobilized'
509      }
510     test message{
511      out:TheActor sends to system actCoordinator.outactCoordinator.oeReportOnCrisis(AdtCrisisID,
        AdtComment)
512     }
513     oracle{
514      variables{
515       AMessage:ptString
516      }
517      constraints{
518      AMessage = 'The crisis comment has been updated !'
519      TheActor.inactAuthenticated.ieMessage(AMessage)
520     }
521    }
522    }
523 //-------------------------------------------------------------------------
524  test step ts19oeCloseCrisis order 19{
525     variables{
526       TheActor : actCoordinator
527       AdtCrisisID : dtCrisisID
528      }
529     constraints{
530       TheActor=TheSystem.rnactCoordinator
531       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
532       ->any2(true)
533       //and AdtCrisisID.value= '1'
534      }
535     test message{
536      out:TheActor sends to system actCoordinator.outactCoordinator.oeCloseCrisis(AdtCrisisID)
537     }
538     oracle{
539      variables {
540       AMessage:ptString
541      }
542      constraints{
543      AMessage = 'The crisis is now closed !'
544      TheActor.inactAuthenticated.ieMessage(AMessage)
545     }
```

```
546     }
547    }
548   }
549  }
550 }
```

Listing C.62: Messir Spec. file tc-testcase01.msr.

C.63 File ./src-gen/messir-spec/test/tci-testcase01-instance01.msr

```
1 package lu.uni.lassy.excalibur.examples.icrash.tests.testcase01.instance01 {
2
3 import lu.uni.lassy.messir.libraries.string
4 import lu.uni.lassy.messir.libraries.primitives
5 import lu.uni.lassy.messir.libraries.math
6 import lu.uni.lassy.messir.libraries.calendar
7
8 import icrash.concepts.primarytypes.associations
9 import icrash.concepts.primarytypes.classes
10 import icrash.concepts.primarytypes.datatypes
11 import lu.uni.lassy.excalibur.examples.icrash.tests.testcase01
12 import icrash.environment
13
14 Test Model {
15  test case instance instance01:testcase01{
16 //-------------------------------------------------------
17   test step instance tsi01:testcase01.ts01oeCreateSystemAndEnvironment{
18    variables {
19     theCreator:testcase01.ts01oeCreateSystemAndEnvironment.Creator = "theCreator"
20     AqtyComCompanies : testcase01.ts01oeCreateSystemAndEnvironment.AqtyComCompanies="4"
21     }
22     oracle {
23      satisfaction = "true"
24     }
25     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
26    }
27 //-------------------------------------------------------
28   test step instance tsi02: testcase01.ts02oeSetClock{
29    variables {
30     theClock:testcase01.ts02oeSetClock.TheActor = "theClock"
31     ACurrentClock : testcase01.ts02oeSetClock.ACurrentClock= "2017:11:24 - 03:20:00"
32     }
33     oracle {
34      satisfaction = "true"
35     }
36     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
37    }
38 //-------------------------------------------------------
39   test step instance tsi03: testcase01.ts03oeLogin{
40    variables {
41     bill:testcase01.ts03oeLogin.TheActor="bill"
42     AdtLogin : testcase01.ts03oeLogin.AdtLogin= "icrashadmin"
43     AdtPassword : testcase01.ts03oeLogin.AdtPassword= "7WXC1359"
44     }
45     oracle {
46      satisfaction = "true"
47      received message {
48       AMessage : testcase01.ts03oeLogin.AMessage= 'You are logged ! Welcome ...'
49       tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
50      }
51     }
52     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
53    }
54 //-------------------------------------------------------
55   test step instance tsi04: testcase01.ts04oeAddCoordinator{
56    variables {
57     reuse tsi03.bill as testcase01.ts04oeAddCoordinator.TheActor
58     AdtCoordinatorID : testcase01.ts04oeAddCoordinator.AdtCoordinatorID = "1"
59     AdtLogin :testcase01.ts04oeAddCoordinator.AdtLogin= "steve"
```

```
60     AdtPassword : testcase01.ts04oeAddCoordinator.AdtPassword = "pwdMessirExcalibur2017"
61    }
62    oracle {
63      satisfaction = "true"
64      received message {
65      tsi03.bill received from system actAdministrator.inactAdministrator.ieCoordinatorAdded()
66      }
67    }
68     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
69    }
70 //-------------------------------------------------------
71   test step instance tsi05: testcase01.ts05oeLogout{
72    variables {
73     reuse tsi03.bill as testcase01.ts05oeLogout.TheActor
74    }
75    oracle {
76      satisfaction = "true"
77      received message {
78      AMessage : testcase01.ts05oeLogout.AMessage= 'You are logged out ! Good Bye ...'
79      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
80      }
81    }
82     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
83    }
84 //-------------------------------------------------------
85   test step instance tsi06: testcase01.ts06oeSetClock02{
86    variables {
87     reuse tsi02.theClock as testcase01.ts06oeSetClock02.TheActor
88     ACurrentClock : testcase01.ts06oeSetClock02.ACurrentClock= "2017:11:26 - 10:15:00"
89    }
90    oracle {
91      satisfaction = "true"
92    }
93     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
94    }
95 //-------------------------------------------------------
96   test step instance tsi07: testcase01.ts07oeAlert1{
97    variables {
98     tango:testcase01.ts07oeAlert1.TheActor ="tango"
99     AetHumanKind : testcase01.ts07oeAlert1.AetHumanKind = "witness"
100    AdtDate : testcase01.ts07oeAlert1.AdtDate = "2017:11:26"
101    AdtTime : testcase01.ts07oeAlert1.AdtTime = "10:10:16"
102    AdtPhoneNumber : testcase01.ts07oeAlert1.AdtPhoneNumber = "+3524666445252"
103    AdtGPSLocation : testcase01.ts07oeAlert1.AdtGPSLocation = "49.627675:6.159590"
104    AdtComment : testcase01.ts07oeAlert1.AdtComment = "3 cars involved in an accident."
105    }
106   oracle {
107     satisfaction = "true"
108     received message {
109     AdtSMS : testcase01.ts07oeAlert1.AdtSMS= 'Your alert has been registered. We will handle it and
        keep you informed'
110     tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
111
112     }
113   }
114    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
115   }
116
117 //-------------------------------------------------------
118   test step instance tsi08: testcase01.ts08oeSetClock03{
119    variables {
120     reuse tsi02.theClock as testcase01.ts08oeSetClock03.ACurrentClock
121     ACurrentClock : testcase01.ts08oeSetClock03.ACurrentClock = "2017:11:26 - 10:30:00"
122    }
123    oracle {
124      satisfaction = "true"
125    }
126     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
127    }
128 //-------------------------------------------------------
```

```
129  test step instance tsi09: testcase01.ts09oeSollicitateCrisisHandling{
130   variables {
131    reuse tsi02.theClock as testcase01.ts09oeSollicitateCrisisHandling.TheActor
132    reuse tsi03.bill as testcase01.ts09oeSollicitateCrisisHandling.TheAdministrator
133   }
134   oracle {
135     satisfaction = "true"
136     received message {
137      steve:testcase01.ts09oeSollicitateCrisisHandling.TheCoordinator ="steve"
138      AMessageForCrisisHandlers : testcase01.ts09oeSollicitateCrisisHandling.
       AMessageForCrisisHandlers= 'There are alerts pending since more than the defined delay. Please
       REACT !'
139
140     tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
141     tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
142          }
143   }
144    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
145   }
146
147 //-----------------------------------------------------
148  test step instance tsi10: testcase01.ts10oeLogin02{
149   variables {
150    reuse tsi09.steve as testcase01.ts10oeLogin02.TheActor
151    AdtLogin : testcase01.ts10oeLogin02.AdtLogin = "steve"
152    AdtPassword : testcase01.ts10oeLogin02.AdtPassword= "pwdMessirExcalibur2017"
153   }
154   oracle {
155     satisfaction = "true"
156     received message {
157     AMessage : testcase01.ts10oeLogin02.AMessage= 'You are logged ! Welcome ...'
158     tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
159
160     }
161   }
162   test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
163   }
164 //-----------------------------------------------------
165  test step instance tsi11: testcase01.ts11oeGetCrisisSet{
166   variables {
167    reuse tsi09.steve as testcase01.ts11oeGetCrisisSet.TheActor
168    AetCrisisStatus : testcase01.ts11oeGetCrisisSet.AetCrisisStatus = "pending"
169   }
170   oracle {
171     satisfaction = "true"
172     received message {
173     ActCrisis : testcase01.ts11oeGetCrisisSet.ActCrisis= "crisis with ID 1 details"
174     tsi09.steve received from system actCoordinator.inactCoordinator.ieSendACrisis(ActCrisis)
175     }
176   }
177   test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
178   }
179 //-----------------------------------------------------
180  test step instance tsi12: testcase01.ts12oeSetCrisisHandler{
181   variables {
182    reuse tsi09.steve as testcase01.ts12oeSetCrisisHandler.TheActor
183    AdtCrisisID : testcase01.ts12oeSetCrisisHandler.AdtCrisisID = "1"
184
185     reuse tsi07.tango as testcase01.ts12oeSetCrisisHandler.TheComCompany
186
187   }
188   oracle {
189     satisfaction = "true"
190     received message {
191     AMessage : testcase01.ts12oeSetCrisisHandler.AMessage= 'You are now considered as handling the
       crisis !'
192     AdtSMS : testcase01.ts12oeSetCrisisHandler.AdtSMS= 'The handling of your alert by our services
          is in progress !'
```

```
193      AdtPhoneNumber : testcase01.ts12oeSetCrisisHandler.AdtPhoneNumber= "+3524666445252"
194
195      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
196      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
197
198      }
199    }
200     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
201    }
202 //-------------------------------------------------------
203    test step instance tsi13: testcase01.ts13oeSetClock04{
204     variables {
205      reuse tsi02.theClock as testcase01.ts13oeSetClock04.TheActor
206       ACurrentClock : testcase01.ts13oeSetClock04.ACurrentClock = "2017:11:26 - 10:45:00"
207     }
208     oracle {
209       satisfaction = "true"
210     }
211     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
212     }
213 //-------------------------------------------------------
214    test step instance tsi14: testcase01.ts14oeValidateAlert{
215     variables {
216      reuse tsi09.steve as testcase01.ts14oeValidateAlert.TheActor
217      AdtAlertID : testcase01.ts14oeValidateAlert.AdtAlertID = "1"
218      }
219     oracle {
220       satisfaction = "true"
221       received message {
222       AMessage : testcase01.ts14oeValidateAlert.AMessage= 'The Alert is now declared as valid !'
223       tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
224
225       }
226     }
227     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
228     }
229 //-------------------------------------------------------
230    test step instance tsi15: testcase01.ts15oeAlert2{
231     variables {
232      reuse tsi07.tango as testcase01.ts15oeAlert2.TheActor
233      AetHumanKind : testcase01.ts15oeAlert2.AetHumanKind ="witness"
234      AdtDate : testcase01.ts15oeAlert2.AdtDate= "2017:11:26"
235      AdtTime : testcase01.ts15oeAlert2.AdtTime= "10:20:00"
236      AdtPhoneNumber : testcase01.ts15oeAlert2.AdtPhoneNumber= "+3524666445000"
237      AdtGPSLocation : testcase01.ts15oeAlert2.AdtGPSLocation= "49.627095:6.160251"
238      AdtComment : testcase01.ts15oeAlert2.AdtComment= "A car crash just happened."
239     }
240     message {
241      tsi07.tango sent to system testcase01.ts15oeAlert2.out : actComCompany.outactComCompany.oeAlert(
           AetHumanKind,AdtDate,AdtTime,AdtPhoneNumber,AdtGPSLocation,AdtComment)
242
243     }
244     oracle {
245       satisfaction = "true"
246       received message {
247       AdtSMS : testcase01.ts15oeAlert2.AdtSMS= 'Your alert has been registered. We will handle it and
          keep you informed'
248       tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
249
250       }
251     }
252     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
253     }
254 //-------------------------------------------------------
255    test step instance tsi16: testcase01.ts16oeSetClock05{
256     variables {
257      reuse tsi02.theClock as testcase01.ts16oeSetClock05.TheActor
258      ACurrentClock : testcase01.ts16oeSetClock05.ACurrentClock = "2017:11:26 - 12:45:00"
259     }
260     oracle {
```

```
261      satisfaction = "true"
262      received message {
263
264      }
265    }
266     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
267    }
268 //-----------------------------------------------------
269   test step instance tsi17: testcase01.ts17oeSetCrisisStatus{
270    variables {
271     reuse tsi09.steve as testcase01.ts17oeSetCrisisStatus.TheActor
272     AdtCrisisID : testcase01.ts17oeSetCrisisStatus.AdtCrisisID = "1"
273     AetCrisisStatus : testcase01.ts17oeSetCrisisStatus.AetCrisisStatus= "solved"
274    }
275    oracle {
276      satisfaction = "true"
277      received message {
278      AMessage : testcase01.ts17oeSetCrisisStatus.AMessage= "The crisis status has been updated !"
279      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
280      }
281    }
282     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
283    }
284 //-----------------------------------------------------
285   test step instance tsi18: testcase01.ts18oeReportOnCrisis{
286    variables {
287     reuse tsi09.steve as testcase01.ts18oeReportOnCrisis.TheActor
288     AdtCrisisID : testcase01.ts18oeReportOnCrisis.AdtCrisisID = "1"
289     AdtComment : testcase01.ts18oeReportOnCrisis.AdtComment= "3 victims sent to hospital, 2 cars
        evacuated and 4 rescue unit mobilized"
290    }
291    oracle {
292      satisfaction = "true"
293      received message {
294      AMessage : testcase01.ts18oeReportOnCrisis.AMessage= 'The crisis comment has been updated !'
295      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
296
297      }
298    }
299     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
300    }
301 //-----------------------------------------------------
302   test step instance tsi19: testcase01.ts19oeCloseCrisis{
303    variables {
304     reuse tsi09.steve as testcase01.ts19oeCloseCrisis.TheActor
305     AdtCrisisID : testcase01.ts19oeCloseCrisis.AdtCrisisID = "1"
306    }
307    oracle {
308      satisfaction = "true"
309      received message {
310      AMessage : testcase01.ts19oeCloseCrisis.AMessage= 'The crisis is now closed !'
311
312     tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
313
314      }
315    }
316     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
317    }
318
319   }
320 //-----------------------------------------------------
321 //-----------------------------------------------------
322 //-----------------------------------------------------
323  test case instance instance01Part01:testcase01{
324 //-----------------------------------------------------
325   test step instance tsi01:testcase01.ts01oeCreateSystemAndEnvironment{
326    variables {
327     theCreator:testcase01.ts01oeCreateSystemAndEnvironment.Creator = "theCreator"
328     AqtyComCompanies : testcase01.ts01oeCreateSystemAndEnvironment.AqtyComCompanies="4"
329     }
```

```
330     oracle {
331      satisfaction = "true"
332     }
333     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
334    }
335 //-----------------------------------------------------
336   test step instance tsi02: testcase01.ts02oeSetClock{
337    variables {
338     theClock:testcase01.ts02oeSetClock.TheActor = "theClock"
339     ACurrentClock : testcase01.ts02oeSetClock.ACurrentClock= "2017:11:24 - 03:20:00"
340     }
341     oracle {
342      satisfaction = "true"
343     }
344     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
345    }
346 //-----------------------------------------------------
347   test step instance tsi03: testcase01.ts03oeLogin{
348    variables {
349     bill:testcase01.ts03oeLogin.TheActor="bill"
350     AdtLogin : testcase01.ts03oeLogin.AdtLogin= "icrashadmin"
351     AdtPassword : testcase01.ts03oeLogin.AdtPassword= "7WXC1359"
352     }
353     oracle {
354      satisfaction = "true"
355      received message {
356       AMessage : testcase01.ts03oeLogin.AMessage= 'You are logged ! Welcome ...'
357       tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
358      }
359     }
360     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
361    }
362 //-----------------------------------------------------
363   test step instance tsi04: testcase01.ts04oeAddCoordinator{
364    variables {
365     reuse tsi03.bill as testcase01.ts04oeAddCoordinator.TheActor
366     AdtCoordinatorID : testcase01.ts04oeAddCoordinator.AdtCoordinatorID = "1"
367     AdtLogin :testcase01.ts04oeAddCoordinator.AdtLogin= "steve"
368     AdtPassword : testcase01.ts04oeAddCoordinator.AdtPassword = "pwdMessirExcalibur2017"
369    }
370    oracle {
371      satisfaction = "true"
372      received message {
373      tsi03.bill received from system actAdministrator.inactAdministrator.ieCoordinatorAdded()
374      }
375    }
376     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
377    }
378 //-----------------------------------------------------
379   test step instance tsi05: testcase01.ts05oeLogout{
380    variables {
381     reuse tsi03.bill as testcase01.ts05oeLogout.TheActor
382    }
383    oracle {
384      satisfaction = "true"
385      received message {
386      AMessage : testcase01.ts05oeLogout.AMessage= 'You are logged out ! Good Bye ...'
387      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
388      }
389    }
390     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
391    }
392 //-----------------------------------------------------
393   test step instance tsi06: testcase01.ts06oeSetClock02{
394    variables {
395     reuse tsi02.theClock as testcase01.ts06oeSetClock02.TheActor
396     ACurrentClock : testcase01.ts06oeSetClock02.ACurrentClock= "2017:11:26 - 10:15:00"
397    }
398    oracle {
399      satisfaction = "true"
```

```
400    }
401     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
402    }
403 //-------------------------------------------------------
404   test step instance tsi07: testcase01.ts07oeAlert1{
405    variables {
406     tango:testcase01.ts07oeAlert1.TheActor ="tango"
407     AetHumanKind : testcase01.ts07oeAlert1.AetHumanKind = "witness"
408     AdtDate : testcase01.ts07oeAlert1.AdtDate = "2017:11:26"
409     AdtTime : testcase01.ts07oeAlert1.AdtTime = "10:10:16"
410     AdtPhoneNumber : testcase01.ts07oeAlert1.AdtPhoneNumber = "+3524666445252"
411     AdtGPSLocation : testcase01.ts07oeAlert1.AdtGPSLocation = "49.627675:6.159590"
412     AdtComment : testcase01.ts07oeAlert1.AdtComment = "3 cars involved in an accident."
413     }
414    oracle {
415      satisfaction = "true"
416      received message {
417      AdtSMS : testcase01.ts07oeAlert1.AdtSMS= 'Your alert has been registered. We will handle it and
         keep you informed'
418      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
419
420      }
421    }
422     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
423    }
424
425 //-------------------------------------------------------
426   test step instance tsi08: testcase01.ts08oeSetClock03{
427    variables {
428     reuse tsi02.theClock as testcase01.ts08oeSetClock03.ACurrentClock
429     ACurrentClock : testcase01.ts08oeSetClock03.ACurrentClock = "2017:11:26 - 10:30:00"
430    }
431    oracle {
432      satisfaction = "true"
433    }
434     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
435    }
436 //-------------------------------------------------------
437   test step instance tsi09: testcase01.ts09oeSollicitateCrisisHandling{
438    variables {
439     reuse tsi02.theClock as testcase01.ts09oeSollicitateCrisisHandling.TheActor
440     reuse tsi03.bill as testcase01.ts09oeSollicitateCrisisHandling.TheAdministrator
441    }
442    oracle {
443      satisfaction = "true"
444      received message {
445       steve:testcase01.ts09oeSollicitateCrisisHandling.TheCoordinator ="steve"
446       AMessageForCrisisHandlers : testcase01.ts09oeSollicitateCrisisHandling.
       AMessageForCrisisHandlers= 'There are alerts pending since more than the defined delay. Please
       REACT !'
447
448      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
449      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
450           }
451    }
452     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
453    }
454    }
455
456 //-------------------------------------------------------
457 //-------------------------------------------------------
458 //-------------------------------------------------------
459  test case instance instance01Part02:testcase01{
460
461   test step instance tsi10: testcase01.ts10oeLogin02{
462    variables {
463     steve : testcase01.ts10oeLogin02.TheActor
464     AdtLogin : testcase01.ts10oeLogin02.AdtLogin = "steve"
```

```
465     AdtPassword : testcase01.ts10oeLogin02.AdtPassword= "pwdMessirExcalibur2017"
466    }
467    oracle {
468      satisfaction = "true"
469      received message {
470      AMessage : testcase01.ts10oeLogin02.AMessage= 'You are logged ! Welcome ...'
471      steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
472
473      }
474    }
475    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
476    }
477 //-----------------------------------------------------
478   test step instance tsi11: testcase01.ts11oeGetCrisisSet{
479    variables {
480     reuse tsi10.steve as testcase01.ts11oeGetCrisisSet.TheActor
481     AetCrisisStatus : testcase01.ts11oeGetCrisisSet.AetCrisisStatus = "pending"
482    }
483    oracle {
484      satisfaction = "true"
485      received message {
486      ActCrisis : testcase01.ts11oeGetCrisisSet.ActCrisis= "crisis with ID 1 details"
487      tsi10.steve received from system actCoordinator.inactCoordinator.ieSendACrisis(ActCrisis)
488      }
489    }
490    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
491    }
492 //-----------------------------------------------------
493   test step instance tsi12: testcase01.ts12oeSetCrisisHandler{
494    variables {
495     reuse tsi10.steve as testcase01.ts12oeSetCrisisHandler.TheActor
496     AdtCrisisID : testcase01.ts12oeSetCrisisHandler.AdtCrisisID = "1"
497    }
498    oracle {
499      satisfaction = "true"
500      received message {
501      tango : testcase01.ts12oeSetCrisisHandler.TheComCompany
502      AMessage : testcase01.ts12oeSetCrisisHandler.AMessage= 'You are now considered as handling the
       crisis !'
503      AdtSMS : testcase01.ts12oeSetCrisisHandler.AdtSMS= 'The handling of your alert by our services
       is in progress !'
504      AdtPhoneNumber : testcase01.ts12oeSetCrisisHandler.AdtPhoneNumber= "+3524666445252"
505
506      tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
507      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
508
509      }
510    }
511     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
512    }
513 //-----------------------------------------------------
514   test step instance tsi13: testcase01.ts13oeSetClock04{
515    variables {
516      theClock : testcase01.ts13oeSetClock04.TheActor
517      ACurrentClock : testcase01.ts13oeSetClock04.ACurrentClock = "2017:11:26 - 10:45:00"
518    }
519    oracle {
520      satisfaction = "true"
521    }
522    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
523    }
524 //-----------------------------------------------------
525   test step instance tsi14: testcase01.ts14oeValidateAlert{
526    variables {
527     reuse tsi10.steve as testcase01.ts14oeValidateAlert.TheActor
528     AdtAlertID : testcase01.ts14oeValidateAlert.AdtAlertID = "1"
529     }
530    oracle {
531      satisfaction = "true"
532      received message {
```

```
533      AMessage : testcase01.ts14oeValidateAlert.AMessage= 'The Alert is now declared as valid !'
534      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
535
536      }
537    }
538     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
539    }
540 //-----------------------------------------------------
541   test step instance tsi15: testcase01.ts15oeAlert2{
542    variables {
543     reuse tsi12.tango as testcase01.ts15oeAlert2.TheActor
544     AetHumanKind : testcase01.ts15oeAlert2.AetHumanKind ="witness"
545     AdtDate : testcase01.ts15oeAlert2.AdtDate= "2017:11:26"
546     AdtTime : testcase01.ts15oeAlert2.AdtTime= "10:20:00"
547     AdtPhoneNumber : testcase01.ts15oeAlert2.AdtPhoneNumber= "+3524666445000"
548     AdtGPSLocation : testcase01.ts15oeAlert2.AdtGPSLocation= "49.627095:6.160251"
549     AdtComment : testcase01.ts15oeAlert2.AdtComment= "A car crash just happened."
550    }
551    message {
552     tsi12.tango sent to system testcase01.ts15oeAlert2.out : actComCompany.outactComCompany.oeAlert(
       AetHumanKind,AdtDate,AdtTime,AdtPhoneNumber,AdtGPSLocation,AdtComment)
553
554    }
555    oracle {
556      satisfaction = "true"
557      received message {
558      AdtSMS : testcase01.ts15oeAlert2.AdtSMS= 'Your alert has been registered. We will handle it and
       keep you informed'
559      tsi12.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
560
561      }
562    }
563     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
564    }
565 //-----------------------------------------------------
566   test step instance tsi16: testcase01.ts16oeSetClock05{
567    variables {
568     reuse tsi13.theClock as testcase01.ts16oeSetClock05.TheActor
569     ACurrentClock : testcase01.ts16oeSetClock05.ACurrentClock = "2017:11:26 - 12:45:00"
570    }
571    oracle {
572      satisfaction = "true"
573      received message {
574
575      }
576    }
577     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
578    }
579 //-----------------------------------------------------
580   test step instance tsi17: testcase01.ts17oeSetCrisisStatus{
581    variables {
582     reuse tsi10.steve as testcase01.ts17oeSetCrisisStatus.TheActor
583     AdtCrisisID : testcase01.ts17oeSetCrisisStatus.AdtCrisisID = "1"
584     AetCrisisStatus : testcase01.ts17oeSetCrisisStatus.AetCrisisStatus= "solved"
585    }
586    oracle {
587      satisfaction = "true"
588      received message {
589      AMessage : testcase01.ts17oeSetCrisisStatus.AMessage= "The crisis status has been updated !"
590      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
591      }
592    }
593     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
594    }
595 //-----------------------------------------------------
596   test step instance tsi18: testcase01.ts18oeReportOnCrisis{
597    variables {
598     reuse tsi10.steve as testcase01.ts18oeReportOnCrisis.TheActor
599     AdtCrisisID : testcase01.ts18oeReportOnCrisis.AdtCrisisID = "1"
600     AdtComment : testcase01.ts18oeReportOnCrisis.AdtComment= "3 victims sent to hospital, 2 cars
```

```
      evacuated and 4 rescue unit mobilized"
601    }
602    oracle {
603      satisfaction = "true"
604      received message {
605      AMessage : testcase01.ts18oeReportOnCrisis.AMessage= 'The crisis comment has been updated !'
606      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
607
608      }
609    }
610     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
611    }
612 //-----------------------------------------------------
613   test step instance tsi19: testcase01.ts19oeCloseCrisis{
614    variables {
615     reuse tsi10.steve as testcase01.ts19oeCloseCrisis.TheActor
616     AdtCrisisID : testcase01.ts19oeCloseCrisis.AdtCrisisID = "1"
617    }
618    oracle {
619      satisfaction = "true"
620      received message {
621      AMessage : testcase01.ts19oeCloseCrisis.AMessage= 'The crisis is now closed !'
622
623     tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
624
625      }
626    }
627     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
628    }
629
630   }
631  }
632
633 }
```

Listing C.63: Messir Spec. file tci-testcase01-instance01.msr.

C.64 File ./src-gen/messir-spec/usecases/usecase-suDeployAndRun.msr

```
1 package icrash.usecases.suDeployAndRun {
2  import icrash.concepts.primarytypes.datatypes
3  import icrash.environment
4  import icrash.usecases.suGlobalCrisisHandling
5  import icrash.usecases.ugAdministrateTheSystem
6  import icrash.usecases.subfunctions
7
8  Use Case Model {
9   use case system summary suDeployAndRun() {
10    actor actAdministrator[primary,active]
11    actor actMsrCreator[secondary,active]
12    actor actCoordinator[secondary,active,multiple]
13    actor actActivator[secondary,proactive]
14    actor actComCompany[secondary,active]
15
16    reuse oeCreateSystemAndEnvironment[1..1]
17    reuse ugAdministrateTheSystem[1..*]
18    reuse suGlobalCrisisHandling[1..*]
19    reuse oeSetClock[1..*]
20    reuse oeSollicitateCrisisHandling[0..*]
21    reuse oeAlert[1..*]
22
23    step a: actMsrCreator executes oeCreateSystemAndEnvironment
24    step b: actAdministrator executes ugAdministrateTheSystem
25    step c: actComCompany executes oeAlert
26    step d: actActivator executes oeSetClock
27    step ^e: actActivator executes oeSollicitateCrisisHandling
28    step f: actCoordinator executes suGlobalCrisisHandling
```

```
29
30    ordering constraint
31     "step (a) must be always the first step."
32    ordering constraint
33     "step (f) can be executed by different actCoordinator actors."
34    ordering constraint
35     "if (e) then previously (d)."
36   }
37 //-------------------------------------------------------
38 //-------------------------------------------------------
39 //-------------------------------------------------------
40   use case instance uciSimpleAndComplete : suDeployAndRun {
41    actors {
42     theCreator : actMsrCreator
43     theClock : actActivator
44     bill : actAdministrator
45     tango : actComCompany
46     steve : actCoordinator
47    }
48    use case steps {
49 //-------------------------------------------------------
50     theCreator
51     executed instanceof subfunction
52        oeCreateSystemAndEnvironment("4"){}
53 //-------------------------------------------------------
54     theClock
55     executed instanceof subfunction
56        oeSetClock("2017:11:24 - 03:20:00"){}
57 //-------------------------------------------------------
58     bill
59     executed instanceof subfunction
60        oeLogin("icrashadmin","7WXC1359"){
61         ieMessage('You are logged ! Welcome ...') returned to bill
62        }
63 //-------------------------------------------------------
64     bill
65     executed instanceof subfunction
66        oeAddCoordinator("1","steve","pwdMessirExcalibur2017"){
67         ieCoordinatorAddedreturned returned to bill
68        }
69 //-------------------------------------------------------
70     bill
71     executed instanceof subfunction
72        oeLogout{
73         ieMessage('You are logged out ! Good Bye ...') returned to bill
74        }
75 //-------------------------------------------------------
76     theClock
77     executed instanceof subfunction
78        oeSetClock("2017:11:26 - 10:15:00"){}
79 //-------------------------------------------------------
80     tango
81     executed instanceof subfunction
82        oeAlert("witness","2017:11:26","10:10:16","+3524666445252",
83            "49.627675:6.159590","3 cars involved in an accident."){
84          ieSmsSend("+3524666445252","Your alert has been registered. We will handle it and keep you
     informed") returned to tango
85        }
86 //-------------------------------------------------------
87     theClock
88     executed instanceof subfunction
89        oeSetClock("2017:11:26 - 10:30:00"){}
90 //-------------------------------------------------------
91     theClock
92     executed instanceof subfunction
93        oeSollicitateCrisisHandling{
94         ieMessage("There are alerts pending since more than the defined delay. Please REACT !")
95         returned to bill
96         ieMessage("There are alerts pending since more than the defined delay. Please REACT !")
97         returned to steve
```

```
98        }
99  //-------------------------------------------------
100     steve
101     executed instanceof subfunction
102        oeLogin("steve","pwdMessirExcalibur2017"){
103         ieMessage('You are logged ! Welcome ...') returned to steve
104        }
105 //-------------------------------------------------
106     steve
107     executed instanceof subfunction
108        oeGetCrisisSet("pending"){
109         ieSendACrisis("crisis with ID 1 details") returned to steve
110        }
111 //-------------------------------------------------
112     steve
113     executed instanceof subfunction
114        oeSetCrisisHandler("1"){
115         ieSmsSend("+3524666445252","The handling of your alert by our services is in progress !")
116         returned to tango
117         ieMessage("You are now considered as handling the crisis !")
118         returned to steve
119        }
120 //-------------------------------------------------
121     theClock
122     executed instanceof subfunction
123        oeSetClock("2017:11:26 - 10:45:00"){}
124 //-------------------------------------------------
125     steve
126     executed instanceof subfunction
127        oeValidateAlert("1"){
128         ieMessage('The Alert is now declared as valid !')
129         returned to steve
130        }
131 //-------------------------------------------------
132     tango
133     executed instanceof subfunction
134        oeAlert("witness","2017:11:26","10:20:00","+3524666445000",
135            "49.627095:6.160251","A car crash just happened."){
136          ieSmsSend("+3524666445000","Your alert has been registered. We will handle it and keep you
       informed") returned to tango
137        }
138 //-------------------------------------------------
139     theClock
140     executed instanceof subfunction
141        oeSetClock("2017:11:26 - 12:45:00"){}
142 //-------------------------------------------------
143     steve
144     executed instanceof subfunction
145        oeSetCrisisStatus("1","solved"){
146         ieMessage('The crisis status has been updated !')
147         returned to steve
148        }
149 //-------------------------------------------------
150     steve
151     executed instanceof subfunction
152        oeReportOnCrisis("1","3 victims sent to hospital, 2 cars evacuated and 4 rescue unit
       mobilized"){
153         ieMessage('The crisis comment has been updated !')
154         returned to steve
155        }
156 //-------------------------------------------------
157     steve
158     executed instanceof subfunction
159        oeCloseCrisis("1"){
160         ieMessage('The crisis is now closed !')
161         returned to steve
162        }
163
164    }
165   }
```

```
166 //-------------------------------------------------------
167 //-------------------------------------------------------
168 //-------------------------------------------------------
169  use case instance uciSimpleAndCompletePart01 : suDeployAndRun{
170
171   actors {
172    theCreator : actMsrCreator
173    theClock : actActivator
174    bill : actAdministrator
175    tango : actComCompany
176    steve : actCoordinator
177   }
178   use case steps {
179 //-------------------------------------------------------
180    theCreator
181    executed instanceof subfunction
182       oeCreateSystemAndEnvironment("4"){}
183 //-------------------------------------------------------
184    theClock
185    executed instanceof subfunction
186       oeSetClock("2017:11:24 - 03:20:00"){}
187 //-------------------------------------------------------
188    bill
189    executed instanceof subfunction
190       oeLogin("icrashadmin","7WXC1359"){
191        ieMessage('You are logged ! Welcome ...') returned to bill
192       }
193 //-------------------------------------------------------
194    bill
195    executed instanceof subfunction
196       oeAddCoordinator("1","steve","pwdMessirExcalibur2017"){
197        ieCoordinatorAddedreturned returned to bill
198       }
199 //-------------------------------------------------------
200    bill
201    executed instanceof subfunction
202       oeLogout{
203        ieMessage('You are logged out ! Good Bye ...') returned to bill
204       }
205 //-------------------------------------------------------
206    theClock
207    executed instanceof subfunction
208       oeSetClock("2017:11:26 - 10:15:00"){}
209 //-------------------------------------------------------
210    tango
211    executed instanceof subfunction
212       oeAlert("witness","2017:11:26","10:10:16","+3524666445252",
213           "49.627675:6.159590","3 cars involved in an accident."){
214         ieSmsSend("+3524666445252","Your alert has been registered. We will handle it and keep you
    informed") returned to tango
215       }
216 //-------------------------------------------------------
217    theClock
218    executed instanceof subfunction
219       oeSetClock("2017:11:26 - 10:30:00"){}
220 //-------------------------------------------------------
221    theClock
222    executed instanceof subfunction
223       oeSollicitateCrisisHandling{
224        ieMessage("There are alerts pending since more than the defined delay. Please REACT !")
225        returned to bill
226        ieMessage("There are alerts pending since more than the defined delay. Please REACT !")
227        returned to steve
228       }
229   }
230  }
231 //-------------------------------------------------------
232 //-------------------------------------------------------
233 //-------------------------------------------------------
234  use case instance uciSimpleAndCompletePart02 : suDeployAndRun{
```

```
235   actors {
236    theCreator : actMsrCreator
237    theClock : actActivator
238    bill : actAdministrator
239    tango : actComCompany
240    steve : actCoordinator
241   }
242   use case steps {
243
244 //-----------------------------------------------------
245    steve
246    executed instanceof subfunction
247       oeLogin("steve","pwdMessirExcalibur2017"){
248        ieMessage('You are logged ! Welcome ...') returned to steve
249       }
250 //-----------------------------------------------------
251    steve
252    executed instanceof subfunction
253       oeGetCrisisSet("pending"){
254        ieSendACrisis("crisis with ID 1 details") returned to steve
255       }
256 //-----------------------------------------------------
257    steve
258    executed instanceof subfunction
259       oeSetCrisisHandler("1"){
260        ieSmsSend("+3524666445252","The handling of your alert by our services is in progress !")
261        returned to tango
262        ieMessage("You are now considered as handling the crisis !")
263        returned to steve
264       }
265 //-----------------------------------------------------
266    theClock
267    executed instanceof subfunction
268       oeSetClock("2017:11:26 - 10:45:00"){}
269 //-----------------------------------------------------
270    steve
271    executed instanceof subfunction
272       oeValidateAlert("1"){
273        ieMessage('The Alert is now declared as valid !')
274        returned to steve
275       }
276 //-----------------------------------------------------
277    tango
278    executed instanceof subfunction
279       oeAlert("witness","2017:11:26","10:20:00","+3524666445000",
280           "49.627095:6.160251","A car crash just happened."){
281         ieSmsSend("+3524666445000","Your alert has been registered. We will handle it and keep you
      informed") returned to tango
282       }
283 //-----------------------------------------------------
284    theClock
285    executed instanceof subfunction
286       oeSetClock("2017:11:26 - 12:45:00"){}
287 //-----------------------------------------------------
288    steve
289    executed instanceof subfunction
290       oeSetCrisisStatus("1","solved"){
291        ieMessage('The crisis status has been updated !')
292        returned to steve
293       }
294 //-----------------------------------------------------
295    steve
296    executed instanceof subfunction
297       oeReportOnCrisis("1","3 victims sent to hospital, 2 cars evacuated and 4 rescue unit
      mobilized"){
298        ieMessage('The crisis comment has been updated !')
299        returned to steve
300       }
301 //-----------------------------------------------------
302    steve
```

```
303     executed instanceof subfunction
304        oeCloseCrisis("1"){
305         ieMessage('The crisis is now closed !')
306         returned to steve
307        }
308
309    }
310   }
311  }
312 }
```

Listing C.64: Messir Spec. file usecase-suDeployAndRun.msr.

C.65 File ./src-gen/messir-spec/usecases/usecase-suGlobalCrisisHandling.msr

```
1 package icrash.usecases.suGlobalCrisisHandling {
2  import lu.uni.lassy.messir.libraries.primitives
3  import icrash.environment
4  import icrash.usecases.subfunctions
5  import icrash.usecases.ugSecurelyUseSystem
6  import icrash.usecases.ugManageCrisis
7  import icrash.usecases.ugMonitor
8
9  Use Case Model {
10  use case system summary
11   suGlobalCrisisHandling() {
12   actor actCoordinator[primary,active]
13
14   reuse ugSecurelyUseSystem[1..*]
15   reuse ugMonitor[1..*]
16   reuse ugManageCrisis[1..*]
17
18   step a: actCoordinator
19       executes ugSecurelyUseSystem
20   step b: actCoordinator
21       executes ugMonitor
22   step c: actCoordinator
23       executes ugManageCrisis
24
25   ordering constraint
26    "steps (a) (b) and (c) executions are interleaved
27    (steps (b) and (c) have their protocol constrained by steps of (a))."
28   ordering constraint
29     "steps (a) (b) and (c) can be executed multiple times."
30 }
31 }}
```

Listing C.65: Messir Spec. file usecase-suGlobalCrisisHandling.msr.

C.66 File ./src-gen/messir-spec/usecases/usecase-ugAdministrateTheSystem.msr

```
1 package icrash.usecases.ugAdministrateTheSystem {
2
3  import icrash.environment
4  import icrash.usecases.ugSecurelyUseSystem
5  import icrash.usecases.subfunctions
6
7  Use Case Model {
8
9   use case system usergoal
10   ugAdministrateTheSystem() {
11   actor actAdministrator[primary,active]
12
13   reuse ugSecurelyUseSystem[1..*]
```

```
14    reuse oeAddCoordinator[1..*]
15    reuse oeDeleteCoordinator[0..*]
16    reuse oeRankDownCoordinator[0..*]
17    reuse oeAddPointOfInterest[1..*]
18    reuse oeEditPointOfInterest[0..*]
19    reuse oeDeletePointOfInterest[0..*]
20
21    step a: actAdministrator
22      executes ugSecurelyUseSystem
23    step b: actAdministrator
24      executes oeAddPointOfInterest
25    step c : actAdministrator
26      executes oeEditPointOfInterest
27    step d : actAdministrator
28      executes oeDeletePointOfInterest
29    step e: actAdministrator
30       executes oeAddCoordinator
31    step f: actAdministrator
32       executes oeDeleteCoordinator
33    step g: actAdministrator
34      executes oeRankDownCoordinator
35
36    ordering constraint
37      "steps (a) (b) (c) (d) (e) (f) and (g) executions are interleaved
38      (steps (b) (c) (d) (e) (f) and (g) have their protocol constrained
39       by steps of (a))."
40    ordering constraint
41      "steps (a) (b) (c) (d) (e) (f) (g) can be executed multiple times."
42   }
43  }
44 }
```

Listing C.66: Messir Spec. file usecase-ugAdministrateTheSystem.msr.

C.67 File ./src-gen/messir-spec/usecases/usecase-ugManageCrisis.msr

```
1 package icrash.usecases.ugManageCrisis {
2
3  import icrash.environment
4  import icrash.usecases.subfunctions
5
6  Use Case Model {
7
8   use case system usergoal ugManageCrisis() {
9    actor actCoordinator[primary, active]
10
11    reuse oeValidateAlert[0..*]
12    reuse oeSetCrisisStatus[0..*]
13    reuse oeSetCrisisHandler[0..*]
14    reuse oeReportOnCrisis[0..*]
15    reuse oeCloseCrisis[0..*]
16    reuse oeInvalidateAlert[0..*]
17
18    step a: actCoordinator executes oeValidateAlert
19    step b: actCoordinator executes oeSetCrisisStatus
20    step c: actCoordinator executes oeSetCrisisHandler
21    step d: actCoordinator executes oeReportOnCrisis
22    step f: actCoordinator executes oeCloseCrisis
23    step g: actCoordinator executes oeInvalidateAlert
24
25    ordering constraint "managing a crisis is doing one of the indicated use cases."
26
27   }
28
29  }
30 }
```

Listing C.67: Messir Spec. file usecase-ugManageCrisis.msr.

C.68 File ./src-gen/messir-spec/usecases/usecase-ugMonitor.msr

```
package icrash.usecases.ugMonitor {

 import icrash.environment
 import icrash.usecases.subfunctions

 Use Case Model {
  use case system usergoal ugMonitor() {
   actor icrash.environment.actCoordinator[primary,active]

   reuse oeGetCrisisSet[0..*]
   reuse oeGetAlertsSet[0..*]

   step a: icrash.environment.actCoordinator executes oeGetAlertsSet
   step b: icrash.environment.actCoordinator executes oeGetCrisisSet
  }
 }
}
```

Listing C.68: Messir Spec. file usecase-ugMonitor.msr.

C.69 File ./src-gen/messir-spec/usecases/usecase-ugSecurelyUseSystem.msr

```
package icrash.usecases.ugSecurelyUseSystem {

import icrash.environment
import icrash.usecases.subfunctions

Use Case Model {

use case system usergoal
 ugSecurelyUseSystem() {

 actor actAuthenticated[primary,active]
 actor actAdministrator[secondary]

 reuse oeLogin[1..*]
 reuse oeFillCaptcha[0..*]
 reuse oeResetPassword[0..1]
 reuse oeLogout[1..1]

 step a: actAuthenticated
    executes oeLogin
 step b: actAuthenticated
    executes oeFillCaptcha
 step c: actAuthenticated
  executes oeResetPassword
 step d: actAuthenticated
    executes oeLogout

 ordering constraint
  "step (a) or step (c) is always the first one.
  step (a) must always precede step (d).
  step (b) is only needed when step (a) fails 3 times in a row.
  step (d) is always the last step"

  }
 }
}
```

Listing C.69: Messir Spec. file usecase-ugSecurelyUseSystem.msr.

C.70 File ./src-gen/messir-spec/usecases/usecaseinstance-ugAdministrateTheSystem-uciugAdministrateTheSystem.msr

```
1 package usecases.uciugAdministrateTheSystem {
2  import icrash.usecases.ugAdministrateTheSystem
3  import icrash.usecases.ugSecurelyUseSystem
4   import icrash.concepts.primarytypes.datatypes
5   import icrash.environment
6   import icrash.usecases.ugAdministrateTheSystem
7   import icrash.usecases.subfunctions
8
9  Use Case Model {
10
11   use case instance uciugAdministrateTheSystem : ugAdministrateTheSystem{
12    actors {
13
14     jorge:actAdministrator
15
16    }
17    use case steps {
18      jorge executed usecases.uciugSecurelyUseSystem.uciugSecurelyUseSystem()
19      jorge executed instanceof subfunction oeAddCoordinator("coordID1", "steve", "abcdefg"){
20       ieCoordinatorAdded() returned to jorge
21      }
22      jorge executed instanceof subfunction oeRankDownCoordinator("coordID1"){
23       ieRankEdited() returned to jorge
24      }
25
26       }
27
28        }
29
30   }
31 }
```

Listing C.70: Messir Spec. file usecaseinstance-ugAdministrateTheSystem-uciugAdministrateTheSystem.msr.

C.71 File ./src-gen/messir-spec/usecases/usecaseinstance-ugAdministrateTheSystem-uciugAdministrateTheSystemPointOfIntere

```
1 package usecases.uciugAdministrateTheSystemPointOfInterest {
2  import icrash.usecases.ugAdministrateTheSystem
3  import icrash.usecases.subfunctions
4
5  Use Case Model {
6
7   use case instance uciugAdministrateTheSystemPointOfInterest : ugAdministrateTheSystem{
8    actors {
9     bill:icrash.environment.actAdministrator
10   }
11   use case steps {
12 //------------------------------
13    bill
14    executed instanceof subfunction
15       oeLogin("icrashadmin","7WXC1359"){
16        ieMessage('You are logged ! Welcome ...') returned to bill
17       }
18 //------------------------------
19    bill
20    executed instanceof subfunction
21       oeAddPointOfInterest("Police station","55.65448:1.548648","Police station: open from 8:30
      until 17:30"){
22        iePointOfInterestAdded returned to bill
23       }
24 //------------------------------
```

```
25     bill
26     executed instanceof subfunction
27        oeEditPointOfInterest("1","Police station : open from 9:00 until 17:00"){
28         iePointOfInterestEdited returned to bill
29        }
30 //-----------------------------
31     bill
32     executed instanceof subfunction
33        oeDeletePointOfInterest("1"){
34         iePointOfInterestDeleted returned to bill
35        }
36 //-----------------------------
37     bill
38     executed instanceof subfunction
39        oeLogout{
40         ieMessage('You are logged out ! Good Bye ...') returned to bill
41        }
42    }
43
44    }
45   }
46 }
```

Listing C.71: Messir Spec. file usecaseinstance-ugAdministrateTheSystem-uciugAdministrateTheSystemPointOfInterest.msr.

C.72 File ./src-gen/messir-spec/usecases/usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystem.msr

```
1 package usecases.uciugSecurelyUseSystem {
2  import icrash.usecases.ugSecurelyUseSystem
3   import icrash.usecases.ugSecurelyUseSystem
4   import icrash.concepts.primarytypes.datatypes
5   import icrash.environment
6   import icrash.usecases.suGlobalCrisisHandling
7   import icrash.usecases.ugAdministrateTheSystem
8   import icrash.usecases.subfunctions
9
10  Use Case Model {
11
12 //-----------------------------
13   use case instance uciugSecurelyUseSystem : ugSecurelyUseSystem {
14    actors {
15     bill:actAuthenticated
16    }
17    use case steps {
18 //-----------------------------
19     bill
20     executed instanceof subfunction
21        oeLogin("icrashadmin","7WXC1359"){
22         ieMessage('You are logged ! Welcome ...') returned to bill
23        }
24 //-----------------------------
25     bill
26     executed instanceof subfunction
27        oeLogout{
28         ieMessage('You are logged out ! Good Bye ...') returned to bill
29        }
30    }
31   }
32  }
33 }
```

Listing C.72: Messir Spec. file usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystem.msr.

C.73 File ./src-gen/messir-spec/usecases/usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemFailLogin.msr

```
package usecases.uciugSecurelyUseSystemFailLogin {
 import icrash.usecases.ugSecurelyUseSystem
  import icrash.concepts.primarytypes.datatypes
  import icrash.environment
  import icrash.usecases.subfunctions

 Use Case Model {

  use case instance uciugSecurelyUseSystemFailLogin : ugSecurelyUseSystem{
   actors {
    Theuser:actAuthenticated
   }
   use case steps {
//------------------------------
   Theuser
   executed instanceof subfunction
     oeLogin("failed","login") {
   ieMessage('Your password is not correct. Wrong attempts : 1') returned to Theuser
      }
//------------------------------
   Theuser
   executed instanceof subfunction
     oeLogin("failed","login") {
   ieMessage('Your password is not correct. Wrong attempts : 2') returned to Theuser
      }
//------------------------------
   Theuser
   executed instanceof subfunction
     oeLogin("failed","login") {
   ieMessage('Your password is not correct. Wrong attempts : 3.') returned to Theuser
   ieMessage('You need to fill out a captcha to be able to login now. You can also do a password
       reset.') returned to Theuser
      }
//------------------------------
   Theuser
   executed instanceof subfunction
     oeFillCaptcha("wrong captcha") {
   ieMessage('Your captcha is not correct. Try again.') returned to Theuser
      }
//------------------------------
   Theuser
   executed instanceof subfunction
     oeFillCaptcha("successful captcha") {
   ieMessage('Your captcha is correct. You may try to login again.') returned to Theuser
      }
//------------------------------
    Theuser
    executed instanceof subfunction
       oeLogin("Theuser","7WXC1359"){
        ieMessage('You are logged ! Welcome ...') returned to Theuser
       }
//------------------------------
    Theuser
    executed instanceof subfunction
       oeLogout{
        ieMessage('You are logged out ! Good Bye ...') returned to Theuser
       }
   }
  }
 }
}
```

Listing C.73: Messir Spec. file usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemFailLogin.msr.

C.74 File ./src-gen/messir-spec/usecases/usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemFailLoginResetPassword.ms

```
1 package usecases.uciugSecurelyUseSystemFailLoginResetPassword {
2  import icrash.usecases.ugSecurelyUseSystem
3   import icrash.concepts.primarytypes.datatypes
4   import icrash.environment
5   import icrash.usecases.subfunctions
6
7  Use Case Model {
8
9   use case instance uciugSecurelyUseSystemFailLoginResetPassword : ugSecurelyUseSystem{
10   actors {
11    Theuser:actAuthenticated
12    Theadmin:actAdministrator
13   }
14   use case steps {
15 //------------------------------
16   Theuser
17   executed instanceof subfunction
18     oeLogin("failed","login") {
19   ieMessage('Your password is not correct. Wrong attempts : 1') returned to Theuser
20      }
21 //------------------------------
22   Theuser
23   executed instanceof subfunction
24     oeLogin("failed","login") {
25   ieMessage('Your password is not correct. Wrong attempts : 2') returned to Theuser
26      }
27 //------------------------------
28   Theuser
29   executed instanceof subfunction
30     oeLogin("failed","login") {
31   ieMessage('Your password is not correct. Wrong attempts : 3.') returned to Theuser
32   ieMessage('You need to fill out a captcha to be able to login now. You can also do a password
         reset.') returned to Theuser
33      }
34 //------------------------------
35   Theuser
36   executed instanceof subfunction
37     oeResetPassword("Theuser") {
38   ieMessage('Your password has been reset. Check your mails to be able to reconnect') returned to
        Theuser
39   ieMessage2('The password of "Theuser" has been reset ') returned to Theadmin
40      }
41 //------------------------------
42    Theuser
43    executed instanceof subfunction
44       oeLogin("Theuser","NewPassword"){
45        ieMessage('You are logged ! Welcome ...') returned to Theuser
46       }
47 //------------------------------
48    Theuser
49    executed instanceof subfunction
50       oeLogout{
51        ieMessage('You are logged out ! Good Bye ...') returned to Theuser
52       }
53   }
54  }
55 }
56 }
```

Listing C.74: Messir Spec. file usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemFailLoginResetPassword.msr.

C.75 File ./src-gen/messir-spec/usecases/usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemWithDirectResetPasswo

```
1 package usecases.uciugSecurelyUseSystemithDirectResetPassword {
2  import icrash.usecases.ugSecurelyUseSystem
3   import icrash.concepts.primarytypes.datatypes
4   import icrash.environment
5   import icrash.usecases.suGlobalCrisisHandling
6   import icrash.usecases.ugAdministrateTheSystem
7   import icrash.usecases.subfunctions
8
9  Use Case Model {
10
11   use case instance uciugSecurelyUseSystemWithDirectResetPassword : ugSecurelyUseSystem{
12    actors {
13     Theuser:actAuthenticated
14     Theadmin:actAdministrator
15
16    }
17    use case steps {
18 //------------------------------
19     Theuser
20     executed instanceof subfunction
21        oeResetPassword("Theuser"){
22         ieMessage('Your password has been reset. Check your mails to be able to reconnect') returned
       to Theuser
23         ieMessage2('The password of the user : "Theuser" has been reset.') returned to Theadmin
24        }
25
26 //------------------------------
27     Theuser
28     executed instanceof subfunction
29        oeLogin("Theuser","7WXC1359"){
30         ieMessage('You are logged ! Welcome ...') returned to Theuser
31        }
32 //------------------------------
33     Theuser
34     executed instanceof subfunction
35        oeLogout{
36         ieMessage('You are logged out ! Good Bye ...') returned to Theuser
37        }
38
39      }
40   }
41  }
42 }
```

Listing C.75: Messir Spec. file usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystemWithDirectResetPassword.msr.

Appendix D

Listing of the Prolog Files Referenced in the Operation Model Specification

D.1 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactActiva oeSetClock.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------
msrop(outactActivator,
   oeSetClock,
   [preProtocol,Self,
    AcurrentClock
    ],
   []):-
/* Pre Protocol:*/
/* PreP01 */
 msrVar(ctState,TheSystem),
 msrVar(ptBoolean,AvpStarted),

 msrNav([Self],[rnActor,rnSystem],[TheSystem]),

 msrNav([Self],[rnActor,rnSystem,vpStarted],[AvpStarted]),
 AvpStarted = [ptBoolean,true],

 msrNav([TheSystem],
    [clock,lt,[AcurrentClock]],
    [[ptBoolean,true]])
 .

msrop(outactActivator,
   oeSetClock,
   [preFunctional,Self,
    AcurrentClock
    ],
   []):-
/* Pre Functional:*/
/* PreF01 */
true.

msrop(outactActivator,
   oeSetClock,
   [post,Self,
    AcurrentClock
    ],
    []):-

```

```
44 msrVar(ctState,TheSystem),
45
46/* Post Functional:*/
47
48 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
49
50/* PostF01 */
51 msrNav([TheSystem],
52     [msmAtPost,clock],
53     [AcurrentClock]),
54
55 /* Post Protocol:*/
56/* PostP01 */
57 true
58 .
```

Listing D.1: Prolog file outactActivator-oeSetClock.pl.

D.2 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAc oeSollicitateCrisisHandling.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%------------------------------------------------------
6
7msrop(outactActivator,
8   oeSollicitateCrisisHandling,
9   [preProtocol,Self
10    ],
11   []):-
12/* Pre Protocol:*/
13 msrVar(ctState,TheSystem),
14 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
15
16 msrVarCol(ctCrisis,_,ColctCrisisToHandle),
17
18/* PreP01 */
19 msrNav([TheSystem],
20    [vpStarted],
21    [[ptBoolean,true]]),
22
23/* PreP02 */
24 msrNav([TheSystem],
25    [rnctCrisis,msrSelect,
26     handlingDelayPassed,[]
27    ],
28    ColctCrisisToHandle),
29
30 msrNav(ColctCrisisToHandle,
31    [msrSize,geq,[[ptInteger,1]]],
32    [[ptBoolean,true]])
33.
34
35msrop(outactActivator,
36   oeSollicitateCrisisHandling,
37   [preFunctional,Self
38    ],
39   []):-
40/* Pre Functional:*/
41/* PreF01 */
42true.
43
44msrop(outactActivator,
45   oeSollicitateCrisisHandling,
46   [post,Self
47    ],
```

```
48     []):-
49
50  msrVar(ctState,TheSystem),
51  msrVar(dtComment,AMessageForCrisisHandlers),
52  msrVar(dtDateAndTime, TheClock),
53  msrVarCol(ctCrisis,_,ColctCrisisToAllocateIfPossible),
54
55 /* Post Functional:*/
56  msrNav([Self],[rnActor,rnSystem],[TheSystem]),
57
58  /* PostF01 */
59  msrNav([TheSystem],
60     [rnctCrisis,msrSelect,
61      maxHandlingDelayPassed,[]
62     ],
63     ColctCrisisToAllocateIfPossible),
64
65 msrNav(ColctCrisisToAllocateIfPossible,
66     [msrForAll,isAllocatedIfPossible,[]],
67     [[ptBoolean,true]]),
68
69  /* PostF02 */
70  msrNav([TheSystem],
71     [rnctCrisis,msrSelect,
72      handlingDelayPassed,[]
73     ],
74     ColctCrisisToHandle),
75
76  msrNav(ColctCrisisToHandle,
77      [msrColSubtract,[ColctCrisisToAllocateIfPossible]
78      ],
79      ColctCrisisToRemind),
80
81  (msrNav(ColctCrisisToRemind,
82      [msrSize,geq,[[ptInteger,1]]],
83      [[ptBoolean,true]])
84   -> (msrNav([AMessageForCrisisHandlers],
85       [value],
86       [[ptString,'There are alerts pending since more than the defined delay. Please REACT !']]),
87
88     msrNav([TheSystem],
89        [rnactAdministrator,rnInterfaceIN,
90         ieMessage,[AMessageForCrisisHandlers]
91        ],
92        [[ptBoolean,true]]),
93
94     msrNav([TheSystem],
95        [rnactCoordinator,msrForAll,rnInterfaceIN,
96         ieMessage,[AMessageForCrisisHandlers]
97        ],
98        [[ptBoolean,true]])
99    )
100 ; true
101 ),
102
103 /* Post Protocol:*/
104 /* PostP01 */
105 msrNav([TheSystem],
106    [clock],
107    [TheClock]),
108
109 msrNav([TheSystem],
110     [msmAtPost,vpLastReminder],
111     [TheClock])
112 .
```

Listing D.2: Prolog file outactActivator-oeSollicitateCrisisHandling.pl.

D.3 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAdoeAddCoordinator.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------
msrop(outactAdministrator,
   oeAddCoordinator,
   [preProtocol,Self,
    AdtCoordinatorID,
    AdtLogin,
    AdtPassword
    ],
   []):-
/* Pre Protocol:*/
 msrVar(ctState,TheSystem),
 msrVar(actAdministrator,TheActor),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),

/* PreP01 */
 msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]]),

/* PreP02 */
 msrNav([TheActor],
    [rnctAuthenticated,vpIsLogged],
    [[ptBoolean,true]])

 .

msrop(outactAdministrator,
   oeAddCoordinator,
   [preFunctional,Self,
    AdtCoordinatorID,
    AdtLogin,
    AdtPassword
    ],
   []):-
/* Pre Functional:*/
 msrVar(ctState,TheSystem),
 msrVar(actAdministrator,TheActor),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),
/* PreF01 */
 msrNav([TheSystem],
     [rnctCoordinator,
     msrSelect,id,eq,[AdtCoordinatorID]],
     ColctCoordinators),
 msrNav(ColctCoordinators,
     [msrIsEmpty],
     [[ptBoolean,true]])
 .

msrop(outactAdministrator,
   oeAddCoordinator,
   [post,Self,
    AdtCoordinatorID,
    AdtLogin,
    AdtPassword
    ],
    []):-

/* Post Functional:*/
 msrVar(ctState,TheSystem),
 msrVar(actAdministrator,TheActor),
```

```
67 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
68 msrNav([Self],[rnActor],[TheActor]),
69
70 msrVar(actCoordinator,TheactCoordinator),
71 msrVar(ctCoordinator,ThectCoordinator),
72
73/* PostF01 */
74 msrNav([TheactCoordinator],
75     [init,[]],
76     [[ptBoolean,true]]),
77
78/* PostF02 */
79 msrNav([ThectCoordinator],
80     [init,[AdtCoordinatorID,AdtLogin,AdtPassword]],
81     [[ptBoolean,true]]),
82
83/* PostF03 */
84 msrNav([TheactCoordinator],
85     [msmAtPost,rnctCoordinator],
86     [ThectCoordinator]),
87
88/* PostF04 */
89  msrNav([ThectCoordinator],
90     [msmAtPost,rnactAuthenticated],
91     [TheactCoordinator]),
92
93/* PostF05 */
94  msrNav([TheActor],
95     [rnInterfaceIN,
96      ieCoordinatorAdded,[]],
97     [[ptBoolean,true]]),
98
99 /* Post Protocol:*/
100/* PostP01 */
101 true
102 .
```

Listing D.3: Prolog file outactAdministrator-oeAddCoordinator.pl.

D.4 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAdmin oeDeleteCoordinator.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactAdministrator,
7   oeDeleteCoordinator,
8   [preProtocol,Self,
9    AdtCoordinatorID
10   ],
11  []):-
12/* Pre Protocol:*/
13 msrVar(ctState,TheSystem),
14 msrVar(actAdministrator,TheActor),
15 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
16 msrNav([Self],[rnActor],[TheActor]),
17
18/* PreP01 */
19 msrNav([TheSystem],
20    [vpStarted],
21    [[ptBoolean,true]]),
22
23 msrNav([TheActor],
24    [rnctAuthenticated,vpIsLogged],
25    [[ptBoolean,true]])
26.
```

```
27
28msrop(outactAdministrator,
29   oeDeleteCoordinator,
30   [preFunctional,Self,
31    AdtCoordinatorID
32    ],
33   []):-
34/* Pre Functional:*/
35 msrVar(ctState,TheSystem),
36 msrVar(actAdministrator,TheActor),
37 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
38 msrNav([Self],[rnActor],[TheActor]),
39
40/* PreF01 */
41 msrNav([TheSystem],
42     [rnctCoordinator,
43     msrSelect,id,eq,[AdtCoordinatorID]],
44     ColctCoordinators),
45
46 msrNav(ColctCoordinators,
47     [msrSize,eq,[[ptInteger,1]]],
48     [[ptBoolean,true]]).
49
50msrop(outactAdministrator,
51   oeDeleteCoordinator,
52   [post,Self,
53    AdtCoordinatorID
54    ],
55    []):-
56
57/* Post Functional:*/
58 msrVar(ctState,TheSystem),
59 msrVar(actAdministrator,TheActor),
60 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
61 msrNav([Self],[rnActor],[TheActor]),
62
63/* PostF01 */
64 msrNav([TheSystem],
65    [rnctCoordinator,
66     msrSelect,id,eq,[AdtCoordinatorID]],
67    [ThectCoordinator]),
68
69 msrNav([ThectCoordinator],
70     [rnactCoordinator,msrForAll,msrIsKilled],
71     [[ptBoolean,true]]),
72
73 msrNav([ThectCoordinator],
74     [msrIsKilled],
75     [[ptBoolean,true]]),
76
77 /* PostF02 */
78 msrNav([TheActor],
79     [rnInterfaceIN,
80     ieCoordinatorDeleted,[]
81     ],
82     [[ptBoolean,true]]),
83
84 /* Post Protocol:*/
85/* PostP01 */
86 true
87 .
```

Listing D.4: Prolog file outactAdministrator-oeDeleteCoordinator.pl.

D.5 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAu oeLogin.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------
msrop(outactAuthenticated,
   oeLogin,
   [preProtocol,Self,
    AdtLogin,
    AdtPassword
    ],
   []):-
/* Pre Protocol:*/
 msrVar(ctState,TheSystem),
 msrVar(actAuthenticated,TheactAuthenticated),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheactAuthenticated]),

 /* PreP01 */
 msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]]),

 msrNav([TheactAuthenticated],
    [rnctAuthenticated,vpIsLogged],
    [[ptBoolean,false]])
 .

msrop(outactAuthenticated,
   oeLogin,
   [preFunctional,Self,
    AdtLogin,
    AdtPassword
    ],
   []):-
/* Pre Functional:*/
/* PreF01 */
true
.

msrop(outactAuthenticated,
   oeLogin,
   [post,Self,
    AdtLogin,
    AdtPassword
    ],
    []):-

 msrVar(ctState,TheSystem),
 msrVar(actAuthenticated,TheactAuthenticated),

 msrVar(ptString,AptStringMessageForTheactAuthenticated),
 msrVar(ptString,AptStringMessageForTheactAdministrator),

/* Post Functional:*/

 msrNav([Self],[rnActor],[TheactAuthenticated]),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),

/* PostF01 */

 ( (msrNav([TheactAuthenticated],
      [rnctAuthenticated,pwd],
      [AdtPassword]),
   msrNav([TheactAuthenticated],
      [rnctAuthenticated,login],
      [AdtLogin])
  )
 -> ( msrNav([AptStringMessageForTheactAuthenticated],
       [eq,[[ptString,'You are logged ! Welcome ...']]],
       [[ptBoolean,true]]),
```

```
72    msrNav([TheactAuthenticated],
73       [rnInterfaceIN,
74        ieMessage,[AptStringMessageForTheactAuthenticated]],
75       [[ptBoolean,true]])
76   )
77 ; ( msrNav([AptStringMessageForTheactAuthenticated],
78       [eq,[[ptString,'Wrong identification information ! Please try again ...']]],
79       [[ptBoolean,true]]),
80    msrNav([TheactAuthenticated],
81       [rnInterfaceIN,
82        ieMessage,[AptStringMessageForTheactAuthenticated]],
83       [[ptBoolean,true]]),
84
85    msrNav([AptStringMessageForTheactAdministrator],
86       [eq,[[ptString,'Intrusion tentative !']]],
87       [[ptBoolean,true]]),
88    msrNav([TheSystem],
89       [rnactAdministrator,rnInterfaceIN,
90        ieMessage,[AptStringMessageForTheactAdministrator]],
91       [[ptBoolean,true]])
92    )
93 ),
94
95 /* Post Protocol:*/
96/* PostP01 */
97 ( (msrNav([TheactAuthenticated],
98      [rnctAuthenticated,pwd],
99      [AdtPassword]),
100   msrNav([TheactAuthenticated],
101      [rnctAuthenticated,login],
102      [AdtLogin])
103  )
104 -> (msrNav([TheactAuthenticated],
105       [rnctAuthenticated,msmAtPost,vpIsLogged],
106       [[ptBoolean,true]])
107    )
108 ; true
109 )
110 .
```

Listing D.5: Prolog file outactAuthenticated-oeLogin.pl.

D.6 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAu oeLogout.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactAuthenticated,
7   oeLogout,
8   [preProtocol,Self
9    ],
10   []):-
11/* Pre Protocol:*/
12 msrVar(ctState,TheSystem),
13 msrVar(actAuthenticated,TheActor),
14 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
15 msrNav([Self],[rnActor],[TheActor]),
16
17/* PreP01 */
18 msrNav([TheSystem],
19    [vpStarted],
20    [[ptBoolean,true]]),
21
22 msrNav([TheActor],
23    [rnctAuthenticated,vpIsLogged],
```

```
24     [[ptBoolean,true]])
25  .
26
27msrop(outactAuthenticated,
28    oeLogout,
29    [preFunctional,Self
30     ],
31    []):-
32/* Pre Functional:*/
33/* PreF01 */
34true
35.
36
37msrop(outactAuthenticated,
38    oeLogout,
39    [post,Self
40     ],
41     []):-
42
43 msrVar(ctState,TheSystem),
44 msrVar(actAuthenticated,TheactAuthenticated),
45
46 msrVar(ptString,AptStringMessageForTheactAuthenticated),
47
48/* Post Functional:*/
49 msrNav([Self],[rnActor],[TheactAuthenticated]),
50 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
51
52/* PostF01 */
53 msrNav([AptStringMessageForTheactAuthenticated],
54     [eq,[[ptString,'You are logged out ! Good Bye ...']]],
55     [[ptBoolean,true]]),
56 msrNav([TheactAuthenticated],
57     [rnInterfaceIN,
58      ieMessage,[AptStringMessageForTheactAuthenticated]],
59     [[ptBoolean,true]]),
60
61 /* Post Protocol:*/
62/* PostP01 */
63msrNav([TheactAuthenticated],
64     [rnctAuthenticated,msmAtPost,vpIsLogged],
65     [[ptBoolean,false]])
66.
```

Listing D.6: Prolog file outactAuthenticated-oeLogout.pl.

D.7 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactComC oeAlert.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------
6nico(A):-
7 trace,
8 write('here'),
9 write('\n').
10
11msrop(outactComCompany,
12    oeAlert,
13    [preProtocol,Self,
14     AetHumanKind,
15     AdtDate,
16     AdtTime,
17     AdtPhoneNumber,
18     AdtGPSLocation,
19     AdtComment
```

```
20     ],
21    []):-
22/* Pre Protocol:*/
23 msrVar(ctState,TheSystem),
24 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
25/* PreP01 */
26 msrNav([TheSystem],
27     [vpStarted],
28     [[ptBoolean,true]])
29 .
30
31msrop(outactComCompany,
32    oeAlert,
33    [preFunctional,Self,
34     AetHumanKind,
35     AdtDate,
36     AdtTime,
37     AdtPhoneNumber,
38     AdtGPSLocation,
39     AdtComment
40     ],
41    []):-
42/* Pre Functional:*/
43/* PreF01 */
44 msrVar(ctState,TheSystem),
45 msrNav([Self],
46      [msmAtPre,rnActor,rnSystem],
47      [TheSystem]),
48
49 ( msrNav([TheSystem],[clock,date,gt,[AdtDate]],[[ptBoolean,true]])
50  ; (msrNav([TheSystem],[clock,date,eq,[AdtDate]],[[ptBoolean,true]])
51   , msrNav([TheSystem],[clock,time,gt,[AdtTime]],[[ptBoolean,true]])
52   )
53 )
54.
55
56msrop(outactComCompany,
57    oeAlert,
58    [post,Self,
59     AetHumanKind,
60     AdtDate,
61     AdtTime,
62     AdtPhoneNumber,
63     AdtGPSLocation,
64     AdtComment
65     ],
66     []):-
67
68 msrVar(ctState,TheSystem),
69 msrVar(ctHuman,ActHuman),
70 msrVar(actComCompany,TheactComCompany),
71 msrVar(ctAlert,ActAlert),
72 msrVar(dtDateAndTime,AAlertInstant),
73 msrVar(etAlertStatus,AetAlertStatus),
74% msrVar(ctAlert,ActAlertNearBy),
75 msrVar(ctCrisis,ActCrisis),
76 msrVar(dtCrisisID,AdtCrisisID),
77% msrVar(etCrisisType,AetCrisisType),
78 msrVar(etCrisisStatus,AetCrisisStatus),
79 msrVar(dtDateAndTime,ACrisisInstant),
80 msrVar(dtComment,ACrisisdtComment),
81% msrVar(ptString,AptStringMessage),
82 msrVar(dtSMS,AdtSMS),
83 msrVar(dtAlertID,AdtAlertID),
84
85% msrVar(ptInteger,TheNextptIntegerValue),
86% msrVar(ptInteger,UpdatedNextptIntegerValue),
87% msrVar(inactComCompany,TheComCompanyIN),
88% msrVar(dtComment,TheCommentStored),
89% msrVar(dtString,TheCommentStoreddtString),
```

```
90
91/* Post Functional:*/
92
93 msrNav([Self],[rnActor],[TheactComCompany]),
94 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
95
96/* PostF01 */
97 msrNav([TheSystem],
98     [nextValueForAlertID],
99     [PrenextValueForAlertID]),
100 msrNav([PrenextValueForAlertID],
101     [add,[[dtInteger,[[value,[ptInteger,1]]],[]]]],
102     [PostnextValueForAlertID]),
103 msrNav([TheSystem],
104     [msmAtPost,nextValueForAlertID],
105     [PostnextValueForAlertID]),
106
107 /* PostF02 */
108 msrNav([AAlertInstant],[date],[AdtDate]),
109 msrNav([AAlertInstant],[time],[AdtTime]),
110
111 msrNav([AetAlertStatus],
112     [],
113     [[etAlertStatus,pending]]),
114
115 msrNav([TheSystem],
116     [nextValueForAlertID,
117     todtString,[],eq,[AdtAlertID]],
118     [[ptBoolean,true]]),
119
120 msrNav([ActAlert],
121     [init,[AdtAlertID,
122        AetAlertStatus,
123        AdtGPSLocation,
124        AAlertInstant,
125        AdtComment]],
126     [[ptBoolean,true]]),
127
128 /* PostF03 */
129 msrNav([TheSystem],
130     [rnctAlert,
131      msrSelect,location,isNearTo,[AdtGPSLocation]],
132      ColctAlertsNearBy),
133
134 ( (msrNav(ColctAlertsNearBy,
135      [msrIsEmpty],
136      [[ptBoolean,true]])
137   )
138 -> (
139    msrNav([TheSystem],
140       [nextValueForCrisisID],
141       [PrenextValueForCrisisID]),
142    msrNav([PrenextValueForCrisisID],
143       [add,[[dtInteger,[[value,[ptInteger,1]]],[]]]],
144       [PostnextValueForCrisisID]),
145    msrNav([TheSystem],
146       [msmAtPost,nextValueForCrisisID],
147       [PostnextValueForCrisisID]),
148
149   msrNav([TheSystem],
150       [nextValueForCrisisID,
151       todtString,[],eq,[AdtCrisisID]],
152       [[ptBoolean,true]]),
153
154   msrNav([AdtCrisisType],[],[[etCrisisType,small]]),
155   msrNav([AetCrisisStatus],[],[[etCrisisStatus,pending]]),
156   msrNav([ACrisisInstant],[],[AAlertInstant]),
157   msrNav([ACrisisdtComment],
158       [value],
159       [[ptString,'no reporting yet defined']]),
```

```
160   msrNav([ActCrisis],[init,[AdtCrisisID,
161                 AdtCrisisType,
162                 AetCrisisStatus,
163                 AdtGPSLocation,
164                 ACrisisInstant,
165                 ACrisisdtComment]],
166       [[ptBoolean,true]])
167
168    )
169 ; (
170   msrNav(ColctAlertsNearBy,
171      [rnTheCrisis,msrAny,msrTrue],
172      [ActCrisis])
173  )
174 ),
175
176 /* PostF04 */
177
178 msrNav([ActAlert],
179     [msmAtPost,rnTheCrisis],
180     [ActCrisis]),
181
182/* PostF05 */
183
184 msrNav([TheSystem],
185    [rnctHuman,
186     msrSelect,id,eq,[AdtPhoneNumber]],
187    HumanCol1),
188
189 msrNav(HumanCol1,
190    [msrSelect,kind,etEq,[AetHumanKind]],
191    HumanCol2),
192
193 (msrNav(HumanCol2,[msrIsEmpty],[[ptBoolean,true]])
194 -> (msrNav([ActHuman],
195       [init,[AdtPhoneNumber,AetHumanKind]],
196       [[ptBoolean,true]]),
197   msrNav([ActHuman],
198       [msmAtPost,rnactComCompany],
199       [TheactComCompany])
200   )
201 ; msrNav(HumanCol2,
202     [msrAny],
203     [ActHuman])
204 ),
205
206msrNav([ActHuman],
207    [rnSignaled,msrIncluding,[ActAlert]],
208    ColAlerts),
209
210msrNav([ActHuman],
211    [msmAtPost,rnSignaled],
212    ColAlerts),
213
214/* PostF06 */
215msrNav([AdtSMS],
216    [value],
217    [[ptString,'Your alert has been registered. We will handle it and keep you informed']]),
218msrNav([TheactComCompany],
219    [rnInterfaceIN,
220    ieSmsSend,[AdtPhoneNumber,
221          AdtSMS]],[[ptBoolean,true]]),
222
223/*
224
225 */
226
227 /* Post Protocol:*/
228/* PostP01 */
229 true
```

```
230 .
```

Listing D.7: Prolog file outactComCompany-oeAlert.pl.

D.8 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCoordi oeGetAlertsSet.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactCoordinator,
7   oeGetAlertsSet,
8   [preProtocol,Self,
9    AetAlertStatus
10   ],
11   []):-
12/* Pre Protocol:*/
13 msrVar(ctState,TheSystem),
14 msrVar(actCoordinator,TheActor),
15 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
16 msrNav([Self],[rnActor],[TheActor]),
17
18/* PreP01 */
19 msrNav([TheSystem],
20    [vpStarted],
21    [[ptBoolean,true]]),
22
23 msrNav([TheActor],
24    [rnctAuthenticated,vpIsLogged],
25    [[ptBoolean,true]])
26.
27
28msrop(outactCoordinator,
29   oeGetAlertsSet,
30   [preFunctional,Self,
31    AetAlertStatus
32    ],
33   []):-
34/* Pre Functional:*/
35/* PreF01 */
36true
37 .
38
39msrop(outactCoordinator,
40   oeGetAlertsSet,
41   [post,Self,
42    AetAlertStatus
43    ],
44    []):-
45
46/* Post Functional:*/
47 msrVar(ctState,TheSystem),
48 msrVar(actCoordinator,TheActor),
49 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
50 msrNav([Self],[rnActor],[TheActor]),
51
52/* PostF01 */
53 msrNav([TheSystem],
54    [rnctAlert,
55     msrSelect,
56     status,etEq,[AetAlertStatus]],
57    ColAlertSet),
58
59 msrNav(ColAlertSet,
60     [msrForAll,isSentToCoordinator,[TheActor]],
61     [[ptBoolean,true]]),
```

```
62
63 /* Post Protocol:*/
64/* PostP01 */
65 true
66 .
```

Listing D.8: Prolog file outactCoordinator-oeGetAlertsSet.pl.

D.9 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCo oeInvalidateAlert.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactCoordinator,
7   oeInvalidateAlert,
8   [preProtocol,Self,
9    AdtAlertID
10    ],
11   []):-
12/* Pre Protocol:*/
13 msrVar(ctState,TheSystem),
14 msrVar(actCoordinator,TheActor),
15 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
16 msrNav([Self],[rnActor],[TheActor]),
17
18/* PreP01 */
19 msrNav([TheSystem],
20    [vpStarted],
21    [[ptBoolean,true]]),
22
23/* PreP02 */
24 msrNav([TheActor],
25    [rnctAuthenticated,vpIsLogged],
26    [[ptBoolean,true]])
27.
28
29msrop(outactCoordinator,
30   oeInvalidateAlert,
31   [preFunctional,Self,
32    AdtAlertID
33    ],
34   []):-
35/* Pre Functional:*/
36 msrVar(ctState,TheSystem),
37 msrVar(actCoordinator,TheActor),
38
39 msrVar(dtAlertID,AdtAlertID),
40
41 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
42 msrNav([Self],[rnActor],[TheActor]),
43
44/* PreF01 */
45 msrNav([TheSystem],
46    [rnctAlert,
47     msrSelect,
48     id,eq,[AdtAlertID]
49    ],
50    ColAlert),
51
52 msrNav(ColAlert,
53     [msrSize,eq,[[ptInteger,1]]],
54     [[ptBoolean,true]])
55 .
56
57msrop(outactCoordinator,
```

```
58    oeInvalidateAlert,
59    [post,Self,
60     AdtAlertID
61     ],
62     []):-
63
64/* Post Functional:*/
65 msrVar(ctState,TheSystem),
66 msrVar(actCoordinator,TheActor),
67
68 msrVar(ctAlert,TheAlert),
69 msrVar(dtAlertID,AdtAlertID),
70
71 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
72 msrNav([Self],[rnActor],[TheActor]),
73
74/* PostF01 */
75 msrNav([TheSystem],
76    [rnctAlert,
77     msrSelect,
78     id,eq,[AdtAlertID]],
79    [TheAlert]),
80
81 msrNav([TheAlert],
82     [msmAtPost,status],
83     [[etAlertStatus,invalid]]),
84
85/* PostF02 */
86 msrNav([TheActor],
87     [rnInterfaceIN,
88     ieMessage,[[ptString,'The alert is now declared as invalid !']]
89     ],
90     [[ptBoolean,true]]),
91
92/* Post Protocol:*/
93/* PostP01 */
94 true
95 .
```

Listing D.9: Prolog file outactCoordinator-oeInvalidateAlert.pl.

D.10 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCoor oeReportOnCrisis.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------
6msrop(outactCoordinator,
7   oeReportOnCrisis,
8   [preProtocol,Self,
9    AdtCrisisID,
10   AdtComment
11   ],
12   []):-
13/* Pre Protocol:*/
14 msrVar(ctState,TheSystem),
15 msrVar(actCoordinator,TheActor),
16 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
17 msrNav([Self],[rnActor],[TheActor]),
18
19/* PreP01 */
20 msrNav([TheSystem],
21    [vpStarted],
22    [[ptBoolean,true]]),
23
24 msrNav([TheActor],
```

```
    [rnctAuthenticated,vpIsLogged],
    [[ptBoolean,true]])
.

msrop(outactCoordinator,
   oeReportOnCrisis,
   [preFunctional,Self,
    AdtCrisisID,
    AdtComment
    ],
   []):-
/* Pre Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),

msrVar(dtCrisisID,AdtCrisisID),

msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PreF01 */
msrNav([TheSystem],
    [rnctCrisis,
     msrSelect,
     id,eq,[AdtCrisisID]
    ],
    ColCrisis),

msrNav(ColCrisis,
     [msrSize,eq,[[ptInteger,1]]],
     [[ptBoolean,true]])
.

msrop(outactCoordinator,
   oeReportOnCrisis,
   [post,Self,
    AdtCrisisID,
    AdtComment
    ],
    []):-

/* Post Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),

msrVar(ctCrisis,TheCrisis),
msrVar(dtCrisisID,AdtCrisisID),
msrVar(dtComment,AdtComment),

msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
msrNav([TheSystem],
    [rnctCrisis,
     msrSelect,
     id,eq,[AdtCrisisID]],
    [TheCrisis]),

msrNav([TheCrisis],
     [msmAtPost,comment],
     [AdtComment]),

msrNav([TheActor],
     [rnInterfaceIN,
     ieMessage,[[ptString,'The crisis comment has been updated !']]
     ],
     [[ptBoolean,true]]),

/* Post Protocol:*/
```

```
95/* PostP01 */
96 true
97 .
```

Listing D.10: Prolog file outactCoordinator-oeReportOnCrisis.pl.

D.11 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCoor oeSetCrisisHandler.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------
6msrop(outactCoordinator,
7   oeSetCrisisHandler,
8   [preProtocol,Self,
9    AdtCrisisID
10   ],
11  []):-
12/* Pre Protocol:*/
13 msrVar(ctState,TheSystem),
14 msrVar(actCoordinator,TheActor),
15 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
16 msrNav([Self],[rnActor],[TheActor]),
17
18/* PreP01 */
19 msrNav([TheSystem],
20    [vpStarted],
21    [[ptBoolean,true]]),
22
23 msrNav([TheActor],
24    [rnctAuthenticated,vpIsLogged],
25    [[ptBoolean,true]])
26.
27
28msrop(outactCoordinator,
29   oeSetCrisisHandler,
30   [preFunctional,Self,
31    AdtCrisisID
32    ],
33   []):-
34/* Pre Functional:*/
35 msrVar(ctState,TheSystem),
36 msrVar(actCoordinator,TheActor),
37
38 msrVar(dtCrisisID,AdtCrisisID),
39
40 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
41 msrNav([Self],[rnActor],[TheActor]),
42
43/* PreF01 */
44 msrNav([TheSystem],
45    [rnctCrisis,
46     msrSelect,
47     id,eq,[AdtCrisisID]
48    ],
49    ColCrisis),
50
51 msrNav(ColCrisis,
52     [msrSize,eq,[[ptInteger,1]]],
53     [[ptBoolean,true]])
54 .
55
56msrop(outactCoordinator,
57   oeSetCrisisHandler,
58   [post,Self,
59    AdtCrisisID
```

```
60     ],
61     []):-
62
63 /* Post Functional:*/
64 msrVar(ctState,TheSystem),
65 msrVar(actCoordinator,TheActor),
66 msrVar(ctCoordinator,TheCoordinator),
67 msrVar(ctCoordinator,TheCurrentHandler),
68
69 msrVar(ctCrisis,TheCrisis),
70 msrVar(dtCrisisID,AdtCrisisID),
71
72 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
73 msrNav([Self],[rnActor],[TheActor]),
74
75 /* PostF01 */
76 msrNav([TheSystem],
77    [rnctCrisis,
78     msrSelect,
79     id,eq,[AdtCrisisID]],
80    [TheCrisis]),
81
82 msrNav([TheCrisis],
83     [msmAtPost,status],
84     [[etCrisisStatus,handled]]),
85
86 msrNav([TheActor],
87     [rnctCoordinator],
88     [TheCoordinator]),
89 msrNav([TheCrisis],
90     [msmAtPost,rnHandler],
91     [TheCoordinator]),
92
93 msrNav([TheActor],
94     [rnInterfaceIN,
95     ieMessage,[[ptString,'You are now considered as handling the crisis !']]
96     ],
97     [[ptBoolean,true]]),
98
99 /* PostF02 */
100 msrNav([TheCrisis],
101     [rnAlerts,msrForAll,isSentToCoordinator,[TheActor]],
102     [[ptBoolean,true]]),
103
104 /* PostF03 */
105 ( msrNav([TheCrisis],
106      [rnHandler,msrSize,eq,[[ptInteger,1]]],
107      [[ptBoolean,true]])
108  -> (msrNav([TheCrisis],
109        [rnHandler],
110        [TheCurrentHandler]),
111    msrNav([TheCurrentHandler],
112        [rnactCoordinator,rnInterfaceIN,
113        ieMessage,[[ptString,'One of the crisis you were handling is now handled by one of your
    colleagues!']]
114        ],
115        [[ptBoolean,true]])
116    )
117  ; true
118 ),
119
120  /* PostF04 */
121 msrNav([TheCrisis],
122     [rnAlerts,rnSignaler,msrForAll,isAcknowledged,[]],
123     [[ptBoolean,true]]),
124
125 /* Post Protocol:*/
126 /* PostP01 */
127 true
```

```
128 .
```

Listing D.11: Prolog file outactCoordinator-oeSetCrisisHandler.pl.

D.12 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCoor oeSetCrisisStatus.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactCoordinator,
7   oeSetCrisisStatus,
8   [preProtocol,Self,
9    AdtCrisisID,
10   AetCrisisStatus
11   ],
12  []):-
13/* Pre Protocol:*/
14 msrVar(ctState,TheSystem),
15 msrVar(actCoordinator,TheActor),
16 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
17 msrNav([Self],[rnActor],[TheActor]),
18
19/* PreP01 */
20 msrNav([TheSystem],
21    [vpStarted],
22    [[ptBoolean,true]]),
23
24 msrNav([TheActor],
25    [rnctAuthenticated,vpIsLogged],
26    [[ptBoolean,true]])
27.
28
29msrop(outactCoordinator,
30   oeSetCrisisStatus,
31   [preFunctional,Self,
32    AdtCrisisID,
33    AetCrisisStatus
34    ],
35   []):-
36/* Pre Functional:*/
37 msrVar(ctState,TheSystem),
38 msrVar(actCoordinator,TheActor),
39
40 msrVar(dtCrisisID,AdtCrisisID),
41
42 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
43 msrNav([Self],[rnActor],[TheActor]),
44
45/* PreF01 */
46 msrNav([TheSystem],
47    [rnctCrisis,
48     msrSelect,
49     id,eq,[AdtCrisisID]
50    ],
51    ColCrisis),
52
53 msrNav(ColCrisis,
54     [msrSize,eq,[[ptInteger,1]]],
55     [[ptBoolean,true]])
56 .
57
58msrop(outactCoordinator,
59   oeSetCrisisStatus,
60   [post,Self,
61    AdtCrisisID,
```

```
62    AetCrisisStatus
63    ],
64    []):-
65
66/* Post Functional:*/
67 msrVar(ctState,TheSystem),
68 msrVar(actCoordinator,TheActor),
69
70 msrVar(ctCrisis,TheCrisis),
71 msrVar(dtCrisisID,AdtCrisisID),
72 msrVar(etCrisisStatus,AetCrisisStatus),
73
74 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
75 msrNav([Self],[rnActor],[TheActor]),
76
77/* PostF01 */
78 msrNav([TheSystem],
79    [rnctCrisis,
80     msrSelect,
81     id,eq,[AdtCrisisID]],
82    [TheCrisis]),
83
84 msrNav([TheCrisis],
85     [msmAtPost,status],
86     [AetCrisisStatus]),
87
88 msrNav([TheActor],
89     [rnInterfaceIN,
90     ieMessage,[[ptString,'The crisis status has been updated !']]
91     ],
92     [[ptBoolean,true]]),
93
94/* Post Protocol:*/
95/* PostP01 */
96 true
97 .
```

Listing D.12: Prolog file outactCoordinator-oeSetCrisisStatus.pl.

D.13 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactC oeSetCrisisType.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------------
6msrop(outactCoordinator,
7   oeSetCrisisType,
8   [preProtocol,Self,
9    AdtCrisisID,
10   AetCrisisType
11   ],
12   []):-
13/* Pre Protocol:*/
14 msrVar(ctState,TheSystem),
15 msrVar(actCoordinator,TheActor),
16 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
17 msrNav([Self],[rnActor],[TheActor]),
18
19/* PreP01 */
20 msrNav([TheSystem],
21    [vpStarted],
22    [[ptBoolean,true]]),
23
24 msrNav([TheActor],
25    [rnctAuthenticated,vpIsLogged],
26    [[ptBoolean,true]])
```

```
.

msrop(outactCoordinator,
   oeSetCrisisType,
   [preFunctional,Self,
    AdtCrisisID,
    AetCrisisType
    ],
   []):-
/* Pre Functional:*/
 msrVar(ctState,TheSystem),
 msrVar(actCoordinator,TheActor),

 msrVar(dtCrisisID,AdtCrisisID),

 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),

/* PreF01 */
 msrNav([TheSystem],
    [rnctCrisis,
     msrSelect,
     id,eq,[AdtCrisisID]
    ],
    ColCrisis),

 msrNav(ColCrisis,
     [msrSize,eq,[[ptInteger,1]]],
     [[ptBoolean,true]])
 .

msrop(outactCoordinator,
   oeSetCrisisType,
   [post,Self,
    AdtCrisisID,
    AetCrisisType
    ],
    []):-

/* Post Functional:*/
 msrVar(ctState,TheSystem),
 msrVar(actCoordinator,TheActor),

 msrVar(ctCrisis,TheCrisis),
 msrVar(dtCrisisID,AdtCrisisID),
 msrVar(etCrisisType,AetCrisisType),

 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
 msrNav([TheSystem],
    [rnctCrisis,
     msrSelect,
     id,eq,[AdtCrisisID]],
    [TheCrisis]),

 msrNav([TheCrisis],
     [msmAtPost,type],
     [AetCrisisType]),

 msrNav([TheActor],
     [rnInterfaceIN,
     ieMessage,[[ptString,'The crisis type has been updated !']]
     ],
     [[ptBoolean,true]]),

/* Post Protocol:*/
/* PostP01 */
 true
```

```
.
```

Listing D.13: Prolog file outactCoordinator-oeSetCrisisType.pl.

D.14 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactC oeValidateAlert.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------
msrop(outactCoordinator,
   oeValidateAlert,
   [preProtocol,Self,
    AdtAlertID
    ],
   []):-
/* Pre Protocol:*/
 msrVar(ctState,TheSystem),
 msrVar(actCoordinator,TheActor),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),

/* PreP01 */
 msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]]),

 msrNav([TheActor],
    [rnctAuthenticated,vpIsLogged],
    [[ptBoolean,true]])
.

msrop(outactCoordinator,
   oeValidateAlert,
   [preFunctional,Self,
    AdtAlertID
    ],
   []):-
/* Pre Functional:*/
 msrVar(ctState,TheSystem),
 msrVar(actCoordinator,TheActor),

 msrVar(dtAlertID,AdtAlertID),

 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
 msrNav([Self],[rnActor],[TheActor]),

/* PreF01 */
 msrNav([TheSystem],
    [rnctAlert,
     msrSelect,
     id,eq,[AdtAlertID]
    ],
    ColAlerts),

 msrNav(ColAlerts,
     [msrSize,eq,[[ptInteger,1]]],
     [[ptBoolean,true]])
 .

msrop(outactCoordinator,
   oeValidateAlert,
   [post,Self,
    AdtAlertID
    ],
    []):-
```

```
62
63/* Post Functional:*/
64 msrVar(ctState,TheSystem),
65 msrVar(actCoordinator,TheActor),
66
67 msrVar(ctAlert,TheAlert),
68 msrVar(dtAlertID,AdtAlertID),
69
70 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
71 msrNav([Self],[rnActor],[TheActor]),
72
73/* PostF01 */
74 msrNav([TheSystem],
75    [rnctAlert,
76     msrSelect,
77     id,eq,[AdtAlertID]],
78    [TheAlert]),
79
80 msrNav([TheAlert],
81     [msmAtPost,status],
82     [[etAlertStatus,valid]]),
83
84 msrNav([TheActor],
85     [rnInterfaceIN,
86     ieMessage,[[ptString,'The Alert is now declared as valid !']]
87     ],
88     [[ptBoolean,true]]),
89
90 /* Post Protocol:*/
91/* PostP01 */
92 true
93 .
```

Listing D.14: Prolog file outactCoordinator-oeValidateAlert.pl.

D.15 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactMsrC oeCreateSystemAndEnvironment.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5/*
6/**************************************
7MSRCreatorActor
8**************************************/
9
10/*** createSystemAndEnvironment ***/
11
12msrop(outactMsrCreator,
13   oeCreateSystemAndEnvironment,
14   [preFunctional,_Self,_AqtyComCompanies],
15   []):-
16 true.
17
18msrop(outactMsrCreator,
19   oeCreateSystemAndEnvironment,
20   [preProtocol,_Self,_AqtyComCompanies],
21   []):-
22 true.
23
24msrop(outactMsrCreator,
25   oeCreateSystemAndEnvironment,
26   [post,_Self,AqtyComCompanies],
27   []):-
28
29 msrVar(ctState,TheSystem),
30 msrVar(actMsrCreator,AactMsrCreator),
```

```
31 msrVar(actAdministrator,AactAdministrator),
32
33 msrVar(dtInteger, AnextValueForAlertID),
34 msrVar(dtInteger, AnextValueForCrisisID),
35 msrVar(dtDateAndTime, Aclock),
36 msrVar(dtSecond, AcrisisReminderPeriod),
37 msrVar(dtSecond, AmaxCrisisReminderPeriod),
38 msrVar(ptBoolean, AvpStarted),
39
40 /* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
41 msrNav([AnextValueForAlertID],
42     [value,eq,[[ptInteger,1]]],
43     [[ptBoolean,true]]),
44
45 msrNav([AnextValueForCrisisID],
46     [value,eq,[[ptInteger,1]]],
47     [[ptBoolean,true]]),
48
49msrNav([Aclock],
50    [date,year,value],
51    [[ptInteger,1970]]),
52msrNav([Aclock],
53    [date,month,value],
54    [[ptInteger,01]]),
55msrNav([Aclock],
56    [date,day,value],
57    [[ptInteger,01]]),
58
59msrNav([Aclock],
60    [time,hour,value],
61    [[ptInteger,00]]),
62msrNav([Aclock],
63    [time,minute,value],
64    [[ptInteger,00]]),
65msrNav([Aclock],
66    [time,second,value],
67    [[ptInteger,00]]),
68
69 msrNav([AcrisisReminderPeriod],
70     [value,eq,[[ptInteger,300]]],
71     [[ptBoolean,true]]),
72
73 msrNav([AmaxCrisisReminderPeriod],
74     [value,eq,[[ptInteger,1200]]],
75     [[ptBoolean,true]]),
76
77 msrNav([AvpStarted],
78     [],
79     [[ptBoolean,true]]),
80
81 msrNav([TheSystem],
82     [init,[AnextValueForAlertID,
83        AnextValueForCrisisID,
84        Aclock,
85        AcrisisReminderPeriod,
86        AmaxCrisisReminderPeriod,
87        Aclock,
88        AvpStarted]
89        ],
90     [[ptBoolean,true]]),
91
92/* PostF02*/
93 msrNav([AactMsrCreator],
94     [init,[]],
95     [[ptBoolean,true]]),
96
97 /* PostF03 */
98 msrVarCol(actComCompany,AqtyComCompanies,AactComCompanyCol),
99
100 msrNav(AactComCompanyCol,
```

```
101     [msrForAll,init,[]],
102     [[ptBoolean,true]]),
103
104 /* PostF04*/
105 msrNav([AactAdministrator],
106     [init,[]],
107     [[ptBoolean,true]]),
108
109 /* PostF05*/
110 msrVar(actActivator,AactActivator),
111 msrNav([AactActivator],
112     [init,[]],
113     [[ptBoolean,true]]),
114
115/* PostF06 */
116 msrVar(ctAdministrator,ActAdministrator),
117 msrVar(dtLogin,AdtLogin),
118 msrVar(dtPassword,AdtPassword),
119
120 msrNav([AdtLogin],
121     [value,eq,[[ptString,'icrashadmin']]],
122     [[ptBoolean,true]]),
123
124 msrNav([AdtPassword],
125     [value,eq,[[ptString,'7WXC1359']]],
126     [[ptBoolean,true]]),
127
128 msrNav([ActAdministrator],
129     [init,[AdtLogin,AdtPassword]],
130     [[ptBoolean,true]]),
131
132 /* PostF07*/
133 msrNav([ActAdministrator],
134     [msmAtPost,rnactAuthenticated],
135     [AactAdministrator]),
136
137/* Post Protocol:*/
138/* PostP01 */
139true
140.
```

Listing D.15: Prolog file outactMsrCreator-oeCreateSystemAndEnvironment.pl.

D.16 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctAdministrator-init.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctAdministrator,init,[Self,
7          Alogin,
8          Apwd],
9   Result):-
10(
11msrVar(ctAdministrator,Self),
12
13/* Post F01 */
14msrNav([Self],[login],[Alogin]),
15msrNav([Self],[pwd],[Apwd]),
16msrNav([Self],[vpIsLogged],[[ptBoolean,false]]),
17
18/* Post F02 */
19 msrNav([Self],[msrIsNew],[Self])
20)
21-> Result = [ptBoolean,true]
22; Result = [ptBoolean,false]
```

```
23.
```

Listing D.16: Prolog file PrimaryTypesClasses-ctAdministrator-init.pl.

D.17 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctAlert-init.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctAlert,init,[Self,
7          Aid,
8          Astatus,
9          Alocation,
10         Ainstant,
11         Acomment],
12   Result):-
13
14/* Post F01 */
15(
16msrVar(ctAlert,Self),
17
18msrNav([Self],[id],[Aid]),
19msrNav([Self],[status],[Astatus]),
20msrNav([Self],[location],[Alocation]),
21msrNav([Self],[instant],[Ainstant]),
22msrNav([Self],[comment],[Acomment]),
23
24/* Post F02 */
25 msrNav([Self],[msrIsNew],[Self])
26)
27-> Result = [ptBoolean,true]
28; Result = [ptBoolean,false]
29.
```

Listing D.17: Prolog file PrimaryTypesClasses-ctAlert-init.pl.

D.18 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctAlert-isSentToCoordinator.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctAlert,isSentToCoordinator,[Self,AactCoordinator],
7   Result):-
8
9/* Post F01 */
10(
11 msrNav([AactCoordinator],
12     [rnInterfaceIN,ieSendAnAlert,[Self]],
13     [[ptBoolean,true]])
14)
15-> Result = [ptBoolean,true]
16; Result = [ptBoolean,false]
17.
```

Listing D.18: Prolog file PrimaryTypesClasses-ctAlert-isSentToCoordinator.pl.

D.19 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctAuthenticated-init.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctAuthenticated,init,[Self,
              Alogin,
              Apwd],
   Result):-

/* Post F01 */
(
msrVar(ctAuthenticated,Self),

msrNav([Self],[login],[Alogin]),
msrNav([Self],[pwd],[Apwd]),
msrNav([Self],[vpIsLogged],[[ptBoolean,false]]),

/* Post F02 */
 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.19: Prolog file PrimaryTypesClasses-ctAuthenticated-init.pl.

D.20 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctCoordinator-init.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctCoordinator,init,[Self,
          Aid,
          Alogin,
          Apwd],
   Result):-

/* Post F01 */
(
msrVar(ctCoordinator,Self),

msrNav([Self],[id],[Aid]),
msrNav([Self],[login],[Alogin]),
msrNav([Self],[pwd],[Apwd]),
msrNav([Self],[vpIsLogged],[[ptBoolean,false]]),

/* Post F02 */
 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.20: Prolog file PrimaryTypesClasses-ctCoordinator-init.pl.

D.21 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctCrisis-handlingDelayPassed.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
5
6msrop(ctCrisis,handlingDelayPassed,[Self],
7   Result):-
8
9/* Post F01 */
10(
11 msrVar(ctState,TheSystem),
12 msrVar(dtInteger,CurrentClockSecondsQty),
13 msrVar(dtInteger,LastReminderSecondsQty),
14 msrVar(dtSecond,CrisisReminderPeriod),
15
16 msrNav([Self],[rnSystem],[TheSystem]),
17
18 msrNav([Self],
19     [status],
20     [[etCrisisStatus,pending]]),
21
22 msrNav([TheSystem],
23    [clock,toSecondsQty,[]],
24    [CurrentClockSecondsQty]),
25
26 msrNav([TheSystem],
27    [vpLastReminder,toSecondsQty,[]],
28    [LastReminderSecondsQty]),
29
30 msrNav([TheSystem],
31    [crisisReminderPeriod],
32    [CrisisReminderPeriod]),
33
34 msrNav([CurrentClockSecondsQty],
35    [sub,[LastReminderSecondsQty],
36     gt, [CrisisReminderPeriod]
37     ],
38    [[ptBoolean,true]])
39
40)
41-> Result = [ptBoolean,true]
42; Result = [ptBoolean,false]
43.
```

Listing D.21: Prolog file PrimaryTypesClasses-ctCrisis-handlingDelayPassed.pl.

D.22 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctCrisis-init.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctCrisis,init,[Self,
7          Aid,
8          Atype,
9          Astatus,
10         Alocation,
11         Ainstant,
12         Acomment],
13   Result):-
14
15/* Post F01 */
16(
17msrVar(ctCrisis,Self),
18
19msrNav([Self],[id],[Aid]),
20msrNav([Self],[type],[Atype]),
21msrNav([Self],[status],[Astatus]),
22msrNav([Self],[location],[Alocation]),
23msrNav([Self],[instant],[Ainstant]),
```

```
msrNav([Self],[comment],[Acomment]),

/* Post F02 */
 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.22: Prolog file PrimaryTypesClasses-ctCrisis-init.pl.

D.23 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctCrisis-isAllocatedIfPossible.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctCrisis,isAllocatedIfPossible,[Self],
   Result):-
(
 msrVar(ctState,TheSystem),
msrNav([Self],[rnSystem],[TheSystem]),

 msrVar(actCoordinator,TheCoordinatorActor),
 msrVar(ctCoordinator,TheCoordinator),
 msrVar(ptString,TheMessage),
 msrVar(ptString,TheCrisisIDptString),

 (
 /* Post F01 */
msrNav([Self],
    [maxHandlingDelayPassed,[]],
    [[ptBoolean,true]]),

 ( msrNav([TheSystem],
     [rnactCoordinator,msrIsEmpty],
     [[ptBoolean,false]])
  -> (
    /* Post F02 */
    msrNav([TheSystem],
       [rnactCoordinator,msrAny,msrTrue],
       [TheCoordinatorActor]),

    msrNav([TheCoordinatorActor],
        [rnctCoordinator],
        [TheCoordinator]),

    msrNav([Self],
        [msmAtPost,rnHandler],
        [TheCoordinator]),

     msrNav([Self],
        [msmAtPost,status],
        [[etCrisisStatus,handled]]),

     msrNav([Self],
        [id,value],
        [TheCrisisIDptString]),

    msrNav([[ptString,'You are now considered as handling the crisis having ID: ']],
        [ptStringConcat,[TheCrisisIDptString]],
        [TheMessage]),

    msrNav([TheCoordinatorActor],
        [rnInterfaceIN,
        ieMessage,[TheMessage]
```

```
55          ],
56          [[ptBoolean,true]])
57     )
58 ; ( /* Post F03 */
59     msrNav([TheSystem],
60        [rnactAdministrator,msrForAll,rnInterfaceIN,
61        ieMessage,[[ptString,'Please add new coordinators to handle pending crisis !']]],
62        [[ptBoolean,true]])
63   )
64 )
65 )
66)
67-> Result = [ptBoolean,true]
68; Result = [ptBoolean,false]
69.
```

Listing D.23: Prolog file PrimaryTypesClasses-ctCrisis-isAllocatedIfPossible.pl.

D.24 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctCrisis-isSentToCoordinator.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctCrisis,isSentToCoordinator,[Self,AactCoordinator],
7    Result):-
8
9/* Post F01 */
10(
11 msrNav([AactCoordinator],
12      [rnInterfaceIN,ieSendACrisis,[Self]],
13      [[ptBoolean,true]])
14)
15-> Result = [ptBoolean,true]
16; Result = [ptBoolean,false]
17.
```

Listing D.24: Prolog file PrimaryTypesClasses-ctCrisis-isSentToCoordinator.pl.

D.25 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctCrisis-maxHandlingDelayPassed.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctCrisis,maxHandlingDelayPassed,[Self],
7    Result):-
8
9/* Post F01 */
10(
11 msrVar(ctState,TheSystem),
12 msrVar(dtInteger,CurrentClockSecondsQty),
13 msrVar(dtInteger,CrisisInstantSecondsQty),
14 msrVar(dtSecond,MaxCrisisReminderPeriod),
15
16 msrNav([Self],[rnSystem],[TheSystem]),
17
18 msrNav([Self],
19      [status],
20      [[etCrisisStatus,pending]]),
21
22 msrNav([TheSystem],
```

```
23     [clock,toSecondsQty,[]],
24     [CurrentClockSecondsQty]),
25
26  msrNav([Self],
27     [instant,toSecondsQty,[]],
28     [CrisisInstantSecondsQty]),
29
30  msrNav([TheSystem],
31     [maxCrisisReminderPeriod],
32     [MaxCrisisReminderPeriod]),
33
34  msrNav([CurrentClockSecondsQty],
35     [sub,[CrisisInstantSecondsQty],
36      gt, [MaxCrisisReminderPeriod]
37      ],
38     [[ptBoolean,true]])
39
40 )
41 -> Result = [ptBoolean,true]
42 ; Result = [ptBoolean,false]
43 .
```

Listing D.25: Prolog file PrimaryTypesClasses-ctCrisis-maxHandlingDelayPassed.pl.

D.26 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctHuman-init.pl

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 /* DISCONTIGUOUS PREDICATES */
3 :- multifile msrop/4.
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6 msrop(ctHuman,init,[Self,
7           Aid,
8           Akind],
9    Result):-
10
11 /* Post F01 */
12 (
13 msrVar(ctHuman,Self),
14
15 msrNav([Self],[id],[Aid]),
16 msrNav([Self],[kind],[Akind]),
17
18 /* Post F02 */
19  msrNav([Self],[msrIsNew],[Self])
20 )
21 -> Result = [ptBoolean,true]
22 ; Result = [ptBoolean,false]
23 .
```

Listing D.26: Prolog file PrimaryTypesClasses-ctHuman-init.pl.

D.27 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses ctHuman-isAcknowledged.pl

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 /* DISCONTIGUOUS PREDICATES */
3 :- multifile msrop/4.
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6 msrop(ctHuman,isAcknowledged,[Self],Result):-
7
8 /* Post F01 */
9 (msrVar(dtPhoneNumber,AdtPhoneNumber),
10  msrVar(dtSMS,AdtSMS),
```

```
11
12 msrNav([Self],
13     [id,eq,[AdtPhoneNumber]],
14     [[ptBoolean,true]]),
15 msrNav([AdtSMS],
16     [value,eq,[[ptString,'The handling of your alert by our services is in progress !']]],
17     [[ptBoolean,true]]),
18 msrNav([Self],
19     [rnactComCompany,rnInterfaceIN,ieSmsSend,[AdtPhoneNumber,AdtSMS]],
20     [[ptBoolean,true]])
21)
22-> Result = [ptBoolean,true]
23; Result = [ptBoolean,false]
24.
```

Listing D.27: Prolog file PrimaryTypesClasses-ctHuman-isAcknowledged.pl.

D.28 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla ctState-init.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctState,init,[Self,
7          AnextValueForAlertID,
8          AnextValueForCrisisID,
9          Aclock,
10         AcrisisReminderPeriod,
11         AmaxCrisisReminderPeriod,
12         AvpLastReminder,
13         AvpStarted],
14   Result):-
15
16 /* Post F01 */
17(
18 msrVar(ctState,Self),
19
20 msrNav([Self],[nextValueForAlertID],[AnextValueForAlertID]),
21 msrNav([Self],[nextValueForCrisisID],[AnextValueForCrisisID]),
22 msrNav([Self],[clock],[Aclock]),
23 msrNav([Self],[crisisReminderPeriod],[AcrisisReminderPeriod]),
24 msrNav([Self],[maxCrisisReminderPeriod],[AmaxCrisisReminderPeriod]),
25 msrNav([Self],[vpLastReminder],[AvpLastReminder]),
26 msrNav([Self],[vpStarted],[AvpStarted]),
27
28 msrNav([Self],[msrIsNew],[Self])
29)
30-> Result = [ptBoolean,true]
31; Result = [ptBoolean,false]
32.
```

Listing D.28: Prolog file PrimaryTypesClasses-ctState-init.pl.

D.29 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtAlertID-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(dtAlertID,is,[AdtValue],Result):-
7% msd01
8msrVar(ptBoolean,TheResult),
```

```
(
 ( msrNav([AdtValue],
    [value,length,[],gt,[[ptInteger,0]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,length,[],leq,[[ptInteger,20]]],
    [[ptBoolean,true]])
 )
 -> (TheResult = [ptBoolean,true])
 ; (TheResult = [ptBoolean,false])
),
TheResult = Result
.

/*
| ?- X = [dtAlertID,[],[[dtString,[[value,[ptString,'0123456789']]],[]]]],
msrNav([X],[is,[]],[Result]).

X = [dtAlertID,[],[[dtString,[[value,[ptString,'0123456789']]],[]]]],
Result = [ptBoolean,true] ?

yes

| ?- X = [dtAlertID,[],[[dtString,[[value,[ptString,'01234567890123456789012345678 9']]],[]]]],
msrNav([X],[is,[]],[Result]).

X = [dtAlertID,[],[[dtString,[[value,[ptString,'01234567890123456789012345678 9']]],[]]]],
Result = [ptBoolean,false] ?

yes
*/
```

Listing D.29: Prolog file PrimaryTypesDatatypes-dtAlertID-is.pl.

D.30 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDataty dtComment-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% dtComment

%msd01
msrop(dtComment,is,[AdtValue],Result):-
 msrVar(ptBoolean,TheResult),
 msrVar(ptInteger,MaxLength),
(
  (
    (
     MaxLength = [ptInteger,160],
     msrNav([AdtValue],
        [value,length,[],leq,[MaxLength]],
        [[ptBoolean,true]])
    )
     -> TheResult = [ptBoolean,true]
     ; TheResult = [ptBoolean,false]
  )
),
 Result = TheResult
.

/*
| ?- X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg ! Please help ...']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg ! Please help ...']]],[]]]],
Result = [ptBoolean,true] ?
yes
```

```
32
33| ?- X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg when I was running with my dog
      to go to the skate park because my friends called me on my mobile phone and told me that a skate
       star was doing triple back flips.']]],[]]]],
34msrNav([X],[is,[]],[Result]).
35X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg when I was running with my dog to go
      to the skate park because my friends called me on my mobile phone and told me that a skate star
      was doing triple back flips.']]],[]]]],
36Result = [ptBoolean,false] ?
37yes
38*/
```

Listing D.30: Prolog file PrimaryTypesDatatypes-dtComment-is.pl.

D.31 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtCoordinatorID-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(dtCoordinatorID,is,[AdtValue],Result):-
7% msd01
8 msrVar(ptBoolean,TheResult),
9(
10 ( msrNav([AdtValue],
11    [value,length,[],gt,[[ptInteger,0]]],
12    [[ptBoolean,true]]),
13   msrNav([AdtValue],
14    [value,length,[],leq,[[ptInteger,5]]],
15    [[ptBoolean,true]])
16 )
17 -> (TheResult = [ptBoolean,true])
18 ; (TheResult = [ptBoolean,false])
19),
20TheResult = Result
21.
```

Listing D.31: Prolog file PrimaryTypesDatatypes-dtCoordinatorID-is.pl.

D.32 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtCrisisID-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(dtCrisisID,is,[AdtValue],Result):-
7% msd01
8 msrVar(ptBoolean,TheResult),
9(
10 ( msrNav([AdtValue],
11    [value,length,[],gt,[[ptInteger,0]]],
12    [[ptBoolean,true]]),
13   msrNav([AdtValue],
14    [value,length,[],leq,[[ptInteger,10]]],
15    [[ptBoolean,true]])
16 )
17 -> (TheResult = [ptBoolean,true])
18 ; (TheResult = [ptBoolean,false])
19),
20TheResult = Result
21.
22/*
```

```
| ?- X = [dtCrisisID,[],[[dtString,[[value,[ptString,'0123456789']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtCrisisID,[],[[dtString,[[value,[ptString,'0123456789']]],[]]]],
Result = [ptBoolean,true] ?
yes

| ?- X = [dtCrisisID,[],[[dtString,[[value,[ptString,'0123456789a']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtCrisisID,[],[[dtString,[[value,[ptString,'0123456789a']]],[]]]],
Result = [ptBoolean,false] ?
yes
*/
```

Listing D.32: Prolog file PrimaryTypesDatatypes-dtCrisisID-is.pl.

D.33 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDataty dtGPSLocation-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% dtPhoneNumber

% msd01
msrop(dtGPSLocation,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
  (
     msrNav([AdtValue],
        [latitude,is,[]],
        [[ptBoolean,true]]),
     msrNav([AdtValue],
        [longitude,is,[]],
        [[ptBoolean,true]])
  )
 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
),

 Result = TheResult
.
```

Listing D.33: Prolog file PrimaryTypesDatatypes-dtGPSLocation-is.pl.

D.34 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDataty dtGPSLocation-isNearTo.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% dtGPSLocation

msrop(dtGPSLocation,isNearTo,[Self,AdtValue],Result):-
msrVar(ptBoolean,TheResult),
msrVar(dtReal,EarthRadius),
msrVar(dtReal,MaxDistance),

msrVar(dtLatitude,ComparedLatitude),
msrVar(dtLongitude,ComparedLongitude),

msrVar(dtReal,R1),msrVar(dtReal,R1a),
msrVar(dtReal,R2),msrVar(dtReal,R2a),
```

```
18
19(
20  (
21    (
22     % msd01
23     msrNav([EarthRadius],[value],[[ptReal,6371]]),
24     msrNav([MaxDistance],[value],[[ptReal,100]]),
25
26     msrNav([AdtValue],[latitude],[ComparedLatitude]),
27     msrNav([AdtValue],[longitude],[ComparedLongitude]),
28
29     msrNav([Self],[latitude,sin,[]],[R1a]),
30     msrNav([AdtValue],[latitude,sin,[],mul,[R1a]],[R1]),
31
32     msrNav([Self],[latitude,cos,[]],[R2a]),
33     msrNav([AdtValue],[latitude,cos,[],mul,[R2a]],[R2]),
34
35     msrNav([AdtValue],[longitude],[ComparedLongitude]),
36     msrNav([Self],[longitude,sub,[ComparedLongitude],cos,[],mul,[R2],
37            add,[R1],
38            acos,[],
39            mul,[EarthRadius],
40            sub,[MaxDistance],
41            value,leq,[[ptReal,0]]],
42        [[ptBoolean,true]])
43    )
44     -> TheResult = [ptBoolean,true]
45     ; TheResult = [ptBoolean,false]
46  )
47),
48 Result = TheResult
49.
```

Listing D.34: Prolog file PrimaryTypesDatatypes-dtGPSLocation-isNearTo.pl.

D.35 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtLatitude-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6% msd01
7msrop(dtLatitude,is,[AdtValue],Result):-
8msrVar(ptBoolean,TheResult),
9(
10 ( msrNav([AdtValue],
11    [value,geq,[[ptReal,-90.0]]],
12    [[ptBoolean,true]]),
13   msrNav([AdtValue],
14    [value,leq,[[ptReal,+90.0]]],
15    [[ptBoolean,true]])
16 )
17 -> (TheResult = [ptBoolean,true])
18 ; (TheResult = [ptBoolean,false])
19),
20Result = TheResult
21.
```

Listing D.35: Prolog file PrimaryTypesDatatypes-dtLatitude-is.pl.

D.36 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtLogin-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%% dtComment
6
7%msd01
8msrop(dtLogin,is,[AdtValue],Result):-
9 msrVar(ptBoolean,TheResult),
10 msrVar(ptInteger,MaxLength),
11(
12  (
13    (
14     MaxLength = [ptInteger,20],
15     msrNav([AdtValue],
16        [value,length,[],leq,[MaxLength]],
17        [[ptBoolean,true]])
18    )
19     -> TheResult = [ptBoolean,true]
20     ; TheResult = [ptBoolean,false]
21  )
22),
23 Result = TheResult
24.
25/*
26| ?- X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567']]],[]]]],
27msrNav([X],[is,[]],[Result]).
28X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567']]],[]]]],
29Result = [ptBoolean,true] ?
30yes
31
32| ?- X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567a']]],[]]]],
33msrNav([X],[is,[]],[Result]).
34X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567a']]],[]]]],
35Result = [ptBoolean,false] ?
36yes
37*/
```

Listing D.36: Prolog file PrimaryTypesDatatypes-dtLogin-is.pl.

D.37 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDataty dtLongitude-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6%% dtPhoneNumber
7
8% msd01
9msrop(dtLongitude,is,[AdtValue],Result):-
10msrVar(ptBoolean,TheResult),
11(
12 ( msrNav([AdtValue],
13    [value,geq,[[ptReal,-180.0]]],
14    [[ptBoolean,true]]),
15   msrNav([AdtValue],
16    [value,leq,[[ptReal,+180.0]]],
17    [[ptBoolean,true]])
18 )
19 -> (TheResult = [ptBoolean,true])
20 ; (TheResult = [ptBoolean,false])
21),
22
23 Result = TheResult
24.
```

Listing D.37: Prolog file PrimaryTypesDatatypes-dtLongitude-is.pl.

D.38 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtPassword-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% dtComment

%msd01
msrop(dtPassword,is,[AdtValue],Result):-
 msrVar(ptBoolean,TheResult),
 msrVar(ptInteger,MinLength),
(
  (
    (
     MinLength = [ptInteger,6],
     msrNav([AdtValue],
        [value,length,[],geq,[MinLength]],
        [[ptBoolean,true]])
    )
     -> TheResult = [ptBoolean,true]
     ; TheResult = [ptBoolean,false]
  )
),
 Result = TheResult
.
/*
| ?- X = [dtPassword,[],[[dtString,[[value,[ptString,'012345']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtPassword,[],[[dtString,[[value,[ptString,'012345']]],[]]]],
Result = [ptBoolean,true] ?
yes

| ?- X = [dtPassword,[],[[dtString,[[value,[ptString,'01234']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtPassword,[],[[dtString,[[value,[ptString,'01234']]],[]]]],
Result = [ptBoolean,false] ?
yes
*/
```

Listing D.38: Prolog file PrimaryTypesDatatypes-dtPassword-is.pl.

D.39 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesDat dtPhoneNumber-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% dtPhoneNumber

% msd01
msrop(dtPhoneNumber,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
 ( msrNav([AdtValue],
    [value,length,[],gt,[[ptInteger,4]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,length,[],leq,[[ptInteger,30]]],
    [[ptBoolean,true]])
 )

 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
```

```
),
 Result = TheResult
.
/*
| ?- X = [dtPhoneNumber,[],[[dtString,[[value,[ptString,'(+352) 46 66 44 60 00']]],[]]]],
msrNav([X],[is,[]],[Result]).

X = [dtPhoneNumber,[],[[dtString,[[value,[ptString,'(+352) 46 66 44 60 00']]],[]]]],

Result = [ptBoolean,true] ?

yes
*/
```

Listing D.39: Prolog file PrimaryTypesDatatypes-dtPhoneNumber-is.pl.

D.40 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses etAlertStatus-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% etAlertStatus

% msd01
msrop(etAlertStatus,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
  (
     member(AdtValue,[pending, valid, invalid])
  )
 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
),
 Result = TheResult
.
```

Listing D.40: Prolog file PrimaryTypesDatatypes-etAlertStatus-is.pl.

D.41 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClasses etCrisisStatus-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% etCrisisStatus

% msd01
msrop(etCrisisStatus,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
  (
     member(AdtValue,[pending, handled, solved, closed])
  )
 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
),
 Result = TheResult
.
```

Listing D.41: Prolog file PrimaryTypesDatatypes-etCrisisStatus-is.pl.

D.42 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla etCrisisType-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% etCrisisType

% msd01
msrop(etCrisisType,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
  (
     member(AdtValue,[small, medium, huge])
  )
 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
),
 Result = TheResult
.
```

Listing D.42: Prolog file PrimaryTypesDatatypes-etCrisisType-is.pl.

D.43 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesCla etHumanKind-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% etHumanKind

% msd01
msrop(etHumanKind,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
  (
     member(AdtValue,[witness,victim,anonymous])
  )
 -> TheResult = [ptBoolean,true]
 ; TheResult = [ptBoolean,false]
),
 Result = TheResult
.
```

Listing D.43: Prolog file PrimaryTypesDatatypes-etHumanKind-is.pl.

D.44 File ./src-gen/prolog-ref-spec/Operations/Concepts/SecondaryTypesD dtSMS-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% dtComment

%msd01
msrop(dtSMS,is,[AdtValue],Result):-
 msrVar(ptBoolean,TheResult),
 msrVar(ptInteger,MaxLength),
(
  (
```

```
13     (
14      MaxLength = [ptInteger,160],
15      msrNav([AdtValue],
16         [value,length,[],leq,[MaxLength]],
17         [[ptBoolean,true]])
18     )
19      -> TheResult = [ptBoolean,true]
20      ; TheResult = [ptBoolean,false]
21   )
22),
23 Result = TheResult
24.
```

Listing D.44: Prolog file SecondaryTypesDatatypes-dtSMS-is.pl.

Glossary

abstract actor an actor that is not 22

actor An actor is a person, organization, or external system that plays a role in one or more interactions with the system 18

direct actor an actor that interacts directly with the system. It thus belongs to the environment. 22

indirect actor an actor that interacts indirectly with the system through a direct actor. It thus belongs the domain but not to the environment. 22

system operation a functionality of the system that can be triggered by a message sent by an actor belonging to the environment. 18

Bibliography

[1] Guelfi, N.: Messir: A Scientific Method for the Software Engineer. to be published (2017)

[2] Armour, F., Miller, G.: Advanced Use Case Modeling: Software Systems. Addison-Wesley (2001)

[3] ISO/IEC: ISO/IEC 25010 - Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models. (2011) ISO/IEC 13211-1.