Laboratory for Advanced Software Systems
University of Luxembourg

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

(*Report type: Specification*)

Thursday 1st December, 2016 - 01:48

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 Messir 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 Messir 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.

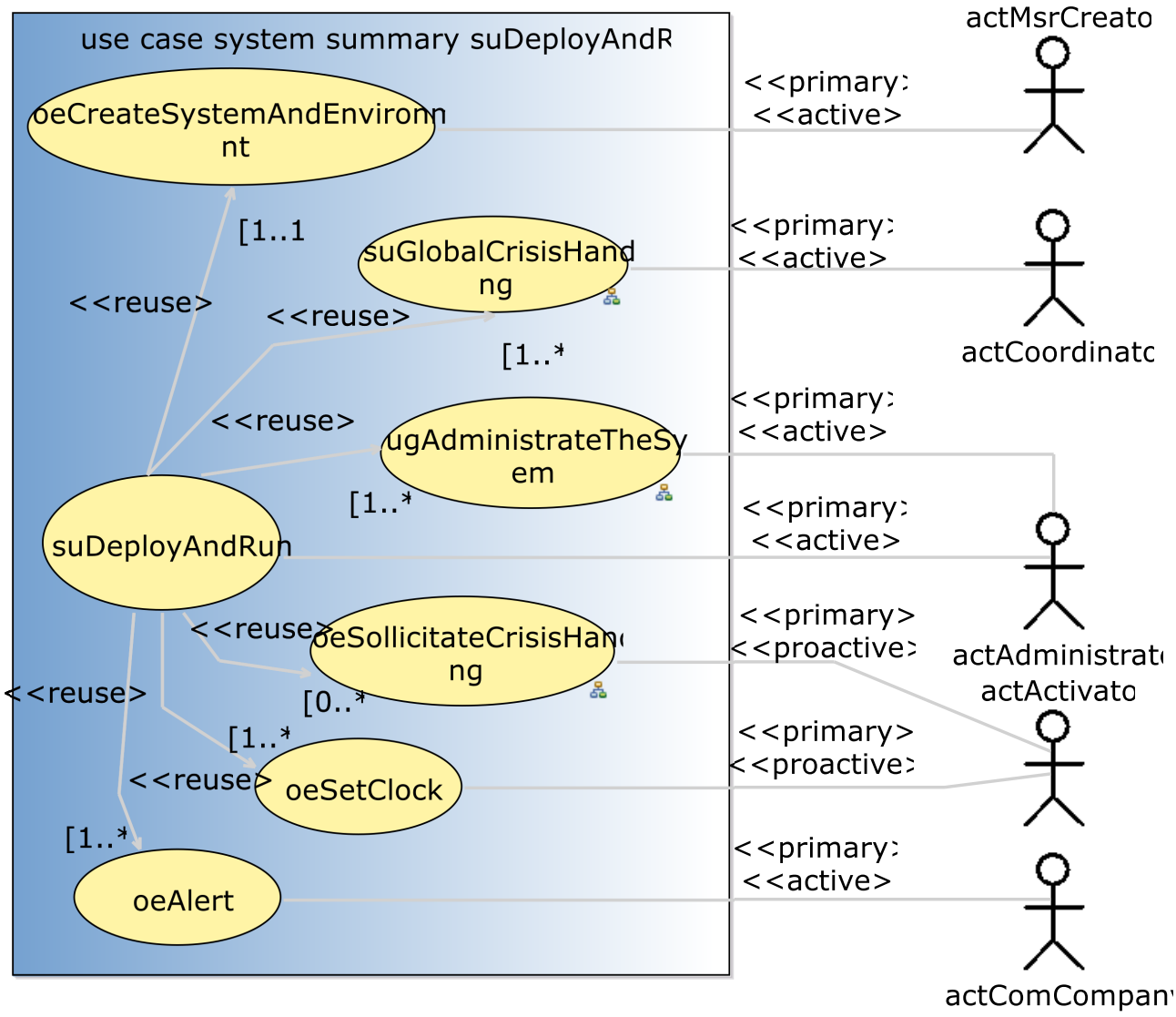

Figure 2.1: suDeployAndRun summary use case

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..*]
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.
Main Steps	
a	the actor actAdministrator executes the ugSecurelyUseSystem use case
b	the actor actAdministrator executes the oeAddCoordinator use case
c	the actor actAdministrator executes the oeDeleteCoordinator 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 2.3 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..*]
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)	

continues in next page ...

... Use-Case Description table continuation

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	
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 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.

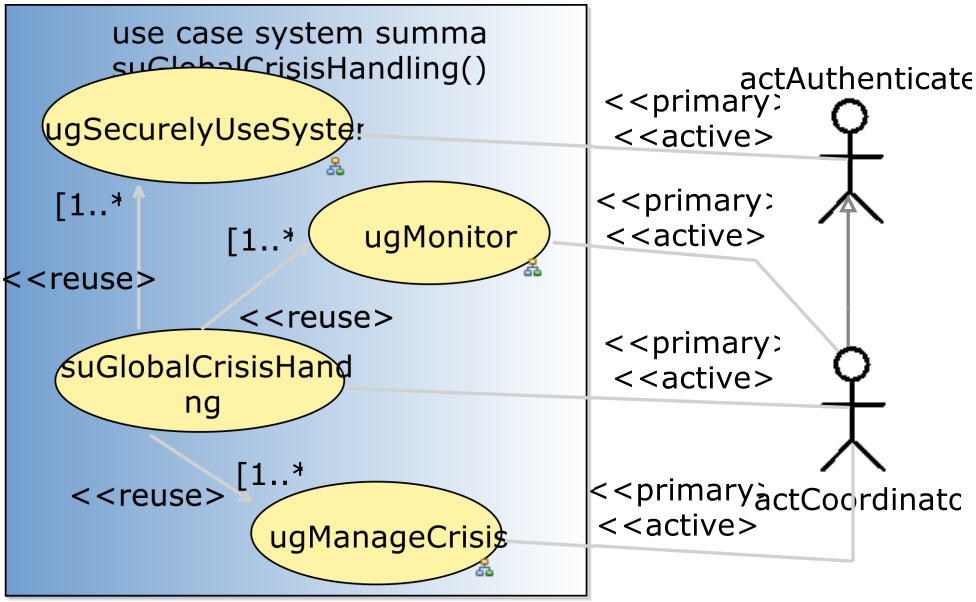

Figure 2.2: suGlobalCrisisHandling user goal use case

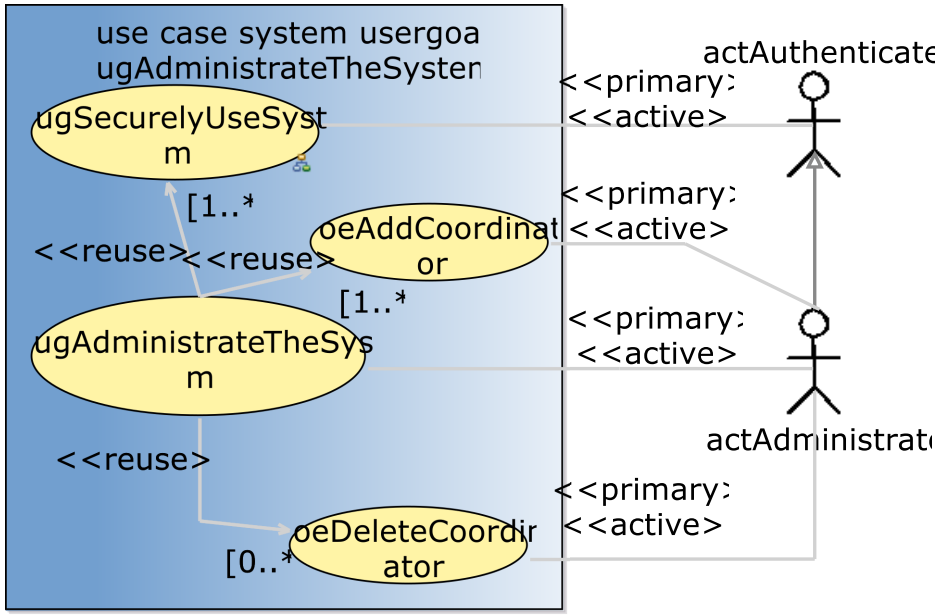

Figure 2.3: ugAdministrateTheSystem user goal use case

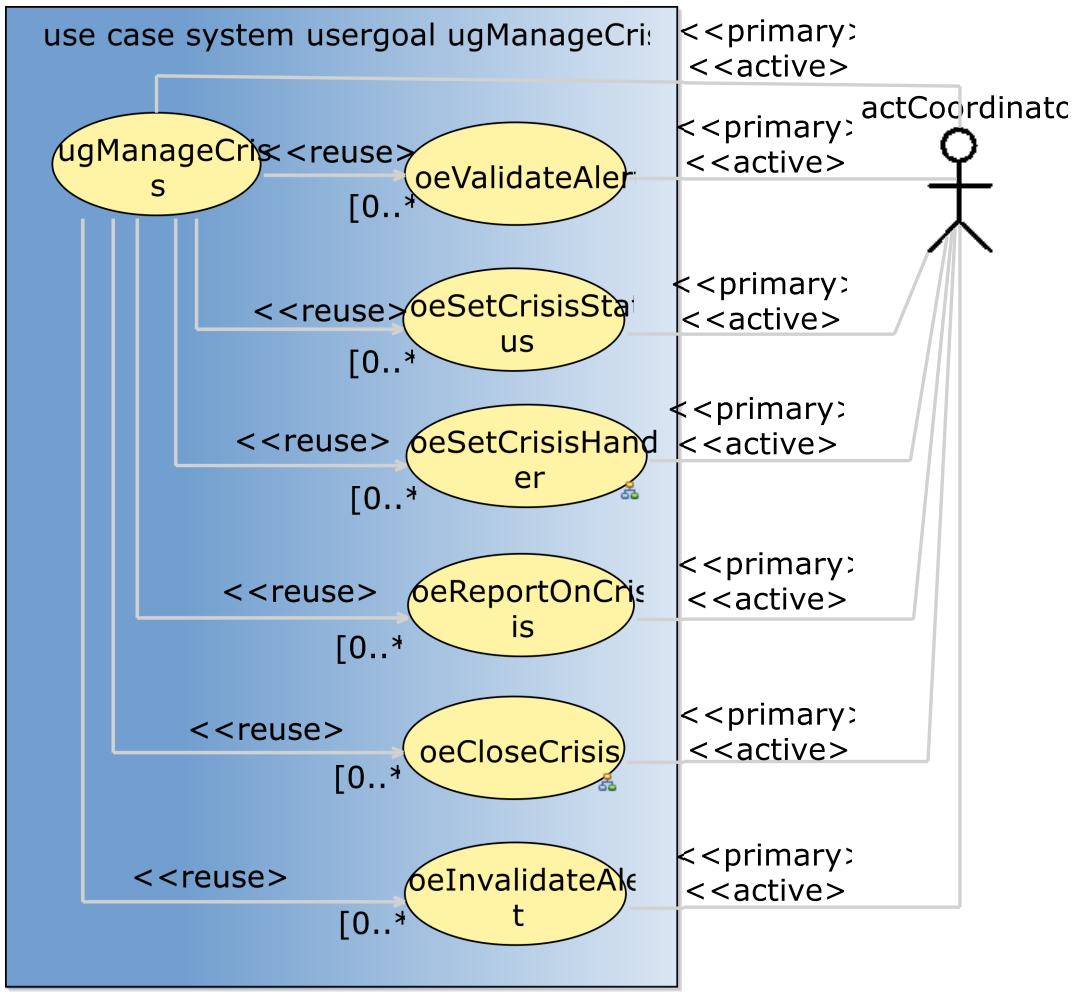

Figure 2.4: ugManageCrisis user goal use case

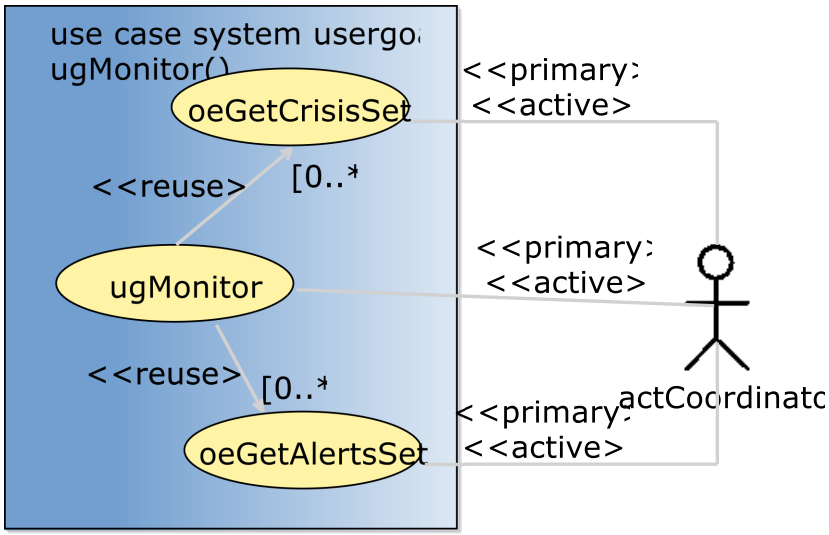

Figure 2.5: ugMonitor user goal use case

USE-CASE DESCRIPTION	
Name	ugSecurelyUseSystem
Scope	system
Level	usergoal
Primary actor(s)	
1	`actAuthenticated[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.	
Reuse	
1	`oeLogin [1..1]`
2	`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 `oeLogout` use case
Steps Ordering Constraints	
1	step (a) must always precede step (b).

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

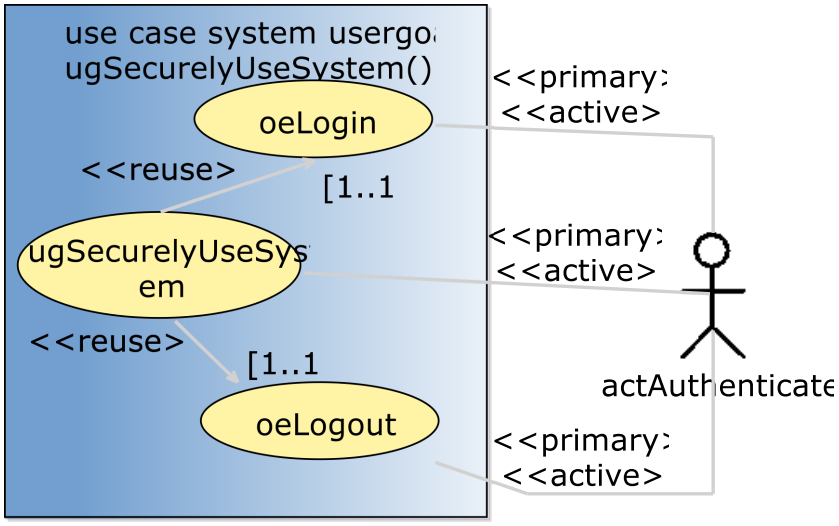

Figure 2.6: ugSecurelyUseSystem user goal use case

2.3.1.7 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

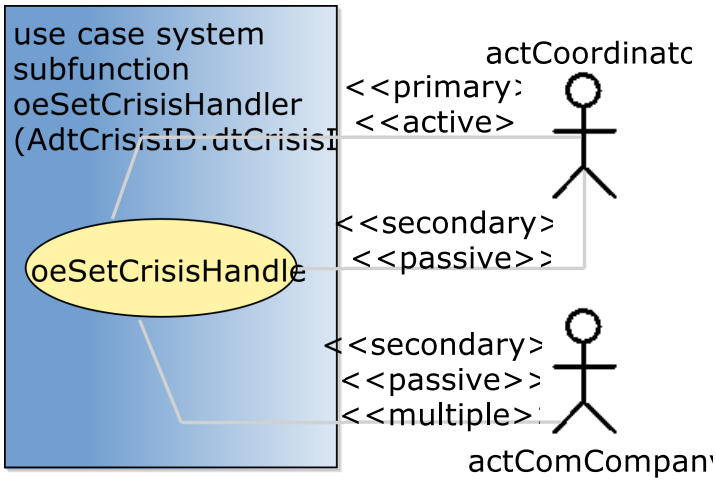

Figure 2.7: oeSetCrisisHandler subfunction use case

2.3.1.8 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)	
1	the iCrash system has been deployed.
2	there exist some crisis still pending and for which no solicitation has been sent to the administrator and the coordinators for more than a predefined maximum delay.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	a simple text message ieMessage('There are alerts not treated since more than the defined delay. Please REACT !') is sent to the system administrator and to all the coordinators of the environment for each crisis that is known to be not handled and for which no solicitation has been sent to the administrator and the coordinators for more than a predefined maximum delay.')
2	the reminder period for the concerned crisis is initialized.

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

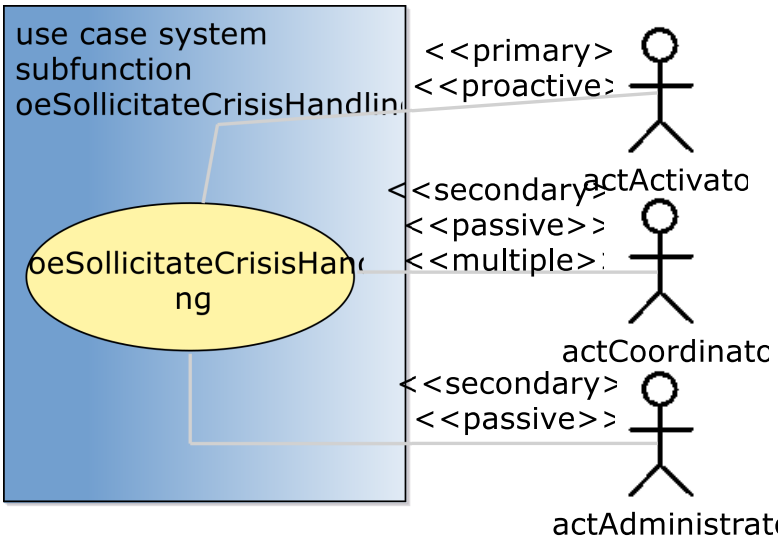

Figure 2.8: oeSollicitateCrisisHandling subfunction use case

2.3.2 Use Case Instance(s)

2.3.2.1 Use-Case Instance - uciSimpleAndComplete:suDeployAndRun

Figure 2.9

2.3.2.2 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 2.10 shows the sequence diagram representing the first part of a simple and complete use case instance for the summary use case `suDeployAndRun`.

2.3.2.3 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 2.11 shows the sequence diagram representing the second part of a simple and complete use


System
theCreator
: actMsrCreator
theClock
: actActivator
bill
: actAdminis
oeCreateSystemAndEnvironment("4")
out01
oeSetClock("2017:11:24 - 03:20:00")
out02
oeLogin("icrashadmin","7WXC1359")
ieMessage(You are logged ! Welcome ...)
oeAddCoordinator("1","steve","pwdMessirExcalibur2017")
ieCoordinatorAddedreturned()
oeLogout()
ieMessage(You are logged out ! Good Bye ...)
oeSetClock("2017:11:26 - 10:15:00")
out06
oeAlert("witness","2017:11:26","10:10:16","+352466445252","49.627675:6:159
ieSmsSend(+3524666445252,Your alert has been registered. We will har
oeSetClock("2017:11:26 - 10:30:00")
out08
oeSollicitateCrisisHandling()
out09
ieMessage(There are alerts pending since more than the defined delay. Pl
ieMessage(There are alerts pending since more than th
oeLogin("steve","pwdMessirExca
ieMessage(You are logged ! We
oeGetCrisisSet("pending
ieSendACrisis(crisis with ID 1
oeSetCrisisHandler("1"

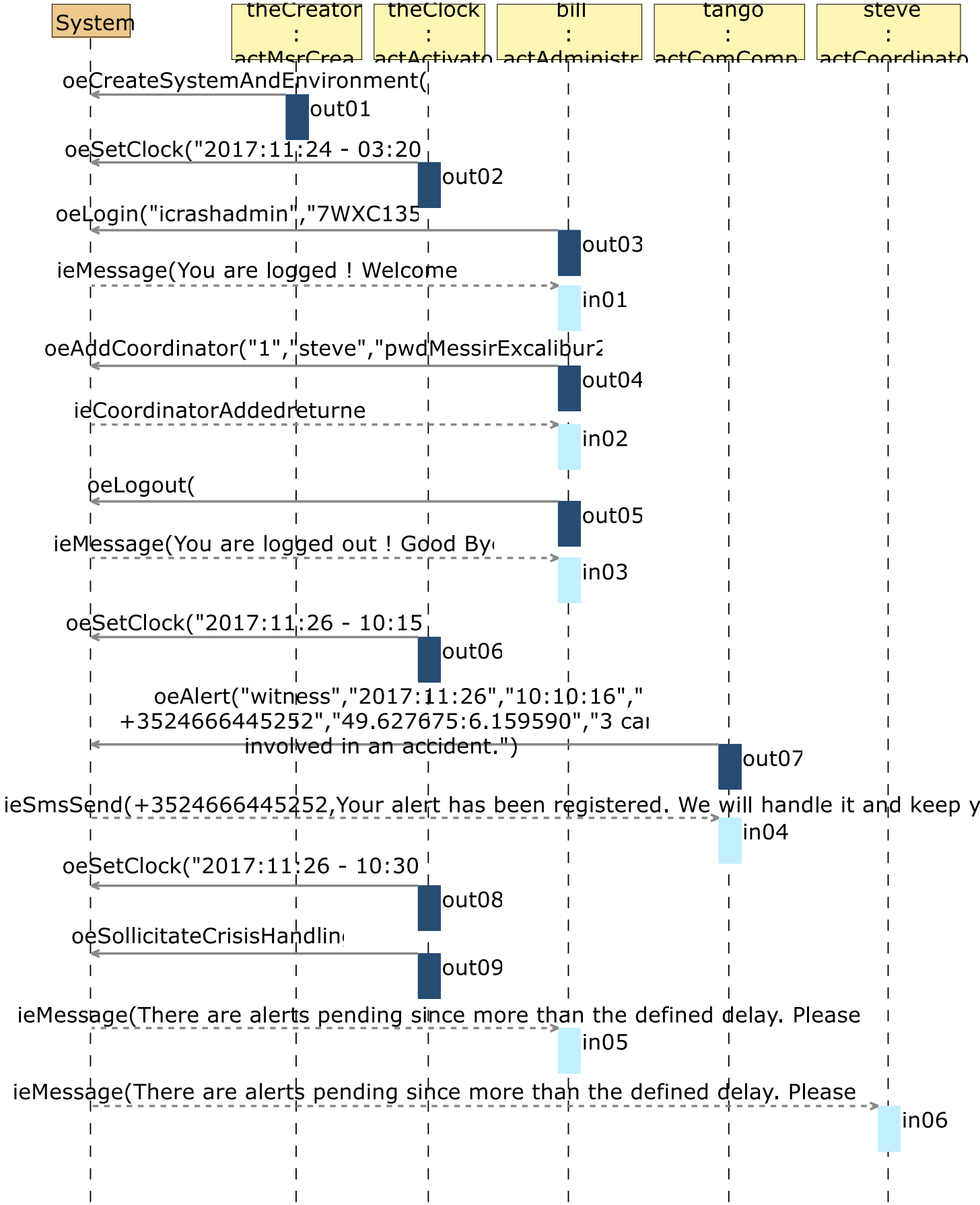

Figure 2.10: uci-suDeployAndRun-uciSimpleAndComplete-Part01

case instance for the summary use case `suDeployAndRun`.

2.3.2.4 Use-Case Instance - uciugSecurelyUseSystem:ugSecurelyUseSystem

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

Figure 2.12

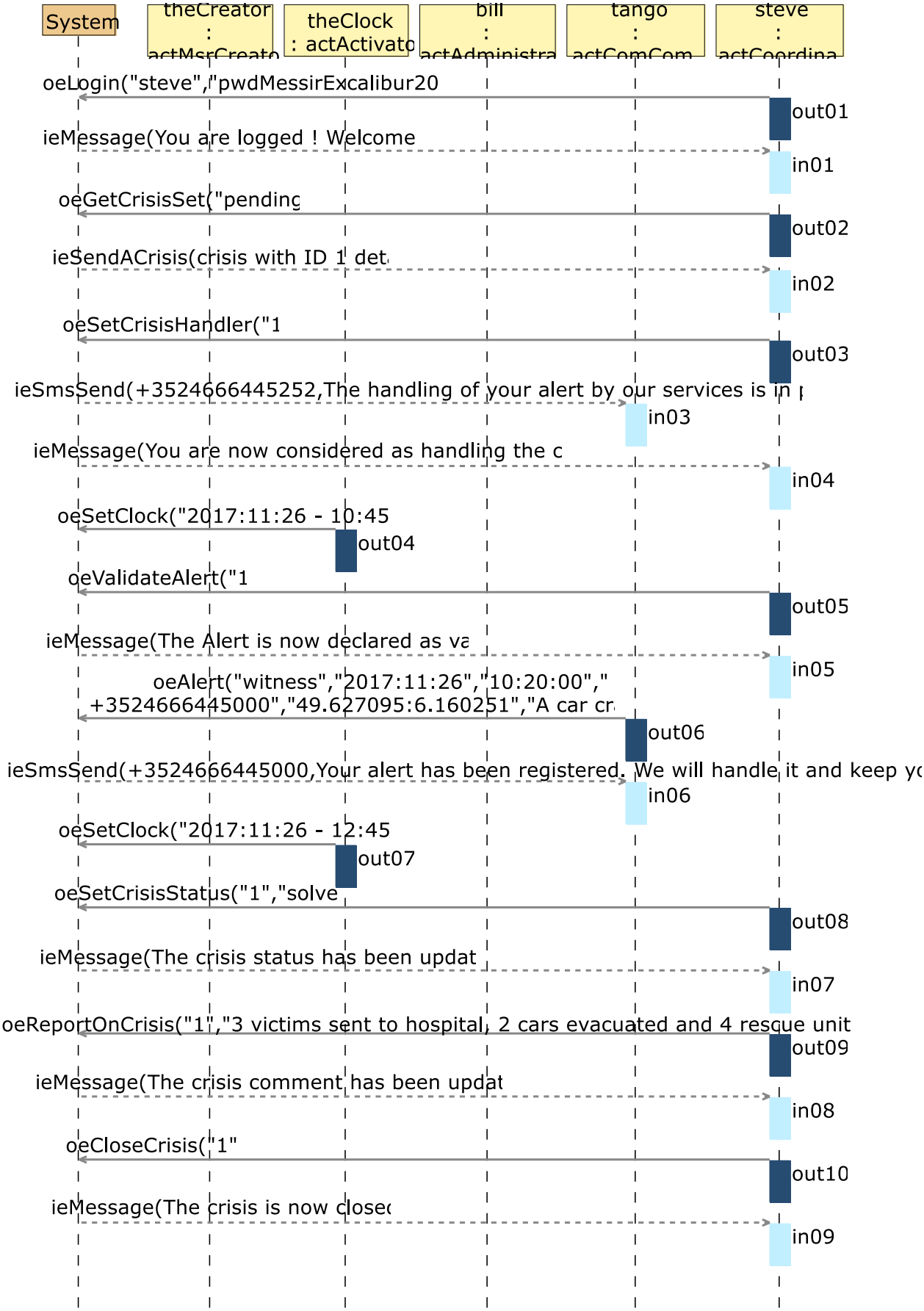

Figure 2.11: uci-suDeployAndRun-uciSimpleAndComplete-Part02 use case instance sequence diagram

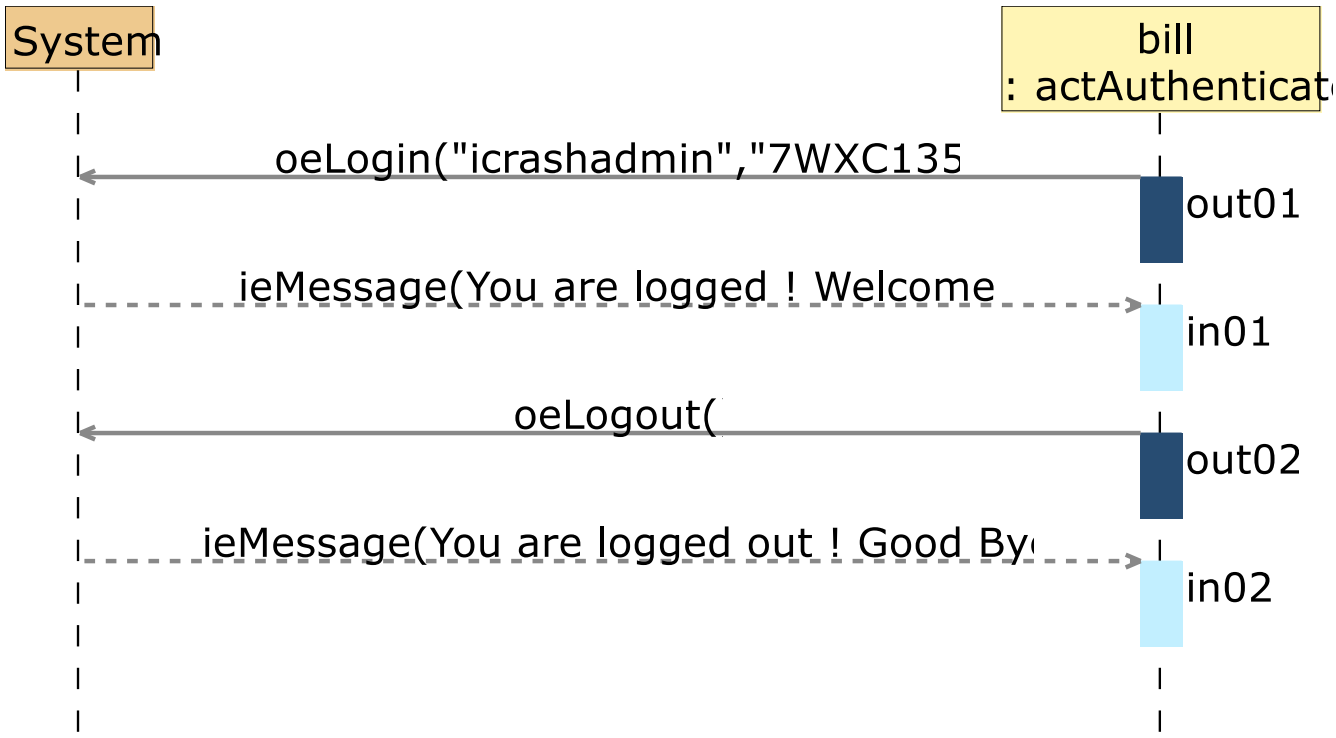
System
bill
: actAuthenticat
oeLogin("icrashadmin","7WXC135
out01
ieMessage(You are logged ! Welcome
in01
oeLogout(
out02
ieMessage(You are logged out ! Good By
in02

Figure 2.12:

Chapter 3

Environment Model

We provide below the view(s) defined for the Messir 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

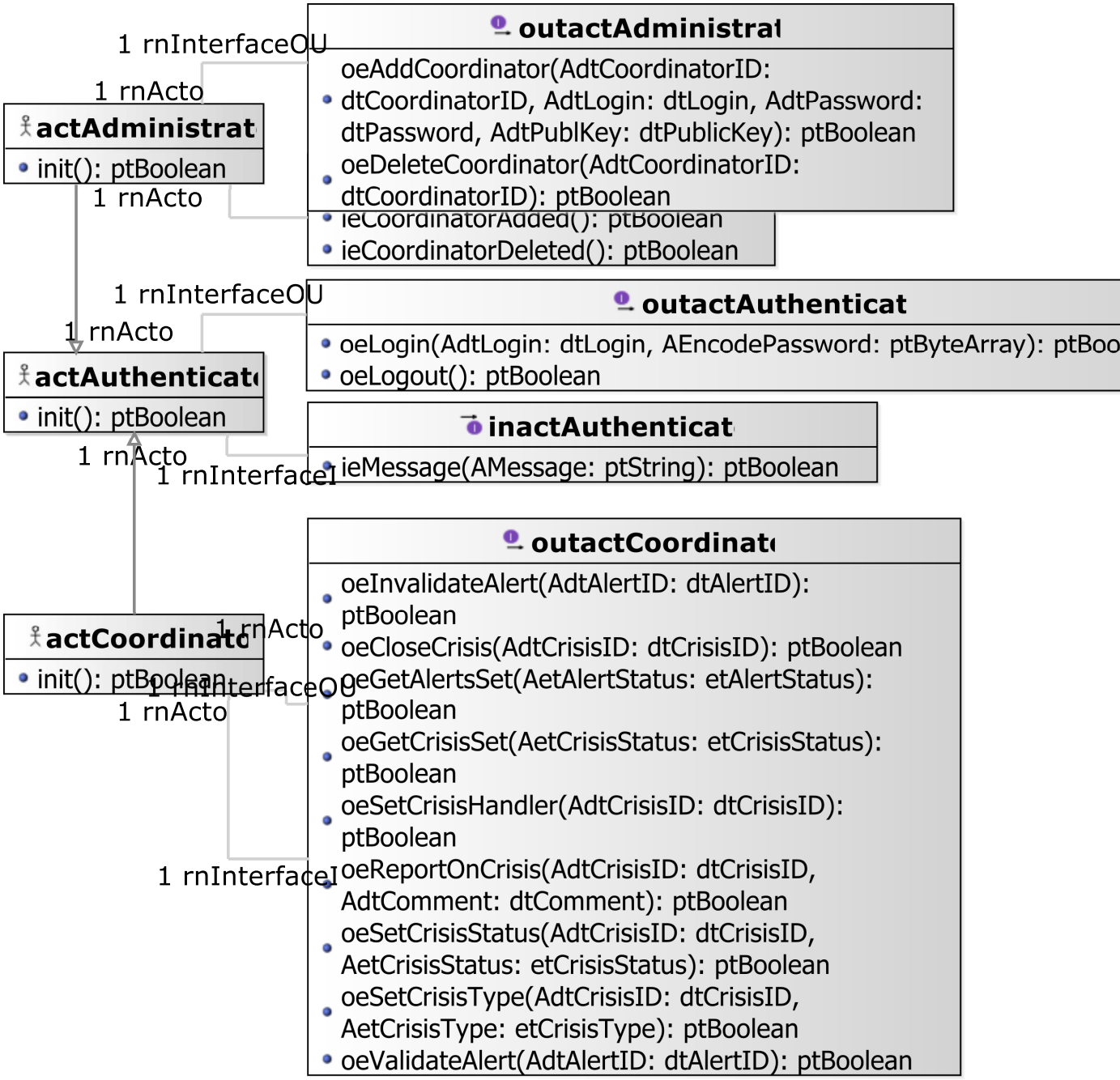

Figure 3.1: Environment Model - Local View 01. environment model local view - Part 1.

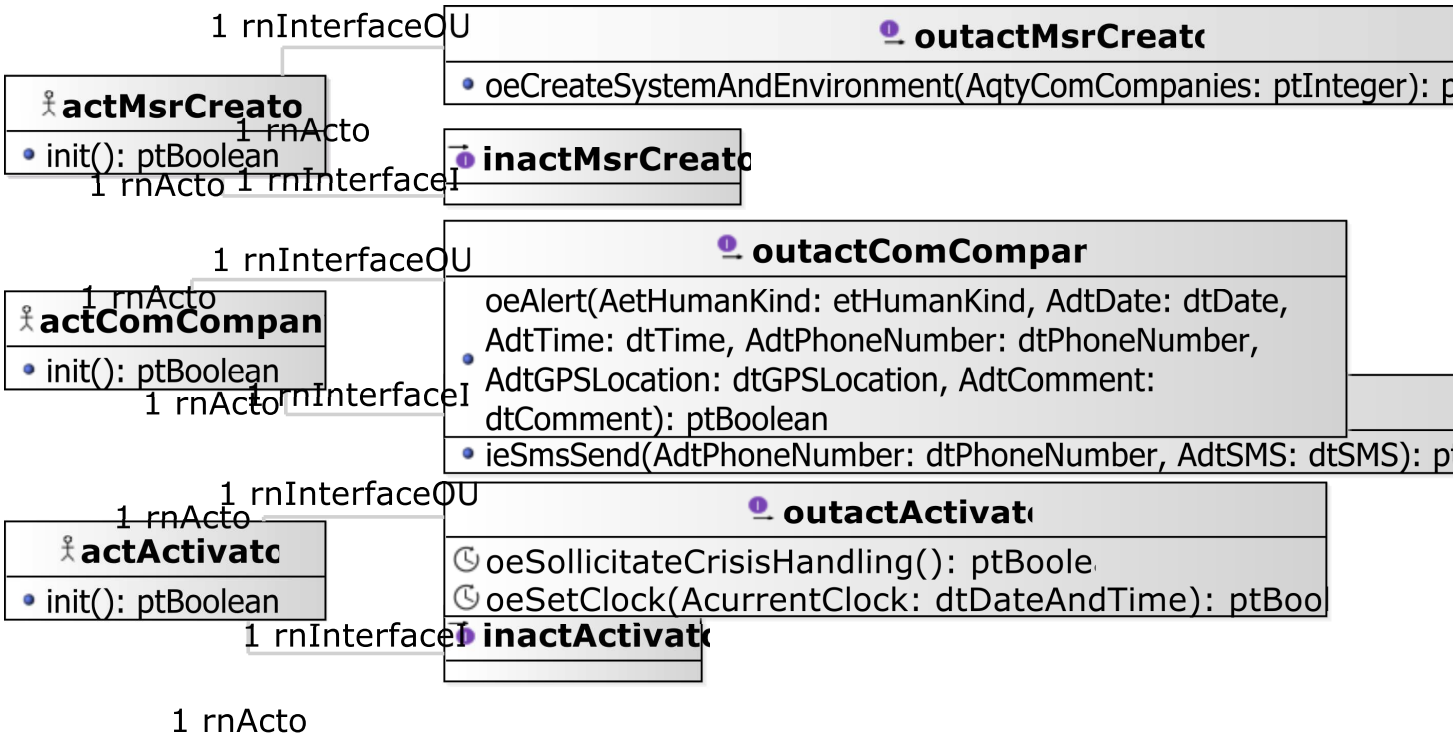

Figure 3.2: Environment Model - Local View 02. environment model local view - Part 2.

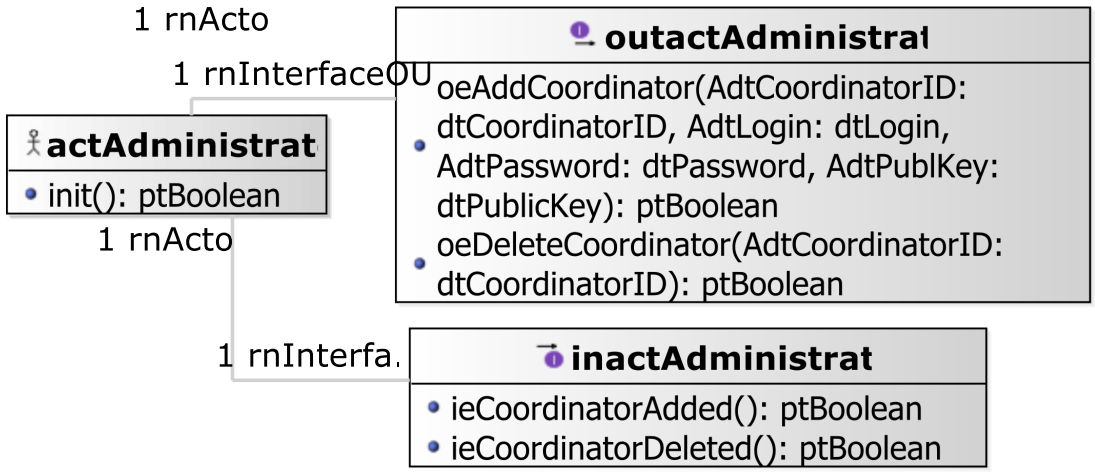

Figure 3.3: Environment Model - Local View 03. administrator actor environment model view.

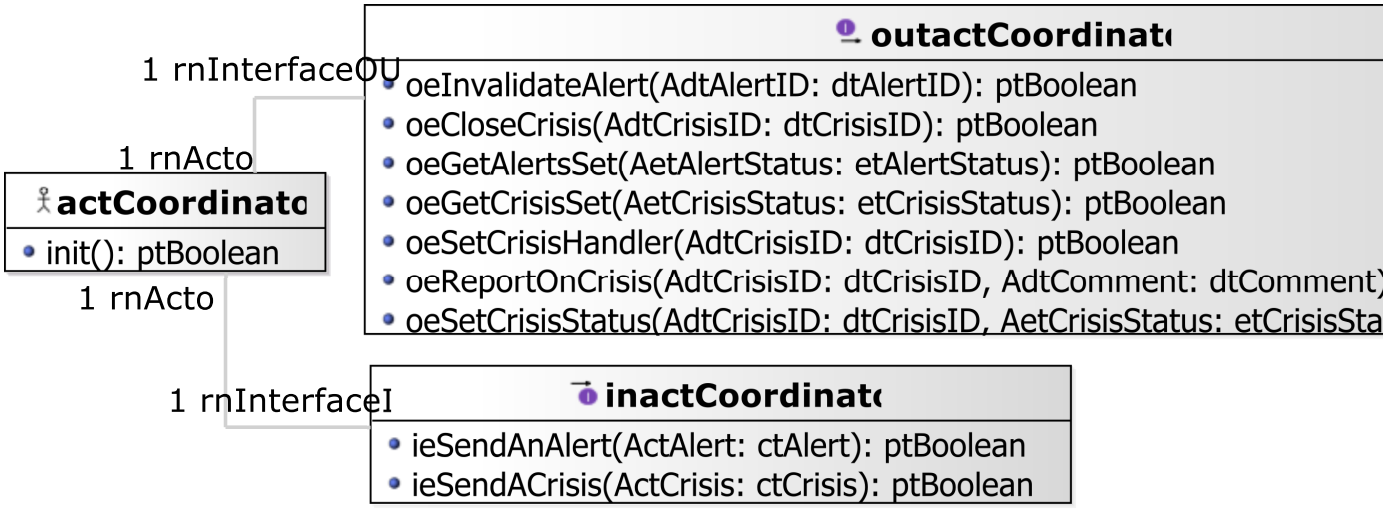

Figure 3.4: Environment Model - Local View 04. coordinator actor environment model view.

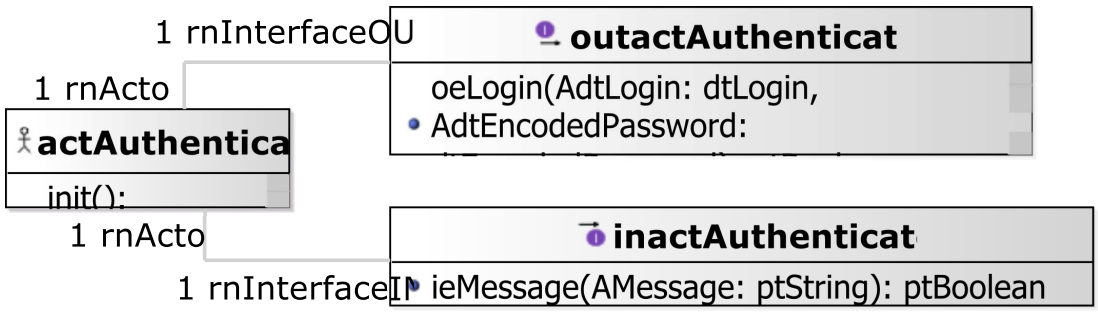

Figure 3.5: Environment Model - Local View 05. authenticated actor environment model local view.

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

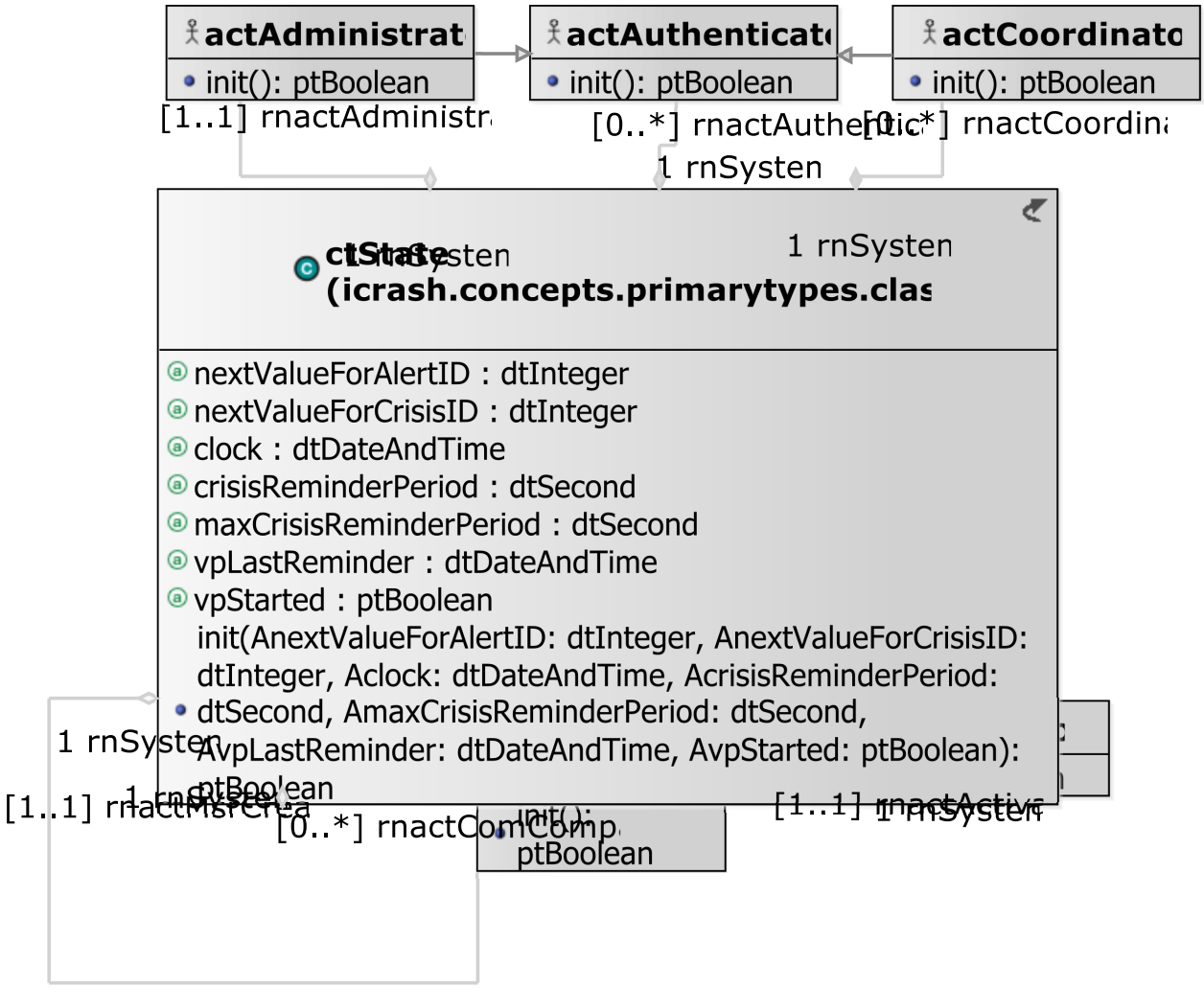

Figure 3.6: Environment Model - Global View 01. em-gv-01 environment model global view.

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, AdtPublKey:dtPublicKey):ptB** 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.
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.

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, AdtEncodedPassword:dtEncodedPassword):ptBo** 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.

continues in next page ...

. . . Actor table continuation

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.
OUT 6	**oeReportOnCrisis(AdtCrisisID:dtCrisisID, AdtComment:dtComment):ptBoolean** sent to update the textual information available for a specific handled crisis.
OUT 7	**oeSetCrisisStatus(AdtCrisisID:dtCrisisID, AetCrisisStatus:etCrisisStatus** sent to define the handling status of a specific crisis.
OUT 8	**oeSetCrisisType(AdtCrisisID:dtCrisisID, AetCrisisType:etCrisisType):ptBo** 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

continues in next page . . .

. . . Actor table continuation

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 Global view 01

Figure 4.5 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 4.6

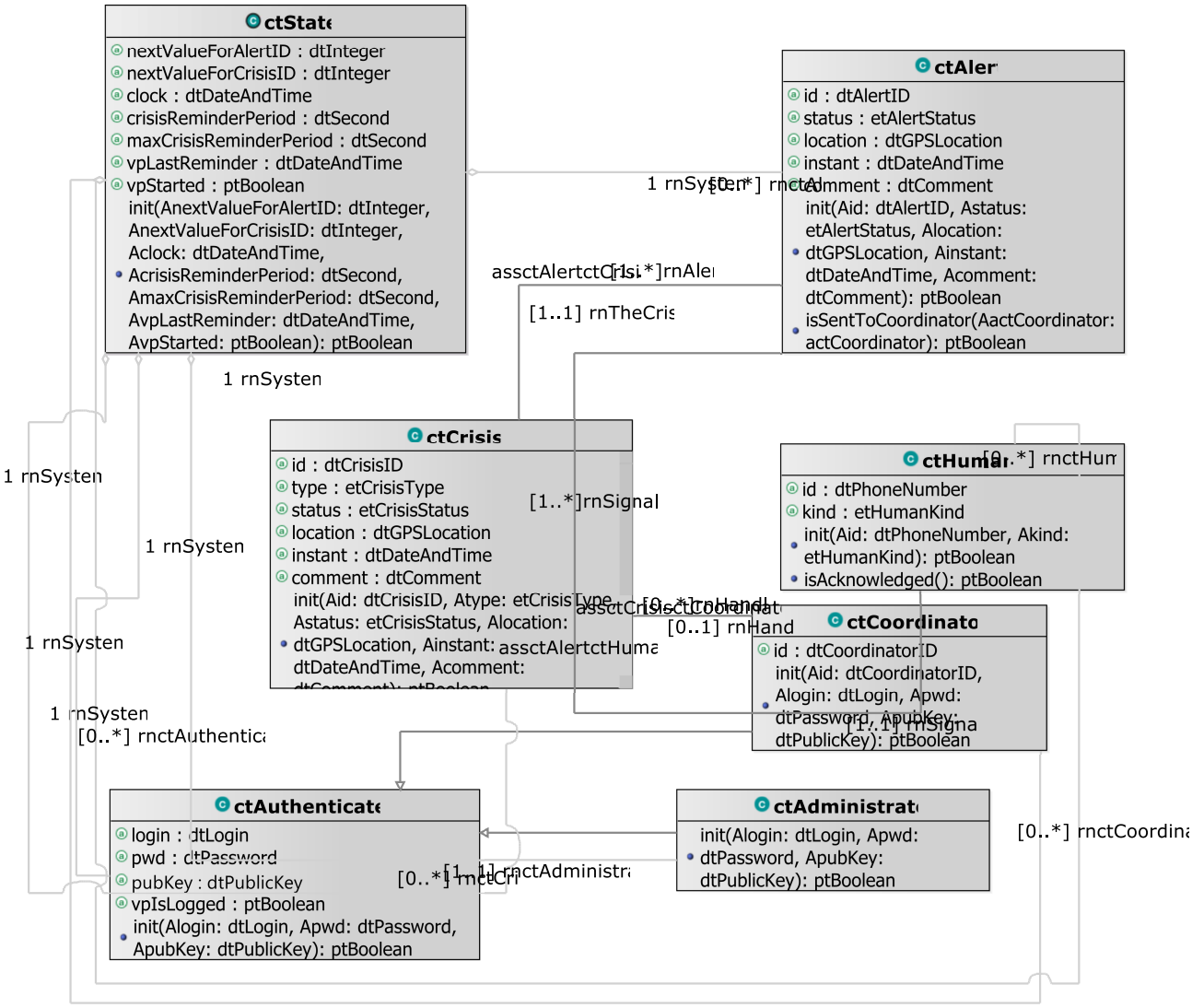

Figure 4.1: Concept Model - PrimaryTypes-Classes local view 01. Local view of all the primary types class types .

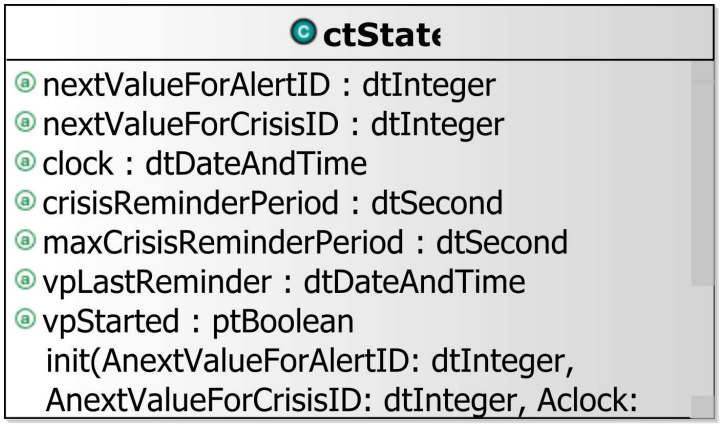

Figure 4.2: Concept Model - PrimaryTypes-Classes local view 02. local view of the ctState primary type.

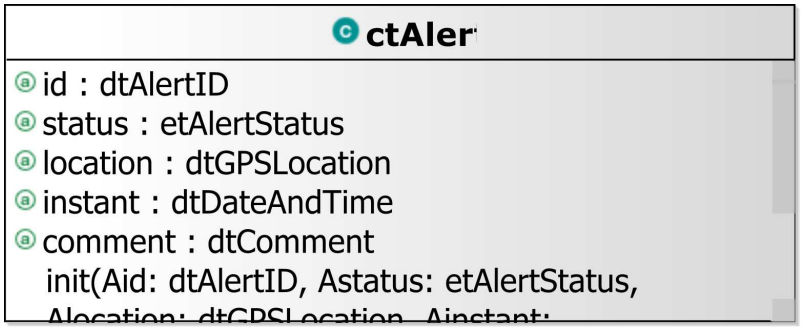

Figure 4.3: Concept Model - PrimaryTypes-Classes local view 03. local view of the ctAlert primary type.

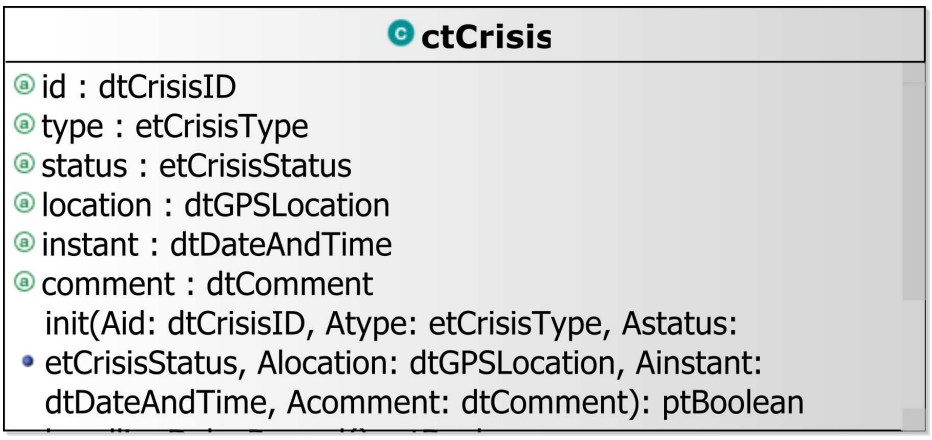

Figure 4.4: Concept Model - PrimaryTypes-Classes local view 04. local view of the ctCrisis primary type.

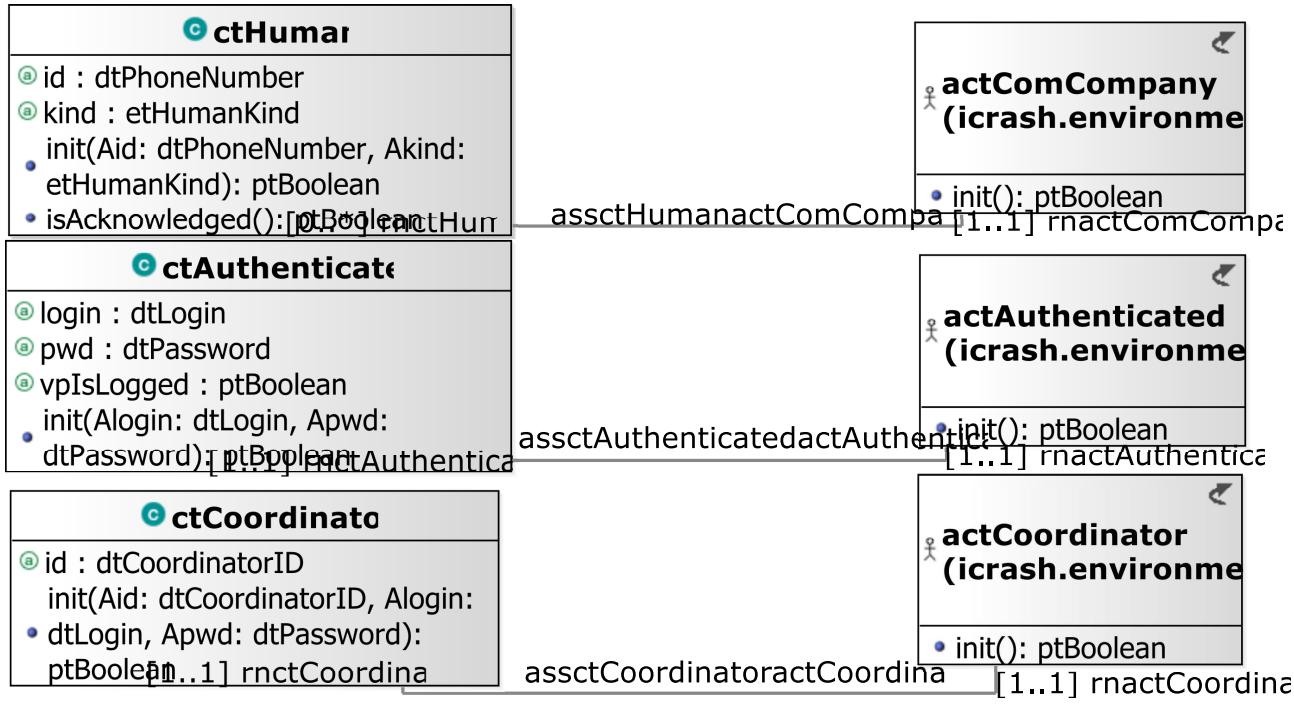

Figure 4.5: Concept Model - PrimaryTypes-Classes global view 01. Primary types class types global view - cm-pt-ct-gv-01 .

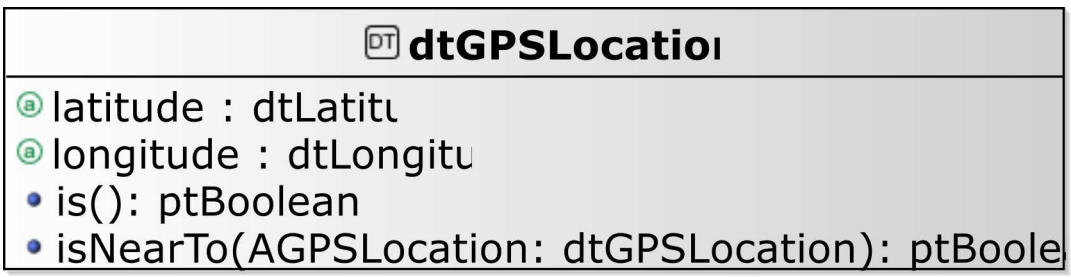

Figure 4.6: Concept Model - PrimaryTypes-Datatypes local view 06. .

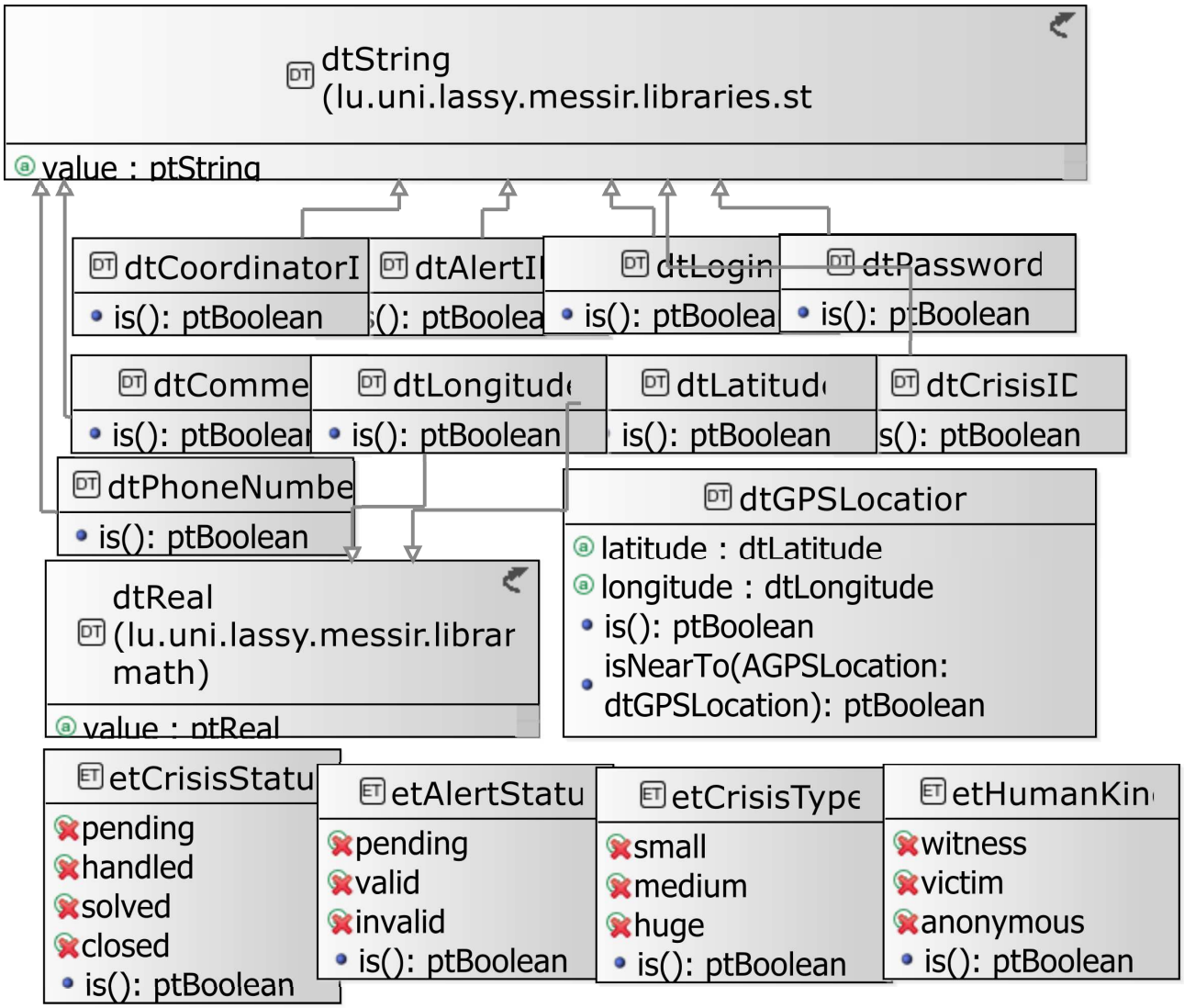

Figure 4.7: Concept Model - PrimaryTypes-Datatypes global view 01. global view of primary types datatype types - cm-pt-dt-gv-01 .

4.2.2 Global view 01

Figure 4.7 shows a global view on the *iCrash* primary types datatype types.

4.3 SecondaryTypes-Datatypes

4.3.1 Local view 01

Figure 4.8 shows the local view of the secondary types datatype types.

Figure 4.8: Concept Model - SecondaryTypes-Datatypes local view 01. 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.	
extends	icrash.concepts.primarytypes.classes.ctAuthenticated
operation	**init(Alogin:dtLogin, Apwd:dtPassword, ApubKey:dtPublicKey):ptBool**
	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, ApubKey:dtPublicKey):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.	
extends	icrash.concepts.primarytypes.classes.ctAuthenticated
attribute	**id: dtCoordinatorID** a unique identification information.
operation	**init(Aid:dtCoordinatorID, Alogin:dtLogin, Apwd:dtPassword, ApubKey:dtPublicKey):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**

continues in next page . . .

... Classes table continuation

	used to determine if the crisis stood too longly in a pending status since last reminder.
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.
ctKeyPair	
The class that stores pair of keys and allows to encode or decode a message	
attribute	**privKey: dtPrivateKey** used to encode password on a client side
attribute	**pubKey: dtPublicKey** used to decode encoded password on the server side
operation	**decodeMsg():ptBoolean** decodes encoded message
operation	**getKeys():ptBoolean** generates a pair of keys
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** `nextValueForAlertID: dtInteger`: used to associate each alert declared with a unique idenitification value.
attribute	**nextValueForCrisisID: dtInteger**

continues in next page . . .

... Classes table continuation

	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: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.
dtByteArray	
attribute	**value: ptString**
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.
dtEncodedPassword	
extends	dtByteArray
operation	**eq():ptBoolean**

continues in next page ...

... Datatypes table continuation

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	**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.
dtPassword	
a password string used to authentify an *iCrash* user	
extends	dtString
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.
dtPrivateKey	
operation	**is():ptBoolean**
dtPublicKey	
operation	**is():ptBoolean**

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.
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.
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.

Undirected 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.

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: Messir (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:

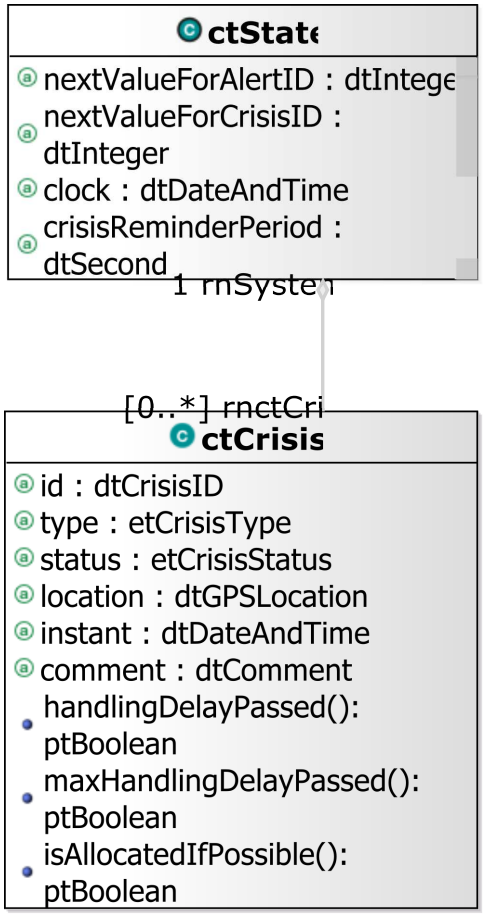

Figure 5.1: oeSollicitateCrisisHandling operation scope

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, AdtPublKey)
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 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 5.4 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 /* 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 5.4: Messir (MCL-oriented) specification of the operation *oeDeleteCoordinator*.

5.3 Environment - Out Interface Operation Scheme for actAuthenticated

5.3.1 Operation Model for oeLogin

The `oeLogin` operation has the following properties:

OPERATION
oeLogin sent to request authorization to request access secured system operations.
Parameters

continues in next page . . .

... Operation table continuation

1	**AdtLogin: dtLogin** first information used to determine accessibility rights for the actual actor.
2	**AdtEncodedPassword: dtEncodedPassword** 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)
Pre-Condition (functional)	
PreF 1	none
Post-Condition (functional)	
PostF 1	if the login and encodede 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 environement are notified of an intrusion temptative.
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)

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: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
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  let AptStringMessageForTheactAdministrator:ptString in
22
23  self.rnActor.rnSystem = TheSystem
24  and self.rnActor = TheactAuthenticated
25
26  and /* PostF01 */
27    if (TheactAuthenticated.rnctAuthenticated.pwd
28      = TheSystem.ctKeyPairs.decodeMsg()
29      and TheactAuthenticated.rnctAuthenticated.login
30        = AdtLogin
```

```
31        )
32      then (AptStringMessageForTheactAuthenticated.eq('You are logged ! Welcome ...')
33         and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
34         )
35      else (AptStringMessageForTheactAuthenticated
36         .eq('Wrong identification information ! Please try again ...')
37         and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
38         and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
39         and TheSystem.rnactAdministrator
40           .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
41         )
42      endif}
43
44  /* Post Protocol:*/
45  postP{ let TheSystem: ctState in
46   let TheactAuthenticated:actAuthenticated in
47
48   self.rnActor.rnSystem = TheSystem
49   and self.rnActor = TheactAuthenticated
50  /* PostP01 */
51   if (TheactAuthenticated.rnctAuthenticated.pwd = ctKeyPairs.decodeMsg(AEncodedPassword)
52     and TheactAuthenticated.rnctAuthenticated.login = AdtLogin
53     )
54   then (TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = true)
55   else true
56   endif}
```

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

5.3.2 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 5.6 provides the Messir (MCL-oriented) specification of the operation.

```
1
```

```
/* Pre Protocol:*/
preP{let TheSystem: ctState in
 let TheActor:actAdministrator in
 self.rnActor.rnSystem = TheSystem
 and self.rnActor = TheActor

/* PreP01 */
 and TheSystem.vpStarted = true
/* PreP02 */
 and TheActor.rnctAuthenticated.vpIsLogged = true}

/* Pre Functional:*/
preF{/* PreF01 */
true}

/* Post Functional:*/
postF{let TheSystem: ctState in
 let TheactAuthenticated:actAuthenticated in
 let AptStringMessageForTheactAuthenticated: ptString in

 self.rnActor.rnSystem = TheSystem
 and self.rnActor = TheactAuthenticated

 /* PostF01 */
 AptStringMessageForTheactAuthenticated.eq('You are logged out ! Good Bye ...')
 and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)}

/* Post Protocol:*/
postP{ let TheSystem: ctState in
 let TheactAuthenticated:actAuthenticated in

 self.rnActor.rnSystem = TheSystem
 and self.rnActor = TheactAuthenticated.asSet
/* PostP01 */
 TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = false}
```

Listing 5.6: 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**

continues in next page . . .

...Operation table continuation

	the GPS position of the phone at the date and time the message was sent.
6	**AdtComment: dtComment** 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 5.7 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)
```

```
      and TheSystem.clock.time.gt(AdtTime)
     )
   )}

/* Post Functional:*/
postF{let TheSystem: ctState in

 let ActHuman:ctHuman in
 let TheactComCompany:actComCompany in
 let ActAlert:ctAlert in
 let AAlertInstant:dtDateAndTime in
 let AetAlertStatus:etAlertStatus in
 let ActAlertNearBy:ctAlert in
 let ActCrisis:ctCrisis in
 let AdtCrisisID:dtCrisisID in
 let AetCrisisType:etCrisisType in
 let AetCrisisStatus:etCrisisStatus in
 let ACrisisInstant:dtDateAndTime in
 let ACrisisdtComment:dtComment in
 let AptStringMessage:ptString in
 let AdtSMS:dtSMS in
 let AdtAlertID:dtAlertID in

 self.rnActor.rnSystem = TheSystem
 and self.rnActor = TheactComCompany
/* PostF01 */
 TheSystem.nextValueForAlertID=PrenextValueForAlertID
 and PrenextValueForAlertID.add(1) = PostnextValueForAlertID
 and TheSystem@post.nextValueForAlertID = PostnextValueForAlertID

 /* PostF02 */
and AAlertInstant.date=AdtDate
and AAlertInstant.time=AdtTime

and AetAlertStatus=pending

and TheSystem.nextValueForAlertID.todtString().eq(AdtAlertID)

and ActAlert.init(AdtAlertID,
         AetAlertStatus,
         AdtGPSLocation,
         AAlertInstant,
         AdtComment)

 /* PostF03 */
and TheSystem.rnctAlert.select(location.isNearTo(AdtGPSLocation)) = ColctAlertsNearBy
and if (ColctAlertsNearBy->size()=0)
  then (TheSystem.nextValueForCrisisID = PrenextValueForCrisisID
     and PrenextValueForCrisisID.add(1) = PostnextValueForCrisisID
     and TheSystem@post.nextValueForCrisisID = PostnextValueForCrisisID
     and TheSystem.nextValueForCrisisID.todtString().eq(AdtCrisisID)
     and AdtCrisisType = small
     and AetCrisisStatus = pending
     and ACrisisInstant= AAlertInstant
     and ACrisisdtComment = 'no reporting yet defined'
     and ActCrisis.init( AdtCrisisID,
               AdtCrisisType,
               AetCrisisStatus,
               AdtGPSLocation,
               ACrisisInstant,
               ACrisisdtComment)
     )
 else (ColctAlertsNearBy.rnTheCrisis->msrAny(true) = ActCrisis)
 endif

 /* PostF04 */
and ActAlert@post.rnTheCrisis = ActCrisis

/* PostF05 */
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 5.7: Messir (MCL-oriented) specification of the operation *oeAlert*.

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

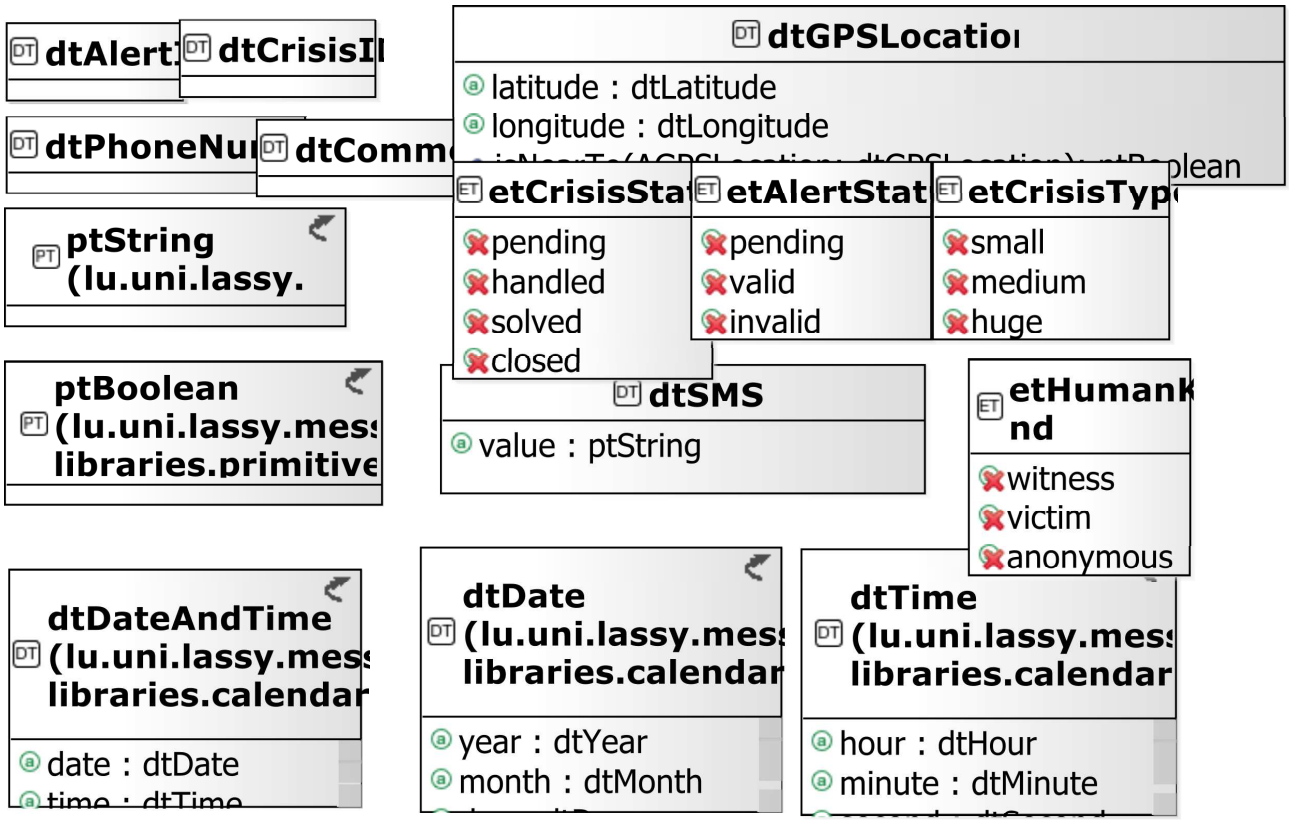

Figure 5.2: oeAlert operation scope

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

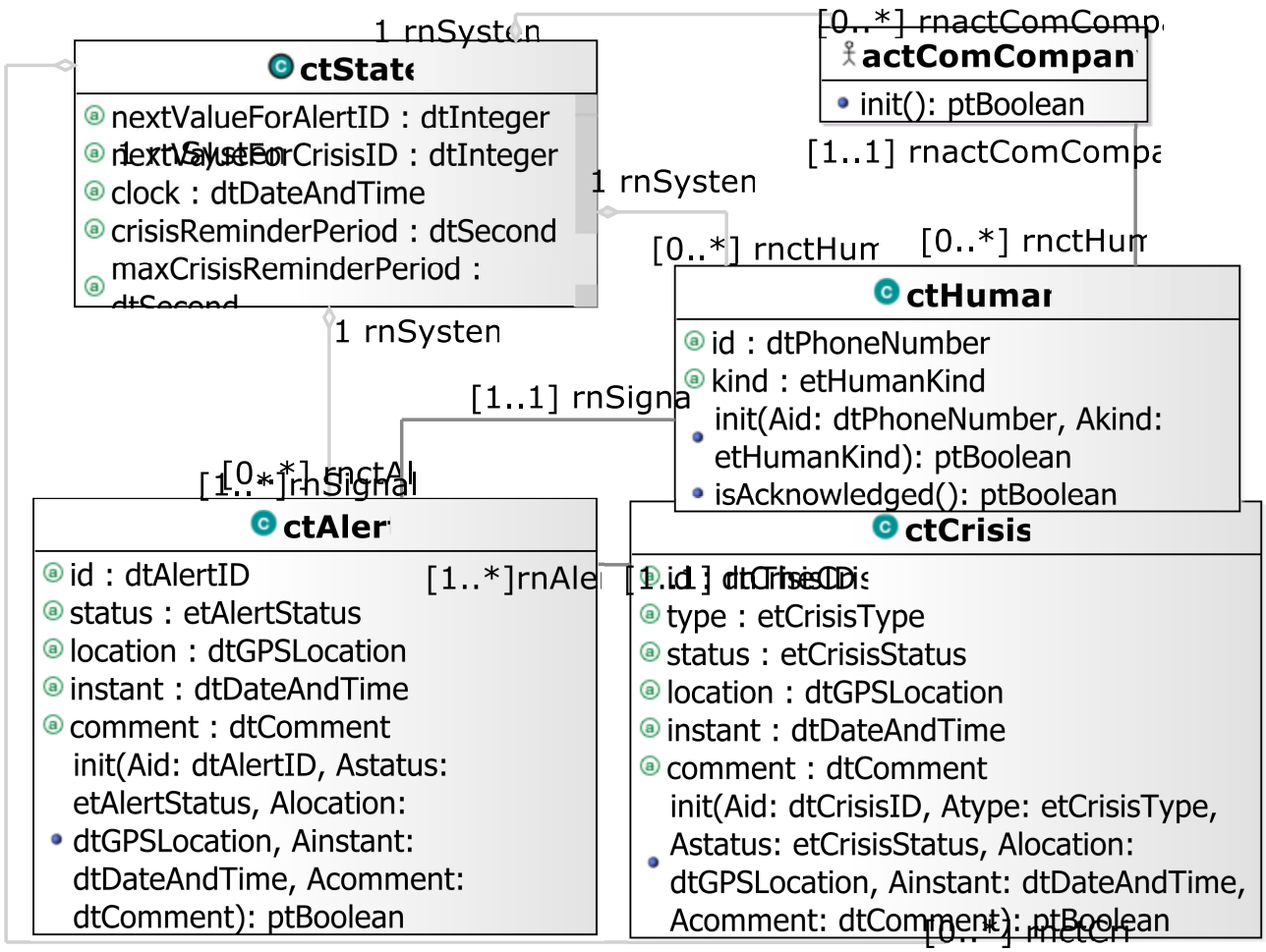

Figure 5.3: oeAlert operation scope

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 actor is informed about the satisfaction of its request.
Post-Condition (protocol)	
PostP 1	none

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)	

continues in next page . . .

... Operation table continuation

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 for the actor sending the message `ieSendACrisis`. (cf. specification of `isSentToCoordinator` predicate given for the `ctCrisis` type.
Post-Condition (protocol)	
PostP 1	none

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	

continues in next page ...

. . . Operation table continuation

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
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.

continues in next page . . .

... Operation table continuation

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.

continues in next page ...

. . . Operation table continuation

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)	
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

continues in next page . . .

... Operation table continuation

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)	
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'.

continues in next page ...

...Operation table continuation

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 5.8 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  let ApublicKey:dtPublicKey in
19
20  /* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
21  AnextValueForAlertID.value.eq(1)
22  and AnextValueForCrisisID.value.eq(1)
23  and Aclock.date.year.value = 1970
24  and Aclock.date.month.value = 01
25  and Aclock.date.day.value = 01
26  and Aclock.time.hour.value = 00
27  and Aclock.time.minute.value = 00
28  and Aclock.time.second.value = 00
29
30  and AcrisisReminderPeriod.value.eq(300)
31  and AmaxCrisisReminderPeriod.value.eq(1200)
32  and AvpStarted = true
33  and TheSystem.init(AnextValueForAlertID,
34             AnextValueForCrisisID,
35             Aclock,
36             AcrisisReminderPeriod,
37             AmaxCrisisReminderPeriod,
38             Aclock,
39             AvpStarted
40            )
41  /* PostF02*/
42  and AactMsrCreator.init()
43  /* PostF03 */
44  and let AactComCompanyCol: Bag(actComCompany) in
45  AactComCompanyCol->size() = AqtyComCompanies
46  AactComCompanyCol-> forAll(init())
47  /* PostF04*/
48  and AactAdministrator.init()
49  and TheSystem.ctKeyPair.initForDecoding(ApublicKey, '[B@6979e8cb')
50  /* PostF05*/
51  and let AactActivator:actActivator in
```

```
52  AactActivator.init()
53 /* PostF06 */
54  and let ActAdministrator:ctAdministrator in
55    let AdtLogin:dtLogin in
56    let AdtEncodedPassword:dtEncodedPassword in
57    AdtLogin.value.eq('icrashadmin')
58    and AdtEncodedPassword.value.eq(TheSystem.ctKeyPair.decodeMsg())
59    and ActAdministrator.init(AdtLogin,AdtPassword)
60  /* PostF07*/
61  and ActAdministrator@post.rnactAuthenticated = AactAdministrator}
62
63 /* Post Protocol:*/
64 postP{ true}
```

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

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

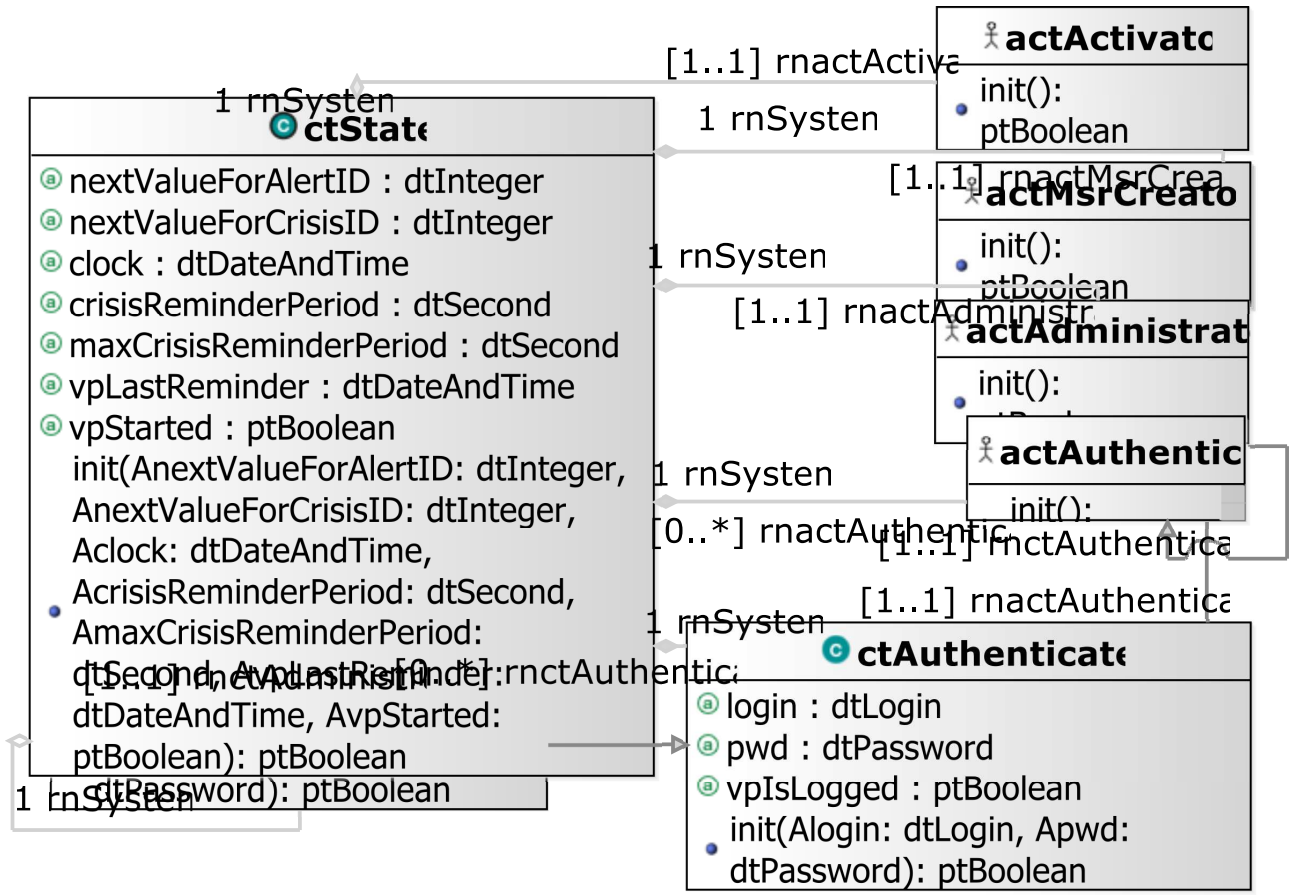

Figure 5.4: oeCreateSystemAndEnvironment operation scope

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

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 5.9 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.pubKey = ApublKey
10 and Self.vpIsLogged = false
11
12 /* Post F02 */
13 and (Self.oclIsNew and self = Self)
14 )
15 then (result = true)
16 else (result = false)
17 endif}
```

Listing 5.9: 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.	
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 5.10 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 5.10: 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	
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 5.11 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 5.11: 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	
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
4	**ApubKey: dtPublicKey** used to initialize the public key 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 5.12 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.pubKey = ApubKey
11 and Self.vpIsLogged = false
12 /* Post F02 */
13 and (Self.oclIsNew and self = Self)
14 )
15 then (result = true)
16 else (result = false)
17 endif}
```

Listing 5.12: 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
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 5.13 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 5.13: Messir (MCL-oriented) specification of the operation *init*.

5.12.2 Operation Model for handlingDelayPassed

The `handlingDelayPassed` operation has the following properties:

Operation

continues in next page . . .

... Operation table continuation

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 5.14 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 5.14: 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 5.15 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheSystem:ctState in
```

```
let CurrentClockSecondsQty:dtInteger in
let CrisisInstantSecondsQty:dtInteger in
let MaxCrisisReminderPeriod:dtSecond in
if
( /* Post F01 */
 self.rnSystem = TheSystem
 and self.status = pending
 and TheSystem.clock.toSecondsQty() = CurrentClockSecondsQty
 and Self.instant.toSecondsQty() = CrisisInstantSecondsQty
 and TheSystem.maxCrisisReminderPeriod = MaxCrisisReminderPeriod
 and CurrentClockSecondsQty.sub(CrisisInstantSecondsQty)
              .gt(MaxCrisisReminderPeriod)
)
then (result = true)
else (result = false)
endif}
```

Listing 5.15: 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 5.16 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{if
(
/* Post F01 */
 AactCoordinator.rnInterfaceIN.ieSendACrisis(self)
)
then (result = true)
else (result = false)
endif}
```

Listing 5.16: 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
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 5.17 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 5.17: 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

continues in next page . . .

... Operation table continuation

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	
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 5.18 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 5.18: 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 ctKeyPair

5.14.1 Operation Model for intiForEncoding

The `intiForEncoding` operation has the following properties:

OPERATION
intiForEncoding
Parameters
1 **AprivKey: dtPrivateKey**
2 **AdecodedPwd: ptString**
Return type
ptBoolean
Post-Condition (functional)
PostF 1

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

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

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

5.14.2 Operation Model for initForDecoding

The `initForDecoding` operation has the following properties:

OPERATION
initForDecoding
Parameters
1 **ApubKey: dtPublicKey**
2 **AencodedPwd: dtEncodedPassword**

continues in next page . . .

... Operation table continuation

Return type
ptBoolean
Post-Condition (functional)
PostF 1

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

```
1
2  /* Post Functional:*/
3  postF{if
4  (
5  let Self:ctKeyPair in
6  /* Post F01 */
7  Self.pubKey = ApubKey
8  and Self.encodedPwd = AencodedPwd
9
10 /* Post F02 */
11 and (Self.oclIsNew and self = Self)
12 )
13 then (result = true)
14 else (result = false)
15 endif}
```

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

5.14.3 Operation Model for getKeys

The `getKeys` operation has the following properties:

OPERATION
getKeys
Return type
ptBoolean
Post-Condition (functional)
PostF 1

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

```
1
2  /* Post Functional:*/
3  postF{let ThePublicKey:dtPublicKey in
4  let ThePrivateKey:dtPrivateKey in
5  if
6  ( /* Post F01 */
7  self.pubKey = ThePublicKey
8  and self.privKey = ThePrivateKey
9  )
10 then (result = true)
11 else (result = false)
```

```
12 endif}
```

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

5.14.4 Operation Model for encodeMsg

The `encodeMsg` operation has the following properties:

OPERATION
encodeMsg
Return type
ptBoolean
Post-Condition (functional)
PostF 1

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

```
1
2  /* Post Functional:*/
3  postF{let ThePrivateKey:dtPrivateKey in
4  let ThePassword:dtPassword in
5  if
6  ( /* Post F01 */
7  self.pubKey = ThePublicKey
8  and self.privKey = ThePrivateKey
9  )
10 then (result = true)
11 else (result = false)
12 endif}
```

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

5.14.5 Operation Model for decodeMsg

The `decodeMsg` operation has the following properties:

OPERATION
decodeMsg
Return type
ptBoolean
Post-Condition (functional)
PostF 1

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

```
1
```

```
/* Post Functional:*/
postF{let ThePublicKey:dtPublicKey in
let ThePrivateKey:dtPrivateKey in
if
( /* Post F01 */
self.pubKey = ThePublicKey
and self.privKey = ThePrivateKey
)
then (result = true)
else (result = false)
endif}
```

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

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

```

/* Post Functional:*/
postF{if
(
/* Post F01 */
```

```
let Self:ctState in

Self.nextValueForAlertID = AnextValueForAlertID
and Self.nextValueForCrisisID = AnextValueForCrisisID
and Self.clock = Aclock
and Self.crisisReminderPeriod = AcrisisReminderPeriod
and Self.maxCrisisReminderPeriod = AmaxCrisisReminderPeriod
and Self.vpLastReminder = AvpLastReminder
and Self.vpStarted = AvpStarted

and (Self.oclIsNew and self = Self)
)
then (result = true)
else (result = false)
endif}
```

Listing 5.24: 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.	
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 5.25 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(20)
   )
   then (TheResult = true)
   else (TheResult = false)
   endif
   result = TheResult
  )}
```

Listing 5.25: 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 5.26 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 5.26: 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 5.27 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 5.27: 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	
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 5.28 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(10)
7       )
8       then (TheResult = true)
9       else (TheResult = false)
10      endif
11      result = TheResult
12     )}
```

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

5.20 Primary Types - Operation Schemes for Datatype dtGPSLocation

5.20.1 Operation Model for is

The `is` operation has the following properties:

OPERATION

continues in next page . . .

... Operation table continuation

is
used to determine which 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 5.29 provides the Messir (MCL-oriented) specification of the operation.

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

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

5.20.2 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 5.30 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in true
4      let EarthRadius: dtReal in
5      let MaxDistance: dtReal in
6      let ComparedLatitude: dtLatitude in
7      let ComparedLongitude: dtLongitude in
8      let R1: dtReal in let R1a: dtReal in
9      let R2: dtReal in let R2a: dtReal in
10
11     ( if
12      ( EarthRadius.value = 6371
13       and MaxDistance.value = 100
14
15       and AdtValue.latitude = ComparedLatitude
16       and AdtValue.longitude = ComparedLongitude
17       and Self.latitude.sin() = R1a
18       and AdtValue.latitude.sin().mul(R1a) = R1
19       and Self.latitude.cos() = R2a
20       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 5.30: Messir (MCL-oriented) specification of the operation *isNearTo*.

5.21 Primary Types - Operation Schemes for Datatype dtLatitude

5.21.1 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 5.31 provides the Messir (MCL-oriented) specification of the operation.

```
1
2  /* Post Functional:*/
3  postF{let TheResult: ptBoolean in
4      ( if
5       ( AdtValue.value.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 5.31: Messir (MCL-oriented) specification of the operation *is*.

5.22 Primary Types - Operation Schemes for Datatype dtLogin

5.22.1 Operation Model for is

The `is` operation has the following properties:

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

5.23 Primary Types - Operation Schemes for Datatype dtLongitude

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 dtLongitude.
Return type
ptBoolean
Post-Condition (functional)

continues in next page . . .

... Operation table continuation

PostF 1	is true if the value is a real in the interval [-180.0 , +180.0].

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

5.24 Primary Types - Operation Schemes for Datatype dtPassword

5.24.1 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 5.34 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 5.34: Messir (MCL-oriented) specification of the operation *is*.

5.25 Primary Types - Operation Schemes for Datatype dtPhoneNumber

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 dtPhoneNumber.
Return type
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 5.35 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 5.35: Messir (MCL-oriented) specification of the operation *is*.

5.26 Primary Types - Operation Schemes for Enumeration etAlertStatus

5.26.1 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 5.36 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let TheResult: ptBoolean in
    ( if
     ( self = pending
      or self = valid
      or self = invalid
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )}
```

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

5.27 Primary Types - Operation Schemes for Enumeration etCrisisStatus

5.27.1 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 5.37 provides the Messir (MCL-oriented) specification of the operation.

```

/* Post Functional:*/
postF{let TheResult: ptBoolean in
    ( if
     ( self = pending
      or self = handled
      or self = solved
      or self = closed
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )}
```

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

5.28 Primary Types - Operation Schemes for Enumeration etCrisisType

5.28.1 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 5.38 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 5.38: Messir (MCL-oriented) specification of the operation *is*.

5.29 Primary Types - Operation Schemes for Enumeration etHumanKind

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: `witness,victim,anonym`

The listing 5.39 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 5.39: Messir (MCL-oriented) specification of the operation *is*.

5.30 Secondary Types - Operation Schemes for Classes

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

5.31 Secondary Types - Operation Schemes for Datatype dtSMS

5.31.1 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 5.40 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 5.40: Messir (MCL-oriented) specification of the operation *is*.

5.32 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.oeCreat

The `testcase01-ts01oeCreateSystemAndEnvironment-actMsrCreator.outactMsrCreator.oeCre` 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

```
}

constraints{
 TheActor=TheSystem.rnactActivator->any2(true)
 ACurrentClock.date.year.value = 2017
 ACurrentClock.date.month.value = 11
 ACurrentClock.date.day.value = 24
 ACurrentClock.time.hour.value = 15
 ACurrentClock.time.minute.value = 20
 ACurrentClock.time.second.value = 00
}

oracle{
 constraints{
  true
 }
}
```

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, AdtEncodedPassword)**
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   AdtEncodedPassword:dtEncodedPassword
6  }
7
8  constraints{
9   TheActor=TheSystem.rnactAdministrator->any2(true)
10  AdtLogin.value.eq('icrashadmin')
11  AdtEncodedPassword.value.eq('[B@6979e8cb')
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.oeAddCoord

The `testcase01-ts04oeAddCoordinator-actAdministrator.outactAdministrator.oeAddC` 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, AdtPublicKey)**
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	

continues in next page . . .

. . . Test Step table continuation

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
 AdtPublicKey:dtPublicKey
}

constraints{
 TheActor = TheSystem.rnactAdministrator->any2(true)
 AdtCoordinatorID.value.eq('1')
 AdtLogin.value.eq('steve')
 AdtPassword.value.eq('pwdMessirExcalibur2017')
 AdtPublicKey.value.eq('Sun RSA public key, 2048 bits ...')
}

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.

continues in next page . . .

... Test Step table continuation

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.

```
1
2  variables{
3   TheActor : actAdministrator
4  }
5
6  constraints{
7   TheActor = TheSystem.rnactAdministrator->any2(true)
8  }
9
10 oracle{
11  variables{
12   AMessage:ptString
13  }
14  constraints{
15   AMessage = 'You are logged out ! Good Bye ...'
16   TheActor.inactAdministrator.ieMessage(AMessage)
17  }
18 }
```

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.

continues in next page ...

. . . Test Step table continuation

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.

```
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 = 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

continues in next page . . .

... Test Step table continuation

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'.
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.

```

variables{
 TheActor : actComCompany
 AetHumanKind:etHumanKind
 AdtDate:dtDate
 AdtTime:dtTime
 AdtPhoneNumber:dtPhoneNumber
 AdtGPSLocation:dtGPSLocation
 AdtComment:dtComment
}

constraints{
 TheActor = TheSystem.rnactComCompany->any2(true)
 AetHumanKind = witness
 AdtDate.year.value = 2017
 AdtDate.month.value = 11
 AdtDate.day.value = 26
 AdtTime.hour.value = 10
 AdtTime.minute.value = 10
 AdtTime.second.value = 16
 AdtPhoneNumber.value = '+3524666445252'
 AdtGPSLocation.latitude.value = 49.627675
 AdtGPSLocation.longitude.value = 6.159590
 AdtComment.value = '3 cars involved in an accident.'
}

oracle{
 variables{
  AdtSMS:dtSMS
 }
 constraints{
  AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
  TheActor.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
 }
}
```

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	
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.oeSollicitateCrisisH

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

Test Step

continues in next page . . .

... Test Step table continuation

ts09oeSollicitateCrisisHandling	
test the proactive sollication to handle an alert.	
Test Sent Message	
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, AdtEncodedPassword)**
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
 AdtEncodedPassword:dtEncodedPassword
```

```
6  }
7
8  constraints{
9   TheActor = TheSystem.rnactCoordinator->select(a | a.rnctCoordinator.login.value.eq('steve'))->any2
        (true)
10  AdtLogin.value.eq('steve')
11  AdtEncodedPassword.value.eq('[B@6979e8cb')
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.oeGetCrisisSe` 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{
8   TheActor=TheSystem.rnactCoordinator
9         ->select(a | a.rnctCoordinator.login.value.eq('steve'))
10        ->any2(true)
11  AetCrisisStatus = pending
12 }
13
14 oracle{
15  variables{
16   ActCrisis:ctCrisis
17  }
18  constraints{
19   TheActor.inactCoordinator.ieSendACrisis(ActCrisis)
20  }
21 }
```

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.oeSetCrisisHa` 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**

continues in next page . . .

... Test Step table continuation

	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
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	
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 6.13 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 = 45
14  ACurrentClock.time.second.value = 00
15 }
16
17 oracle{
18  constraints{
19   true
20  }
21 }
```

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

continues in next page . . .

... Test Step table continuation

Test Sent Message	
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
```

```
 AdtPhoneNumber:dtPhoneNumber
 AdtGPSLocation:dtGPSLocation
 AdtComment:dtComment
}

constraints{
 TheActor = TheSystem.rnactComCompany->any2(true)
 AetHumanKind = witness
 AdtDate.year.value = 2017
 AdtDate.month.value = 11
 AdtDate.day.value = 26
 AdtTime.hour.value = 10
 AdtTime.minute.value = 20
 AdtTime.second.value = 00
 AdtPhoneNumber.value = '+3524666445000'
 AdtGPSLocation.latitude.value = 49.627095
 AdtGPSLocation.longitude.value = 6.160251
 AdtComment.value = 'A car crash just happened.'
}

oracle{
 variables{
  AdtSMS:dtSMS
 }
 constraints{
  AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
  TheActor.actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
 }
}
```

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.

```
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 = 12
13  ACurrentClock.time.minute.value = 45
14  ACurrentClock.time.second.value = 00
15 }
16
17 oracle{
18  constraints{
19   true
20  }
21 }
```

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.oeSetCrisisSta` 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.oeReportOnCris

The `testcase01-ts18oeReportOnCrisis-actCoordinator.outactCoordinator.oeReportOnC` 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* .

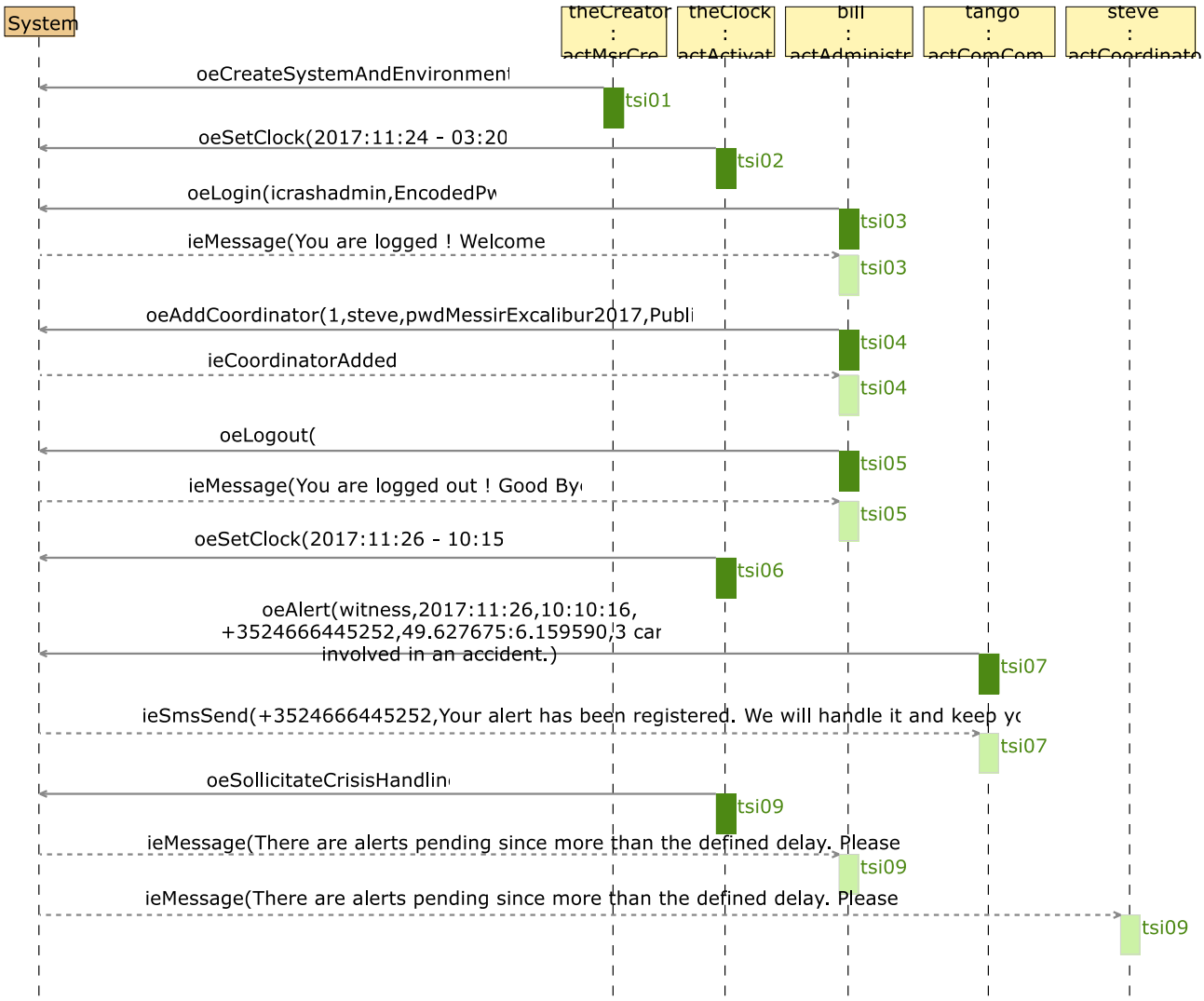

Figure 6.1: tci-testcase01-instance01-Part01 testcase instance sequence diagram

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* .

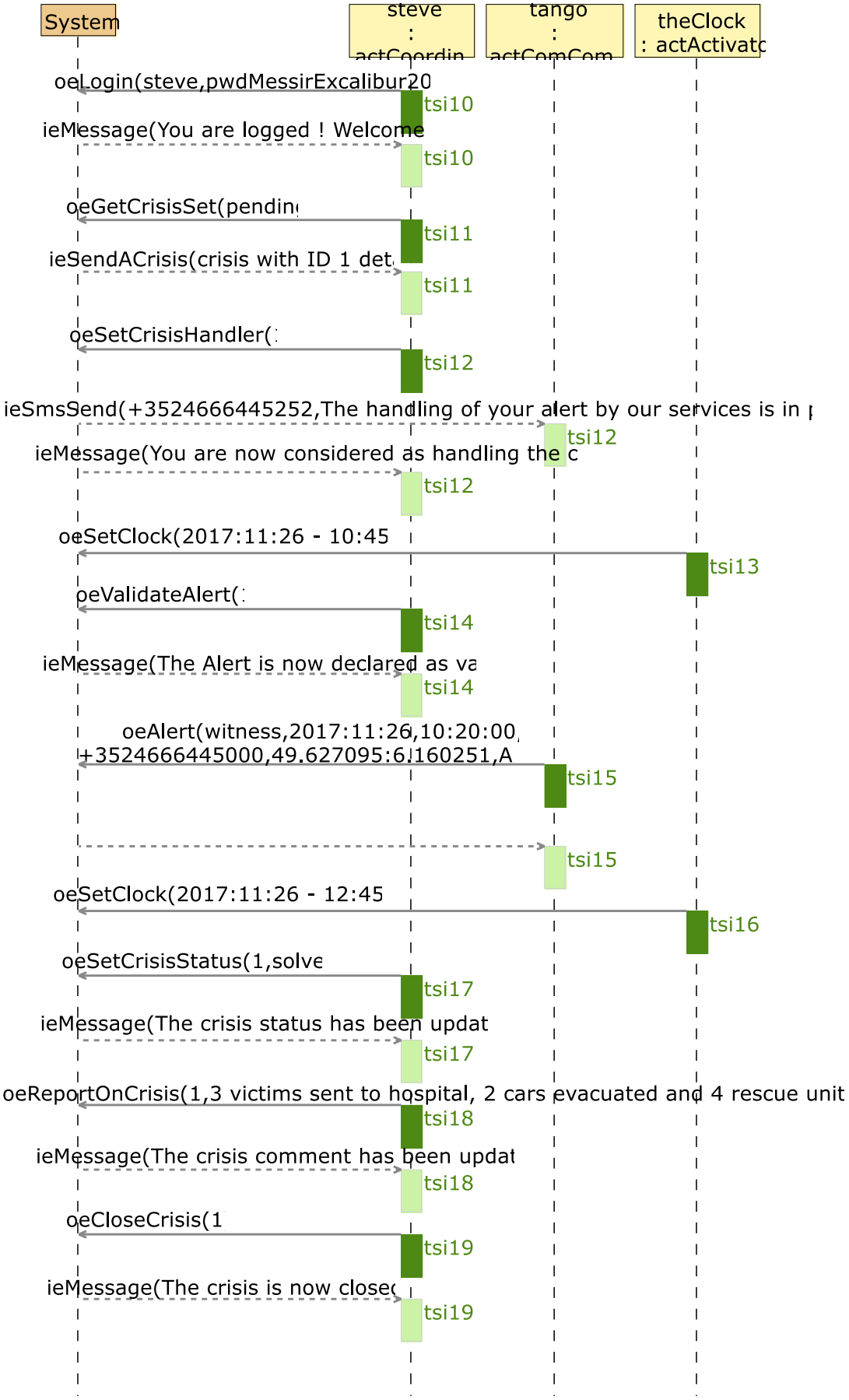

Figure 6.2: tci-testcase01-instance01-Part02 testcase instance sequence diagram

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 Case Instances

A.1.1 Undocumented User-Goal Level Use Case Instances

- usecases.uciugSecurelyUseSystem.uciugSecurelyUseSystem

A.1.2 Undocumented Use Case Instance Views

- uci-uciSimpleAndComplete
- uci-uciugSecurelyUseSystem

A.2 Undocumented Primary Types

A.2.1 Undocumented Primary Datatype Types

- icrash.concepts.primarytypes.datatypes.dtByteArray
- icrash.concepts.primarytypes.datatypes.dtEncodedPassword
- icrash.concepts.primarytypes.datatypes.dtPrivateKey
- icrash.concepts.primarytypes.datatypes.dtPublicKey

A.3 Undocumented Concept Model Views

- cm-pt-dt-lv-02-dtGPSLocation

A.4 Undocumented Operation Specifications

- icrash.concepts.primarytypes.classes.ctKeyPair.decodeMsg
- icrash.concepts.primarytypes.classes.ctKeyPair.encodeMsg
- icrash.concepts.primarytypes.classes.ctKeyPair.getKeys
- icrash.concepts.primarytypes.classes.ctKeyPair.initForDecoding

- icrash.concepts.primarytypes.classes.ctKeyPair.intiForEncoding
- icrash.concepts.primarytypes.datatypes.dtEncodedPassword.eq
- icrash.concepts.primarytypes.datatypes.dtPrivateKey.fromString
- icrash.concepts.primarytypes.datatypes.dtPrivateKey.getFromFile
- icrash.concepts.primarytypes.datatypes.dtPrivateKey.is
- icrash.concepts.primarytypes.datatypes.dtPrivateKey.toStringVal
- icrash.concepts.primarytypes.datatypes.dtPublicKey.fromString
- icrash.concepts.primarytypes.datatypes.dtPublicKey.is
- icrash.concepts.primarytypes.datatypes.dtPublicKey.toStringVal

A.5 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

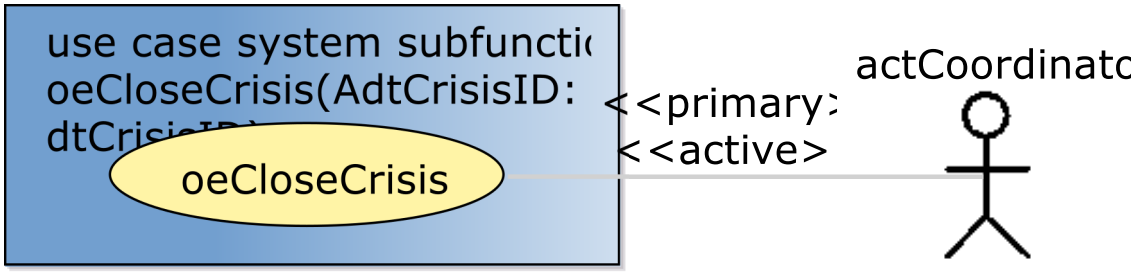

Figure B.1: oeCloseCrisis subfunction use case

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, AdtPublKey:dtPublicKey):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, AdtPublKey)
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-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.6: Messir Spec. file environment-actAdministrator-oeDeleteCoordinator.msr.

C.7 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, AEncodedPassword:
      dtEncodedPassword):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 }
26 preF{
27 /* PreF01 */
28 true
29 }
30 postF{
31  let TheSystem: ctState in
32  let TheactAuthenticated:actAuthenticated in
33  let AptStringMessageForTheactAuthenticated: ptString in
34  let AptStringMessageForTheactAdministrator:ptString in
35
36  self.rnActor.rnSystem = TheSystem
37  and self.rnActor = TheactAuthenticated
38
39  and /* PostF01 */
40    if (TheactAuthenticated.rnctAuthenticated.pwd
41      = TheSystem.ctKeyPairs.decodeMsg()
42      and TheactAuthenticated.rnctAuthenticated.login
43        = AdtLogin
44      )
45    then (AptStringMessageForTheactAuthenticated.eq('You are logged ! Welcome ...')
46       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
47       )
48    else (AptStringMessageForTheactAuthenticated
49       .eq('Wrong identification information ! Please try again ...')
50       and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
51       and AptStringMessageForTheactAdministrator.eq('Intrusion tentative !')
52       and TheSystem.rnactAdministrator
53         .rnInterfaceIN^ieMessage(AptStringMessageForTheactAdministrator)
54       )
55    endif
56 }
57 postP{
58  let TheSystem: ctState in
59  let TheactAuthenticated:actAuthenticated in
60
61  self.rnActor.rnSystem = TheSystem
62  and self.rnActor = TheactAuthenticated
63 /* PostP01 */
64  if (TheactAuthenticated.rnctAuthenticated.pwd = ctKeyPairs.decodeMsg(AEncodedPassword)
65    and TheactAuthenticated.rnctAuthenticated.login = AdtLogin
66    )
67  then (TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = true)
68  else true
69  endif
70 }
```

```
71 prolog{"src/Operations/Environment/OUT/outactAuthenticated-oeLogin.pl"}
72 }
73 /*-------------------------------------------------------------------------------*/
74
75 operation: actAuthenticated.outactAuthenticated.oeLogout():ptBoolean{
76
77 preP{
78  let TheSystem: ctState in
79  let TheActor:actAdministrator in
80  self.rnActor.rnSystem = TheSystem
81  and self.rnActor = TheActor
82
83 /* PreP01 */
84  and TheSystem.vpStarted = true
85 /* PreP02 */
86  and TheActor.rnctAuthenticated.vpIsLogged = true
87 }
88 preF{
89 /* PreF01 */
90 true
91 }
92 postF{
93  let TheSystem: ctState in
94  let TheactAuthenticated:actAuthenticated in
95  let AptStringMessageForTheactAuthenticated: ptString in
96
97  self.rnActor.rnSystem = TheSystem
98  and self.rnActor = TheactAuthenticated
99
100  /* PostF01 */
101  AptStringMessageForTheactAuthenticated.eq('You are logged out ! Good Bye ...')
102  and TheactAuthenticated.rnInterfaceIN^ieMessage(AptStringMessageForTheactAuthenticated)
103 }
104 postP{
105  let TheSystem: ctState in
106  let TheactAuthenticated:actAuthenticated in
107
108  self.rnActor.rnSystem = TheSystem
109  and self.rnActor = TheactAuthenticated.asSet
110 /* PostP01 */
111  TheactAuthenticated.rnctAuthenticated@post.vpIsLogged = false
112 }
113 prolog{"src/Operations/Environment/OUT/outactAuthenticated-oeLogout.pl"}
114 }
115 }
116 }
```

Listing C.7: Messir Spec. file environment-actAuthenticated.msr.

C.8 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
```

```
operation: actComCompany.outactComCompany.oeAlert(
 AetKind:etHumanKind,
 AdtMyDate:dtDate,
 AdtTime:dtTime,
 AdtPhoneNumber:dtPhoneNumber,
 AdtGPSLocation:dtGPSLocation,
 AdtComment:dtComment
):ptBoolean{

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

/* PreP01 */
 and TheSystem.vpStarted = true
}
preF{
 let TheSystem: ctState in
 self.rnActor.rnSystem = TheSystem

/* PreF01 */
 and (TheSystem.clock.date.gt(AdtDate)
    or (TheSystem.clock.date.eq(AdtDate)
      and TheSystem.clock.time.gt(AdtTime)
     )
   )
}
postF{
 let TheSystem: ctState in

 let ActHuman:ctHuman in
 let TheactComCompany:actComCompany in
 let ActAlert:ctAlert in
 let AAlertInstant:dtDateAndTime in
 let AetAlertStatus:etAlertStatus in
 let ActAlertNearBy:ctAlert in
 let ActCrisis:ctCrisis in
 let AdtCrisisID:dtCrisisID in
 let AetCrisisType:etCrisisType in
 let AetCrisisStatus:etCrisisStatus in
 let ACrisisInstant:dtDateAndTime in
 let ACrisisdtComment:dtComment in
 let AptStringMessage:ptString in
 let AdtSMS:dtSMS in
 let AdtAlertID:dtAlertID in

 self.rnActor.rnSystem = TheSystem
 and self.rnActor = TheactComCompany
/* PostF01 */
 TheSystem.nextValueForAlertID=PrenextValueForAlertID
 and PrenextValueForAlertID.add(1) = PostnextValueForAlertID
 and TheSystem@post.nextValueForAlertID = PostnextValueForAlertID

 /* PostF02 */
and AAlertInstant.date=AdtDate
and AAlertInstant.time=AdtTime

and AetAlertStatus=pending

and TheSystem.nextValueForAlertID.todtString().eq(AdtAlertID)

and ActAlert.init(AdtAlertID,
         AetAlertStatus,
         AdtGPSLocation,
         AAlertInstant,
         AdtComment)

 /* PostF03 */
and TheSystem.rnctAlert.select(location.isNearTo(AdtGPSLocation)) = ColctAlertsNearBy
and if (ColctAlertsNearBy->size()=0)
```

```
 then (TheSystem.nextValueForCrisisID = PrenextValueForCrisisID
    and PrenextValueForCrisisID.add(1) = PostnextValueForCrisisID
    and TheSystem@post.nextValueForCrisisID = PostnextValueForCrisisID
    and TheSystem.nextValueForCrisisID.todtString().eq(AdtCrisisID)
    and AdtCrisisType = small
    and AetCrisisStatus = pending
    and ACrisisInstant= AAlertInstant
    and ACrisisdtComment = 'no reporting yet defined'
    and ActCrisis.init( AdtCrisisID,
              AdtCrisisType,
              AetCrisisStatus,
              AdtGPSLocation,
              ACrisisInstant,
              ACrisisdtComment)
    )
 else (ColctAlertsNearBy.rnTheCrisis->msrAny(true) = ActCrisis)
 endif

 /* PostF04 */
and ActAlert@post.rnTheCrisis = ActCrisis

/* PostF05 */
and TheSystem.rnctHuman->select(id.eq(AdtPhoneNumber)) = HumanCol1

and HumanCol1->select(kind.etEq(AetHumanKind)) = HumanCol2
and if (HumanCol2->msrIsEmpty)
  then (ActHuman.init(AdtPhoneNumber,AetHumanKind)
    and ActHuman@post.rnactComCompany = TheactComCompany
    )
  else (HumanCol2->any(true) = ActHuman)
  endif

 and ActHuman.rnSignaled->msrIncluding(ActAlert) = ColAlerts

 and ActHuman@post.rnSignaled = ColAlerts

/* PostF06 */
AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
and TheactComCompany.rnInterfaceIN^ieSmsSend(AdtPhoneNumber,AdtSMS)
}
 /* Post Protocol:*/
/* PostP01 */
postP{true}

prolog{"src/Operations/Environment/OUT/outactComCompany-oeAlert.pl"}
}
}
}
```

Listing C.8: Messir Spec. file environment-actComCompany.msr.

C.9 File ./src-gen/messir-spec/operations/environment/environment-actCoordinator-oeCloseCrisis.msr

```
package icrash.operations.environment.actCoordinator.oeCloseCrisis {

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.environment

Operation Model {

operation: actCoordinator.outactCoordinator.oeCloseCrisis(AdtCrisisID:dtCrisisID):ptBoolean{
prolog{"src/Operations/Environment/OUT/outactCoordinator-oeCloseCrisis.pl"}
}
```

```
15 }
16 }
```

Listing C.9: Messir Spec. file environment-actCoordinator-oeCloseCrisis.msr.

C.10 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.10: Messir Spec. file environment-actCoordinator-oeGetAlertsSet.msr.

C.11 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.environment
9
10 Operation Model {
11
12 operation: actCoordinator.outactCoordinator.oeGetCrisisSet(AetCrisisStatus:etCrisisStatus):ptBoolean
      {
13 prolog{"src/Operations/Environment/OUT/outactCoordinator-oeGetCrisisSet.pl"}
14 }
15 }
16 }
```

Listing C.11: Messir Spec. file environment-actCoordinator-oeGetCrisisSet.msr.

C.12 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.12: Messir Spec. file environment-actCoordinator-oeInvalidateAlert.msr.

C.13 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.13: Messir Spec. file environment-actCoordinator-oeReportOnCrisis.msr.

C.14 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.14: Messir Spec. file environment-actCoordinator-oeSetCrisisHandler.msr.

C.15 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.15: Messir Spec. file environment-actCoordinator-oeSetCrisisStatus.msr.

C.16 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.16: Messir Spec. file environment-actCoordinator-oeSetCrisisType.msr.

C.17 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.17: Messir Spec. file environment-actCoordinator-oeValidateAlert.msr.

C.18 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.18: Messir Spec. file environment-actMsrCreator-init.msr.

C.19 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  let ApublicKey:dtPublicKey in
29
30  /* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
31  AnextValueForAlertID.value.eq(1)
32  and AnextValueForCrisisID.value.eq(1)
33  and Aclock.date.year.value = 1970
34  and Aclock.date.month.value = 01
35  and Aclock.date.day.value = 01
36  and Aclock.time.hour.value = 00
37  and Aclock.time.minute.value = 00
38  and Aclock.time.second.value = 00
39
40  and AcrisisReminderPeriod.value.eq(300)
41  and AmaxCrisisReminderPeriod.value.eq(1200)
42  and AvpStarted = true
43  and TheSystem.init(AnextValueForAlertID,
44            AnextValueForCrisisID,
45            Aclock,
46            AcrisisReminderPeriod,
```

```
47            AmaxCrisisReminderPeriod,
48            Aclock,
49            AvpStarted
50           )
51  /* PostF02*/
52  and AactMsrCreator.init()
53  /* PostF03 */
54  and let AactComCompanyCol: Bag(actComCompany) in
55  AactComCompanyCol->size() = AqtyComCompanies
56  AactComCompanyCol-> forAll(init())
57  /* PostF04*/
58  and AactAdministrator.init()
59  and TheSystem.ctKeyPair.initForDecoding(ApublicKey, '[B@6979e8cb')
60  /* PostF05*/
61  and let AactActivator:actActivator in
62  AactActivator.init()
63 /* PostF06 */
64  and let ActAdministrator:ctAdministrator in
65    let AdtLogin:dtLogin in
66    let AdtEncodedPassword:dtEncodedPassword in
67    AdtLogin.value.eq('icrashadmin')
68    and AdtEncodedPassword.value.eq(TheSystem.ctKeyPair.decodeMsg())
69    and ActAdministrator.init(AdtLogin,AdtPassword)
70  /* PostF07*/
71  and ActAdministrator@post.rnactAuthenticated = AactAdministrator}
72 postP{true}
73
74 prolog{ "src/Operations/Environment/OUT/outactMsrCreator-oeCreateSystemAndEnvironment.pl"}
75
76 }
77 }
78
79 }
```

Listing C.19: Messir Spec. file environment-actMsrCreator-oeCreateSystemAndEnvironment.msr.

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

```
1 package icrash.environment{
2
3 import icrash.concepts.primarytypes.datatypes
4 import icrash.concepts.primarytypes.classes
5 import icrash.concepts.secondarytypes.datatypes
6 import lu.uni.lassy.messir.libraries.primitives
7 import lu.uni.lassy.messir.libraries.math
8 import 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
24       role rnactAdministrator
25       cardinality [1..1]
26       extends actAuthenticated {
27
28    operation init():ptBoolean
29
30    output interface outactAdministrator{
31
```

```
32    operation oeAddCoordinator(
33           AdtCoordinatorID:dtCoordinatorID ,
34           AdtLogin:dtLogin ,
35           AdtPassword:dtPassword ,
36           AdtPublKey:dtPublicKey):ptBoolean
37
38    operation oeDeleteCoordinator(
39           AdtCoordinatorID:dtCoordinatorID ):ptBoolean
40   }
41
42   input interface inactAdministrator{
43
44    operation ieCoordinatorAdded():ptBoolean
45    operation ieCoordinatorDeleted():ptBoolean
46    }
47  }
48
49  actor actCoordinator
50      role rnactCoordinator
51      cardinality [0..*]
52      extends actAuthenticated{
53
54   operation init():ptBoolean
55
56   output interface outactCoordinator{
57    operation oeInvalidateAlert(AdtAlertID:dtAlertID ):ptBoolean
58    operation oeCloseCrisis(AdtCrisisID:dtCrisisID ):ptBoolean
59    operation oeGetAlertsSet(AetAlertStatus:etAlertStatus ):ptBoolean
60    operation oeGetCrisisSet(AetCrisisStatus:etCrisisStatus ):ptBoolean
61    operation oeSetCrisisHandler(AdtCrisisID:dtCrisisID ):ptBoolean
62    operation oeReportOnCrisis(
63           AdtCrisisID:dtCrisisID ,
64           AdtComment:dtComment
65           ):ptBoolean
66    operation oeSetCrisisStatus(
67           AdtCrisisID:dtCrisisID ,
68           AetCrisisStatus:etCrisisStatus
69           ):ptBoolean
70    operation oeSetCrisisType(
71                   AdtCrisisID:dtCrisisID ,
72                   AetCrisisType:etCrisisType
73                   ):ptBoolean
74    operation oeValidateAlert(AdtAlertID:dtAlertID ):ptBoolean
75   }
76
77   input interface inactCoordinator{
78    operation ieSendAnAlert(ActAlert:ctAlert ):ptBoolean
79    operation ieSendACrisis(ActCrisis:ctCrisis ):ptBoolean
80    }
81  }
82
83  actor actComCompany role rnactComCompany cardinality [0..*]{
84
85   operation init():ptBoolean
86
87   output interface outactComCompany{
88    operation oeAlert(
89          AetHumanKind:etHumanKind ,
90          AdtDate:dtDate ,
91          AdtTime:dtTime ,
92          AdtPhoneNumber:dtPhoneNumber ,
93          AdtGPSLocation:dtGPSLocation ,
94          AdtComment:dtComment
95          ):ptBoolean
96   }
97
98   input interface inactComCompany{
99    operation ieSmsSend(AdtPhoneNumber:dtPhoneNumber ,
100            AdtSMS:dtSMS
101            ):ptBoolean
```

```
    }
  }

  actor actAuthenticated role rnactAuthenticated cardinality [0..*]{

   operation init():ptBoolean

   output interface outactAuthenticated{
    operation oeLogin(AdtLogin:dtLogin , AdtEncodedPassword:dtEncodedPassword):ptBoolean
    operation oeLogout():ptBoolean
   }

   input interface inactAuthenticated{
    operation ieMessage(AMessage:ptString):ptBoolean
    }
  }

  actor actActivator[proactive] role rnactActivator cardinality [1..1]{

   operation init():ptBoolean

   output interface outactActivator{
    proactive operation oeSollicitateCrisisHandling():ptBoolean
    proactive operation oeSetClock(AcurrentClock:dtDateAndTime ):ptBoolean
   }

   input interface inactActivator{
    }
  }
 }
}
```

Listing C.20: Messir Spec. file environment.msr.

C.21 File ./src-gen/messir-spec/concepts/primarytypes-associations.msr

```
package icrash.concepts.primarytypes.associations {

import icrash.concepts.primarytypes.datatypes
import icrash.concepts.primarytypes.classes
import icrash.environment
import lu.uni.lassy.messir.libraries.primitives

 Concept Model {

  Primary Types{

// Internal

association assctAlertctCrisis
 ctAlert(rnAlerts)[1..*]
 ctCrisis (rnTheCrisis)[1..1]

association assctAlertctHuman
 ctAlert(rnSignaled)[1..*]
 ctHuman (rnSignaler)[1..1]

association assctCrisisctCoordinator
 ctCrisis(rnHandled)[0..*]
 ctCoordinator(rnHandler)[0..1]

// With Actors

   association assctHumanactComCompany
      ctHuman(rnctHuman)[0..*]
      actComCompany(rnactComCompany)[1..1]

```

```
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   }
41  }
42 }
```

Listing C.21: Messir Spec. file primarytypes-associations.msr.

C.22 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  ApublKey:dtPublicKey
14  ):ptBoolean{
15 postF{
16 if
17 (
18 let Self:ctAdministrator in
19 /* Post F01 */
20 Self.login(Alogin)
21 and Self.pwd = Apwd
22 and Self.pubKey = ApublKey
23 and Self.vpIsLogged = false
24
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-ctAdministrator-init.pl"}
33  }
34 }
35 }
```

Listing C.22: Messir Spec. file primarytypes-classes-ctAdministrator.msr.

C.23 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.23: Messir Spec. file primarytypes-classes-ctAlert.msr.

C.24 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
       , ApubKey:dtPublicKey):ptBoolean{
10 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctAuthenticated-init.pl"}
11 }
12 }
13
```

```
14 }
```

Listing C.24: Messir Spec. file primarytypes-classes-ctAuthenticated.msr.

C.25 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, ApubKey:dtPublicKey):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.pubKey = ApubKey
20 and Self.vpIsLogged = false
21 /* Post F02 */
22 and (Self.oclIsNew and self = Self)
23 )
24 then (result = true)
25 else (result = false)
26 endif}
27 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCoordinator-init.pl"}
28 }
29 }
30 }
```

Listing C.25: Messir Spec. file primarytypes-classes-ctCoordinator.msr.

C.26 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
79  and TheSystem.maxCrisisReminderPeriod = MaxCrisisReminderPeriod
80  and CurrentClockSecondsQty.sub(CrisisInstantSecondsQty)
81                .gt(MaxCrisisReminderPeriod)
82 )
83 then (result = true)
84 else (result = false)
85 endif
86 }
87 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctCrisis-
       maxHandlingDelayPassed.pl"}}
88 //---------------------------------------------------------------
89 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.26: Messir Spec. file primarytypes-classes-ctCrisis.msr.

C.27 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctHuman.msr

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

import lu.uni.lassy.messir.libraries.primitives
import icrash.concepts.primarytypes.datatypes

import icrash.concepts.primarytypes.classes

Operation Model {

operation: icrash.concepts.primarytypes.classes.ctHuman.init(Aid:dtPhoneNumber, Akind:etHumanKind):
    ptBoolean
{
postF{
if
(
```

```
/* Post F01 */
let Self:ctHuman in

Self.id = Aid
and Self.kind = Akind

/* Post F02 */
and (Self.oclIsNew and self = Self)
)
then (result = true)
else (result = false)
endif
}
prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctHuman-init.pl"}
}
operation: icrash.concepts.primarytypes.classes.ctHuman.isAcknowledged():ptBoolean{
prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctHuman-isAcknowledged.pl"}
}
}
}
```

Listing C.27: Messir Spec. file primarytypes-classes-ctHuman.msr.

C.28 File ./src-gen/messir-spec/operations/concepts/primarytypes-classes/primarytypes-classes-ctKeyPair.msr

```
package icrash.concepts.primarytypes.classes.operations.classes.ctKeyPair {

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

Operation Model {
//-------------------------------------------------------------
operation: icrash.concepts.primarytypes.classes.ctKeyPair.intiForEncoding(
 AprivKey:dtPrivateKey,
 AdecodedPwd:ptString
):ptBoolean{
postF{
if
(
let Self:ctKeyPair in
/* Post F01 */
Self.privKey = Alogin
and Self.decodedPwd = AdecodedPwd

/* Post F02 */
and (Self.oclIsNew and self = Self)
)
then (result = true)
else (result = false)
endif}}
//-------------------------------------------------------------
operation: icrash.concepts.primarytypes.classes.ctKeyPair.initForDecoding(
 ApubKey:dtPublicKey,
 AencodedPwd:dtEncodedPassword
):ptBoolean{
postF{
if
(
let Self:ctKeyPair in
/* Post F01 */
```

```
43 Self.pubKey = ApubKey
44 and Self.encodedPwd = AencodedPwd
45
46 /* Post F02 */
47 and (Self.oclIsNew and self = Self)
48 )
49 then (result = true)
50 else (result = false)
51 endif}}
52 //-------------------------------------------------------------
53 operation: icrash.concepts.primarytypes.classes.ctKeyPair.getKeys():ptBoolean{
54 postF{
55 let ThePublicKey:dtPublicKey in
56 let ThePrivateKey:dtPrivateKey in
57 if
58 ( /* Post F01 */
59 self.pubKey = ThePublicKey
60 and self.privKey = ThePrivateKey
61 )
62 then (result = true)
63 else (result = false)
64 endif}}
65 //-------------------------------------------------------------
66 operation: icrash.concepts.primarytypes.classes.ctKeyPair.encodeMsg():ptBoolean{
67 postF{
68 let ThePrivateKey:dtPrivateKey in
69 let ThePassword:dtPassword in
70 if
71 ( /* Post F01 */
72 self.pubKey = ThePublicKey
73 and self.privKey = ThePrivateKey
74 )
75 then (result = true)
76 else (result = false)
77 endif}}
78 //-------------------------------------------------------------
79 operation: icrash.concepts.primarytypes.classes.ctKeyPair.decodeMsg():ptBoolean{
80 postF{
81 let ThePublicKey:dtPublicKey in
82 let ThePrivateKey:dtPrivateKey in
83 if
84 ( /* Post F01 */
85 self.pubKey = ThePublicKey
86 and self.privKey = ThePrivateKey
87 )
88 then (result = true)
89 else (result = false)
90 endif}}
91 }
92 }
```

Listing C.28: Messir Spec. file primarytypes-classes-ctKeyPair.msr.

C.29 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   AnextValueForAlertID: dtInteger,
```

```
13   AnextValueForCrisisID: dtInteger ,
14   dtAclock:dtDateAndTime,
15   AcrisisReminderPeriod: dtSecond,
16   AmaxCrisisReminderPeriod: dtSecond ,
17   AvpLastReminder: dtDateAndTime ,
18   AvpStarted:ptBoolean ):ptBoolean{
19 postF{
20 if
21 (
22 /* Post F01 */
23 let Self:ctState in
24
25 Self.nextValueForAlertID = AnextValueForAlertID
26 and Self.nextValueForCrisisID = AnextValueForCrisisID
27 and Self.clock = Aclock
28 and Self.crisisReminderPeriod = AcrisisReminderPeriod
29 and Self.maxCrisisReminderPeriod = AmaxCrisisReminderPeriod
30 and Self.vpLastReminder = AvpLastReminder
31 and Self.vpStarted = AvpStarted
32
33 and (Self.oclIsNew and self = Self)
34 )
35 then (result = true)
36 else (result = false)
37 endif
38 }
39 prolog{"src/Operations/Concepts/PrimaryTypesClasses/PrimaryTypesClasses-ctState-init.pl" }
40  }
41  }
42 }
```

Listing C.29: Messir Spec. file primarytypes-classes-ctState.msr.

C.30 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 clock:dtDateAndTime
17     attribute crisisReminderPeriod:dtSecond
18     attribute maxCrisisReminderPeriod:dtSecond
19     attribute vpLastReminder:dtDateAndTime
20     attribute vpStarted:ptBoolean
21
22     operation init( AnextValueForAlertID:dtInteger,
23             AnextValueForCrisisID:dtInteger,
24              Aclock:dtDateAndTime,
25             AcrisisReminderPeriod:dtSecond ,
26             AmaxCrisisReminderPeriod:dtSecond ,
27             AvpLastReminder:dtDateAndTime ,
28             AvpStarted:ptBoolean ): ptBoolean
29    }
30
31    class ctAlert role rnctAlert cardinality [0..*]{
32     attribute id:dtAlertID
33     attribute status: etAlertStatus
34     attribute location:dtGPSLocation
```

```
35     attribute instant:dtDateAndTime
36     attribute comment:dtComment
37
38     operation init(    Aid:dtAlertID ,
39           Astatus:etAlertStatus ,
40           Alocation:dtGPSLocation ,
41           Ainstant:dtDateAndTime ,
42           Acomment:dtComment ):ptBoolean
43     operation isSentToCoordinator(AactCoordinator:actCoordinator ):ptBoolean
44
45    }
46
47    class ctCrisis role rnctCrisis cardinality [0..*]{
48     attribute id:dtCrisisID
49     attribute type:etCrisisType
50     attribute status: etCrisisStatus
51     attribute location:dtGPSLocation
52     attribute instant:dtDateAndTime
53     attribute comment:dtComment
54
55     operation init(
56           Aid:dtCrisisID ,
57           Atype:etCrisisType ,
58           Astatus:etCrisisStatus ,
59           Alocation:dtGPSLocation ,
60           Ainstant:dtDateAndTime ,
61           Acomment:dtComment ):ptBoolean
62
63     operation handlingDelayPassed():ptBoolean
64     operation maxHandlingDelayPassed():ptBoolean
65     operation isSentToCoordinator(AactCoordinator:actCoordinator ):ptBoolean
66     operation isAllocatedIfPossible():ptBoolean
67    }
68
69    class ctHuman role rnctHuman cardinality [0..*]{
70     attribute id:dtPhoneNumber
71     attribute kind:etHumanKind
72
73     operation init(
74           Aid:dtPhoneNumber ,
75           Akind:etHumanKind ):ptBoolean
76     operation isAcknowledged():ptBoolean
77    }
78
79    class ctAuthenticated
80      role rnctAuthenticated
81      cardinality [0..*]{
82
83     attribute login:dtLogin
84     attribute pwd: dtPassword
85     attribute pubKey: dtPublicKey
86     attribute vpIsLogged:ptBoolean
87
88     operation init(
89           Alogin:dtLogin ,
90           Apwd:dtPassword ,
91           ApubKey:dtPublicKey):ptBoolean
92    }
93
94    class ctCoordinator
95      role rnctCoordinator
96      cardinality [0..*]
97      extends ctAuthenticated{
98
99     attribute id:dtCoordinatorID
100
101    operation init(
102          Aid:dtCoordinatorID ,
103          Alogin:dtLogin ,
104          Apwd:dtPassword ,
```

```
105           ApubKey:dtPublicKey):ptBoolean
106     }
107
108     class ctAdministrator
109       role rnctAdministrator
110       cardinality [1..1]
111       extends ctAuthenticated{
112
113      operation init(
114           Alogin:dtLogin ,
115           Apwd:dtPassword ,
116           ApubKey:dtPublicKey):ptBoolean
117     }
118
119     class ctKeyPair
120       role rnctKeyPair
121       cardinality[1..1]{
122
123      attribute pubKey:dtPublicKey
124      attribute privKey:dtPrivateKey
125      attribute encodedPwd: dtEncodedPassword
126      attribute decodedPwd: ptString
127
128      operation intiForEncoding(AprivKey:dtPrivateKey, AdecodedPwd:ptString):ptBoolean
129      operation initForDecoding(ApubKey:dtPublicKey, AencodedPwd:dtEncodedPassword):ptBoolean
130      operation getKeys():ptBoolean
131
132      operation encodeMsg():ptBoolean
133      operation decodeMsg():ptBoolean
134       }
135   }
136  }
137 }
```

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

C.31 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.31: Messir Spec. file primarytypes-datatypes-dtAlertID.msr.

C.32 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.32: Messir Spec. file primarytypes-datatypes-dtComment.msr.

C.33 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.33: Messir Spec. file primarytypes-datatypes-dtCoordinatorID.msr.

C.34 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.34: Messir Spec. file primarytypes-datatypes-dtCrisisID.msr.

C.35 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   operation: icrash.concepts.primarytypes.datatypes.dtGPSLocation.isNearTo(aGPSLocation:
         dtGPSLocation):ptBoolean{
29   postF{
30     let TheResult: ptBoolean in true
31     let EarthRadius: dtReal in
32     let MaxDistance: dtReal in
33     let ComparedLatitude: dtLatitude in
34     let ComparedLongitude: dtLongitude in
35     let R1: dtReal in let R1a: dtReal in
36     let R2: dtReal in let R2a: dtReal in
37
38     ( if
```

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

Listing C.35: Messir Spec. file primarytypes-datatypes-dtGPSLocation.msr.

C.36 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{
```

```
    let TheResult: ptBoolean in
    let MaxLength: ptInteger in
    ( if
     ( MaxLength = 20
      and AdtValue.value.length().leq(MaxLength)
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )
    }
  prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtLogin-is.pl"}
 }
}
}
```

Listing C.36: Messir Spec. file primarytypes-datatypes-dtLogin.msr.

C.37 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPassword.msr

```
package icrash.operations.concepts.primarytypes.datatypes.dtPassword{

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

Operation Model {

  operation: icrash.concepts.primarytypes.datatypes.dtPassword.is():ptBoolean{
  postF{
    let TheResult: ptBoolean in
    let MinLength: ptInteger in
    ( if
     ( MinLength = 6
      and AdtValue.value.length().geq(MinLength)
     )
     then (TheResult = true)
     else (TheResult = false)
     endif
     result = TheResult
    )
}
  prolog{"src/Operations/Concepts/PrimaryTypesDatatypes/PrimaryTypesDatatypes-dtPassword-is.pl"}
 }
}
}
```

Listing C.37: Messir Spec. file primarytypes-datatypes-dtPassword.msr.

C.38 File ./src-gen/messir-spec/operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPhoneNumber.msr

```
package icrash.operations.concepts.primarytypes.datatypes.dtPhoneNumber{

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

Operation Model {

  operation: icrash.concepts.primarytypes.datatypes.dtPhoneNumber.is():ptBoolean{

  postF{
  let TheResult: ptBoolean in
  ( if
   ( AdtValue.value.length().gt(4)
    and AdtValue.value.length().leq(30)
   )
```

```
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.38: Messir Spec. file primarytypes-datatypes-dtPhoneNumber.msr.

C.39 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.39: Messir Spec. file primarytypes-datatypes-etAlertStatus.msr.

C.40 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.40: Messir Spec. file primarytypes-datatypes-etCrisisStatus.msr.

C.41 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.41: Messir Spec. file primarytypes-datatypes-etCrisisType.msr.

C.42 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.42: Messir Spec. file primarytypes-datatypes-etHumanKind.msr.

C.43 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
12    datatype dtAlertID extends dtString {
13    operation is():ptBoolean
14   }
15   datatype dtCrisisID extends dtString {
16    operation is():ptBoolean
17   }
18   datatype dtLogin extends dtString {
19    operation is():ptBoolean
20   }
21   datatype dtPassword extends dtString {
22    operation is():ptBoolean
23   }
24   datatype dtCoordinatorID extends dtString {
25    operation is():ptBoolean
26   }
27   datatype dtPhoneNumber extends dtString {
28    operation is():ptBoolean
29   }
30   datatype dtComment extends dtString {
31    operation is():ptBoolean
32   }
33   datatype dtLatitude extends dtReal {
34    operation is():ptBoolean
35   }
36   datatype dtLongitude extends dtReal {
37    operation is():ptBoolean
38   }
39   datatype dtGPSLocation {
40    attribute latitude: dtLatitude
41    attribute longitude: dtLongitude
42    operation is():ptBoolean
43    operation isNearTo(AGPSLocation:dtGPSLocation ):ptBoolean
44   }
45
46   datatype dtPublicKey {
47     operation fromString(pubKey : ptString) : ptBoolean
48     operation toStringVal(): ptString
49     operation is():ptBoolean
50
51   }
52
53   datatype dtPrivateKey {
54     operation fromString(privKey : ptString) : ptBoolean
55     operation toStringVal(): ptString
56     operation getFromFile(value : ptString) : ptBoolean
57     operation is():ptBoolean
58   }
59
60   datatype dtEncodedPassword extends dtByteArray {
61
62     external operation eq():ptBoolean
63   }
64
65   datatype dtByteArray {
66     attribute value: ptString
```

```
67
68    }
69
70    enum etCrisisStatus {
71     constants["pending", "handled", "solved","closed"]
72     operation is():ptBoolean
73    }
74    enum etAlertStatus {
75     constants["pending", "valid", "invalid"]
76     operation is():ptBoolean
77    }
78    enum etCrisisType {
79     constants["small", "medium", "huge"]
80     operation is():ptBoolean
81    }
82    enum etHumanKind {
83     constants["witness", "victim", "anonymous"]
84     operation is():ptBoolean
85    }
86   }
87  }
88 }
```

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

C.44 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.44: Messir Spec. file secondarytypes-associations.msr.

C.45 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.45: Messir Spec. file secondarytypes-classes.msr.

C.46 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.46: Messir Spec. file secondarytypes-datatypes.msr.

C.47 File ./src-gen/messir-spec/usecases/subfunctions-usecases.msr

```
 1package icrash.usecases.subfunctions {
 2
 3import lu.uni.lassy.messir.libraries.primitives
 4
 5import icrash.concepts.primarytypes.datatypes
 6import icrash.concepts.primarytypes.classes
 7import icrash.concepts.secondarytypes.datatypes
 8import lu.uni.lassy.messir.libraries.primitives
 9import lu.uni.lassy.messir.libraries.math
10import lu.uni.lassy.messir.libraries.calendar
11
12import icrash.environment
13
14 Use Case Model {
15
16//-------------------------------------------------------
17 use case system subfunction oeAddCoordinator(AdtCoordinatorID:dtCoordinatorID, AdtLogin:dtLogin,
      AdtPassword:dtPassword, AdtPublicKey:dtPublicKey){
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 , AdtEncodedPassword:dtEncodedPassword) {
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
140 }
```

Listing C.47: Messir Spec. file subfunctions-usecases.msr.

C.48 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       AdtEncodedPassword:dtEncodedPassword
67      }
68     constraints{
69       TheActor=TheSystem.rnactAdministrator->any2(true)
70       AdtLogin.value.eq('icrashadmin')
71       AdtEncodedPassword.value.eq('[B@6979e8cb')
72      }
73     test message{
74      out:TheActor sends to system actAdministrator.outactAdministrator.oeLogin(AdtLogin,
      AdtEncodedPassword)
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       AdtPublicKey:dtPublicKey
94      }
95     constraints{
96       TheActor = TheSystem.rnactAdministrator->any2(true)
97       AdtCoordinatorID.value.eq('1')
98       AdtLogin.value.eq('steve')
99       AdtPassword.value.eq('pwdMessirExcalibur2017')
100      AdtPublicKey.value.eq('Sun RSA public key, 2048 bits ...')
101     }
102    test message{
103     out:TheActor
104     sends to system actAdministrator.outactAdministrator.oeAddCoordinator
105                                 (AdtCoordinatorID,
106                                  AdtLogin,
107                                  AdtPassword,
108                                  AdtPublicKey)
109    }
110    oracle{
111     constraints{
112       TheActor.inactAdministrator.ieCoordinatorAdded()
```

```
113      }
114    }
115    }
116 //-------------------------------------------------------------------------
117  test step ts05oeLogout order 05{
118     variables{
119       TheActor : actAdministrator
120      }
121     constraints{
122       TheActor = TheSystem.rnactAdministrator->any2(true)
123      }
124     test message{
125      out:TheActor sends to system actAdministrator.outactAdministrator.oeLogout()
126     }
127     oracle{
128      variables{
129       AMessage:ptString
130      }
131      constraints{
132      AMessage = 'You are logged out ! Good Bye ...'
133      TheActor.inactAdministrator.ieMessage(AMessage)
134     }
135    }
136   }
137 //-------------------------------------------------------------------------
138  test step ts06oeSetClock02 order 06{
139     variables{
140       TheActor:actActivator
141       ACurrentClock:dtDateAndTime
142      }
143     constraints{
144       TheActor=TheSystem.rnactActivator->any2(true)
145       ACurrentClock.date.year.value = 2017
146       ACurrentClock.date.month.value = 11
147       ACurrentClock.date.day.value = 26
148       ACurrentClock.time.hour.value = 10
149       ACurrentClock.time.minute.value = 15
150       ACurrentClock.time.second.value = 00
151      }
152     test message{
153      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
154     }
155     oracle{
156      constraints{
157      true
158     }
159    }
160    }
161 //-------------------------------------------------------------------------
162  test step ts07oeAlert1 order 07{
163     variables{
164       TheActor : actComCompany
165       AetHumanKind:etHumanKind
166       AdtDate:dtDate
167       AdtTime:dtTime
168       AdtPhoneNumber:dtPhoneNumber
169       AdtGPSLocation:dtGPSLocation
170       AdtComment:dtComment
171      }
172     constraints{
173       TheActor = TheSystem.rnactComCompany->any2(true)
174       AetHumanKind = witness
175       AdtDate.year.value = 2017
176       AdtDate.month.value = 11
177       AdtDate.day.value = 26
178       AdtTime.hour.value = 10
179       AdtTime.minute.value = 10
180       AdtTime.second.value = 16
181       AdtPhoneNumber.value = '+3524666445252'
182       AdtGPSLocation.latitude.value = 49.627675
```

```
183       AdtGPSLocation.longitude.value = 6.159590
184       AdtComment.value = '3 cars involved in an accident.'
185      }
186     test message{
187      out:TheActor
188      sends to system actComCompany.outactComCompany.oeAlert( AetHumanKind,
189                                  AdtDate,
190                                  AdtTime,
191                                  AdtPhoneNumber,
192                                  AdtGPSLocation,
193                                  AdtComment)
194     }
195     oracle{
196      variables{
197        AdtSMS:dtSMS
198      }
199      constraints{
200      AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
201      TheActor.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
202     }
203    }
204    }
205 //-------------------------------------------------------------------------
206  test step ts08oeSetClock03 order 08{
207     variables{
208       TheActor:actActivator
209       ACurrentClock:dtDateAndTime
210      }
211    constraints{
212       TheActor=TheSystem.rnactActivator->any2(true)
213       ACurrentClock.date.year.value = 2017
214       ACurrentClock.date.month.value = 11
215       ACurrentClock.date.day.value = 26
216       ACurrentClock.time.hour.value = 10
217       ACurrentClock.time.minute.value = 30
218       ACurrentClock.time.second.value = 00
219      }
220     test message{
221      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
222     }
223     oracle{
224      constraints{
225       true
226      }
227    }
228    }
229 //-------------------------------------------------------------------------
230  test step ts09oeSollicitateCrisisHandling order 09{
231     variables{
232       TheActor : actActivator
233      }
234     constraints{
235       TheActor = TheSystem.rnactActivator->any2(true)
236      }
237     test message{
238      out:TheActor sends to system actActivator.outactActivator.oeSollicitateCrisisHandling()
239     }
240     oracle{
241     variables{
242       TheAdministrator:actAdministrator
243       TheCoordinator:actCoordinator
244       AMessageForCrisisHandlers:ptString
245      }
246      constraints{
247      TheAdministrator = TheSystem.rnactAdministrator->any2(true)
248      TheCoordinator = TheSystem.rnactCoordinator->any2(true)
249      AMessageForCrisisHandlers = 'There are alerts pending since more than the defined delay. Please
         REACT !'
250
251      TheAdministrator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
```

```
252     TheCoordinator.inactAdministrator.ieMessage(AMessageForCrisisHandlers)
253
254 /* this oracle should be written like this:
255
256     oracle{
257     variables{
258       TheAdministrator:actAdministrator
259       AMessageForCrisisHandlers:ptString
260      }
261      constraints{
262      AMessageForCrisisHandlers = 'There are alerts pending since more than the defined delay. Please
        REACT !'
263      TheAdministrator = TheSystem.rnactAdministrator->any2(true)
264
265      TheSystem.rnactCoordinator->forAll(TheCoordinator:actCoordinator | TheCoordinator.
       actAuthenticated.inactAuthenticated.ieMessage(AMessage))
266
267      // receives from system is for step instances
268
269  */
270     }
271    }
272    }
273 //-------------------------------------------------------------------------
274  test step ts10oeLogin02 order 10{
275     variables{
276       TheActor : actCoordinator
277       AdtLogin:dtLogin
278       AdtEncodedPassword:dtEncodedPassword
279      }
280     constraints{
281       TheActor = TheSystem.rnactCoordinator->select(a | a.rnctCoordinator.login.value.eq('steve'))->
       any2(true)
282       AdtLogin.value.eq('steve')
283       AdtEncodedPassword.value.eq('[B@6979e8cb')
284      }
285     test message{
286      out:TheActor sends to system actAuthenticated.outactAuthenticated.oeLogin(AdtLogin,
       AdtEncodedPassword)
287     }
288     oracle{
289      variables{
290        AMessage:ptString
291      }
292      constraints{
293      AMessage = 'You are logged ! Welcome ...'
294      TheActor.inactAuthenticated.ieMessage(AMessage)
295     }
296    }
297    }
298 //-------------------------------------------------------------------------
299  test step ts11oeGetCrisisSet order 11{
300     variables{
301       TheActor : actCoordinator
302       AetCrisisStatus : etCrisisStatus
303      }
304     constraints{
305       TheActor=TheSystem.rnactCoordinator
306       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
307       ->any2(true)
308       AetCrisisStatus = pending
309      }
310     test message{
311      out:TheActor sends to system actCoordinator.outactCoordinator.oeGetCrisisSet(AetCrisisStatus)
312     }
313     oracle{
314 //TODO - make consistent with test step implementation by adding Prolog code for input messages
315     variables{
316       ActCrisis:ctCrisis
317     }
```

```
318     constraints{
319      TheActor.inactCoordinator.ieSendACrisis(ActCrisis)
320     }
321    }
322    }
323 //-------------------------------------------------------------------------
324  test step ts12oeSetCrisisHandler order 12{
325     variables{
326       TheActor : actCoordinator
327       AdtCrisisID : dtCrisisID
328      }
329     constraints{
330       TheActor=TheSystem.rnactCoordinator
331       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
332       ->any2(true)
333       //and AdtCrisisID.value= '1'
334      }
335     test message{
336      out:TheActor sends to system actCoordinator.outactCoordinator.oeSetCrisisHandler(AdtCrisisID)
337     }
338     oracle{
339      variables{
340       AMessage:ptString
341       AdtPhoneNumber:dtPhoneNumber
342       AdtSMS:dtSMS
343       ActAlert:ctAlert
344
345       TheComCompany: actComCompany
346       TheCoordinator:actCoordinator
347      }
348      constraints{
349       AMessage = 'You are now considered as handling the crisis !'
350       AdtSMS.value = 'The handling of your alert by our services is in progress !'
351       TheComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
352       TheCoordinator.inactCoordinator.ieSendAnAlert(ActAlert)
353       TheActor.inactAuthenticated.ieMessage(AMessage)
354     }
355    }
356    }
357 //-------------------------------------------------------------------------
358  test step ts13oeSetClock04 order 13{
359     variables{
360       TheActor:actActivator
361       ACurrentClock:dtDateAndTime
362      }
363     constraints{
364       TheActor=TheSystem.rnactActivator->any2(true)
365       ACurrentClock.date.year.value = 2017
366       ACurrentClock.date.month.value = 11
367       ACurrentClock.date.day.value = 26
368       ACurrentClock.time.hour.value = 10
369       ACurrentClock.time.minute.value = 45
370       ACurrentClock.time.second.value = 00
371      }
372     test message{
373      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
374     }
375     oracle{
376      constraints{
377      true
378     }
379    }
380    }
381 //-------------------------------------------------------------------------
382  test step ts14oeValidateAlert order 14{
383     variables{
384       TheActor : actCoordinator
385       AdtAlertID : dtAlertID
386      }
387     constraints{
```

```
388       TheActor=TheSystem.rnactCoordinator
389       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
390       ->any2(true)
391       //and AdtAlertID.value= '1'
392      }
393     test message{
394      out:TheActor sends to system actCoordinator.outactCoordinator.oeValidateAlert(AdtAlertID)
395     }
396     oracle{
397      variables{
398        AMessage:ptString
399      }
400      constraints{
401      AMessage = 'The Alert is now declared as valid !'
402      TheActor.actAuthenticated.inactAuthenticated.ieMessage(AMessage)
403     }
404    }
405    }
406 //-------------------------------------------------------------------------
407  test step ts15oeAlert2 order 15{
408     variables{
409       TheActor : actComCompany
410       AetHumanKind:etHumanKind
411       AdtDate:dtDate
412       AdtTime:dtTime
413       AdtPhoneNumber:dtPhoneNumber
414       AdtGPSLocation:dtGPSLocation
415       AdtComment:dtComment
416      }
417     constraints{
418       TheActor = TheSystem.rnactComCompany->any2(true)
419       AetHumanKind = witness
420       AdtDate.year.value = 2017
421       AdtDate.month.value = 11
422       AdtDate.day.value = 26
423       AdtTime.hour.value = 10
424       AdtTime.minute.value = 20
425       AdtTime.second.value = 00
426       AdtPhoneNumber.value = '+3524666445000'
427       AdtGPSLocation.latitude.value = 49.627095
428       AdtGPSLocation.longitude.value = 6.160251
429       AdtComment.value = 'A car crash just happened.'
430      }
431     test message{
432      out:TheActor
433      sends to system actComCompany.outactComCompany.oeAlert( AetHumanKind,
434                                  AdtDate,
435                                  AdtTime,
436                                  AdtPhoneNumber,
437                                  AdtGPSLocation,
438                                  AdtComment)
439     }
440     oracle{
441      variables{
442       AdtSMS:dtSMS
443      }
444      constraints{
445      AdtSMS.value = 'Your alert has been registered. We will handle it and keep you informed'
446      TheActor.actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
447     }
448     }
449    }
450 //-------------------------------------------------------------------------
451  test step ts16oeSetClock05 order 16{
452     variables{
453       TheActor:actActivator
454       ACurrentClock:dtDateAndTime
455      }
456     constraints{
457       TheActor=TheSystem.rnactActivator->any2(true)
```

```
458       ACurrentClock.date.year.value = 2017
459       ACurrentClock.date.month.value = 11
460       ACurrentClock.date.day.value = 26
461       ACurrentClock.time.hour.value = 12
462       ACurrentClock.time.minute.value = 45
463       ACurrentClock.time.second.value = 00
464      }
465     test message{
466      out:TheActor sends to system actActivator.outactActivator.oeSetClock(ACurrentClock)
467     }
468     oracle{
469      constraints{
470      true
471     }
472    }
473    }
474 //-------------------------------------------------------------------------
475  test step ts17oeSetCrisisStatus order 17{
476     variables{
477       TheActor : actCoordinator
478       AdtCrisisID : dtCrisisID
479       AetCrisisStatus : etCrisisStatus
480      }
481     constraints{
482       TheActor=TheSystem.rnactCoordinator
483       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
484       ->any2(true)
485       //and AdtCrisisID.value= '1'
486       //and AetCrisisStatus = solved
487      }
488     test message{
489      out:TheActor sends to system actCoordinator.outactCoordinator.oeSetCrisisStatus(AdtCrisisID,
     AetCrisisStatus)
490     }
491     oracle{
492      variables{
493       AMessage:ptString
494      }
495      constraints{
496      AMessage = 'The crisis status has been updated !'
497      TheActor.inactAuthenticated.ieMessage(AMessage)
498     }
499    }
500    }
501 //-------------------------------------------------------------------------
502  test step ts18oeReportOnCrisis order 18{
503     variables{
504       TheActor : actCoordinator
505       AdtCrisisID : dtCrisisID
506       AdtComment : dtComment
507      }
508     constraints{
509       TheActor=TheSystem.rnactCoordinator
510       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
511       ->any2(true)
512       //and AdtCrisisID.value= '1'
513       //and AdtComment.value = '3 victims sent to hospital, 2 cars evacuated and 4 rescue unit
     mobilized'
514      }
515     test message{
516      out:TheActor sends to system actCoordinator.outactCoordinator.oeReportOnCrisis(AdtCrisisID,
     AdtComment)
517     }
518     oracle{
519      variables{
520       AMessage:ptString
521      }
522      constraints{
523      AMessage = 'The crisis comment has been updated !'
524      TheActor.inactAuthenticated.ieMessage(AMessage)
```

```
525     }
526    }
527    }
528 //-------------------------------------------------------------------
529  test step ts19oeCloseCrisis order 19{
530     variables{
531       TheActor : actCoordinator
532       AdtCrisisID : dtCrisisID
533      }
534     constraints{
535       TheActor=TheSystem.rnactCoordinator
536       ->select(a | a.rnctCoordinator.login.value.eq('steve'))
537       ->any2(true)
538       //and AdtCrisisID.value= '1'
539      }
540     test message{
541      out:TheActor sends to system actCoordinator.outactCoordinator.oeCloseCrisis(AdtCrisisID)
542     }
543     oracle{
544      variables {
545       AMessage:ptString
546      }
547      constraints{
548      AMessage = 'The crisis is now closed !'
549      TheActor.inactAuthenticated.ieMessage(AMessage)
550     }
551     }
552    }
553   }
554  }
555 }
```

Listing C.48: Messir Spec. file tc-testcase01.msr.

C.49 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      AdtEncodedPassword : testcase01.ts03oeLogin.AdtEncodedPassword= "[B@6979e8cb"
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      AdtPublicKey : testcase01.ts04oeAddCoordinator.AdtPublicKey = "Sun RSA public key, 2048 bits ...
        "
62     }
63     oracle {
64       satisfaction = "true"
65       received message {
66       tsi03.bill received from system actAdministrator.inactAdministrator.ieCoordinatorAdded()
67       }
68    }
69     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
70    }
71 //-------------------------------------------------------
72    test step instance tsi05: testcase01.ts05oeLogout{
73     variables {
74      reuse tsi03.bill as testcase01.ts05oeLogout.TheActor
75     }
76     oracle {
77       satisfaction = "true"
78       received message {
79       AMessage : testcase01.ts05oeLogout.AMessage= 'You are logged out ! Good Bye ...'
80       tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
81       }
82    }
83     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
84    }
85 //-------------------------------------------------------
86    test step instance tsi06: testcase01.ts06oeSetClock02{
87     variables {
88      reuse tsi02.theClock as testcase01.ts06oeSetClock02.TheActor
89      ACurrentClock : testcase01.ts06oeSetClock02.ACurrentClock= "2017:11:26 - 10:15:00"
90     }
91     oracle {
92       satisfaction = "true"
93     }
94      test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
95     }
96 //-------------------------------------------------------
97    test step instance tsi07: testcase01.ts07oeAlert1{
98     variables {
99      tango:testcase01.ts07oeAlert1.TheActor ="tango"
100     AetHumanKind : testcase01.ts07oeAlert1.AetHumanKind = "witness"
101     AdtDate : testcase01.ts07oeAlert1.AdtDate = "2017:11:26"
102     AdtTime : testcase01.ts07oeAlert1.AdtTime = "10:10:16"
```

```
103     AdtPhoneNumber : testcase01.ts07oeAlert1.AdtPhoneNumber = "+3524666445252"
104     AdtGPSLocation : testcase01.ts07oeAlert1.AdtGPSLocation = "49.627675:6.159590"
105     AdtComment : testcase01.ts07oeAlert1.AdtComment = "3 cars involved in an accident."
106     }
107    oracle {
108      satisfaction = "true"
109      received message {
110      AdtSMS : testcase01.ts07oeAlert1.AdtSMS= 'Your alert has been registered. We will handle it and
         keep you informed'
111      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
112
113      }
114    }
115     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
116    }
117
118 //-------------------------------------------------------
119   test step instance tsi08: testcase01.ts08oeSetClock03{
120    variables {
121     reuse tsi02.theClock as testcase01.ts08oeSetClock03.ACurrentClock
122     ACurrentClock : testcase01.ts08oeSetClock03.ACurrentClock = "2017:11:26 - 10:30:00"
123    }
124    oracle {
125      satisfaction = "true"
126    }
127     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
128    }
129 //-------------------------------------------------------
130   test step instance tsi09: testcase01.ts09oeSollicitateCrisisHandling{
131    variables {
132     reuse tsi02.theClock as testcase01.ts09oeSollicitateCrisisHandling.TheActor
133     steve:testcase01.ts09oeSollicitateCrisisHandling.TheCoordinator ="steve"
134     reuse tsi03.bill as testcase01.ts09oeSollicitateCrisisHandling.TheAdministrator
135    }
136    oracle {
137      satisfaction = "true"
138      received message {
139       AMessageForCrisisHandlers : testcase01.ts09oeSollicitateCrisisHandling.
       AMessageForCrisisHandlers= 'There are alerts pending since more than the defined delay. Please
       REACT !'
140
141      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
142      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
143           }
144    }
145     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
146    }
147
148 //-------------------------------------------------------
149   test step instance tsi10: testcase01.ts10oeLogin02{
150    variables {
151     reuse tsi09.steve as testcase01.ts10oeLogin02.TheActor
152     AdtLogin : testcase01.ts10oeLogin02.AdtLogin = "steve"
153     AdtEncodedPassword : testcase01.ts10oeLogin02.AdtEncodedPassword= "[B@6979e8cb"
154    }
155    oracle {
156      satisfaction = "true"
157      received message {
158      AMessage : testcase01.ts10oeLogin02.AMessage= 'You are logged ! Welcome ...'
159      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
160
161      }
162    }
163    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
164    }
165 //-------------------------------------------------------
166   test step instance tsi11: testcase01.ts11oeGetCrisisSet{
167    variables {
```

```
168     reuse tsi09.steve as testcase01.ts11oeGetCrisisSet.TheActor
169     AetCrisisStatus : testcase01.ts11oeGetCrisisSet.AetCrisisStatus = "pending"
170    }
171    oracle {
172      satisfaction = "true"
173      received message {
174      ActCrisis : testcase01.ts11oeGetCrisisSet.ActCrisis= "crisis with ID 1 details"
175      tsi09.steve received from system actCoordinator.inactCoordinator.ieSendACrisis(ActCrisis)
176      }
177    }
178    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
179    }
180 //-------------------------------------------------------
181   test step instance tsi12: testcase01.ts12oeSetCrisisHandler{
182    variables {
183     reuse tsi09.steve as testcase01.ts12oeSetCrisisHandler.TheActor
184     AdtCrisisID : testcase01.ts12oeSetCrisisHandler.AdtCrisisID = "1"
185
186      reuse tsi07.tango as testcase01.ts12oeSetCrisisHandler.TheComCompany
187
188    }
189    oracle {
190      satisfaction = "true"
191      received message {
192      AMessage : testcase01.ts12oeSetCrisisHandler.AMessage= 'You are now considered as handling the
        crisis !'
193      AdtSMS : testcase01.ts12oeSetCrisisHandler.AdtSMS= 'The handling of your alert by our services
             is in progress !'
194      AdtPhoneNumber : testcase01.ts12oeSetCrisisHandler.AdtPhoneNumber= "+3524666445252"
195
196      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
197      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
198
199      }
200    }
201     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
202    }
203 //-------------------------------------------------------
204   test step instance tsi13: testcase01.ts13oeSetClock04{
205    variables {
206     reuse tsi02.theClock as testcase01.ts13oeSetClock04.TheActor
207      ACurrentClock : testcase01.ts13oeSetClock04.ACurrentClock = "2017:11:26 - 10:45:00"
208    }
209    oracle {
210      satisfaction = "true"
211    }
212    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
213    }
214 //-------------------------------------------------------
215   test step instance tsi14: testcase01.ts14oeValidateAlert{
216    variables {
217     reuse tsi09.steve as testcase01.ts14oeValidateAlert.TheActor
218     AdtAlertID : testcase01.ts14oeValidateAlert.AdtAlertID = "1"
219     }
220    oracle {
221      satisfaction = "true"
222      received message {
223      AMessage : testcase01.ts14oeValidateAlert.AMessage= 'The Alert is now declared as valid !'
224      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
225
226      }
227    }
228     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
229    }
230 //-------------------------------------------------------
231   test step instance tsi15: testcase01.ts15oeAlert2{
232    variables {
233     reuse tsi07.tango as testcase01.ts15oeAlert2.TheActor
234     AetHumanKind : testcase01.ts15oeAlert2.AetHumanKind ="witness"
235     AdtDate : testcase01.ts15oeAlert2.AdtDate= "2017:11:26"
```

```
236     AdtTime : testcase01.ts15oeAlert2.AdtTime= "10:20:00"
237     AdtPhoneNumber : testcase01.ts15oeAlert2.AdtPhoneNumber= "+3524666445000"
238     AdtGPSLocation : testcase01.ts15oeAlert2.AdtGPSLocation= "49.627095:6.160251"
239     AdtComment : testcase01.ts15oeAlert2.AdtComment= "A car crash just happened."
240    }
241    message {
242     tsi07.tango sent to system testcase01.ts15oeAlert2.out : actComCompany.outactComCompany.oeAlert(
           AetHumanKind,AdtDate,AdtTime,AdtPhoneNumber,AdtGPSLocation,AdtComment)
243
244    }
245    oracle {
246      satisfaction = "true"
247      received message {
248      AdtSMS : testcase01.ts15oeAlert2.AdtSMS= 'Your alert has been registered. We will handle it and
         keep you informed'
249      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
250
251      }
252    }
253     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
254    }
255 //-------------------------------------------------------
256   test step instance tsi16: testcase01.ts16oeSetClock05{
257    variables {
258     reuse tsi02.theClock as testcase01.ts16oeSetClock05.TheActor
259     ACurrentClock : testcase01.ts16oeSetClock05.ACurrentClock = "2017:11:26 - 12:45:00"
260    }
261    oracle {
262      satisfaction = "true"
263      received message {
264
265      }
266    }
267     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
268    }
269 //-------------------------------------------------------
270   test step instance tsi17: testcase01.ts17oeSetCrisisStatus{
271    variables {
272     reuse tsi09.steve as testcase01.ts17oeSetCrisisStatus.TheActor
273     AdtCrisisID : testcase01.ts17oeSetCrisisStatus.AdtCrisisID = "1"
274     AetCrisisStatus : testcase01.ts17oeSetCrisisStatus.AetCrisisStatus= "solved"
275    }
276    oracle {
277      satisfaction = "true"
278      received message {
279      AMessage : testcase01.ts17oeSetCrisisStatus.AMessage= "The crisis status has been updated !"
280      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
281      }
282    }
283     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
284    }
285 //-------------------------------------------------------
286   test step instance tsi18: testcase01.ts18oeReportOnCrisis{
287    variables {
288     reuse tsi09.steve as testcase01.ts18oeReportOnCrisis.TheActor
289     AdtCrisisID : testcase01.ts18oeReportOnCrisis.AdtCrisisID = "1"
290     AdtComment : testcase01.ts18oeReportOnCrisis.AdtComment= "3 victims sent to hospital, 2 cars
        evacuated and 4 rescue unit mobilized"
291    }
292    oracle {
293      satisfaction = "true"
294      received message {
295      AMessage : testcase01.ts18oeReportOnCrisis.AMessage= 'The crisis comment has been updated !'
296      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
297
298      }
299    }
300     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
301    }
302 //-------------------------------------------------------
```

```
303   test step instance tsi19: testcase01.ts19oeCloseCrisis{
304    variables {
305     reuse tsi09.steve as testcase01.ts19oeCloseCrisis.TheActor
306     AdtCrisisID : testcase01.ts19oeCloseCrisis.AdtCrisisID = "1"
307    }
308    oracle {
309      satisfaction = "true"
310      received message {
311      AMessage : testcase01.ts19oeCloseCrisis.AMessage= 'The crisis is now closed !'
312
313     tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
314
315      }
316    }
317     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
318    }
319
320   }
321 //-------------------------------------------------------
322 //-------------------------------------------------------
323 //-------------------------------------------------------
324  test case instance instance01Part01:testcase01{
325 //-------------------------------------------------------
326   test step instance tsi01:testcase01.ts01oeCreateSystemAndEnvironment{
327    variables {
328     theCreator:testcase01.ts01oeCreateSystemAndEnvironment.Creator = "theCreator"
329     AqtyComCompanies : testcase01.ts01oeCreateSystemAndEnvironment.AqtyComCompanies="4"
330     }
331     oracle {
332      satisfaction = "true"
333     }
334     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
335    }
336 //-------------------------------------------------------
337   test step instance tsi02: testcase01.ts02oeSetClock{
338    variables {
339     theClock:testcase01.ts02oeSetClock.TheActor = "theClock"
340     ACurrentClock : testcase01.ts02oeSetClock.ACurrentClock= "2017:11:24 - 03:20:00"
341     }
342     oracle {
343      satisfaction = "true"
344     }
345     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
346    }
347 //-------------------------------------------------------
348   test step instance tsi03: testcase01.ts03oeLogin{
349    variables {
350     bill:testcase01.ts03oeLogin.TheActor="bill"
351     AdtLogin : testcase01.ts03oeLogin.AdtLogin= "icrashadmin"
352     AdtEncodedPassword : testcase01.ts03oeLogin.AdtEncodedPassword= "   _   t   9   >   ;      (
              *W!5      YT      "
353     }
354     oracle {
355      satisfaction = "true"
356      received message {
357       AMessage : testcase01.ts03oeLogin.AMessage= 'You are logged ! Welcome ...'
358       tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
359      }
360     }
361     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
362    }
363 //-------------------------------------------------------
364   test step instance tsi04: testcase01.ts04oeAddCoordinator{
365    variables {
366     reuse tsi03.bill as testcase01.ts04oeAddCoordinator.TheActor
367     AdtCoordinatorID : testcase01.ts04oeAddCoordinator.AdtCoordinatorID = "1"
368     AdtLogin :testcase01.ts04oeAddCoordinator.AdtLogin= "steve"
369     AdtPassword : testcase01.ts04oeAddCoordinator.AdtPassword = "pwdMessirExcalibur2017"
370     AdtPublicKey :testcase01.ts04oeAddCoordinator.AdtPublicKey = "Sun RSA public key, 2048 bits ..."
371    }
```

```
372    oracle {
373      satisfaction = "true"
374      received message {
375      tsi03.bill received from system actAdministrator.inactAdministrator.ieCoordinatorAdded()
376      }
377    }
378     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
379    }
380 //-------------------------------------------------------
381   test step instance tsi05: testcase01.ts05oeLogout{
382    variables {
383     reuse tsi03.bill as testcase01.ts05oeLogout.TheActor
384    }
385    oracle {
386      satisfaction = "true"
387      received message {
388      AMessage : testcase01.ts05oeLogout.AMessage= 'You are logged out ! Good Bye ...'
389      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
390      }
391    }
392     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
393    }
394 //-------------------------------------------------------
395   test step instance tsi06: testcase01.ts06oeSetClock02{
396    variables {
397     reuse tsi02.theClock as testcase01.ts06oeSetClock02.TheActor
398     ACurrentClock : testcase01.ts06oeSetClock02.ACurrentClock= "2017:11:26 - 10:15:00"
399    }
400    oracle {
401      satisfaction = "true"
402    }
403     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
404    }
405 //-------------------------------------------------------
406   test step instance tsi07: testcase01.ts07oeAlert1{
407    variables {
408     tango:testcase01.ts07oeAlert1.TheActor ="tango"
409     AetHumanKind : testcase01.ts07oeAlert1.AetHumanKind = "witness"
410     AdtDate : testcase01.ts07oeAlert1.AdtDate = "2017:11:26"
411     AdtTime : testcase01.ts07oeAlert1.AdtTime = "10:10:16"
412     AdtPhoneNumber : testcase01.ts07oeAlert1.AdtPhoneNumber = "+3524666445252"
413     AdtGPSLocation : testcase01.ts07oeAlert1.AdtGPSLocation = "49.627675:6.159590"
414     AdtComment : testcase01.ts07oeAlert1.AdtComment = "3 cars involved in an accident."
415     }
416    oracle {
417      satisfaction = "true"
418      received message {
419      AdtSMS : testcase01.ts07oeAlert1.AdtSMS= 'Your alert has been registered. We will handle it and
         keep you informed'
420      tsi07.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
421
422      }
423    }
424     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
425    }
426
427 //-------------------------------------------------------
428   test step instance tsi08: testcase01.ts08oeSetClock03{
429    variables {
430     reuse tsi02.theClock as testcase01.ts08oeSetClock03.ACurrentClock
431     ACurrentClock : testcase01.ts08oeSetClock03.ACurrentClock = "2017:11:26 - 10:30:00"
432    }
433    oracle {
434      satisfaction = "true"
435    }
436     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
437    }
438 //-------------------------------------------------------
439   test step instance tsi09: testcase01.ts09oeSollicitateCrisisHandling{
440    variables {
```

```
441     reuse tsi02.theClock as testcase01.ts09oeSollicitateCrisisHandling.TheActor
442     steve:testcase01.ts09oeSollicitateCrisisHandling.TheCoordinator ="steve"
443     reuse tsi03.bill as testcase01.ts09oeSollicitateCrisisHandling.TheAdministrator
444    }
445    oracle {
446      satisfaction = "true"
447      received message {
448       AMessageForCrisisHandlers : testcase01.ts09oeSollicitateCrisisHandling.
       AMessageForCrisisHandlers= 'There are alerts pending since more than the defined delay. Please
       REACT !'
449
450      tsi03.bill received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
451      tsi09.steve received from system actAuthenticated.inactAuthenticated.ieMessage(
       AMessageForCrisisHandlers)
452           }
453    }
454     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
455    }
456    }
457
458 //-------------------------------------------------------
459 //-------------------------------------------------------
460 //-------------------------------------------------------
461  test case instance instance01Part02:testcase01{
462
463   test step instance tsi10: testcase01.ts10oeLogin02{
464    variables {
465     steve : testcase01.ts10oeLogin02.TheActor
466     AdtLogin : testcase01.ts10oeLogin02.AdtLogin = "steve"
467     AdtEncodedPassword : testcase01.ts10oeLogin02.AdtEncodedPassword= "[B@6979e8cb"
468    }
469    oracle {
470      satisfaction = "true"
471      received message {
472      AMessage : testcase01.ts10oeLogin02.AMessage= 'You are logged ! Welcome ...'
473      steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
474
475      }
476    }
477    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
478    }
479 //-------------------------------------------------------
480   test step instance tsi11: testcase01.ts11oeGetCrisisSet{
481    variables {
482     reuse tsi10.steve as testcase01.ts11oeGetCrisisSet.TheActor
483     AetCrisisStatus : testcase01.ts11oeGetCrisisSet.AetCrisisStatus = "pending"
484    }
485    oracle {
486      satisfaction = "true"
487      received message {
488      ActCrisis : testcase01.ts11oeGetCrisisSet.ActCrisis= "crisis with ID 1 details"
489      tsi10.steve received from system actCoordinator.inactCoordinator.ieSendACrisis(ActCrisis)
490      }
491    }
492    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
493    }
494 //-------------------------------------------------------
495   test step instance tsi12: testcase01.ts12oeSetCrisisHandler{
496    variables {
497     reuse tsi10.steve as testcase01.ts12oeSetCrisisHandler.TheActor
498     AdtCrisisID : testcase01.ts12oeSetCrisisHandler.AdtCrisisID = "1"
499     tango : testcase01.ts12oeSetCrisisHandler.TheComCompany
500    }
501    oracle {
502      satisfaction = "true"
503      received message {
504      AMessage : testcase01.ts12oeSetCrisisHandler.AMessage= 'You are now considered as handling the
       crisis !'
505      AdtSMS : testcase01.ts12oeSetCrisisHandler.AdtSMS= 'The handling of your alert by our services
```

```
      is in progress !'
506    AdtPhoneNumber : testcase01.ts12oeSetCrisisHandler.AdtPhoneNumber= "+3524666445252"
507
508    tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
509    tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
510
511    }
512   }
513    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
514   }
515 //-------------------------------------------------------
516  test step instance tsi13: testcase01.ts13oeSetClock04{
517   variables {
518    theClock : testcase01.ts13oeSetClock04.TheActor
519    ACurrentClock : testcase01.ts13oeSetClock04.ACurrentClock = "2017:11:26 - 10:45:00"
520   }
521   oracle {
522    satisfaction = "true"
523   }
524   test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
525   }
526 //-------------------------------------------------------
527  test step instance tsi14: testcase01.ts14oeValidateAlert{
528   variables {
529    reuse tsi10.steve as testcase01.ts14oeValidateAlert.TheActor
530    AdtAlertID : testcase01.ts14oeValidateAlert.AdtAlertID = "1"
531    }
532   oracle {
533    satisfaction = "true"
534    received message {
535    AMessage : testcase01.ts14oeValidateAlert.AMessage= 'The Alert is now declared as valid !'
536    tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
537
538    }
539   }
540    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
541   }
542 //-------------------------------------------------------
543  test step instance tsi15: testcase01.ts15oeAlert2{
544   variables {
545    reuse tsi12.tango as testcase01.ts15oeAlert2.TheActor
546    AetHumanKind : testcase01.ts15oeAlert2.AetHumanKind ="witness"
547    AdtDate : testcase01.ts15oeAlert2.AdtDate= "2017:11:26"
548    AdtTime : testcase01.ts15oeAlert2.AdtTime= "10:20:00"
549    AdtPhoneNumber : testcase01.ts15oeAlert2.AdtPhoneNumber= "+3524666445000"
550    AdtGPSLocation : testcase01.ts15oeAlert2.AdtGPSLocation= "49.627095:6.160251"
551    AdtComment : testcase01.ts15oeAlert2.AdtComment= "A car crash just happened."
552   }
553   message {
554    tsi12.tango sent to system testcase01.ts15oeAlert2.out : actComCompany.outactComCompany.oeAlert(
       AetHumanKind,AdtDate,AdtTime,AdtPhoneNumber,AdtGPSLocation,AdtComment)
555
556   }
557   oracle {
558    satisfaction = "true"
559    received message {
560    AdtSMS : testcase01.ts15oeAlert2.AdtSMS= 'Your alert has been registered. We will handle it and
       keep you informed'
561    tsi12.tango received from system actComCompany.inactComCompany.ieSmsSend(AdtPhoneNumber,AdtSMS)
562
563    }
564   }
565    test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
566   }
567 //-------------------------------------------------------
568  test step instance tsi16: testcase01.ts16oeSetClock05{
569   variables {
570    reuse tsi13.theClock as testcase01.ts16oeSetClock05.TheActor
571    ACurrentClock : testcase01.ts16oeSetClock05.ACurrentClock = "2017:11:26 - 12:45:00"
572   }
```

```
573    oracle {
574      satisfaction = "true"
575      received message {
576
577      }
578    }
579     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
580    }
581 //-------------------------------------------------------
582   test step instance tsi17: testcase01.ts17oeSetCrisisStatus{
583    variables {
584     reuse tsi10.steve as testcase01.ts17oeSetCrisisStatus.TheActor
585     AdtCrisisID : testcase01.ts17oeSetCrisisStatus.AdtCrisisID = "1"
586     AetCrisisStatus : testcase01.ts17oeSetCrisisStatus.AetCrisisStatus= "solved"
587    }
588    oracle {
589      satisfaction = "true"
590      received message {
591      AMessage : testcase01.ts17oeSetCrisisStatus.AMessage= "The crisis status has been updated !"
592      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
593      }
594    }
595     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
596    }
597 //-------------------------------------------------------
598   test step instance tsi18: testcase01.ts18oeReportOnCrisis{
599    variables {
600     reuse tsi10.steve as testcase01.ts18oeReportOnCrisis.TheActor
601     AdtCrisisID : testcase01.ts18oeReportOnCrisis.AdtCrisisID = "1"
602     AdtComment : testcase01.ts18oeReportOnCrisis.AdtComment= "3 victims sent to hospital, 2 cars
       evacuated and 4 rescue unit mobilized"
603    }
604    oracle {
605      satisfaction = "true"
606      received message {
607      AMessage : testcase01.ts18oeReportOnCrisis.AMessage= 'The crisis comment has been updated !'
608      tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
609
610      }
611    }
612     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
613    }
614 //-------------------------------------------------------
615   test step instance tsi19: testcase01.ts19oeCloseCrisis{
616    variables {
617     reuse tsi10.steve as testcase01.ts19oeCloseCrisis.TheActor
618     AdtCrisisID : testcase01.ts19oeCloseCrisis.AdtCrisisID = "1"
619    }
620    oracle {
621      satisfaction = "true"
622      received message {
623      AMessage : testcase01.ts19oeCloseCrisis.AMessage= 'The crisis is now closed !'
624
625     tsi10.steve received from system actAuthenticated.inactAuthenticated.ieMessage(AMessage)
626
627      }
628    }
629     test results {pre-protocol = "true" pre-functional = "true" post-functional = "true"}
630    }
631
632   }
633  }
634
635 }
```

Listing C.49: Messir Spec. file tci-testcase01-instance01.msr.

C.50 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","[B@6979e8cb"){
61        ieMessage('You are logged ! Welcome ...') returned to bill
62       }
63 //-------------------------------------------------------
64    bill
65    executed instanceof subfunction
66       oeAddCoordinator("1","steve","pwdMessirExcalibur2017", "Sun RSA public key, 2048 bits ..."){
```

```
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","[B@6979e8cb"){
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","[B@6979e8cb"){
191         ieMessage('You are logged ! Welcome ...') returned to bill
192        }
193 //-------------------------------------------------------
194     bill
195     executed instanceof subfunction
196        oeAddCoordinator("1","steve","pwdMessirExcalibur2017", "Sun RSA public key, 2048 bits..."){
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","[B@6979e8cb"){
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.50: Messir Spec. file usecase-suDeployAndRun.msr.

C.51 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.51: Messir Spec. file usecase-suGlobalCrisisHandling.msr.

C.52 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
17   step a: actAdministrator
18      executes ugSecurelyUseSystem
19   step b: actAdministrator
20      executes oeAddCoordinator
21   step c: actAdministrator
22      executes oeDeleteCoordinator
23
24   ordering constraint
25     "steps (a) (b) and (c) executions are interleaved
26     (steps (b) and (c) have their protocol constrained
27      by steps of (a))."
28   ordering constraint
29     "steps (a) (b) and (c) can be executed multiple times."
30  }
31 }
32 }
```

Listing C.52: Messir Spec. file usecase-ugAdministrateTheSystem.msr.

C.53 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.53: Messir Spec. file usecase-ugManageCrisis.msr.

C.54 File ./src-gen/messir-spec/usecases/usecase-ugMonitor.msr

```
1 package icrash.usecases.ugMonitor {
2
3  import icrash.environment
4  import icrash.usecases.subfunctions
5
6  Use Case Model {
7   use case system usergoal ugMonitor() {
8    actor icrash.environment.actCoordinator[primary,active]
9
10   reuse oeGetCrisisSet[0..*]
11   reuse oeGetAlertsSet[0..*]
12
13   step a: icrash.environment.actCoordinator executes oeGetAlertsSet
14   step b: icrash.environment.actCoordinator executes oeGetCrisisSet
15  }
16 }
17 }
```

Listing C.54: Messir Spec. file usecase-ugMonitor.msr.

C.55 File ./src-gen/messir-spec/usecases/usecase-ugSecurelyUseSystem.msr

```
1 package icrash.usecases.ugSecurelyUseSystem {
2
3 import icrash.environment
4 import icrash.usecases.subfunctions
5
6 Use Case Model {
7
8 use case system usergoal
9  ugSecurelyUseSystem() {
10
11  actor actAuthenticated[primary,active]
12
13  reuse oeLogin[1..1]
14  reuse oeLogout[1..1]
15
16  step a: actAuthenticated
17     executes oeLogin
18  step b: actAuthenticated
19     executes oeLogout
20
```

```
21 ordering constraint
22  "step (a) must always precede step (b)."
23  }
24 }
25 }
```

Listing C.55: Messir Spec. file usecase-ugSecurelyUseSystem.msr.

C.56 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","   _   t   9   >   ;      (         *W!5      YT      "){
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.56: Messir Spec. file usecaseinstance-ugSecurelyUseSystem-uciugSecurelyUseSystem.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/outactActi 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
    ],
    []):-

```

```
msrVar(ctState,TheSystem),

/* Post Functional:*/

msrNav([Self],[rnActor,rnSystem],[TheSystem]),

/* PostF01 */
msrNav([TheSystem],
    [msmAtPost,clock],
    [AcurrentClock]),

 /* Post Protocol:*/
/* PostP01 */
true
 .
```

Listing D.1: Prolog file outactActivator-oeSetClock.pl.

D.2 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outact oeSollicitateCrisisHandling.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------

msrop(outactActivator,
   oeSollicitateCrisisHandling,
   [preProtocol,Self
    ],
   []):-
/* Pre Protocol:*/
msrVar(ctState,TheSystem),
msrNav([Self],[rnActor,rnSystem],[TheSystem]),

msrVarCol(ctCrisis,_,ColctCrisisToHandle),

/* PreP01 */
msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]]),

/* PreP02 */
msrNav([TheSystem],
    [rnctCrisis,msrSelect,
     handlingDelayPassed,[]
    ],
    ColctCrisisToHandle),

msrNav(ColctCrisisToHandle,
    [msrSize,geq,[[ptInteger,1]]],
    [[ptBoolean,true]])
.

msrop(outactActivator,
   oeSollicitateCrisisHandling,
   [preFunctional,Self
    ],
   []):-
/* Pre Functional:*/
/* PreF01 */
true.

msrop(outactActivator,
   oeSollicitateCrisisHandling,
   [post,Self
    ],
```

```
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/outact oeAddCoordinator.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),
```

```
msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

msrVar(actCoordinator,TheactCoordinator),
msrVar(ctCoordinator,ThectCoordinator),

/* PostF01 */
msrNav([TheactCoordinator],
    [init,[]],
    [[ptBoolean,true]]),

/* PostF02 */
msrNav([ThectCoordinator],
    [init,[AdtCoordinatorID,AdtLogin,AdtPassword]],
    [[ptBoolean,true]]),

/* PostF03 */
msrNav([TheactCoordinator],
    [msmAtPost,rnctCoordinator],
    [ThectCoordinator]),

/* PostF04 */
 msrNav([ThectCoordinator],
    [msmAtPost,rnactAuthenticated],
    [TheactCoordinator]),

/* PostF05 */
 msrNav([TheActor],
    [rnInterfaceIN,
     ieCoordinatorAdded,[]],
    [[ptBoolean,true]]),

 /* Post Protocol:*/
/* PostP01 */
true
.
```

Listing D.3: Prolog file outactAdministrator-oeAddCoordinator.pl.

D.4 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactAdn oeDeleteCoordinator.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------
msrop(outactAdministrator,
   oeDeleteCoordinator,
   [preProtocol,Self,
    AdtCoordinatorID
    ],
   []):-
/* Pre Protocol:*/
msrVar(ctState,TheSystem),
msrVar(actAdministrator,TheActor),
msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PreP01 */
msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]]),

msrNav([TheActor],
    [rnctAuthenticated,vpIsLogged],
    [[ptBoolean,true]])
.
```

```
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/outact... 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/outact 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],
```

```
    [[ptBoolean,true]])
 .

msrop(outactAuthenticated,
   oeLogout,
   [preFunctional,Self
    ],
   []):-
/* Pre Functional:*/
/* PreF01 */
true
.

msrop(outactAuthenticated,
   oeLogout,
   [post,Self
    ],
    []):-

 msrVar(ctState,TheSystem),
 msrVar(actAuthenticated,TheactAuthenticated),

 msrVar(ptString,AptStringMessageForTheactAuthenticated),

/* Post Functional:*/
 msrNav([Self],[rnActor],[TheactAuthenticated]),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),

/* PostF01 */
 msrNav([AptStringMessageForTheactAuthenticated],
    [eq,[[ptString,'You are logged out ! Good Bye ...']]],
    [[ptBoolean,true]]),
 msrNav([TheactAuthenticated],
    [rnInterfaceIN,
     ieMessage,[AptStringMessageForTheactAuthenticated]],
    [[ptBoolean,true]]),

 /* Post Protocol:*/
/* PostP01 */
msrNav([TheactAuthenticated],
    [rnctAuthenticated,msmAtPost,vpIsLogged],
    [[ptBoolean,false]])
.
```

Listing D.6: Prolog file outactAuthenticated-oeLogout.pl.

D.7 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCom oeAlert.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------
nico(A):-
 trace,
 write('here'),
 write('\n').

msrop(outactComCompany,
   oeAlert,
   [preProtocol,Self,
    AetHumanKind,
    AdtDate,
    AdtTime,
    AdtPhoneNumber,
    AdtGPSLocation,
    AdtComment
```

```
    ],
   []):-
/* Pre Protocol:*/
 msrVar(ctState,TheSystem),
 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
/* PreP01 */
 msrNav([TheSystem],
    [vpStarted],
    [[ptBoolean,true]])
 .

msrop(outactComCompany,
   oeAlert,
   [preFunctional,Self,
    AetHumanKind,
    AdtDate,
    AdtTime,
    AdtPhoneNumber,
    AdtGPSLocation,
    AdtComment
    ],
   []):-
/* Pre Functional:*/
/* PreF01 */
 msrVar(ctState,TheSystem),
 msrNav([Self],
     [msmAtPre,rnActor,rnSystem],
     [TheSystem]),

 ( msrNav([TheSystem],[clock,date,gt,[AdtDate]],[[ptBoolean,true]])
  ; (msrNav([TheSystem],[clock,date,eq,[AdtDate]],[[ptBoolean,true]])
   , msrNav([TheSystem],[clock,time,gt,[AdtTime]],[[ptBoolean,true]])
   )
 )
.

msrop(outactComCompany,
   oeAlert,
   [post,Self,
    AetHumanKind,
    AdtDate,
    AdtTime,
    AdtPhoneNumber,
    AdtGPSLocation,
    AdtComment
    ],
    []):-

 msrVar(ctState,TheSystem),
 msrVar(ctHuman,ActHuman),
 msrVar(actComCompany,TheactComCompany),
 msrVar(ctAlert,ActAlert),
 msrVar(dtDateAndTime,AAlertInstant),
 msrVar(etAlertStatus,AetAlertStatus),
% msrVar(ctAlert,ActAlertNearBy),
 msrVar(ctCrisis,ActCrisis),
 msrVar(dtCrisisID,AdtCrisisID),
% msrVar(etCrisisType,AetCrisisType),
 msrVar(etCrisisStatus,AetCrisisStatus),
 msrVar(dtDateAndTime,ACrisisInstant),
 msrVar(dtComment,ACrisisdtComment),
% msrVar(ptString,AptStringMessage),
 msrVar(dtSMS,AdtSMS),
 msrVar(dtAlertID,AdtAlertID),

% msrVar(ptInteger,TheNextptIntegerValue),
% msrVar(ptInteger,UpdatedNextptIntegerValue),
% msrVar(inactComCompany,TheComCompanyIN),
% msrVar(dtComment,TheCommentStored),
% 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/outactCoo oeCloseCrisis.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%-----------------------------------------------------
6msrop(outactCoordinator,
7   oeCloseCrisis,
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/* PreP02 */
24 msrNav([TheActor],
25    [rnctAuthenticated,vpIsLogged],
26    [[ptBoolean,true]])
27.
28
29msrop(outactCoordinator,
30   oeCloseCrisis,
31   [preFunctional,Self,
32    AdtCrisisID
33    ],
34   []):-
35/* Pre Functional:*/
36 msrVar(ctState,TheSystem),
37 msrVar(actCoordinator,TheActor),
38
39 msrVar(dtCrisisID,AdtCrisisID),
40
41 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
42 msrNav([Self],[rnActor],[TheActor]),
43
44/* PreF01 */
45 msrNav([TheSystem],
46    [rnctCrisis,
47     msrSelect,
48     id,eq,[AdtCrisisID]
49    ],
50    ColCrisis),
51
52 msrNav(ColCrisis,
53     [msrSize,eq,[[ptInteger,1]]],
54     [[ptBoolean,true]])
55 .
56
57msrop(outactCoordinator,
58   oeCloseCrisis,
59   [post,Self,
60    AdtCrisisID
61    ],
```

```
62    []):-
63
64/* Post Functional:*/
65 msrVar(ctState,TheSystem),
66 msrVar(actCoordinator,TheActor),
67
68 msrVar(ctCrisis,TheCrisis),
69 msrVar(dtCrisisID,AdtCrisisID),
70
71 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
72 msrNav([Self],[rnActor],[TheActor]),
73
74/* PostF01 */
75 msrNav([TheSystem],
76    [rnctCrisis,
77     msrSelect,
78     id,eq,[AdtCrisisID]],
79    [TheCrisis]),
80
81 msrNav([TheCrisis],
82     [msmAtPost,status],
83     [[etCrisisStatus,closed]]),
84
85/* PostF02 */
86 msrNav([TheCrisis],
87     [msmAtPost,rnHandler],
88     []),
89
90 /* PostF03 */
91 msrNav([TheCrisis],
92     [rnAlerts,msrForAll,msrIsKilled],
93     [[ptBoolean,true]]),
94
95/* PostF04 */
96 msrNav([TheActor],
97     [rnInterfaceIN,
98     ieMessage,[[ptString,'The crisis is now closed !']]
99     ],
100    [[ptBoolean,true]]),
101
102/* Post Protocol:*/
103/* PostP01 */
104 true
105 .
```

Listing D.8: Prolog file outactCoordinator-oeCloseCrisis.pl.

D.9 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outact 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 */
```

```
msrNav([TheSystem],
   [vpStarted],
   [[ptBoolean,true]]),

msrNav([TheActor],
   [rnctAuthenticated,vpIsLogged],
   [[ptBoolean,true]])
.

msrop(outactCoordinator,
  oeGetAlertsSet,
  [preFunctional,Self,
   AetAlertStatus
   ],
  []):-
/* Pre Functional:*/
/* PreF01 */
true
 .

msrop(outactCoordinator,
  oeGetAlertsSet,
  [post,Self,
   AetAlertStatus
   ],
   []):-

/* Post Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),
msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
msrNav([TheSystem],
   [rnctAlert,
    msrSelect,
    status,etEq,[AetAlertStatus]],
   ColAlertSet),

msrNav(ColAlertSet,
    [msrForAll,isSentToCoordinator,[TheActor]],
    [[ptBoolean,true]]),

/* Post Protocol:*/
/* PostP01 */
true
.
```

Listing D.9: Prolog file outactCoordinator-oeGetAlertsSet.pl.

D.10 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCo oeGetCrisisSet.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------
msrop(outactCoordinator,
  oeGetCrisisSet,
  [preProtocol,Self,
   AetCrisisStatus
   ],
  []):-
/* 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,
  oeGetCrisisSet,
  [preFunctional,Self,
   AetCrisisStatus
   ],
  []):-
/* Pre Functional:*/
/* PreF01 */
true
 .

msrop(outactCoordinator,
  oeGetCrisisSet,
  [post,Self,
   AetCrisisStatus
   ],
   []):-

/* Post Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),
msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
msrNav([TheSystem],
   [rnctCrisis,
    msrSelect,
    status,etEq,[AetCrisisStatus]],
   ColCrisisSet),

msrNav(ColCrisisSet,
    [msrForAll,isSentToCoordinator,[TheActor]],
    [[ptBoolean,true]]),

 /* Post Protocol:*/
/* PostP01 */
 true
 .
```

Listing D.10: Prolog file outactCoordinator-oeGetCrisisSet.pl.

D.11 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outac oeInvalidateAlert.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------
msrop(outactCoordinator,
  oeInvalidateAlert,
  [preProtocol,Self,
   AdtAlertID
   ],
```

```
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
29 msrop(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
57 msrop(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
```

```
msrNav([TheAlert],
    [msmAtPost,status],
    [[etAlertStatus,invalid]]),

/* PostF02 */
msrNav([TheActor],
    [rnInterfaceIN,
    ieMessage,[[ptString,'The alert is now declared as invalid !']]
    ],
    [[ptBoolean,true]]),

/* Post Protocol:*/
/* PostP01 */
true
.
```

Listing D.11: Prolog file outactCoordinator-oeInvalidateAlert.pl.

D.12 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outac oeReportOnCrisis.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------
msrop(outactCoordinator,
  oeReportOnCrisis,
  [preProtocol,Self,
   AdtCrisisID,
   AdtComment
   ],
  []):-
/* 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,
  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,
```

```
48      msrSelect,
49      id,eq,[AdtCrisisID]
50     ],
51     ColCrisis),
52
53  msrNav(ColCrisis,
54      [msrSize,eq,[[ptInteger,1]]],
55      [[ptBoolean,true]])
56  .
57
58 msrop(outactCoordinator,
59    oeReportOnCrisis,
60    [post,Self,
61     AdtCrisisID,
62     AdtComment
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(dtComment,AdtComment),
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,comment],
86      [AdtComment]),
87
88  msrNav([TheActor],
89      [rnInterfaceIN,
90      ieMessage,[[ptString,'The crisis comment has been updated !']]
91      ],
92      [[ptBoolean,true]]),
93
94 /* Post Protocol:*/
95 /* PostP01 */
96  true
97  .
```

Listing D.12: Prolog file outactCoordinator-oeReportOnCrisis.pl.

D.13 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCo oeSetCrisisHandler.pl

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 /* DISCONTIGUOUS PREDICATES */
3 :- multifile msrop/4.
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5 %-----------------------------------------------------
6 msrop(outactCoordinator,
7    oeSetCrisisHandler,
8    [preProtocol,Self,
9     AdtCrisisID
10    ],
11   []):-
12 /* 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,
  oeSetCrisisHandler,
  [preFunctional,Self,
   AdtCrisisID
   ],
  []):-
/* 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,
  oeSetCrisisHandler,
  [post,Self,
   AdtCrisisID
   ],
   []):-

/* Post Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),
msrVar(ctCoordinator,TheCoordinator),
msrVar(ctCoordinator,TheCurrentHandler),

msrVar(ctCrisis,TheCrisis),
msrVar(dtCrisisID,AdtCrisisID),

msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
msrNav([TheSystem],
   [rnctCrisis,
    msrSelect,
    id,eq,[AdtCrisisID]],
   [TheCrisis]),

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.13: Prolog file outactCoordinator-oeSetCrisisHandler.pl.

D.14 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCo oeSetCrisisStatus.pl

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 /* DISCONTIGUOUS PREDICATES */
3 :- multifile msrop/4.
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5 %-----------------------------------------------------------
6 msrop(outactCoordinator,
7    oeSetCrisisStatus,
8    [preProtocol,Self,
9     AdtCrisisID,
10    AetCrisisStatus
11    ],
12   []):-
13 /* Pre Protocol:*/
14 msrVar(ctState,TheSystem),
15 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,
  oeSetCrisisStatus,
  [preFunctional,Self,
   AdtCrisisID,
   AetCrisisStatus
   ],
  []):-
/* 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,
  oeSetCrisisStatus,
  [post,Self,
   AdtCrisisID,
   AetCrisisStatus
   ],
   []):-

/* Post Functional:*/
msrVar(ctState,TheSystem),
msrVar(actCoordinator,TheActor),

msrVar(ctCrisis,TheCrisis),
msrVar(dtCrisisID,AdtCrisisID),
msrVar(etCrisisStatus,AetCrisisStatus),

msrNav([Self],[rnActor,rnSystem],[TheSystem]),
msrNav([Self],[rnActor],[TheActor]),

/* PostF01 */
msrNav([TheSystem],
   [rnctCrisis,
    msrSelect,
    id,eq,[AdtCrisisID]],
   [TheCrisis]),

msrNav([TheCrisis],
    [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.14: Prolog file outactCoordinator-oeSetCrisisStatus.pl.

D.15 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outactCo oeSetCrisisType.pl

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 /* DISCONTIGUOUS PREDICATES */
3 :- multifile msrop/4.
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5 %-------------------------------------------------------
6 msrop(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]])
27 .
28
29 msrop(outactCoordinator,
30    oeSetCrisisType,
31    [preFunctional,Self,
32     AdtCrisisID,
33     AetCrisisType
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     ],
```

```
    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.15: Prolog file outactCoordinator-oeSetCrisisType.pl.

D.16 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outac 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]),
```

```
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   oeValidateAlert,
30   [preFunctional,Self,
31    AdtAlertID
32    ],
33   []):-
34/* Pre Functional:*/
35 msrVar(ctState,TheSystem),
36 msrVar(actCoordinator,TheActor),
37
38 msrVar(dtAlertID,AdtAlertID),
39
40 msrNav([Self],[rnActor,rnSystem],[TheSystem]),
41 msrNav([Self],[rnActor],[TheActor]),
42
43/* PreF01 */
44 msrNav([TheSystem],
45    [rnctAlert,
46     msrSelect,
47     id,eq,[AdtAlertID]
48    ],
49    ColAlerts),
50
51 msrNav(ColAlerts,
52     [msrSize,eq,[[ptInteger,1]]],
53     [[ptBoolean,true]])
54 .
55
56msrop(outactCoordinator,
57   oeValidateAlert,
58   [post,Self,
59    AdtAlertID
60    ],
61    []):-
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,
```

```
    ieMessage,[[ptString,'The Alert is now declared as valid !']]
    ],
    [[ptBoolean,true]]),

/* Post Protocol:*/
/* PostP01 */
true
.
```

Listing D.16: Prolog file outactCoordinator-oeValidateAlert.pl.

D.17 File ./src-gen/prolog-ref-spec/Operations/Environment/OUT/outac oeCreateSystemAndEnvironment.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
/*************************************
MSRCreatorActor
*************************************/

/*** createSystemAndEnvironment ***/

msrop(outactMsrCreator,
   oeCreateSystemAndEnvironment,
   [preFunctional,_Self,_AqtyComCompanies],
   []):-
true.

msrop(outactMsrCreator,
   oeCreateSystemAndEnvironment,
   [preProtocol,_Self,_AqtyComCompanies],
   []):-
true.

msrop(outactMsrCreator,
   oeCreateSystemAndEnvironment,
   [post,_Self,AqtyComCompanies],
   []):-

msrVar(ctState,TheSystem),
msrVar(actMsrCreator,AactMsrCreator),
msrVar(actAdministrator,AactAdministrator),

msrVar(dtInteger, AnextValueForAlertID),
msrVar(dtInteger, AnextValueForCrisisID),
msrVar(dtDateAndTime, Aclock),
msrVar(dtSecond, AcrisisReminderPeriod),
msrVar(dtSecond, AmaxCrisisReminderPeriod),
msrVar(ptBoolean, AvpStarted),

/* PostF01 -- MUST ALWAYS BE MADE FIRST -- */
msrNav([AnextValueForAlertID],
    [value,eq,[[ptInteger,1]]],
    [[ptBoolean,true]]),

msrNav([AnextValueForCrisisID],
    [value,eq,[[ptInteger,1]]],
    [[ptBoolean,true]]),

msrNav([Aclock],
   [date,year,value],
   [[ptInteger,1970]]),
msrNav([Aclock],
   [date,month,value],
   [[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.17: Prolog file outactMsrCreator-oeCreateSystemAndEnvironment.pl.

D.18 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC 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.18: Prolog file PrimaryTypesClasses-ctAdministrator-init.pl.

D.19 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC 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(
```

```
msrVar(ctAlert,Self),

msrNav([Self],[id],[Aid]),
msrNav([Self],[status],[Astatus]),
msrNav([Self],[location],[Alocation]),
msrNav([Self],[instant],[Ainstant]),
msrNav([Self],[comment],[Acomment]),

/* Post F02 */
 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.19: Prolog file PrimaryTypesClasses-ctAlert-init.pl.

D.20 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass ctAlert-isSentToCoordinator.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctAlert,isSentToCoordinator,[Self,AactCoordinator],
   Result):-

/* Post F01 */
(
 msrNav([AactCoordinator],
     [rnInterfaceIN,ieSendAnAlert,[Self]],
     [[ptBoolean,true]])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.20: Prolog file PrimaryTypesClasses-ctAlert-isSentToCoordinator.pl.

D.21 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass 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]
```

```
24.
```

Listing D.21: Prolog file PrimaryTypesClasses-ctAuthenticated-init.pl.

D.22 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctCoordinator-init.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctCoordinator,init,[Self,
7          Aid,
8          Alogin,
9          Apwd],
10   Result):-
11
12/* Post F01 */
13(
14msrVar(ctCoordinator,Self),
15
16msrNav([Self],[id],[Aid]),
17msrNav([Self],[login],[Alogin]),
18msrNav([Self],[pwd],[Apwd]),
19msrNav([Self],[vpIsLogged],[[ptBoolean,false]]),
20
21/* Post F02 */
22 msrNav([Self],[msrIsNew],[Self])
23)
24-> Result = [ptBoolean,true]
25; Result = [ptBoolean,false]
26.
```

Listing D.22: Prolog file PrimaryTypesClasses-ctCoordinator-init.pl.

D.23 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctCrisis-handlingDelayPassed.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
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.23: Prolog file PrimaryTypesClasses-ctCrisis-handlingDelayPassed.pl.

D.24 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass 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]),
24msrNav([Self],[comment],[Acomment]),
25
26/* Post F02 */
27 msrNav([Self],[msrIsNew],[Self])
28)
29-> Result = [ptBoolean,true]
30; Result = [ptBoolean,false]
31.
```

Listing D.24: Prolog file PrimaryTypesClasses-ctCrisis-init.pl.

D.25 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass ctCrisis-isAllocatedIfPossible.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(ctCrisis,isAllocatedIfPossible,[Self],
7   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]
        ],
        [[ptBoolean,true]])
   )
; ( /* Post F03 */
   msrNav([TheSystem],
      [rnactAdministrator,msrForAll,rnInterfaceIN,
      ieMessage,[[ptString,'Please add new coordinators to handle pending crisis !']]],
      [[ptBoolean,true]])
 )
)
)
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.25: Prolog file PrimaryTypesClasses-ctCrisis-isAllocatedIfPossible.pl.

D.26 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctCrisis-isSentToCoordinator.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctCrisis,isSentToCoordinator,[Self,AactCoordinator],
   Result):-

/* Post F01 */
(
 msrNav([AactCoordinator],
     [rnInterfaceIN,ieSendACrisis,[Self]],
     [[ptBoolean,true]])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.26: Prolog file PrimaryTypesClasses-ctCrisis-isSentToCoordinator.pl.

D.27 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass ctCrisis-maxHandlingDelayPassed.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctCrisis,maxHandlingDelayPassed,[Self],
   Result):-

/* Post F01 */
(
 msrVar(ctState,TheSystem),
 msrVar(dtInteger,CurrentClockSecondsQty),
 msrVar(dtInteger,CrisisInstantSecondsQty),
 msrVar(dtSecond,MaxCrisisReminderPeriod),

 msrNav([Self],[rnSystem],[TheSystem]),

 msrNav([Self],
     [status],
     [[etCrisisStatus,pending]]),

 msrNav([TheSystem],
    [clock,toSecondsQty,[]],
    [CurrentClockSecondsQty]),

 msrNav([Self],
    [instant,toSecondsQty,[]],
    [CrisisInstantSecondsQty]),

 msrNav([TheSystem],
    [maxCrisisReminderPeriod],
    [MaxCrisisReminderPeriod]),

 msrNav([CurrentClockSecondsQty],
    [sub,[CrisisInstantSecondsQty],
     gt, [MaxCrisisReminderPeriod]
     ],
    [[ptBoolean,true]])

)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.27: Prolog file PrimaryTypesClasses-ctCrisis-maxHandlingDelayPassed.pl.

D.28 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctHuman-init.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctHuman,init,[Self,
          Aid,
          Akind],
   Result):-

/* Post F01 */
(
msrVar(ctHuman,Self),

msrNav([Self],[id],[Aid]),
msrNav([Self],[kind],[Akind]),

/* Post F02 */
 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.28: Prolog file PrimaryTypesClasses-ctHuman-init.pl.

D.29 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctHuman-isAcknowledged.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctHuman,isAcknowledged,[Self],Result):-

/* Post F01 */
(msrVar(dtPhoneNumber,AdtPhoneNumber),
 msrVar(dtSMS,AdtSMS),

 msrNav([Self],
     [id,eq,[AdtPhoneNumber]],
     [[ptBoolean,true]]),
 msrNav([AdtSMS],
     [value,eq,[[ptString,'The handling of your alert by our services is in progress !']]],
     [[ptBoolean,true]]),
 msrNav([Self],
     [rnactComCompany,rnInterfaceIN,ieSmsSend,[AdtPhoneNumber,AdtSMS]],
     [[ptBoolean,true]])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.29: Prolog file PrimaryTypesClasses-ctHuman-isAcknowledged.pl.

D.30 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC ctState-init.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(ctState,init,[Self,
          AnextValueForAlertID,
          AnextValueForCrisisID,
          Aclock,
          AcrisisReminderPeriod,
          AmaxCrisisReminderPeriod,
          AvpLastReminder,
          AvpStarted],
   Result):-

 /* Post F01 */
(
 msrVar(ctState,Self),

 msrNav([Self],[nextValueForAlertID],[AnextValueForAlertID]),
 msrNav([Self],[nextValueForCrisisID],[AnextValueForCrisisID]),
 msrNav([Self],[clock],[Aclock]),
 msrNav([Self],[crisisReminderPeriod],[AcrisisReminderPeriod]),
 msrNav([Self],[maxCrisisReminderPeriod],[AmaxCrisisReminderPeriod]),
 msrNav([Self],[vpLastReminder],[AvpLastReminder]),
 msrNav([Self],[vpStarted],[AvpStarted]),

 msrNav([Self],[msrIsNew],[Self])
)
-> Result = [ptBoolean,true]
; Result = [ptBoolean,false]
.
```

Listing D.30: Prolog file PrimaryTypesClasses-ctState-init.pl.

D.31 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData dtAlertID-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(dtAlertID,is,[AdtValue],Result):-
% msd01
msrVar(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,'012345678901234567890123456789']]],[]]]],
msrNav([X],[is,[]],[Result]).
```

```
34
35X = [dtAlertID,[],[[dtString,[[value,[ptString,'01234567890123456789012345678901234567890123456789']]],[]]]],
36Result = [ptBoolean,false] ?
37
38yes
39*/
```

Listing D.31: Prolog file PrimaryTypesDatatypes-dtAlertID-is.pl.

D.32 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesD dtComment-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5%% dtComment
6
7%msd01
8msrop(dtComment,is,[AdtValue],Result):-
9 msrVar(ptBoolean,TheResult),
10 msrVar(ptInteger,MaxLength),
11(
12  (
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.
25
26/*
27| ?- X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg ! Please help ...']]],[]]]],
28msrNav([X],[is,[]],[Result]).
29X = [dtComment,[],[[dtString,[[value,[ptString,'I broke my leg ! Please help ...']]],[]]]],
30Result = [ptBoolean,true] ?
31yes
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.32: Prolog file PrimaryTypesDatatypes-dtComment-is.pl.

D.33 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesD dtCoordinatorID-is.pl

```
1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2/* DISCONTIGUOUS PREDICATES */
3:- multifile msrop/4.
4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6msrop(dtCoordinatorID,is,[AdtValue],Result):-
```

```
% msd01
 msrVar(ptBoolean,TheResult),
(
 ( msrNav([AdtValue],
    [value,length,[],gt,[[ptInteger,0]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,length,[],leq,[[ptInteger,5]]],
    [[ptBoolean,true]])
 )
 -> (TheResult = [ptBoolean,true])
 ; (TheResult = [ptBoolean,false])
),
TheResult = Result
.
```

Listing D.33: Prolog file PrimaryTypesDatatypes-dtCoordinatorID-is.pl.

D.34 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData dtCrisisID-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

msrop(dtCrisisID,is,[AdtValue],Result):-
% msd01
 msrVar(ptBoolean,TheResult),
(
 ( msrNav([AdtValue],
    [value,length,[],gt,[[ptInteger,0]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,length,[],leq,[[ptInteger,10]]],
    [[ptBoolean,true]])
 )
 -> (TheResult = [ptBoolean,true])
 ; (TheResult = [ptBoolean,false])
),
TheResult = Result
.
/*
| ?- 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.34: Prolog file PrimaryTypesDatatypes-dtCrisisID-is.pl.

D.35 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData 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.35: Prolog file PrimaryTypesDatatypes-dtGPSLocation-is.pl.

D.36 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesD 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),

(
  (
    (
     % msd01
     msrNav([EarthRadius],[value],[[ptReal,6371]]),
     msrNav([MaxDistance],[value],[[ptReal,100]]),

     msrNav([AdtValue],[latitude],[ComparedLatitude]),
     msrNav([AdtValue],[longitude],[ComparedLongitude]),

     msrNav([Self],[latitude,sin,[]],[R1a]),
     msrNav([AdtValue],[latitude,sin,[],mul,[R1a]],[R1]),

     msrNav([Self],[latitude,cos,[]],[R2a]),
     msrNav([AdtValue],[latitude,cos,[],mul,[R2a]],[R2]),

     msrNav([AdtValue],[longitude],[ComparedLongitude]),
     msrNav([Self],[longitude,sub,[ComparedLongitude],cos,[],mul,[R2],
            add,[R1],
            acos,[],
            mul,[EarthRadius],
            sub,[MaxDistance],
            value,leq,[[ptReal,0]]],
        [[ptBoolean,true]])
```

```
    )
     -> TheResult = [ptBoolean,true]
     ; TheResult = [ptBoolean,false]
  )
),
 Result = TheResult
.
```

Listing D.36: Prolog file PrimaryTypesDatatypes-dtGPSLocation-isNearTo.pl.

D.37 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData dtLatitude-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% msd01
msrop(dtLatitude,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
 ( msrNav([AdtValue],
    [value,geq,[[ptReal,-90.0]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,leq,[[ptReal,+90.0]]],
    [[ptBoolean,true]])
 )
 -> (TheResult = [ptBoolean,true])
 ; (TheResult = [ptBoolean,false])
),
Result = TheResult
.
```

Listing D.37: Prolog file PrimaryTypesDatatypes-dtLatitude-is.pl.

D.38 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData dtLogin-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% dtComment

%msd01
msrop(dtLogin,is,[AdtValue],Result):-
 msrVar(ptBoolean,TheResult),
 msrVar(ptInteger,MaxLength),
(
  (
    (
     MaxLength = [ptInteger,20],
     msrNav([AdtValue],
        [value,length,[],leq,[MaxLength]],
        [[ptBoolean,true]])
    )
     -> TheResult = [ptBoolean,true]
     ; TheResult = [ptBoolean,false]
  )
),
 Result = TheResult
.
/*
| ?- X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567']]],[]]]],
```

```
msrNav([X],[is,[]],[Result]).
X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567']]],[]]]],
Result = [ptBoolean,true] ?
yes

| ?- X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567a']]],[]]]],
msrNav([X],[is,[]],[Result]).
X = [dtLogin,[],[[dtString,[[value,[ptString,'01234567a']]],[]]]],
Result = [ptBoolean,false] ?
yes
*/
```

Listing D.38: Prolog file PrimaryTypesDatatypes-dtLogin-is.pl.

D.39 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesD dtLongitude-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% dtPhoneNumber

% msd01
msrop(dtLongitude,is,[AdtValue],Result):-
msrVar(ptBoolean,TheResult),
(
 ( msrNav([AdtValue],
    [value,geq,[[ptReal,-180.0]]],
    [[ptBoolean,true]]),
   msrNav([AdtValue],
    [value,leq,[[ptReal,+180.0]]],
    [[ptBoolean,true]])
 )
 -> (TheResult = [ptBoolean,true])
 ; (TheResult = [ptBoolean,false])
),

 Result = TheResult
.
```

Listing D.39: Prolog file PrimaryTypesDatatypes-dtLongitude-is.pl.

D.40 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesD 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.40: Prolog file PrimaryTypesDatatypes-dtPassword-is.pl.

D.41 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesData 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.41: Prolog file PrimaryTypesDatatypes-dtPhoneNumber-is.pl.

D.42 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass 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.42: Prolog file PrimaryTypesDatatypes-etAlertStatus-is.pl.

D.43 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC 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.43: Prolog file PrimaryTypesDatatypes-etCrisisStatus-is.pl.

D.44 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesC 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.44: Prolog file PrimaryTypesDatatypes-etCrisisType-is.pl.

D.45 File ./src-gen/prolog-ref-spec/Operations/Concepts/PrimaryTypesClass 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.45: Prolog file PrimaryTypesDatatypes-etHumanKind-is.pl.

D.46 File ./src-gen/prolog-ref-spec/Operations/Concepts/SecondaryTypesDa dtSMS-is.pl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/* DISCONTIGUOUS PREDICATES */
:- multifile msrop/4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% dtComment

%msd01
msrop(dtSMS,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
.
```

Listing D.46: 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.