

Laboratory for Advanced Software Systems
University of Luxembourg
6, rue R. Coudenhove-Kalergi, Luxembourg
with participation of:
Vicious & Delicious Working Group
Heidi Flinkman, Alexander Eyjolfsson, Yan Medernach

iCrash :
A Crisis Management Case Study
Messir Analysis Document
- v 1.8 -

Tuesday 8th April, 2014 - 12:14

Contents

1	Introduction	5
1.1	Overview	5
1.2	Purpose and recipients of the document	5
1.3	Application Domain	5
1.4	Definitions, acronyms and abbreviations	5
1.5	Document structure	6
2	General Description	7
2.1	Domain Stakeholders	7
2.1.1	Communication Company	7
2.1.2	Humans	7
2.1.3	Coordinators	8
2.1.4	Administrator	8
2.1.5	Creator	8
2.1.6	Activator	9
2.2	System's Actors	9
2.3	Use Cases Model	10
2.3.1	Use Case Diagram(s)	10
2.3.2	Use Case Instance(s)	15
3	Environment Model	21
3.1	Local view 01	21
3.2	Local view 02	21
3.3	Local view 03	21
3.4	Local view 04	21
3.5	Local view 05	21
3.6	Actors and Interfaces Descriptions	23
3.6.1	actDomainExpert Actor	23
3.6.2	actComCompany Actor	25
3.6.3	actEMS Actor	25
3.6.4	actMsrCreator Actor	26
3.6.5	actCoordinator Actor	26
3.6.6	actAdministrator Actor	27
3.6.7	actAuthenticated Actor	27
4	Concept Model	29
4.1	PrimaryTypes-Classes	29
4.1.1	Local view 01	29
4.1.2	Local view 02	29
4.1.3	Local view 03	29
4.1.4	Local view 04	29
4.1.5	Local view 06	29
4.1.6	Global view 01	32
4.2	PrimaryTypes-Datatypes	32
4.2.1	Global view 01	32
4.3	SecondaryTypes-Datatypes	32

4.3.1 Local view 01	32
4.4 Concept Model Types Descriptions	32
4.4.1 Primary types - Class types descriptions	32
4.4.2 Primary types - Datatypes types descriptions	38
4.4.3 Primary types - Association types descriptions	41
4.4.4 Primary types - Aggregation types descriptions	41
4.4.5 Primary types - Composition types descriptions	42
4.4.6 Secondary types - Class types descriptions	42
4.4.7 Secondary types - Datatypes types descriptions	42
4.4.8 Secondary types - Association types descriptions	42
4.4.9 Secondary types - Aggregation types descriptions	42
4.4.10 Secondary types - Composition types descriptions	42
5 Operation Model	43
5.1 Environment - Out Interface Operation Scheme for actAuthenticated	43
5.1.1 Operation Model for oeLogin	43
5.1.2 Operation Model for oeLogout	44
5.2 Environment - Out Interface Operation Scheme for actComCompany	44
5.2.1 Operation Model for oeAlert	44
5.3 Environment - Out Interface Operation Scheme for actMsrCreator	45
5.3.1 Operation Model for oeCreateSystemAndEnvironment	45
5.4 Environment - Actor Operation Scheme for actComCompany	46
5.4.1 Operation Model for init	46
5.5 Primary Types - Operation Schemes for Class ctAdministrator	47
5.5.1 Operation Model for init	47
5.6 Primary Types - Operation Schemes for Class ctAlert	47
5.6.1 Operation Model for init	47
5.7 Primary Types - Operation Schemes for Class ctState	48
5.7.1 Operation Model for init	48
5.8 Primary Types - Operation Schemes for Datatype dtAlertID	49
5.8.1 Operation Model for is	49
5.9 Primary Types - Operation Schemes for Datatype dtComment	49
5.9.1 Operation Model for is	49
5.10 Primary Types - Operation Schemes for Datatype dtPhoneNumber	50
5.10.1 Operation Model for is	50
5.11 Primary Types - Operation Schemes for Enumerations	51
5.12 Secondary Types - Operation Schemes for Classes	51
5.13 Secondary Types - Operation Schemes for Datatype dtSMS	51
5.13.1 Operation Model for is	51
5.14 Secondary Types - Operation Schemes for Enumerations	51
6 Test Model(s)	53
7 Additional Constraints	55
A Use Case Description Tables	57
A.1 Use cases list(s)	57
A.2 Use case Table(s)	59
A.2.1 Summary	59
A.2.2 User-goal	63
A.2.3 Subfunction	71
A.3 Use case instance Table(s)	106
B Messir Specification Files Listing	111
B.1 File /concepts/primarytypes-associations.msr	111
B.2 File /concepts/primarytypes-classes.msr	112
B.3 File /concepts/primarytypes-datatypes.msr	114

B.4 File /concepts/secondarytypes-associations.msr	115
B.5 File /concepts/secondarytypes-classes.msr	116
B.6 File /concepts/secondarytypes-datatypes.msr	116
B.7 File /environment/environment.msr	116
B.8 File /operations/concepts.../primarytypes-classes-ctAdministrator.msr	118
B.9 File /operations/concepts/primarytypes-classes/primarytypes-classes-ctAlert.msr	119
B.10 File /operations/concepts/primarytypes-classes/primarytypes-classes-ctState.msr	119
B.11 File /operations/concepts.../primarytypes-datatypes-dtAlertID.msr	120
B.12 File /operations/concepts.../primarytypes-datatypes-dtComment.msr	120
B.13 File /operations/concepts.../primarytypes-datatypes-dtPhoneNumber.msr	121
B.14 File /operations/concepts/secondarytypes-datatypes/dtSMS.msr	121
B.15 File /operations/environment/environment-actAuthenticated.msr	122
B.16 File /operations/environment/environment-actComCompany.msr	122
B.17 File /operations/environment/environment-actMsrCreator.msr	123

List of Figures

2.1	<i>iCrash</i> suDeployAndRun Use Case Diagram	10
2.2	<i>iCrash</i> suGlobalCrisisHandling Use Case Diagram	11
2.3	<i>iCrash</i> ugManageCrisis Use Case Diagram	12
2.4	<i>iCrash</i> ugAdministrateTheSystem Use Case Diagram	13
2.5	<i>iCrash</i> ugSecurelyUseSystem Use Case Diagram	13
2.6	<i>iCrash</i> ugMonitor Use Case Diagram	14
2.7	<i>iCrash</i> Use Case Diagram: SequenceDiagram Diagram	19
3.1	Environment Model - Local View 01 - Part 1	22
3.2	Environment Model - Local View 02 - Part 2	23
3.3	Environment Model - Local View 03 -	23
3.4	Environment Model - Local View 04 -	24
3.5	Environment Model - Local View 05 -	24
4.1	Concept Model - PrimaryTypes-Classes local view 01 - Local view of the Primary Types Clas .	30
4.2	Concept Model - PrimaryTypes-Classes local view 02 -	31
4.3	Concept Model - PrimaryTypes-Classes local view 03 -	32
4.4	Concept Model - PrimaryTypes-Classes local view 04 -	33
4.5	Concept Model - PrimaryTypes-Classes local view 06 -	34
4.6	Concept Model - PrimaryTypes-Classes global view 01 - Associations between primary type c .	35
4.7	Concept Model - PrimaryTypes-Datatypes global view 01 -	36
4.8	Concept Model - SecondaryTypes-Datatypes local view 01 - Local view of the secondary types da	37
A.1	<i>iCrash</i> Use Case Diagram: oeDeployAndRun Diagram	61
A.2	<i>iCrash</i> Use Case Diagram: ugGlobalCrisisHandling Diagram	63
A.3	<i>iCrash</i> Use Case Diagram: ugAdministrateTheSystem Diagram	65
A.4	<i>iCrash</i> Use Case Diagram: ugSecurelyUseSystem Diagram	67
A.5	<i>iCrash</i> Use Case Diagram: UgManageCrisis Diagram	69
A.6	<i>iCrash</i> Use Case Diagram: UgMonitor Diagram	71
A.7	<i>iCrash</i> Use Case Diagram: oeSollicitateCrisisHandling Diagram	73
A.8	<i>iCrash</i> Use Case Diagram: oeAddUser Diagram	75
A.9	<i>iCrash</i> Use Case Diagram: oeDeleteUser Diagram	77
A.10	<i>iCrash</i> Use Case Diagram: oeRequestEMSAssistance Diagram	79
A.11	<i>iCrash</i> Use Case Diagram: oeSollicitateCrisisHandling Diagram	81
A.12	<i>iCrash</i> Use Case Diagram: oeReportOnCrisis Diagram	83
A.13	<i>iCrash</i> Use Case Diagram: oeInfoFamily Diagram	85
A.14	<i>iCrash</i> Use Case Diagram: oeCloseCrisis Diagram	87
A.15	<i>iCrash</i> Use Case Diagram: oeAlert Diagram	89
A.16	<i>iCrash</i> Use Case Diagram: oeGetAlertSet Diagram	91
A.17	<i>iCrash</i> Use Case Diagram: oeSetCrisisHandler Diagram	93
A.18	<i>iCrash</i> Use Case Diagram: oeLogin Diagram	95
A.19	<i>iCrash</i> Use Case Diagram: oeLogout Diagram	97
A.20	<i>iCrash</i> Use Case Diagram: oeCloseCrisis Diagram	99
A.21	<i>iCrash</i> Use Case Diagram: oeReplyToRequest Diagram	101
A.22	<i>iCrash</i> Use Case Diagram: oeCloseCrisis Diagram	103
A.23	<i>iCrash</i> Use Case Diagram: oeValidateAlert Diagram	105

A.24 <i>iCrash</i> Use Case Diagram: SequenceDiagram Diagram	110
--	-----

Listings

5.1	Messir OCL description of the operation <i>oeAlert</i>	45
5.2	Messir OCL description of the operation <i>oeCreateSystemAndEnvironment</i>	45
5.3	Messir OCL description of the operation <i>init</i>	46
5.4	Messir OCL description of the operation <i>is</i>	49
5.5	Messir OCL description of the operation <i>is</i>	50
5.6	Messir OCL description of the operation <i>is</i>	51
B.1	Messir Spec. file primarytypes-associations.msr	111
B.2	Messir Spec. file primarytypes-classes.msr	112
B.3	Messir Spec. file primarytypes-datatypes.msr	114
B.4	Messir Spec. file secondarytypes-associations.msr	115
B.5	Messir Spec. file secondarytypes-classes.msr	116
B.6	Messir Spec. file secondarytypes-datatypes.msr	116
B.7	Messir Spec. file environment.msr	116
B.8	Messir Spec. file primarytypes-classes-ctAdministrator.msr	118
B.9	Messir Spec. file primarytypes-classes-ctAlert.msr	119
B.10	Messir Spec. file primarytypes-classes-ctState.msr	119
B.11	Messir Spec. file primarytypes-datatypes-dtAlertID.msr	120
B.12	Messir Spec. file primarytypes-datatypes-dtComment.msr	120
B.13	Messir Spec. file primarytypes-datatypes-dtPhoneNumber.msr	121
B.14	Messir Spec. file dtSMS.msr	121
B.15	Messir Spec. file environment-actAuthenticated.msr	122
B.16	Messir Spec. file environment-actComCompany.msr	122
B.17	Messir Spec. file environment-actMsrCreator.msr	123

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 [?]. 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 **MESSIP** methodology [?]. 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 7. 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 **Messip** method, the information provided in this section is intended to present the system for which the **Messip** 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 **Messip** 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 2.3, 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 **Messip** 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 a 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 is related to as a victim or witness.

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

- to provide as much details as possible, even tho these details might contain errors, 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 communicate 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 a coordinator 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

A Creator¹ is a technician who is installing the *iCrash* system on the infrastructure of the ABC company. The objectives of a Creator are:

- to install the *iCrash* system
- to define the values for the initial system's state

¹ In **Messip** each system has a creator stakeholder who is different from a technical system administrator. Although those two roles can be fulfilled by the same person.

- 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 **Messip** 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 **Messip** specification since actor and messages names together with parameters are partly adapted to be consistent with the specification identifiers (see [?] for more details).

Among all the stakeholders presented in the previous section, we can determine five types of *direct actors*²:

- *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 The naming conventions in **Messip** 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.

2.3 Use Cases Model

2.3.1 Use Case Diagram(s)

The figures 2.1, 2.2, 2.3, 2.4 and 2.5 provide **Messip** use casediagrams for the ABC system use cases.

Even though this general description is not formal, follow the advice given in the **Messip** book [?] which proposes to have for each actor an input interface to receive messages from the system and an output one to send messages to the system.

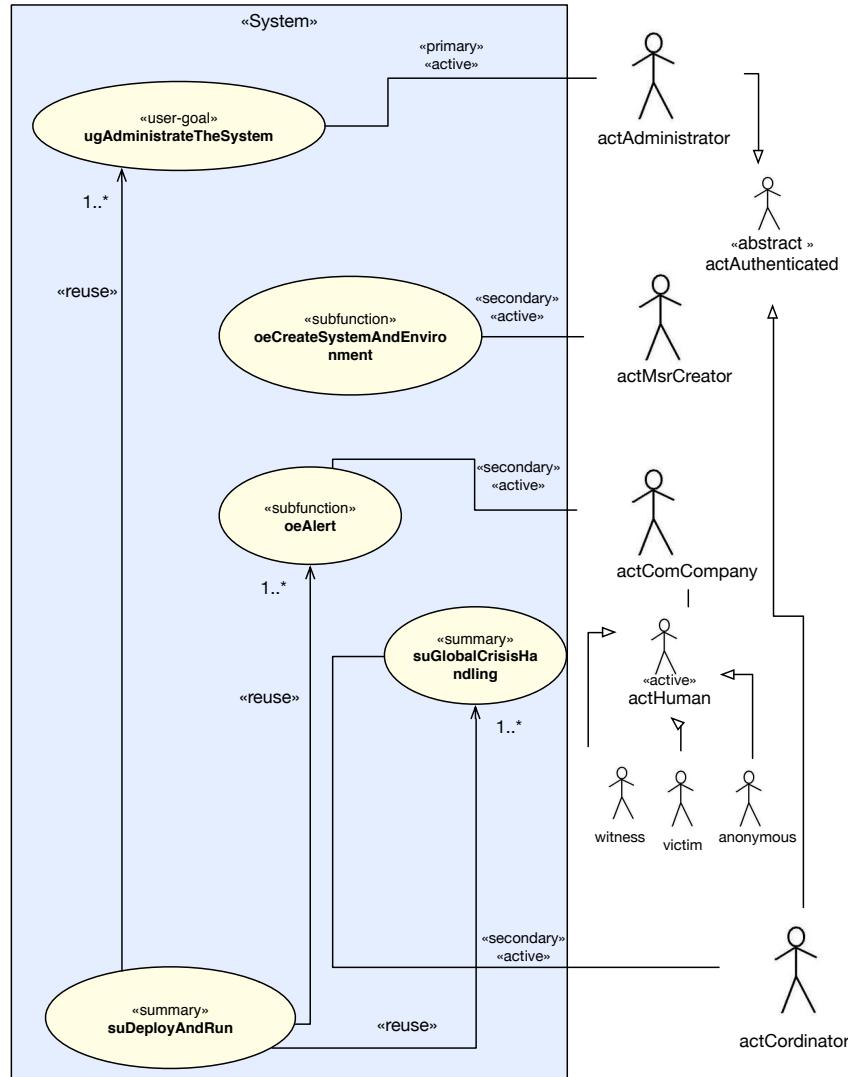


Fig. 2.1 *iCrash* suDeployAndRun Use Case Diagram

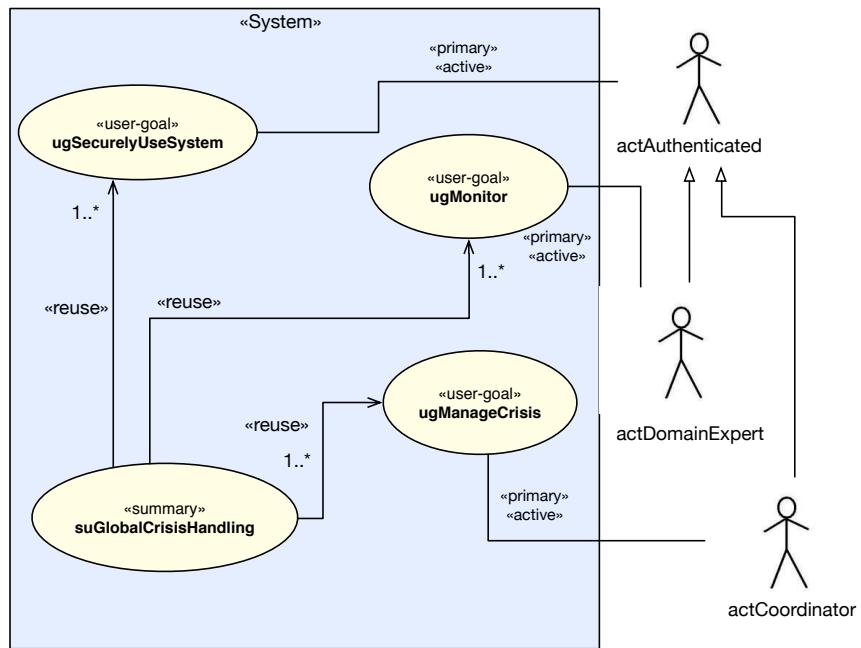


Fig. 2.2 *iCrash* suGlobalCrisisHandling Use Case Diagram

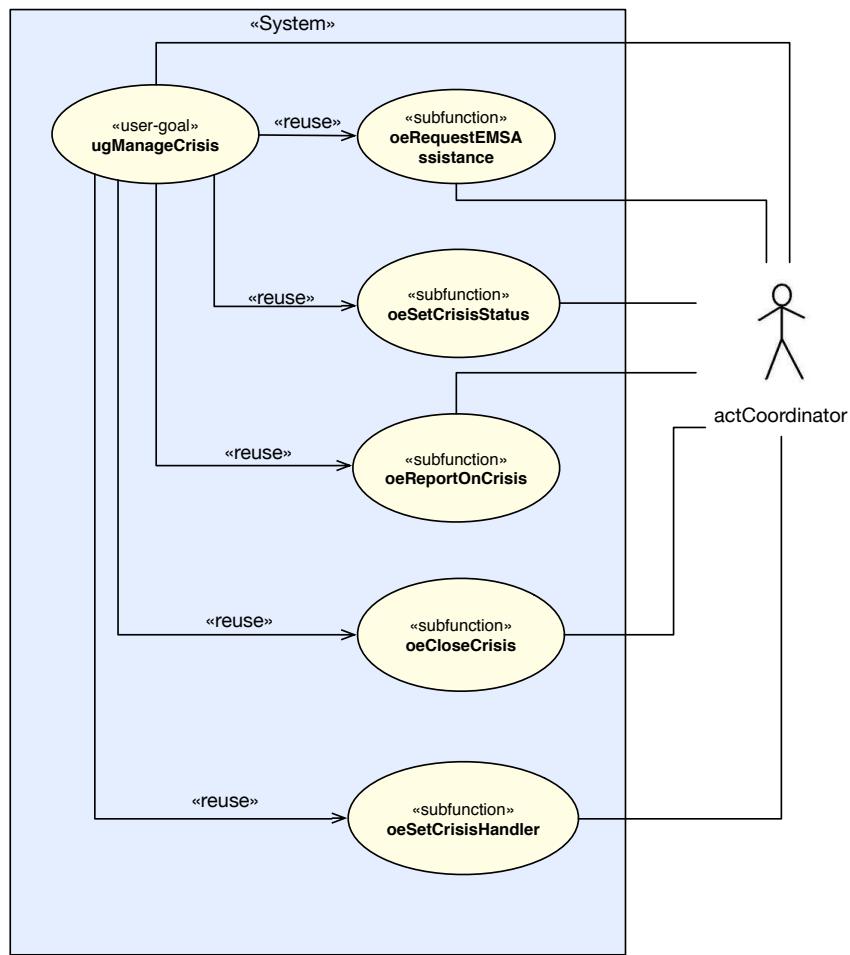
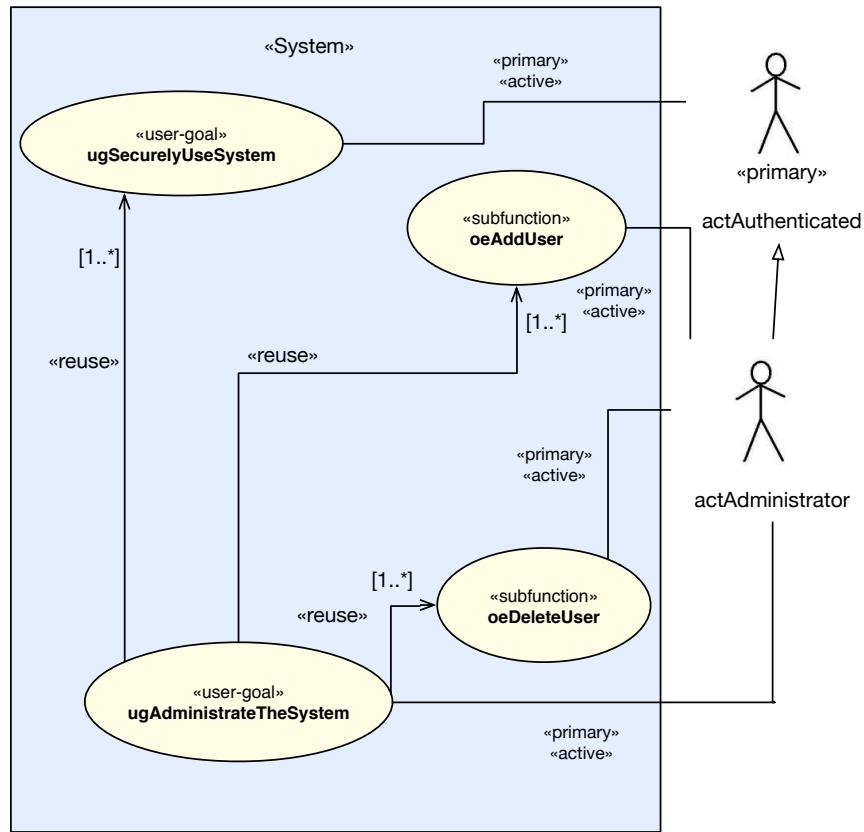
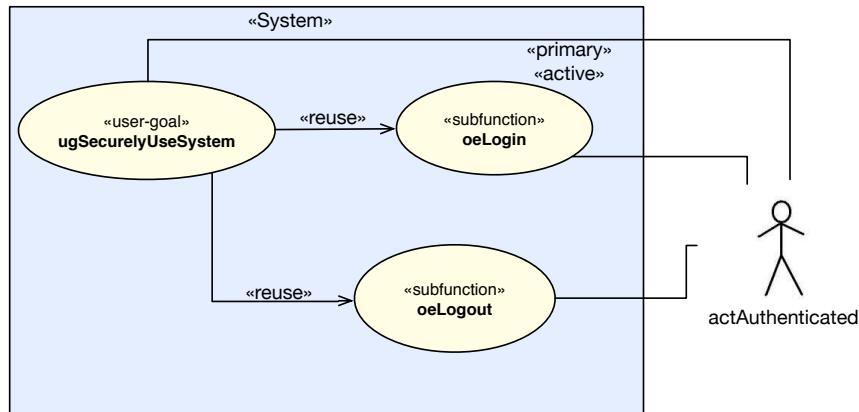


Fig. 2.3 *iCrash* *ugManageCrisis* Use Case Diagram

**Fig. 2.4** *iCrash* **ugAdministrateTheSystem** Use Case Diagram**Fig. 2.5** *iCrash* **ugSecurelyUseSystem** Use Case Diagram

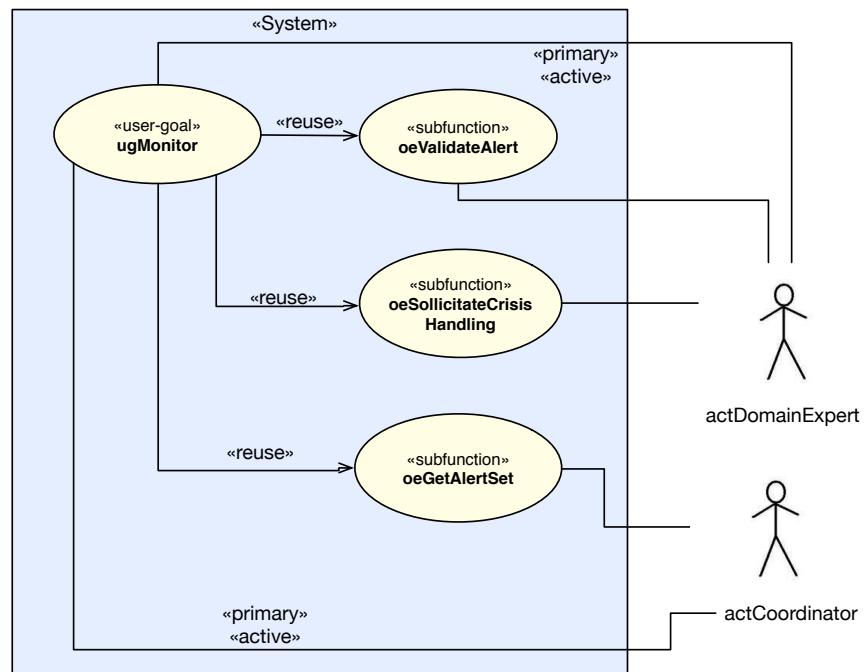


Fig. 2.6 *iCrash ugMonitor* Use Case Diagram

2.3.2 Use Case Instance(s)

The use case instance described below represents a summary level use case instance intended to illustrate a small and complete execution scenario of the *iCrash* system. Figure ?? provides informal graphical view (using a sequence diagram) of the messages sent during the scenario. The use case instance is described textually in the following use case description form as suggested by the **Messip** method and inspired by the standard Cokburn template [?]. We chose to indicate only messages sent by the environment to the system and not reply messages for readability.

USE-CASE INSTANCE	
Name	Simple and Complete DeployAndRun Instance
Instance ID	01
Environmental conditions and assumptions	The necessary IT infrastructure exists to allow for deployment of the <i>iCrash</i> system.
Inputs	inputs are sequences of characters interpreted as string or numbers.
Instance flow description	<p>1 the actMsrCreator actor theCreator sends the message oeCreateSystemAndEnvironment (1) requesting the initialization of the system and its environment (consisting of one administrator identified here by bill) and indicating that the number of communication company actor instances for the system's environment is 1 (identified here by tango).</p> <p>2 the actAdministrator actor bill sends the message oeLogin(icrashadmin, 7WXC1359) to securely connect to the system.</p> <p>3 the actAdministrator actor bill sends the message oeAddUser(1, Steve, pwdMessirExcalibur2017, 1635242A75, 'Vehicular, Pedestrian, Wildlife, Property, Injury', Coordinator) to set up a coordinator (i.e. identified here by steve) and indicating his identification information, ID (i.e. 1) and a password (i.e. pwdMessirExcalibur2017), id from a token(i.e.1635242A75), specifying his domain of expertise(i.e. 'Vehicular, Pedestrian, Wildlife, Property, Injury') and finally he designates the user as a Coordinator(i.e.Coordinato).</p> <p>4 the actAdministrator actor bill sends the message oeAddUser(2, franklin, pwdMessirExcalibur, 1456872B82, Null, DomainExpert) create a domain expert who is one person(i.e. identified here by franklin) and indicating his user identification his identification, ID(i.e.2), password(i.e.pwdMessirExcalibur123), id from a token(i.e.14566872B82), specifying the users domain of expertise as(Null meaning that the user has no specific domain and can therefore only be an EMS user or Domain expert) and finally he designates the user as a domain expert(i.e.DomainExpert).</p> <p>5 the actAdministrator actor bill sends the message oeAddUser(3, barry, pwdEMSExcalibur321, 2436787C82, Null, EMS) to set up a EMS user who is any user in the EMS headquarters(i.e. identified here by barry) and indicating his user identification as ID(i.e.3), password (i.e.pwdEMSExcalibur421), id from a token(i.e.2436787C82), specifying the users domain of expertise as(i.e.Null meaning that the user has no specific domain and can therefore only be an EMS user or DomainExpert) and finally he designates the user as an EMS user(i.e.EMS).</p> <p>6 the actAdministrator actor bill sends the message oeLogout () to disconnect from the system.</p> <p>7 the actComCompany actor tango sends the message oeAlert(witness, 2017-11-26-at-10-10-16AM, +3524666445252, 49.627675-6.159590, '3 cars involved in an accident.') to transfer a declaration of a car crash by a witness indicating specific phone number, the date and time, the GPS coordinates of the witnessed car crash and a short message giving additional details.</p> <p>8 the actDomainExpertactor franklin sends the message oeLogin(franklin, pwdMessirExcalibur123, 687594FAD9) to securely connect to the system entering his login(i.e.franklin), his password(i.e.pwdMessirExcalibur123)and entering his serial key that he reads form a token device given to him by the administrator(i.e.687594FAD9).</p> <p>9 the actEMSactor barry sends the message oeLogin(barry, pwdEMSExcalibur321, 24367872C282) to securely connect to the system entering his login(i.e.barry), his password(i.e.pwdEMSExcalibur321)and entering his serial key that he reads form a token device given to him by the administrator(i.e.24367872C282).</p> <p>10 the actDomainExpert actor franklin sends the message oeValidateAlert(1, 49.627675-6.159590, 2017-11-26-at-10-10-16AM, valid, 'Vehicular, Pedestrian') to validate the crisis and to set a Domain to it.</p> <p>11 the actDomainExpert actor franklin sends the message oeSollicitateCrisisHandling() indicating that there is a declared alert that is still not handled by any coordinator.</p>

continues in next page ...

... Use-Case Instance table continuation

- 12 the actCoordinator actor steve sends the message oeLogin(steve, pwdMessirExcalibur2017, 63524275D9) to securely connect to the system, entering his login(i.e.steve), his password(i.e.pwdMessirExcalibur2017) and his serial Key that he reads form a token device he is given by the administrator(i.e.63524275D9)
- 13 the actCoordinator actor steve sends the message oeGetAlertsSet(valid) to receive information about all the pending crisis.
- 14 the actCoordinator actor steve sends the message oeSetCrisisHandler(1,Medium, in-handling, 49.627095-6.160251, 2017-11-26-at-10-10-16AM) to declare that he is taking care of the alert with the ID equal to 1 which becomes the Crisis Id, sets the crisis type to(i.e.Medium, the crisis type to(i.e.in-handling), enters the GPS coordinates and the date and time.
- 15 the actComCompany actor tango sends the message oeAlert(witness, 2017-11-26-at-10-20-18AM, +3524666445314, 49.627095-6.160251,Please send rescue services.) to transfer a declaration of a car crash by a witness indicating specific the phone number, the date and time, the GPS coordinates of the witnessed car crash and a short message giving additional details. This alert's GPS coordinates match the previous alert sent coordinates in step 7 and the time elapsed between the two alerts is 10 minutes therefore the alert is considered to be originating from the same location.
- 16 the actDomainExpert actor franklin sends the message oeValidateAlert(1, 49.627675-6.159590, 2017-11-26-at-10-10-16AM, valid, 'Vehicular, Pedestrian') to validate the crisis and to set a Domain to it. Because the alert originated from the same location and only 10 min later it an automated message informs the Domain Expert to validate it to the same alert ID as the previous alert.
- 17 the actCoordinator actor steve send the message oeReportOnCrisis(1, ' 2 Alerts received about a car accident at the same location apparently involving 3 vehicles', 2, Medium, Handled,'witness, 2017-11-26-at-10-10-16AM,+3524666445252, 49.627675-6.159590,'3 cars involved in an accident', witness, 2017-11-26-at-10-2-18AM,+3524666445314, 49.627095-6.160251,'Please send rescue services.',3, 3) indicating the crisis ID(i.e.1), entering a specific comment in the comment area(i.e.' 2Alertsrecived about a car accident at the same location apparently involving 3 vehicles'), specifying his user ID(i.e.2), setting the CrisisType(i.e.Medium), indicating the current crisisStatus(handled), entering the previous received alerts, indicating the number of vehicles involved in the accident(i.e.3) and entering the number of victims(i.e.3) because he suspects that there may be 3 victims due to the amount of cars involved in the accident. Entering a preliminary report on the crisis with the information that he presumes are right.
- 18 the actCoordinator actor steve send the message oeRequestEMSAssistance('We have received an Alert of an accident involving 3 cars from 1 victim and 1 witness at the same location please send police and ambulance',1, 49.627095-6.160251, 3,3, Ambulance Police), entering a comment(i.e. ' We have received an Alert of an accident involving 3 cars from 1 victim and 1 witness at the same location please send Police assistance'), indicating the RequestID(i.e.1), entering the GPS location of the accident(i.e.49.627095-6.160251), informing them of the number of cars involved(i.e.3) and informing them of the presumed number of victims(i.e.3) requesting emergency services assistance(i.e.Ambulance, Police).
- 19 the actCoordinator actor steve sends the message oeSetCrisisStatus(1, handled) to change the status of the crisis identified by the ID(i.e.1) to the status(i.e.handled) thus indicating that he has handled the situation for now but it's not solved yet.
- 20 the actEMS actor barry sends the message oeReplyToRequest(1,'Message received, dispatching police and ambulance units.', 1, Handled) indicating the CrisisID(i.e.1), sending the comment(i.e.'Message received, dispatching police and ambulance units.'), to confirm that the request with the ID(i.e.1) has been received and is being handled, indicated by the EMS crisis status(i.e. Handled).

continues in next page ...

...Use-Case Instance table continuation

- 21 the actEMS actor barry sends the message oeReportEMSCrisisStatus(1, 'Units arrived on scene report 4 victims, 3 cars involved in accident. Situation under control 1 victim brought Mercy hospital', solved, Mercy Hospital, Jeremy Springer, John Snow, +32168432) indicating the crisis ID(i.e.1), adding a comment(i.e. 'Units arrived on scene report 4 victims, 3 cars involved in accident. Situation under control 1 brought to Mercy hospital'), specifying the EMS crisis status as solved (i.e.solved), indicating the hospital name as (i.e.Mercy Hospital), indicating the victim name(i.e.Jeremy Springer), indicating the victims ICE contact's name as(i.e.John Snow) and the contacts phone number as (+32168432) to report on the EMS crisis Status.
- 22 the actComCompany actor tango sends message oeInfoFam('Dear John Snow, I regret to inform you that Jeremy Springer was in a car accident involving 3 other cars. Jeremy Springer was brought to the Mercy hospital for examination. Regretfully yours..', Jeremy Springer, +32168432, Mercy Hospital) to inform the victims family members about the state of the victim and in which hospital they reside.
- 23 the actCoordinator actor steve sends the message oeReportOnCrisis(1, ' 2 Alerts received about a car accident at the same location involving 3 vehicles and 4 victims one victim Jeremy Springer was brought to the Mercy hospital.', 2, Medium, Handled,'witness, 2017-11-26-at-10-10-16AM,+3524666445252, 49.627675-6.159590,'3 cars involved in an accident 'witness, 2017-11-26-at-10-2-18AM,+3524666445314, 49.627095-6.160251,'Please send rescue services.",3, 4) to set the report for the crisis with ID equal to 1 that he is handling.
- 24 the actCoordinator actor steve sends the message oeReportOnCrisis(1,'3 victims sent to hospital, 2 cars evacuated and 4 rescue unit mobilized') to set the report for the crisis with ID equal to 1 that he is handling.
- 25 the actCoordinator actor steve sends the message oeCloseCrisis(1) to declare the crisis with ID equal to 1 as closed.

Outputs

at the end of this instance flow the system should have its environment made of one communication company actor instance, one coordinator actor instance, one administrator instance and the creator instance. The system state should contain two alerts defined according to the information received from the communication company, only one crisis that should be considered as solved and being alerted by the two alerts received.



1: oeCreateSystemAndEnvironment (1)

2: oeLogin(icrashadmin, 7WC1359)

3: oeAddUser (1, steve, pwdMessirExcalibur2017, 1635242A75, 'Vehicular, Pedestrian, Wildlife, Property, Injury', Coordinator)

4: oeAddUser (2, franklin, pwdMessirExcalibur, 1456872B82, Null, DomainExpert)

5: oeAddUser (3, barry, pwdEMSExcalibur321, 2436787C82, Null, EMS)

6: oeLogout()

7: oeAlert(witness, 2017-11-26-at-10-10-16AM, +352466445252, 49.627675-6.159590, '3 cars involved in an accident.')

8: oeLogin(franklin, pwdMessirExcalibur123, 687594FAD9)

9: oeLogin(barry, pwdEMSExcalibur321, 24367872C282)

10: oeValidateAlert(1,49.627675-6.159590,2017-11-26-at-10-10-16AM, valid, 'Vehicular, Pedestrian')

11: oeSollicitateCrisisHandling()

12: oeLogin(steve,pwdMessirExcalibur2017, 63524275D9)

tangoOUT

steveOUT

billOUT

franklinOUT

BarryOUT

theCreatorOUT

13: oeGetAlertSet(Valid)

Chapter 3

Environment Model

We provide below the view(s) defined for the **Messip** environment model (cf. [?]) of the system.

3.1 Local view 01

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

3.3 Local view 03

Figure 3.3

3.4 Local view 04

Figure 3.4

3.5 Local view 05

Figure 3.5

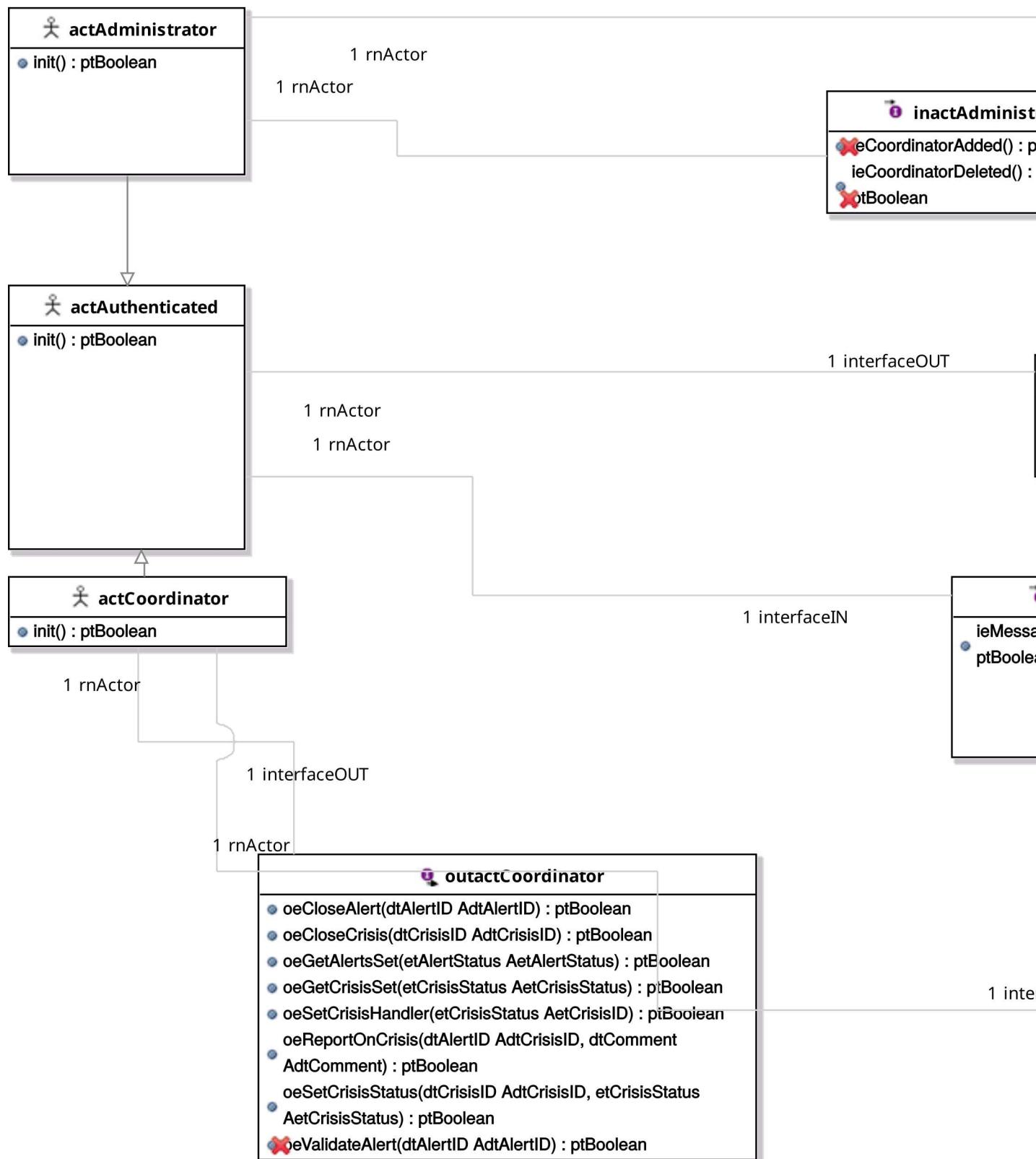


Fig. 3.1 Environment Model - Local View 01. Part 1.

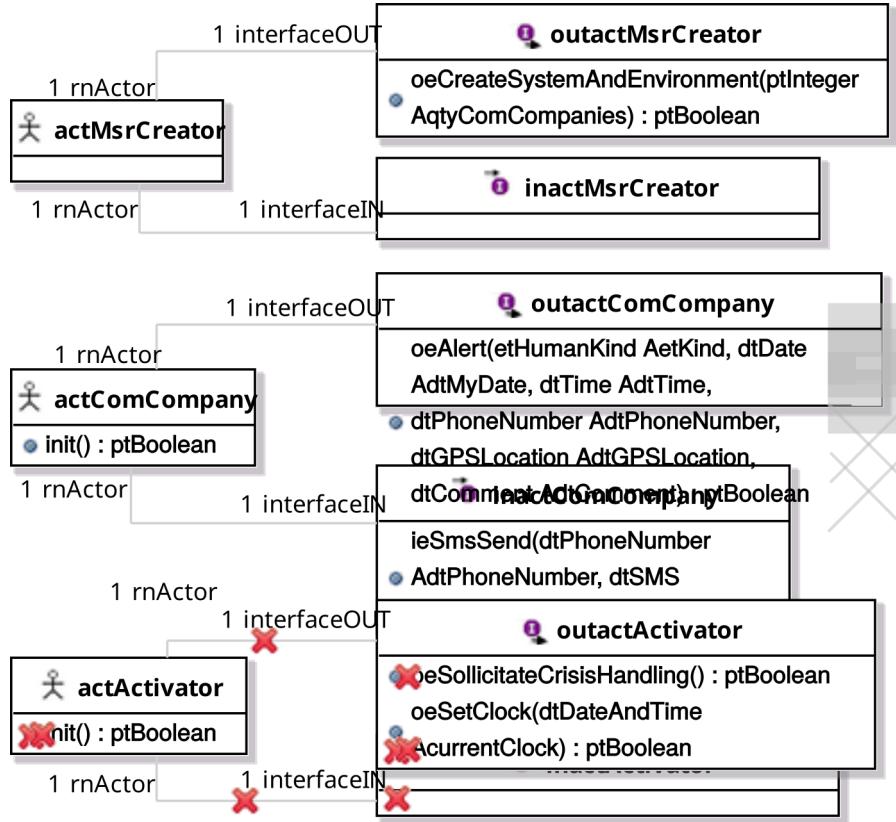


Fig. 3.2 Environment Model - Local View 02. Part 2.

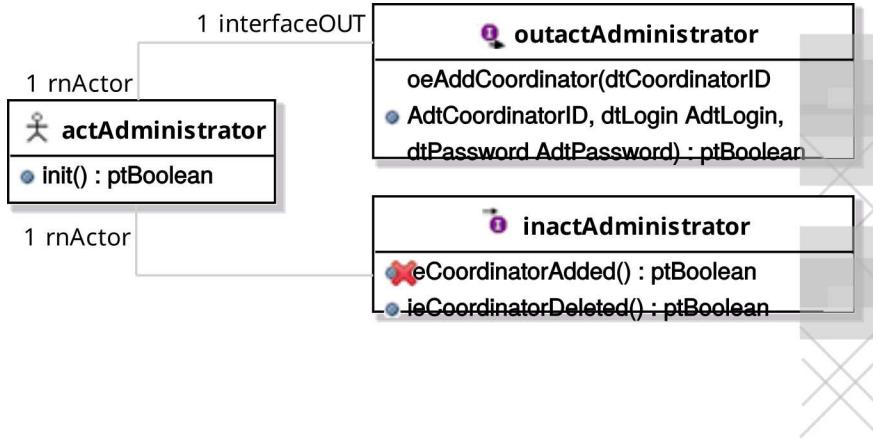


Fig. 3.3 Environment Model - Local View 03. .

3.6 Actors and Interfaces Descriptions

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

3.6.1 *actDomainExpert Actor*

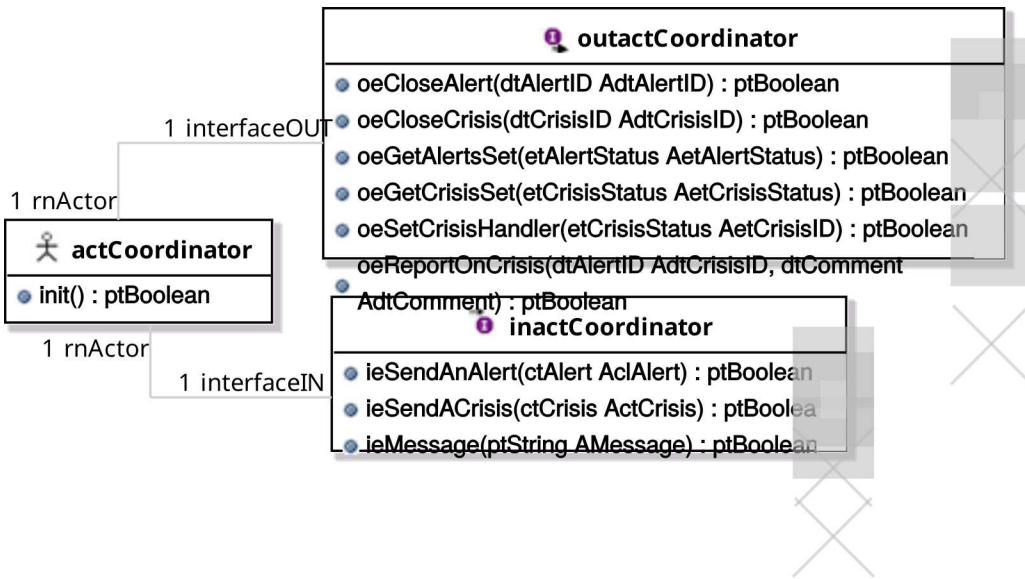


Fig. 3.4 Environment Model - Local View 04..

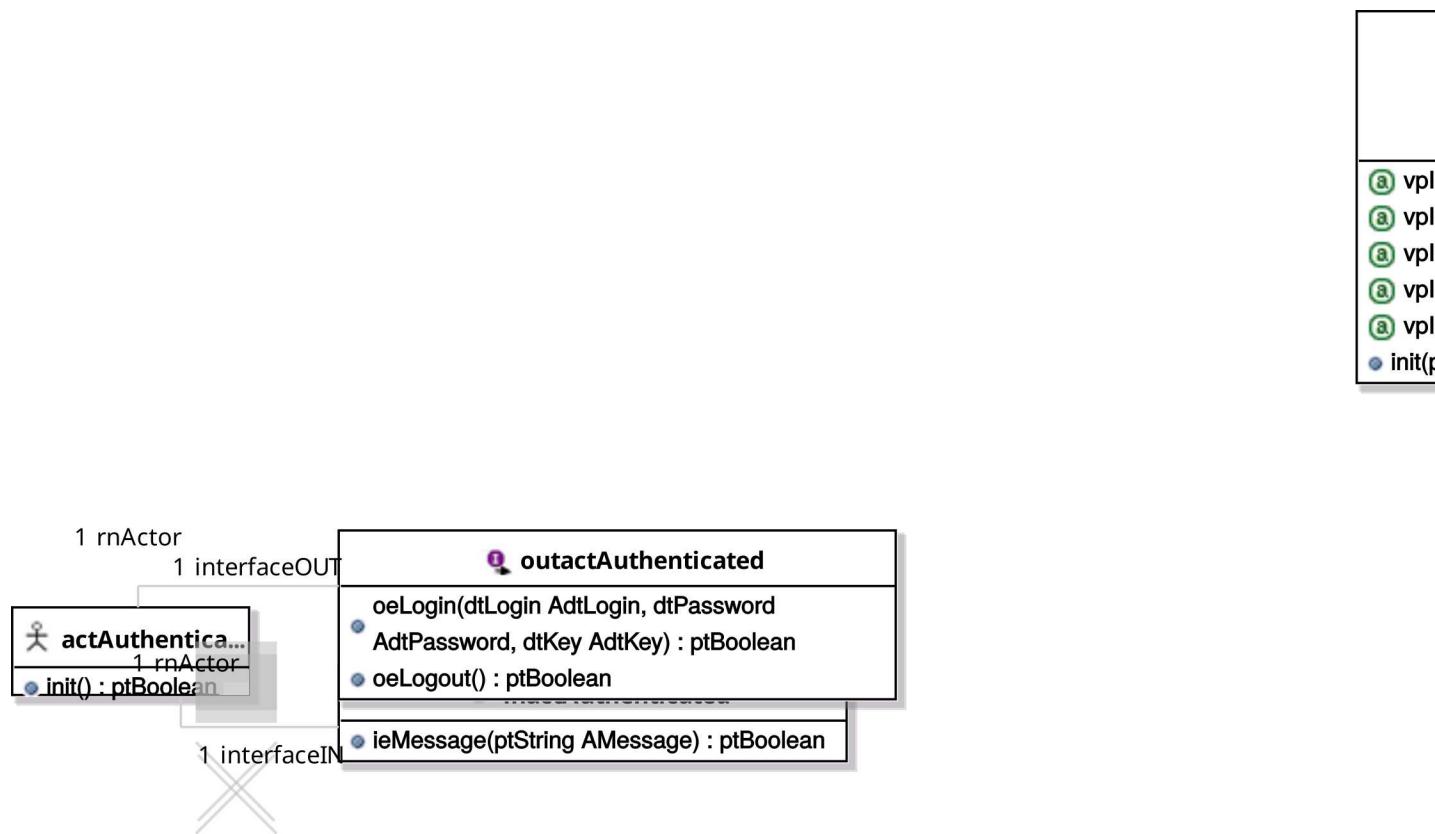


Fig. 3.5 Environment Model - Local View 05..

ACTOR
<i>actDomainExpert</i>
<i>InputInterfaces</i>
IN 1 ieAlertRecieved() :null
<i>OutputInterfaces</i>
OUT 1 oeValidateAlert (dtAlertID AdtAlertID, dtGPSLocation AdtGPSLocation, null AdtDateandTime, etAlertStatus edtAlertStatus, ctDomain ActDomain) :null
OUT 2 oeSollicitateCrisisHandling() :null

3.6.2 *actComCompany* Actor

ACTOR
<i>actComCompany</i>
A Communication Company works as communication link between witness/anonymous/victim and iCrash message exchange and victim's family and iCrash system. Communication Company's objectives are:
- Delivering SMS messages from humans to iCrash phone number - Transmitting SMS messages from iCrash system to Humans (including witness, anonymous, victim, family member) with an SMS enabling device with phone number.
The responsibilities of a Communication Company therefore are:
- Guaranteeing integrity, confidentiality, reliability and availability of the information and message delivery system for any SMS communication between iCrash and Humans.
<i>InputInterfaces</i>
IN 1 ieSmsSend(dtPhoneNumber AdtPhoneNumber, dtSMS AdtSMS) :ptBoolean
<i>OutputInterfaces</i>
OUT 1 oeAlert (etHumanKind AetKind, dtDate AdtMyDate, dtTime AdtTime, dtPhoneNumber AdtPhoneNumber, dtGPSLocation AdtGPSLocation, dtComment AdtComment) :ptBoolean

3.6.3 *actEMS* Actor

ACTOR
<i>actEMS</i>
EMS services is an acronym of Emergency Services. The person operating at EMS end of the system is a human provided with an interface including a login, password and tokenID. The objectives of EMS services are:
- To communicate with a Coordinator at company ABC by receiving and sending messages through iCrash - Report back to iCrash when a rescue unit (and its consistency) has been dispatched to the accident site. - Send information about the accident back to iCrash
Therefore the responsibilities are:
- To be aware how to securely use the iCrash system by possessing information about identification. - Provide the information needed to iCrash in order to send the SMS to correct family number
<i>OutputInterfaces</i>
OUT 1 oeReplyToRequest (dtCrisisID AdtCrisisID, dtComment AdtComment, dtRequestId AdtRequest, etEMSCrisisStatus AdtEMSCrisisStatus, dtHospitalName AdtHospitalName, dtName AdtVictimsName, dtName AdtVictimICEContactName, dtPhoneNumber AdtVictimICEContactPhoneNumber) :ptBoolean
<i>continues in next page ...</i>

...Actor table continuation

OUT 2 oeReportEMSCrisisStatus (dtCrisisID AdtCrisisID, dtComment AdtComment, etEMSCrisisStatus AdtEMSCrisisStatus) :ptBoolean
--

3.6.4 actMsrCreator Actor

ACTOR
actMsrCreator
A Creator is a human working at/for company ABC, possessing high enough technical capabilities to perform the installation of iCrash.
In addition to installation of iCrash, a Creator's role also includes definition of values for intial system's state and environment.
A Creator and Administrator are separate humans and one cannot behold the tasks of both.
OutputInterfaces
OUT 1 oeCreateSystemAndEnvironment (ptInteger AqtyComCompanies) :ptBoolean

3.6.5 actCoordinator Actor

ACTOR
actCoordinator
A Coordinator is a human who has the following objectives: - Monitoring excisting alerts and crisis and to - Managing alerts and crisis until they have been declared resolved
To achieve the objectives, the responsibilities for a Coordinator therefore are:
- Communicating with EMS though iCrash system in order to request for dispatch of vehicles - To be available for crising handling (requests and alerts) - Being able to make a decision if an alert or or crisis can be closed - To be able to use the system safely regarding identification - To send a message to the victim's family through ComCompany containing carefully chosen information about the accident and its victims.
Extends
icrash.environment.actAuthenticated
InputInterfaces
IN 1 ieSendAnAlert (ctAlert AclAlert) :ptBoolean
IN 2 ieSendACrisis (ctCrisis ActCrisis) :ptBoolean
IN 3 ieMessage (ptString AMassage) :ptBoolean
OutputInterfaces
OUT 1 oeCloseAlert (dtAlertID AdtAlertID) :ptBoolean
OUT 2 oeCloseCrisis (dtCrisisID AdtCrisisID) :ptBoolean
OUT 3 oeGetAlertsSet (etAlertStatus AetAlertStatus) :ptBoolean
OUT 4 oeGetCrisisSet (etCrisisStatus AetCrisisStatus) :ptBoolean
OUT 5 oeSetCrisisHandler (etCrisisStatus AetCrisisID) :ptBoolean

continues in next page ...

...Actor table continuation

OUT 6	<code>oeReportOnCrisis(dtAlertID AdtCrisisID, dtComment AdtComment, dtUserID AdtUserID, etCrisisType AdtCrisisType, etCrisisStatus AdtCrisisStatus, dtReceivedMessage AdtReceivedMessage, dtNbrofVehiclesInAccident AdtVehiclesInAccident, dtNbrOfVictims AdtNbrVictims) :ptBoolean</code>
OUT 7	<code>oeSetCrisisStatus(dtCrisisID AdtCrisisID, etCrisisStatus AetCrisisStatus) :ptBoolean</code>
OUT 8	<code>oeValidateAlert(dtAlertID AdtAlertID) :ptBoolean</code>
OUT 9	<code>oeInfoFam(dtComment AdtComment, dtPhoneNumber AdtVictimICEContactPhoneNumber, dtHospitalName AdtHospitalName) :ptBoolean</code>
OUT 10	<code>oeRequestEMSService(dtComment AdtComment, dtRequestID AdtRequest, dtGPSLocation AdtGPSLocation, dtNbrofVehiclesInAccident AdtVehiclesInAccident, dtNbrOfVictims AdtNbrVictims) :ptBoolean</code>

3.6.6 actAdministrator Actor

ACTOR
actAdministrator
Administrating the iCrash system is the responsibility of an Administrator, who is also a human. To fullfill his/her objective:
- Adding users (Coordinators, EMS or DomainExpert) to and from the system and its environment - Deleting users to and from the system and its environment
The responsibilities include:
- Identifying the company ABC employees that are users with access to the system - Be aware of the security policies of company ABC - Be aware of the the system's identification information for secure system usage - Communication with users (Coordinators, EMS or DomainExpert) in order to provide them with identification information to guarantee the secure use of the system.
Extends
icrash.environment.actAuthenticated
InputInterfaces
IN 1 <code>ieUserAdded() :ptBoolean</code>
IN 2 <code>ieUserDeleted() :ptBoolean</code>
OutputInterfaces
OUT 1 <code>oeAddUser(dtUserID AdtUserID, dtLogin AdtLogin, dtPassword AdtPassword, dtTokenID AdtTokenID) :ptBoolean</code>
OUT 2 <code>oeDeleteUser(dtUserID AdtUserID) :ptBoolean</code>

3.6.7 actAuthenticated Actor

ACTOR
actAuthenticated
Authenticated is a super user of iCrash System whom all other users inherit from for safe use of the system.

continues in next page ...

... Actor table continuation

<i>InputInterfaces</i>
IN 1 ieMessage(ptString AMessage) :ptBoolean
<i>OutputInterfaces</i>
OUT 1 oeLogin(dtLogin AdtLogin, dtPassword AdtPassword, dtKey AdtKey):ptBoolean
OUT 2 oeLogout () :ptBoolean

Chapter 4

Concept Model

4.1 PrimaryTypes-Classes

4.1.1 Local view 01

Figure 4.1 shows the primary types class types.

4.1.2 Local view 02

Figure 4.2

4.1.3 Local view 03

Figure 4.3

4.1.4 Local view 04

Figure 4.4

4.1.5 Local view 06

Figure 4.5

Fig. 4.1 Concept Model - PrimaryTypes-Classes local view 01. Local view of the Primary Types Class Types .

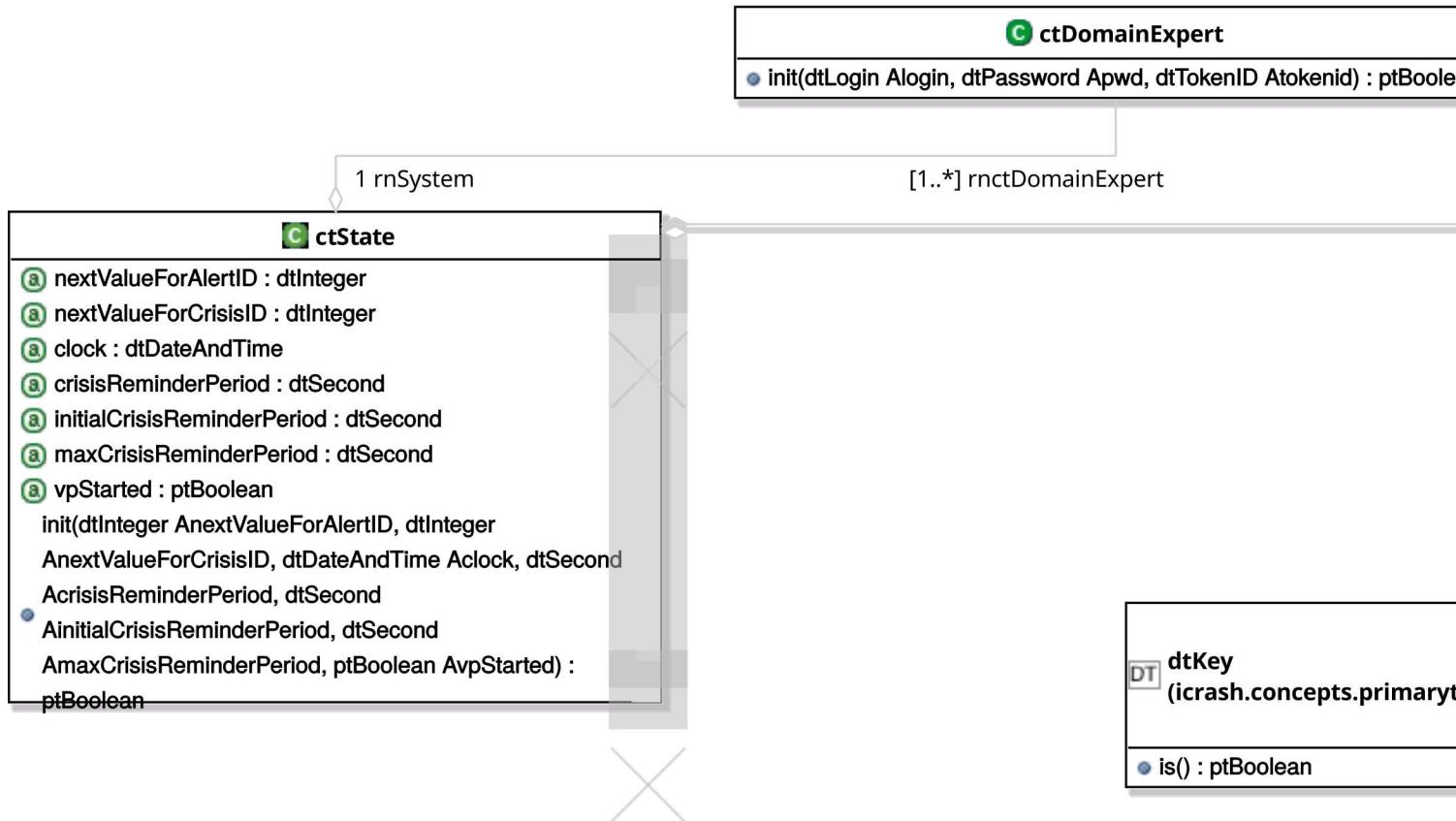


Fig. 4.2 Concept Model - PrimaryTypes-Classes local view 02. .

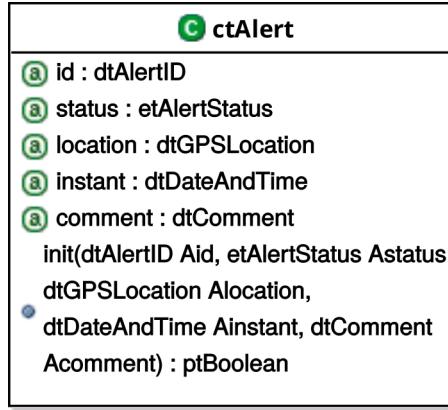


Fig. 4.3 Concept Model - PrimaryTypes-Classes local view 03. .

4.1.6 Global view 01

Figure 4.6 shows the primary types class types together with the association(s) types with the actor classes of the environment model.

4.2 PrimaryTypes-Datatypes

4.2.1 Global view 01

Figure 4.7 shows the primary types datatype types.

4.3 SecondaryTypes-Datatypes

4.3.1 Local view 01

Figure 4.8 shows 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.

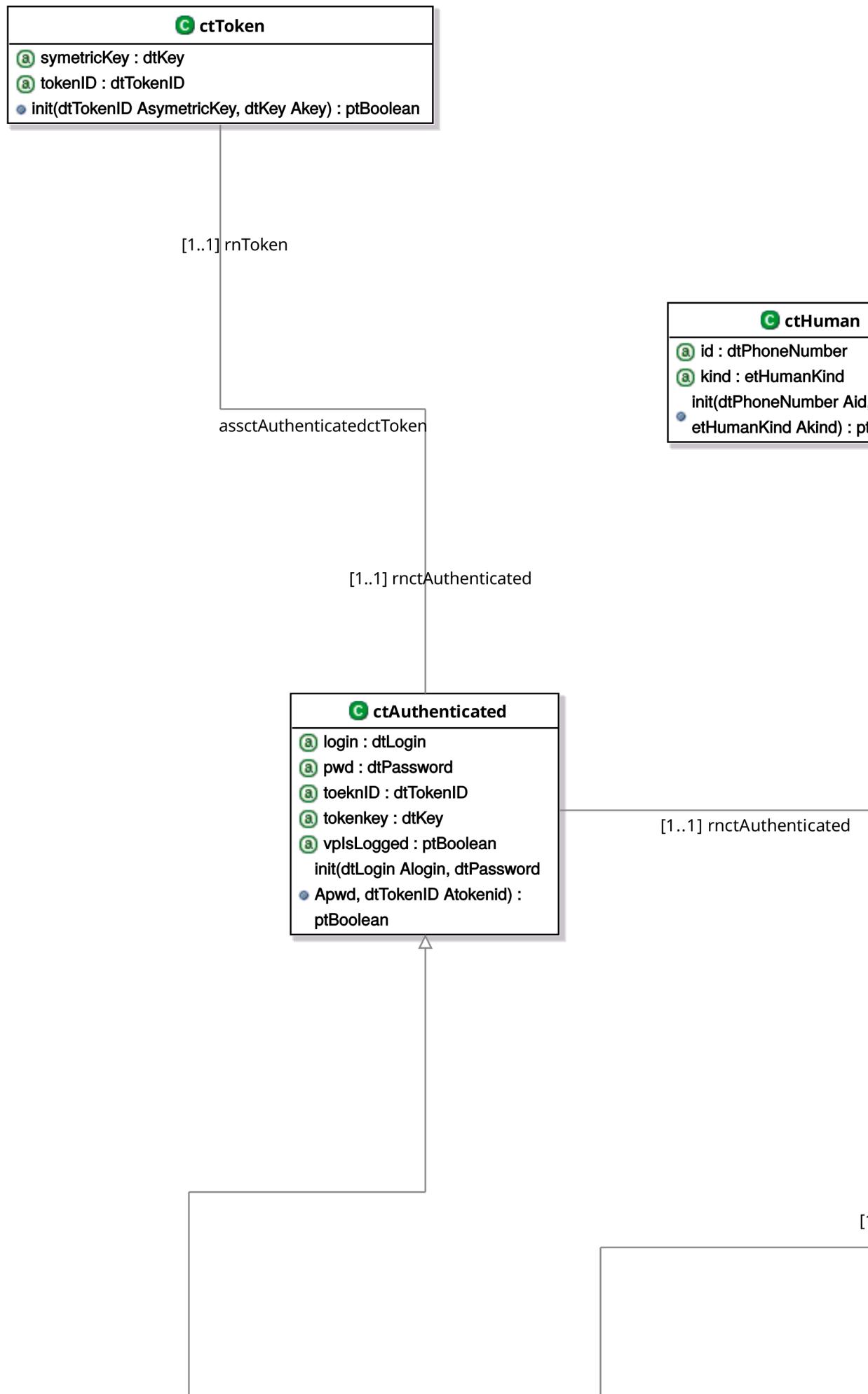
4.4.1 Primary types - Class types descriptions

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.



Fig. 4.4 Concept Model - PrimaryTypes-Classes local view 04. .

Fig. 4.5 Concept Model - PrimaryTypes-Classes local view 06. .



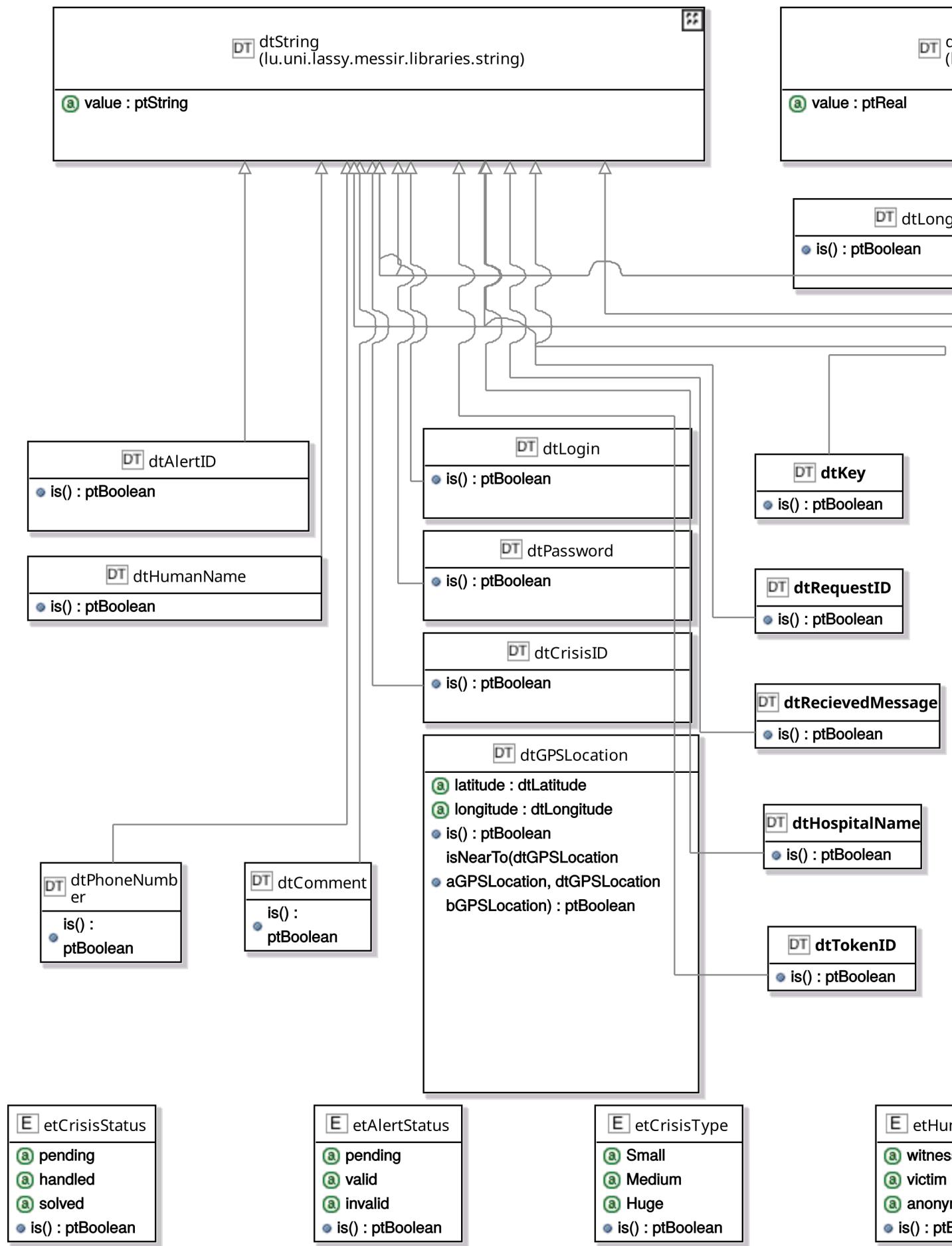


Fig. 4.7 Concept Model - PrimaryTypes-Datatypes global view 01. .

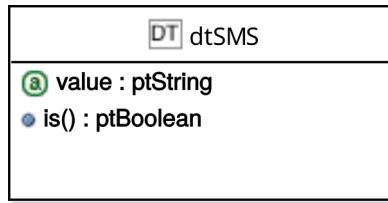


Fig. 4.8 Concept Model - SecondaryTypes-Datatypes local view 01. Local view of the secondary types datatype types.

CLASSES	
<i>ctDomainExpert</i>	
extends	icrash.concepts.primarytypes.classes.ctAuthenticated
attribute	id: dtUserID:
operation	init:
<i>ctCrisis</i>	
This class is a crisis which is initiated from the alerts received by the system.	
attribute	id: this attribute is an ID of the crisis. The type is a string.
attribute	type: this attribute is a status of a crisis. The status is enum and it could take as a value: "small", "medium", "huge".
attribute	status: this attribute is a type of a crisis. The type is enum and it could take as a value: "pending", "handled", "solved".
attribute	location: this attribute is a GPS location of a crisis. The type is dtGPSLocation which contains real numbers for latitude and longitude.
attribute	instant: this attribute is a date and a time when crisis happens.
attribute	comment: this attribute is a miscellaneous text sent as a comment.
operation	init: this operation initializes ctCrisis with necessary attributes.
<i>ctToken</i>	
This class contains information related to a token. Token is used upon logging in in order to provide symmetric encryption and enhanced security.	
attribute	symmetricKey: dtKey:
attribute	tokenID: dtTokenID:
operation	init:
<i>ctHuman</i>	
This class is a human that sends an alert to the system through the communication company.	
attribute	id: this attribute is an ID of a human. The type is a string which covers the phone numbers of human who reports an alert of a crisis.
attribute	kind: this attribute is a status of a human. The status is enum and it could take as a value: "witness", "victim", "anonymous".
operation	init: this operation initializes ctHuman with necessary attributes.
<i>ctCoordinator</i>	
This class is a main user of the system. It inherits from the ctAuthenticated the login information (login name and password).	
extends	icrash.concepts.primarytypes.classes.ctAuthenticated
attribute	id: this attribute is an ID of the coordinator user. The type is a string.
operation	init: this operation initializes ctCoordinator with necessary attributes.
<i>ctState</i>	
This class is the main class that represents the system and the state of the system.	
attribute	nextValueForAlertID: this attribute keeps track of the next value for an ID of an alert.
attribute	nextValueForCrisisID: this attribute keeps track of the next value for an ID of a crisis.
attribute	clock: this attribute keeps the track of time.
attribute	crisisReminderPeriod:
attribute	initialCrisisReminderPeriod:

continues in next page ...

... Classes table continuation

<i>attribute</i>	maxCrisisReminderPeriod:
<i>attribute</i>	vpStarted:
<i>operation</i>	init: this operation initializes ctState with necessary attributes.
ctAdministrator	
This class is in charge of administrating the system and adding/removing coordinator users.	
<i>extends</i>	icrash.concepts.primarytypes.classes.ctAuthenticated
<i>operation</i>	init: this operation initializes ctAdministrator with necessary attributes.
ctAlert	
This class is an alert sent by the Communication Company from the Human to the system and is assigned to a crisis.	
<i>attribute</i>	id: this attribute is an ID of an alert. The type is string.
<i>attribute</i>	status: this attribute is a status of an alert. The type is enum and it could take as a value: "pending", "valid", "invalid".
<i>attribute</i>	location: this attribute is a GPS location of an alert. The type is dtGPSLocation which contains real numbers for latitude and longitude.
<i>attribute</i>	instant: this attribute is a date and a time when alert is received.
<i>attribute</i>	comment: this attribute is a miscellaneous text sent as a comment.
<i>operation</i>	init: this operation initializes ctAlert with necessary attributes.
ctAuthenticated	
This abstract class contains the login information of the system users.	
<i>attribute</i>	login: this attribute is a login name of the system user. The type is a string.
<i>attribute</i>	pwd: this attribute is a lpassword of the system user. The type is a string.
<i>attribute</i>	vpiIsLogged: this boolean attribute takes into account if the user is logged or not.
<i>operation</i>	init: this operation initializes ctAuthenticated with necessary attributes.
ctEMStype	
<i>attribute</i>	vpRequestFireFigher: null:
<i>attribute</i>	vpRequestPolice: null:
<i>attribute</i>	vpRequestAmbulance: null:
<i>operation</i>	init:
ctVictim	
<i>extends</i>	icrash.concepts.primarytypes.classes.ctHuman
<i>attribute</i>	VictimICEPhoneNumber: dtPhoneNumber:
<i>attribute</i>	VictimICEContactName: dtName:
<i>operation</i>	init:
ctDomain	
This class contains information regarding Accident domains, every alert gets assigned one. This helps to categorize accidents and to provide best possible Coordinator with specified knowledge to each crisis. The accident domain is assigned by a Domain Expert.	
<i>attribute</i>	vpiIsPedestrian: ptBoolean:
<i>attribute</i>	vpiIsVehicular: ptBoolean:
<i>attribute</i>	vpiIsWildlife: ptBoolean:
<i>attribute</i>	vpiIsProperty: ptBoolean:
<i>attribute</i>	vpiIsInjury: ptBoolean:
<i>operation</i>	init:

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	
dtHumanName	
This presents the name of the accident victim or for example a witness, whatever the system Coordinator thinks is important to save into the system regarding the crisis.	
<i>extends</i>	null
<i>operation</i>	is:
dtReceivedMessage	
This describes the message/messages received to iCrash.	
<i>extends</i>	lu.uni.lassy.messir.libraries.string.dtString
<i>operation</i>	is:
dtLogin	
Every user (actAdministrator, actCoordinator, actDomainExpert, actEMS) must have a username for logging in.	
<i>extends</i>	null
<i>operation</i>	is:
dtAlertID	
Every alert is auto-assigned an AlertID by the system.	
<i>extends</i>	null
<i>operation</i>	is:
dtKey	
Unique key generated by a token (with a unique tokenID) for users to log into the system using symmetric encryption.	
<i>extends</i>	lu.uni.lassy.messir.libraries.string.dtString
<i>operation</i>	is:
dtPassword	
Every user with a login and token must also have a password. Password is entered upon a login operation.	
<i>extends</i>	null
<i>operation</i>	is:
dtLongitude	
Describes the longitude GPS-coordinates of the accident venue.	
<i>extends</i>	null
<i>operation</i>	is:
dtNbrOfVictims	
<i>extends</i>	null
<i>operation</i>	is:
dtRequestID	
<i>extends</i>	null
<i>operation</i>	is:
dtPhoneNumber	
This datatype includes the phone number of for example witness who sent a message about the accident or phone number of the victim's family.	
<i>extends</i>	null
<i>operation</i>	is:
dtNbrofVehiclesInAccident	
<i>extends</i>	null
<i>operation</i>	is:
dtVictimICEContactName	
<i>extends</i>	null
<i>operation</i>	is:
dtGPSLocation	

continues in next page ...

... Datatypes table continuation

Describes the longitude and latitude GPS-coordinates of the accident location.	
<i>attribute</i>	latitude: dtLatitude:
<i>attribute</i>	longitude: dtLongitude:
<i>operation</i>	is:
<i>operation</i>	isNearTo:
<i>dtName</i>	
<i>extends</i>	null
<i>operation</i>	is:
<i>dtComment</i>	
This describes a comment that can be inserted into an accident report or a message.	
<i>extends</i>	null
<i>operation</i>	is:
<i>dtUserID</i>	
<i>extends</i>	null
<i>operation</i>	is:
<i>dtLatitude</i>	
Describes the latitude GPS-coordinates for the accident location.	
<i>extends</i>	null
<i>operation</i>	is:
<i>dtCrisisID</i>	
Every crisis gets auto-assigned an ID.	
<i>extends</i>	null
<i>operation</i>	is:
<i>dtTokenID</i>	
Every token has a unique tokenID, therefore every human with a login has a unique tokenID.	
<i>extends</i>	lu.uni.lassy.messir.libraries.string.dtString
<i>operation</i>	is:
<i>dtHospitalName</i>	
This describes the name of the possible hospital where the victim's were taken to. This same hospital name is sent to the victim's family for later reunion.	
<i>extends</i>	lu.uni.lassy.messir.libraries.string.dtString
<i>operation</i>	is:
<i>dtVictimICEContactPhoneNumber</i>	
<i>extends</i>	null
<i>operation</i>	is:

ENUMERATIONS***etHumanKind***

Describes the sender of the original message (victim, witness, anonymous).

operation is:***etEMSCrisisStatus****operation* is:***etUserType****operation* is:***etCrisisType***

Describes the type of the crises (Pedestrian, Wildlife, Property, Injury)

operation is:*continues in next page ...*

... Enumerations table continuation

<i>etCrisisStatus</i>
Describes the status of the crisis (pending, active, resolved).
<i>operation</i> is:
<i>etAlertStatus</i>
Describes the status of the alert (valid, invalid).
<i>operation</i> is:

4.4.3 Primary types - Association types descriptions

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

UNDIRECTED ASSOCIATIONS
<i>assctDomainExpertDomainExpert</i>
<i>assctDomainExpertDomainsctDomains</i>
<i>assctHumanactComCompany</i> The initial SMS sent by witness/victim/anonymous gets delivered to iCrash by Communication Company.
<i>assctAlertctCrisis</i> Every alert is turned into a crisis after the alert has been validated by the Coordinator.
<i>assctAlertctHuman</i> Every alert is associated with a human (witness, anonymous, victim) who sent the initial SMS about the accident to iCrash.
<i>assctCrisisctCoordinator</i> Every crisis has as specialist handling it, this specialist is called a Coordinator.
<i>assctCrisisctDomains</i> Every crisis gets assigned a Domain (pedestrian, wildlife, property, injury).
<i>assctEMSEMS</i>
<i>assctAlertctDomains</i> Every crisis must be assigned a domain (Wildlife, Pedestrian, Property, Injury).
<i>assctAuthenticatedctToken</i> Every authenticated user must have a token.
<i>assctAuthenticatedactAuthenticated</i>
<i>assDomainExpertctDomains</i>
<i>assctCoordinatoractCoordinator</i>
<i>assctCoordinatorctDomains</i> Every coordinator is assigned an alert linked to a specific accident domain by the Domain Expert.

4.4.4 Primary types - Aggregation types descriptions

There are no aggregation types for the primary types.

4.4.5 Primary types - Composition types descriptions

There are no composition types for the primary types.

4.4.6 Secondary types - Class types descriptions

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

4.4.7 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	
<i>dtSMS</i>	
This describes the message containing detailed information about the accident and its victims sent to the family.	
<i>attribute</i>	<i>value:</i> null:
<i>operation</i>	<i>is:</i>

4.4.8 Secondary types - Association types descriptions

There are no association types for the secondary types.

4.4.9 Secondary types - Aggregation types descriptions

There are no aggregation types for the secondary types.

4.4.10 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, or in a primary or secondary type (class, datatype or enumeration types). The **Mess1p** OCL code listing is joined to the comment table.

5.1 Environment - Out Interface Operation Scheme for actAuthenticated

5.1.1 Operation Model for oeLogin

The oeLogin operation has the following properties:

OPERATION	
oeLogin	
An authenticated user wants to login to the system to use it safely, this is done by entering a username, password and a key generated by a token.	
Parameters	
1	null AdtLogin
2	null AdtPassword
3	null AdtKey
Return type	
null	
Pre-Condition (functional)	
PreF 1	
Pre-Condition (protocol)	
PreP 1 1. the iCrash system has been deployed. 2. the authenticated user that is trying to login has to exsist within the system.	
Post-Condition (functional)	
PostF 1 The authenticated user has now been logged into the system and can now use it safely.	
Post-Condition (protocol)	
PostP 1 The the protocol variable for the authenticated user executing this opperation, preP must be vpLogged = true	
Example	
ctAdministratorActor actor bill sends the message oeLogin(icrashadmin,7WXC1359) to securly connect to the system.	

5.1.2 Operation Model for oeLogout

The oeLogout operation has the following properties:

OPERATION	
oeLogout	
An AuthenticatedActor aims to disconnect from the system.	
Return type	
null	
Pre-Condition (functional)	
PreF 1	
Pre-Condition (protocol)	
PreP 1 1. the iCrash system has been deployed. 2. The AuthenticatedActor has to be logged in.	
Post-Condition (functional)	
PostF 1 The AutenticatedActor has been disconnected from the system.	
Post-Condition (protocol)	
PostP 1 The the protocol variable for the authenticated user executing this operation, preP must be vpLogged = false	
Example	
the ctAdministratorActor bill sends the message oeLogout() to disconnect from the system.	

5.2 Environment - Out Interface Operation Scheme for actComCompany

5.2.1 Operation Model for oeAlert

The oeAlert operation has the following properties:

OPERATION	
oeAlert	
The ComCompany transfers a declaration of a car crash by an etHumanKind indicating specific information.	
Parameters	
1	null AdtKind
2	null AdtMyDate
3	null AdtTime
4	null AdtPhoneNumber
5	null AdtGPSLocation
6	null AdtComment
Return type	
null	
Pre-Condition (functional)	
PreF 1	
Pre-Condition (protocol)	

continues in next page ...

... Operation table continuation

PreP 1	1. the iCrash system has been deployed. 2. a message has been sent by a etHuman to ComCompany.
<i>Post-Condition (functional)</i>	
PostF 1	A crisis is received by the system.
<i>Post-Condition (protocol)</i>	
PostP 1	A crisis is created.
<i>Example</i>	
the ctComCompanyActor actor tango sends the message oeAlert(witness, 2017-11-26-at-10-10-16AM, +352466445252, 49.627675-6.159590,'3 cars invloved in an accident.')	

The listing 5.1 provides the **Mess1p** OCL description of the operation.

```

1 iyyiyuuouuhuhuhulhu
2 hii
3 preF: if coordinator

```

Listing 5.1 **Mess1p** OCL description of the operation *oeAlert*.

5.3 Environment - Out Interface Operation Scheme for actMsrCreator

5.3.1 Operation Model for *oeCreateSystemAndEnvironment*

The *oeCreateSystemAndEnvironment* operation has the following properties:

OPERATION
<i>oeCreateSystemAndEnvironment</i>
Actor MsrCreator aims to create the iCrash system and enviroment at company ABC.
Parameters
1 ptInteger AqtyComCompanies
Return type
ptBoolean
<i>Pre-Condition (functional)</i>
PreF 1
<i>Pre-Condition (protocol)</i>
PreP 1
<i>Post-Condition (functional)</i>
PostF 1 1. the iCrash system has been deployed.
<i>Post-Condition (protocol)</i>
PostP 1
<i>Example</i>
the ctMSRCreatorActor actor the Creator sends the message <i>oeCreateSystemAndEnviroment</i> (1) requesting the initialization fo the system and its enviroment(made of one administrator identified by clock) and indicating that the number of communication company actor instances for the system's enviroment is 1(identified here by tango)

The listing 5.2 provides the **Mess1p** OCL description of the operation.

```

1  context Environment::outactMsrCreator::
2  oeCreateSystemAndEnvironment(qtyComCompanies: Integer)
3  /*-----*/
4  pre:
5  /* Pre Protocol:*/
6  /* Pref01 */
7  true
8  /* Pre Functional:*/
9  /* PreP01 */
10 and true
11 and true
12 /*-----*/
13 post:
14 let TheState:PrimaryTypesClasses::ctState in
15 let AactMsrCreator:Environment::actMsrCreator in
16
17 /* Post Functional:*/
18 /* PostF01 */
19 TheState._init(1,true)
20
21 /* PostF02 */
22 and TheState.rnactComCompany->size() = qtyComCompanies
23 and TheState.rnactComCompany
24   ->forAll(cca:Environment::actComCompany | cca._init())
25
26 /* PostF03 */
27 and AactMsrCreator._init()
28
29 /* Post Protocol:*/
30 /* PostP01 */
31 and TheState.vpmStarted = true

```

Listing 5.2 **Messip** OCL description of the operation *oeCreateSystemAndEnvironment*.

5.4 Environment - Actor Operation Scheme for actComCompany

5.4.1 Operation Model for *init*

The *init* operation has the following properties:

OPERATION
<i>init</i>
This declares the actor ComCompany.
<i>Return type</i>
null
<i>Pre-Condition (functional)</i>
PreF 1
<i>Post-Condition (functional)</i>
PostF 1
<i>Example</i>

The listing 5.3 provides the **Messip** OCL description of the operation.

```

1  context Environment::actComCompany :: _init(): Boolean
2  /*-----*/
3  pre: true
4  /*-----*/
5  post:
6  let pIN:Environment::inactComCompany in
7  let pOUT:Environment::outactComCompany in
8  if (
9    /* Post Functional:*/
10   /

```

```

11  /* Post F01 */
12  self.oclIsNew()
13  /* Post F02 */
14  and pIN.oclIsNew()
15  and pOUT.oclIsNew()
16  and pIN = self.InterfaceIN
17  and pOUT= self.InterfaceOUT
18  )
19  then result = true
20  else result = false
21  endif
22  -----*/

```

Listing 5.3 Message OCL description of the operation *init*.

5.5 Primary Types - Operation Schemes for Class ctAdministrator

5.5.1 Operation Model for *init*

The *init* operation has the following properties:

OPERATION	
<i>init</i>	This operation initialises the class Administrator.
Parameters	
1	null Alogin
2	null Apwd
Return type	
null	
Pre-Condition (functional)	
PreF 1 True	
Post-Condition (functional)	
PostF 1 True	
Example	

5.6 Primary Types - Operation Schemes for Class ctAlert

5.6.1 Operation Model for *init*

The *init* operation has the following properties:

OPERATION	
<i>init</i>	This operation initialises the class Alert.
Parameters	
1	null Aid
2	null Astatus

continues in next page ...

...Operation table continuation

3	null
	Alocation
4	null
	Ainstant
5	null
	Acomment
<i>Return type</i>	
null	
<i>Pre-Condition (functional)</i>	
PreF 1	True
<i>Post-Condition (functional)</i>	
PostF 1	True
<i>Example</i>	

5.7 Primary Types - Operation Schemes for Class ctState

5.7.1 Operation Model for *init*

The *init* operation has the following properties:

OPERATION	
<i>init</i>	
This operation initialises the class State.	
<i>Parameters</i>	
1 null AnextValueForAlertID	
2 null AnextValueForCrisisID	
3 null Aclock	
4 null AcrisisReminderPeriod	
5 null AinitialCrisisReminderPeriod	
6 null AmaxCrisisReminderPeriod	
7 null AvpStarted	
<i>Return type</i>	
null	
<i>Pre-Condition (functional)</i>	
PreF 1	True
<i>Post-Condition (functional)</i>	
PostF 1	
<i>Example</i>	
True	

5.8 Primary Types - Operation Schemes for Datatype dtAlertID

5.8.1 Operation Model for *is*

The *is* operation has the following properties:

OPERATION
<i>is</i>
Displays the ID for an alert.
<i>Return type</i>
null
<i>Pre-Condition (functional)</i>
PreF 1 True
<i>Post-Condition (functional)</i>
PostF 1 True
<i>Example</i>
The alert ID is:789879

The listing 5.4 provides the **Mess1p** OCL description of the operation.

```

1  context PrimaryTypesDatatypes::dtAlertID :: 
2    is() : Boolean
3    /*-----*/
4    pre:
5      /* Pre Functional:*/
6      /* Pre F01 */
7      true
8    post:
9      let TheResult:Boolean in
10     if
11     (
12       (
13       /* Post Functional:*/
14       /* Post F01 */
15       self.value > 0
16     )
17     then TheResult = true
18     else TheResult = false
19     endif
20     and result = TheResult
21
22   /*-----*/

```

Listing 5.4 **Mess1p** OCL description of the operation *is*.

5.9 Primary Types - Operation Schemes for Datatype dtComment

5.9.1 Operation Model for *is*

The *is* operation has the following properties:

OPERATION
<i>is</i>
A comment can be added or be included in any message received from a human or EMS. Comments can also be used by Coordinators to add comments into reports.
<i>Return type</i>

continues in next page ...

... Operation table continuation

null
<i>Pre-Condition (functional)</i>
PreF 1 True
<i>Post-Condition (functional)</i>
PostF 1 True
<i>Example</i>
A message from a witness to iCrash: 'Please send ambulance'

The listing 5.5 provides the **Mess1p** OCL description of the operation.

```

1 context PrimaryTypesDatatypes::dtComment::
2   is() : Boolean
3   /*-----*/
4   pre:
5     /* Pre Functional:*/
6     /* Pre F01 */
7     true
8   post:
9     let TheResult:Boolean in
10    let MaxLength:Integer in
11    if
12      (
13        /* Post Functional:*/
14        /* Post F01 */
15        MaxLength = 160
16        and self.value.size() <= MaxLength
17      )
18    then TheResult = true
19    else TheResult = false
20  endif
21  and result = TheResult
22 /*-----*/
23

```

Listing 5.5 **Mess1p** OCL description of the operation *is*.

5.10 Primary Types - Operation Schemes for Datatype dtPhoneNumber

5.10.1 Operation Model for *is*

The *is* operation has the following properties:

OPERATION
<i>is</i>
This presents the mobile phone number of the human sending in the initial SMS about the accident as well as the phone number of the family contact.
<i>Return type</i>
null
<i>Pre-Condition (functional)</i>
PreF 1 True
<i>Post-Condition (functional)</i>
PostF 1 True
<i>Example</i>
The phone number of the witness is: 5813935

The listing 5.6 provides the **Mess1p** OCL description of the operation.

```

1 context PrimaryTypesDatatypes:: dtPhoneNumber::
2 is () : Boolean
3 /*-----*/
4 pre:
5 /* Pre Functional:*/
6 /* Pre F01 */
7 true
8
9 post:
10 let TheResult: Boolean in
11 if
12 (
13 /* Post Functional:*/
14 /* Post F01 */
15 self.value.size() = 10
16 )
17 then TheResult = true
18 else TheResult = false
19 endif
20 and result = TheResult
21 /*-----*/

```

Listing 5.6 **Messir** OCL description of the operation *is*.

5.11 Primary Types - Operation Schemes for Enumerations

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

5.12 Secondary Types - Operation Schemes for Classes

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

5.13 Secondary Types - Operation Schemes for Datatype dtSMS

5.13.1 *Operation Model for is*

The *is* operation has the following properties:

OPERATION
<i>is</i>
Return type
null
Pre-Condition (functional)
PreF 1 True
Post-Condition (functional)
PostF 1 True
Example

5.14 Secondary Types - Operation Schemes for Enumerations

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

Chapter 6

Test Model(s)

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

Chapter 7

Additional Constraints

Appendix A

Use Case Description Tables

A.1 Use cases list(s)

- Summary
 - suDeployAndRun
 - suGlobalCrisisHandling
- user-goal
 - ugAdministrateTheSystem
 - ugManageCrisis
 - ugMonitor
 - ugSecurelyUseSystem
- subfunction
 - oeAddCoordinator
 - oeAlert
 - oeCloseAlert
 - oeCloseCrisis
 - oeCreateSystemAndEnvironment
 - oeDeleteCoordinator
 - oeGetAlertsSet
 - oeSetCrisisHandler
 - oeInfoFam
 - oeLogin
 - oeLogout
 - oeReportOnCrisis
 - oeRequestEMSService
 - oeReportEMSCrisisStatus
 - oeReplyToRequest
 - oeSetCrisisStatus
 - oeSollicitateCrisisHandling
 - oeValidateAlert

- suDeployAndRun
 - oeCreateSystemAndEnvironment
 - ugAdministateTheSystem
 - suGlobalCrisisHandling
 - oeAlert
- suGlobalCrisisHandling
 - ugSecurelyUseSystem
 - ugMonitor
 - ugManageCrisis
- ugAdministateTheSystem
 - ugSecurelyUseSystem
 - oeAddUser
 - oeDeleteUser
- ugSecurelyUseSystem
 - oeLogin
 - oeLogout
- ugManageCrisis
 - oeSetCrisisStatus
 - oeSetCrisisHandler
 - oeReportOnCrisis
 - oeRequestEMSAssistance
 - oeCloseCrisis
- ugMonitor
 - oeGetAlertSet
 - oeValidateAlert
 - oeSollicitateCrisisHandling

A.2 Use case Table(s)

A.2.1 Summary

USE-CASE DESCRIPTION	
<i>Name</i>	suDeployAndRun
<i>Scope</i>	System
<i>Altitude</i>	Summary
Parameters	
1	none
Primary actor(s)	
1	actAdministrator[active]
Secondary actor(s)	
1	actMsrCreator[active]
2	actCoordinator[active]
3	actComCompany[active]
Goal(s) description	
The goal is to install the <i>iCrash</i> system on its infrastructure and to exploit its capacities related to the secure administration and efficient handling of car crash crisis situations depending on alerts received.	
Reuse	
1	<u>oeCreateSystemAndEnvironment</u>
2	<u>ugAdministateTheSystem</u> [1..*]
3	<u>suGlobalCrisisHandling</u> [1..*]
4	<u>oeAlert</u> [1..*]
Protocol condition(s)	
1	the <i>iCrash</i> system has never been deployed and used
Pre-condition(s)	
1	none
Main post-condition(s)	
1	<i>iCrash</i> system has been created and able to handled the crisis situations for which it received alerts through the communication company.
Main success steps	
a	the actor actMsrCreator executes the <u>oeCreateSystemAndEnvironment</u> use case
b	the actor actAdministrator executes the <u>ugAdministateTheSystem</u> use case
c	the actor actComCompany executes the <u>oeAlert</u> use case
d	the actor actCoordinator executes the <u>suGlobalCrisisHandling</u> use case
Step Constraints Ordering and Extensions	
1	step (a) must be always the first step.
2	step (b) (c) and (d) executions are interleaved.
3	step (b) (c) (d) can be executed multiple times.
4	step (d) can be executed by different actCoordinator actors.
5	step (d) can be executed as a reaction to step (c).
Additional Information	
none	

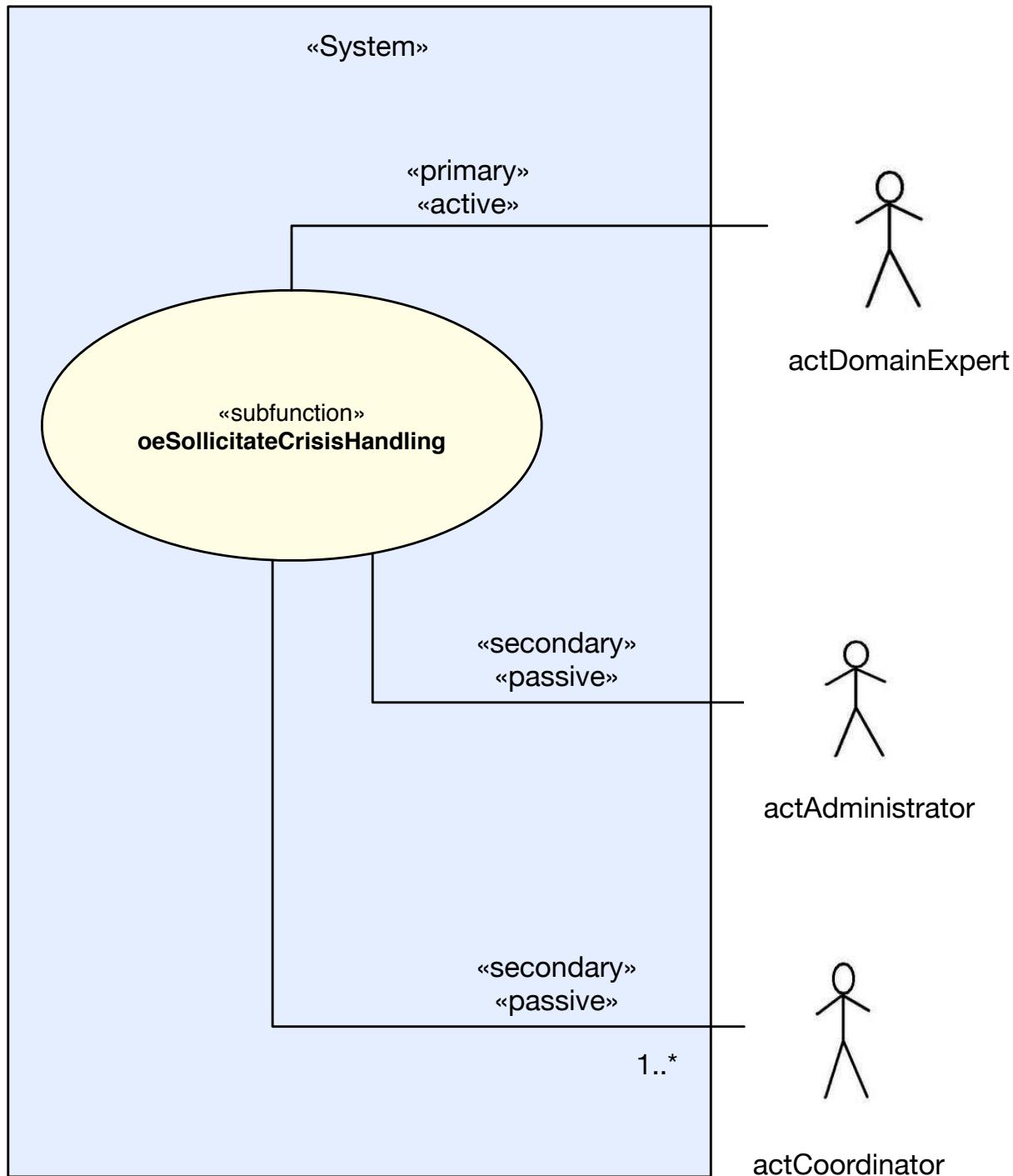


Fig. A.1 *iCrash* Use Case Diagram: **oeDeployAndRun**

USE-CASE DESCRIPTION	
<i>Name</i>	suGlobalCrisisHandling
<i>Scope</i>	System
<i>Altitude</i>	Summary
<i>Parameters</i>	
1	none
<i>Primary actor(s)</i>	
1	actCoordinator[active]
2	actCoordinator[active]
<i>Secondary actor(s)</i>	
1	none
<i>Goal(s) description</i>	
the actCoordinator and actDomainExpert's goals are to monitor the alerts received and the corresponding crises in order to be able to act as necessary to handle the crises.	
<i>Reuse</i>	
1	ugSecurelyUseSystem[1..*]
2	ugMonitor[1..*]
3	ugManageCrisis[1..*]
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been deployed
2	the actCoordinator involved in the use case has been declared by the actor actAdministrator.
3	the actDomainExpert involved in the use case has been declared by the actor actAdministrator.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	modifications have been made by the Domain expert on existing alerts.
2	modifications have been made by the Coordinator on the existing crises.
<i>Main success steps</i>	
a	the actor actCoordinator and the actor actDomainExpert execute the ugSecurelyUseSystem use case
b	the actor actDomainExpert executes the ugMonitor use case
c	the actor actCoordinator executes the ugManageCrisis use case
<i>Step Constraints Ordering and Extensions</i>	
1	steps (a) (b) and (c) executions are interleaved (steps (b) and (c) have there protocol constrained by steps of (a)).
2	step (b) must be executed before (c) can be executed due to domain constraint related to alerts.
3	steps (a) (b) and (c) can be executed multiple times.
<i>Additional Information</i>	
there might be several actors actCoordinator that concurrently execute their suGlobalCrisisHandling use cases.	

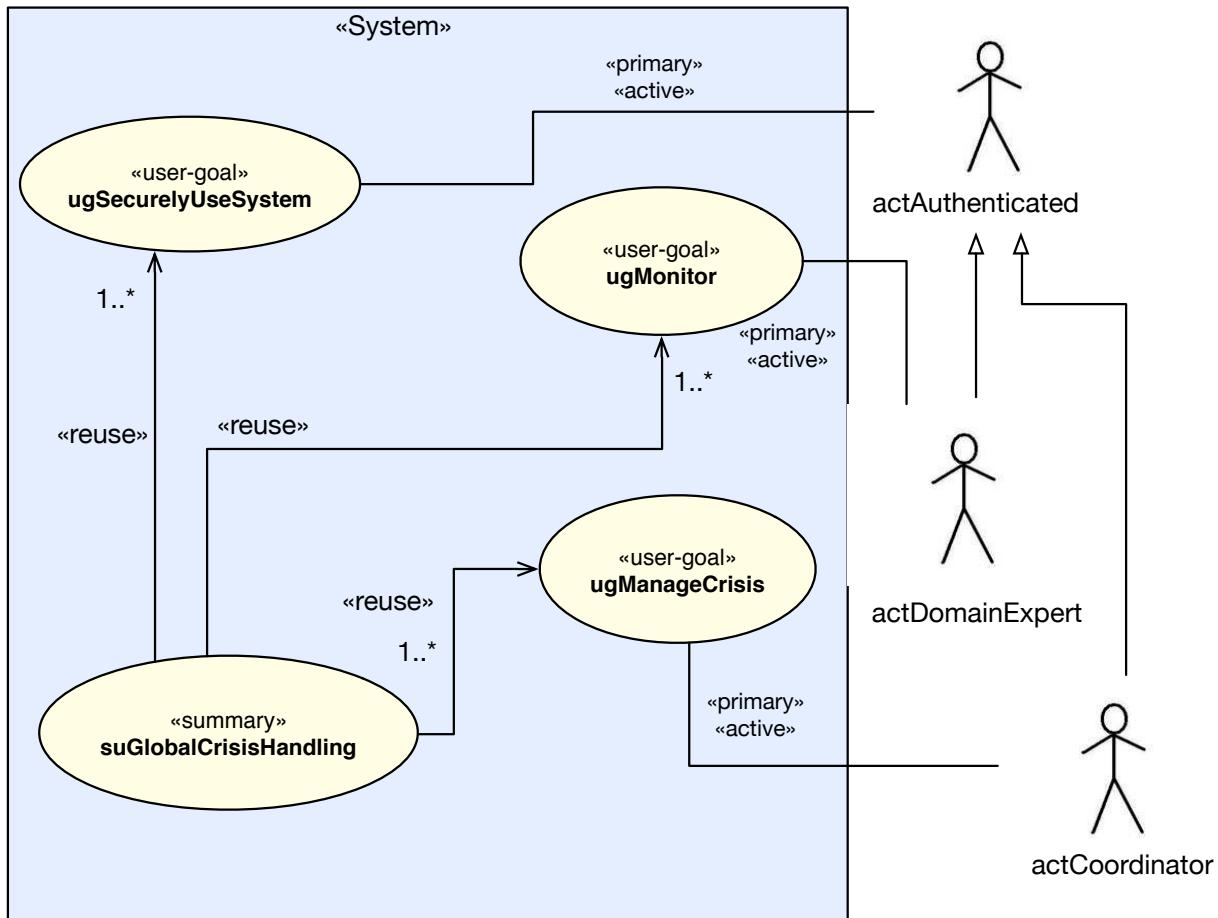
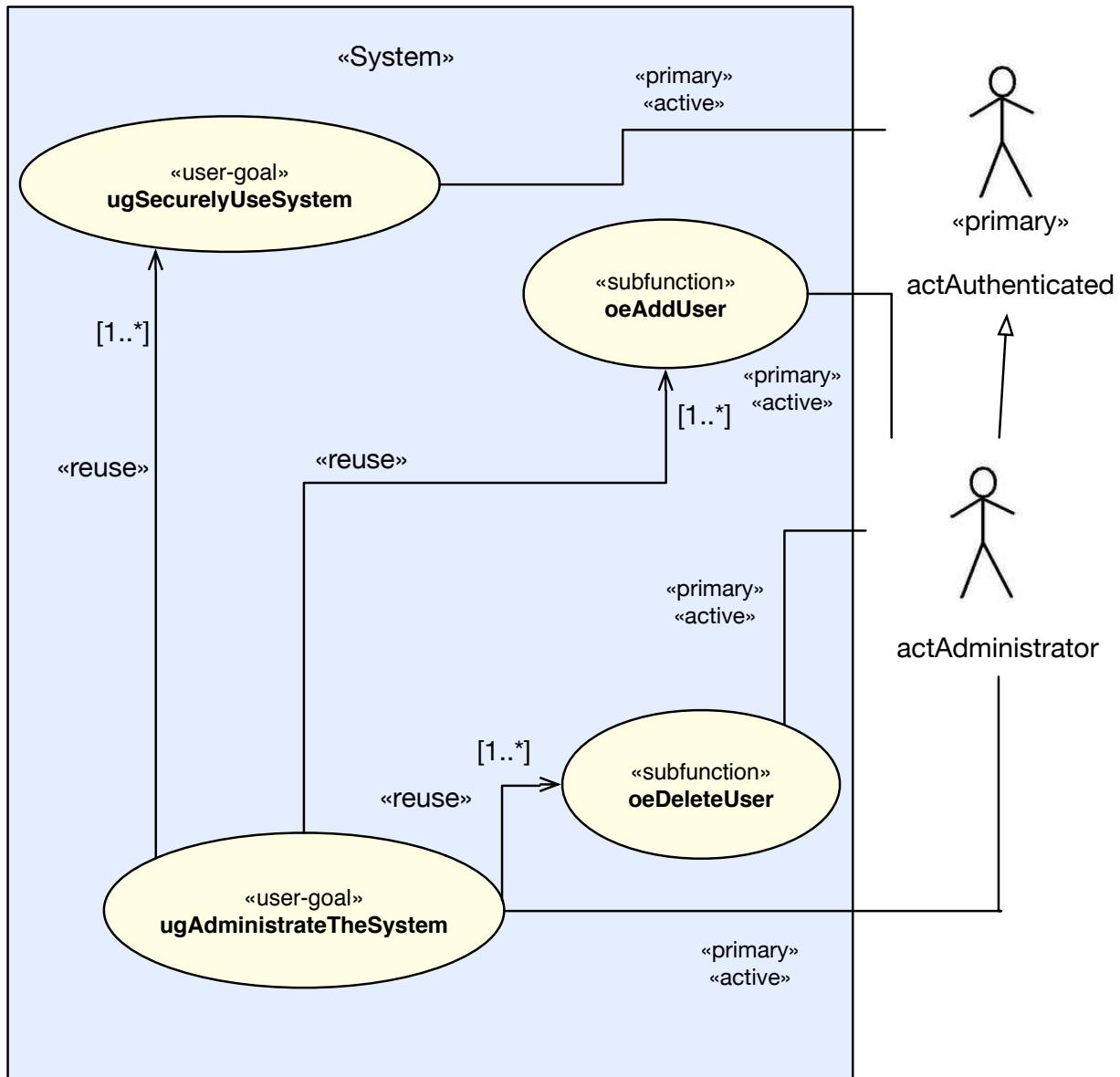


Fig. A.2 *iCrash* Use Case Diagram: ugGlobalCrisisHandling

A.2.2 User-goal

USE-CASE DESCRIPTION	
<i>Name</i>	ugAdministateTheSystem
<i>Scope</i>	System
<i>Altitude</i>	Summary
<i>Parameters</i>	
1	none
<i>Primary actor(s)</i>	
1	actAdministrator[active]
<i>Secondary actor(s)</i>	
1	none
<i>Goal(s) description</i>	
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.	
<i>Reuse</i>	
1	ugSecurelyUseSystem[1..*]
2	oeAddUser[1..*]
3	oeDeleteUser[1..*]
<i>Protocol condition(s)</i>	
1	the iCrash system has been deployed.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	a new user is added to the system and can now securely use the system.
<i>Main success steps</i>	
a	the actor actAdministrator executes the <u>ugSecurelyUseSystem</u> use case.
b	the actor actAdministrator executes the <u>oeAddUser</u> use case.
c	the actor actAdministrator executes the <u>oeDeleteUser</u> use case.
<i>Step Constraints Ordering and Extensions</i>	
1	steps (a) (b) and (c) executions are interleaved (steps (b) and (c) have there protocol constrained by steps of (a)).
2	steps (a) (b) and (c) can be executed multiple times.
<i>Additional Information</i>	
none	

**Fig. A.3** *iCrash* Use Case Diagram: ugAdministateTheSystem

USE-CASE DESCRIPTION	
<i>Name</i>	ugSecurelyUseSystem
<i>Scope</i>	System
<i>Altitude</i>	user-goal
<i>Parameters</i>	
1	none
<i>Primary actor(s)</i>	
1	actAuthenticated[abstract]
<i>Secondary actor(s)</i>	
1	none
<i>Goal(s) description</i>	
the goal is to follow an identification procedure and thus be allowed to safely use the system and its features.	
<i>Reuse</i>	
1	<u>oeLogin</u>
2	<u>oeLogout</u>
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been deployed.
2	the a useraccount has been created for the user trying to securely use the system.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	the actAuthenticated is known by the system not to be logged.
<i>Main success steps</i>	
a	the actor actAuthenticated executes the <u>oeLogin</u> use case.
b	the actor actAuthenticated executes the <u>oeLogout</u> use case
<i>Step Constraints Ordering and Extensions</i>	
1	step (a) must always precede step (b).
<i>Additional Information</i>	
none.	

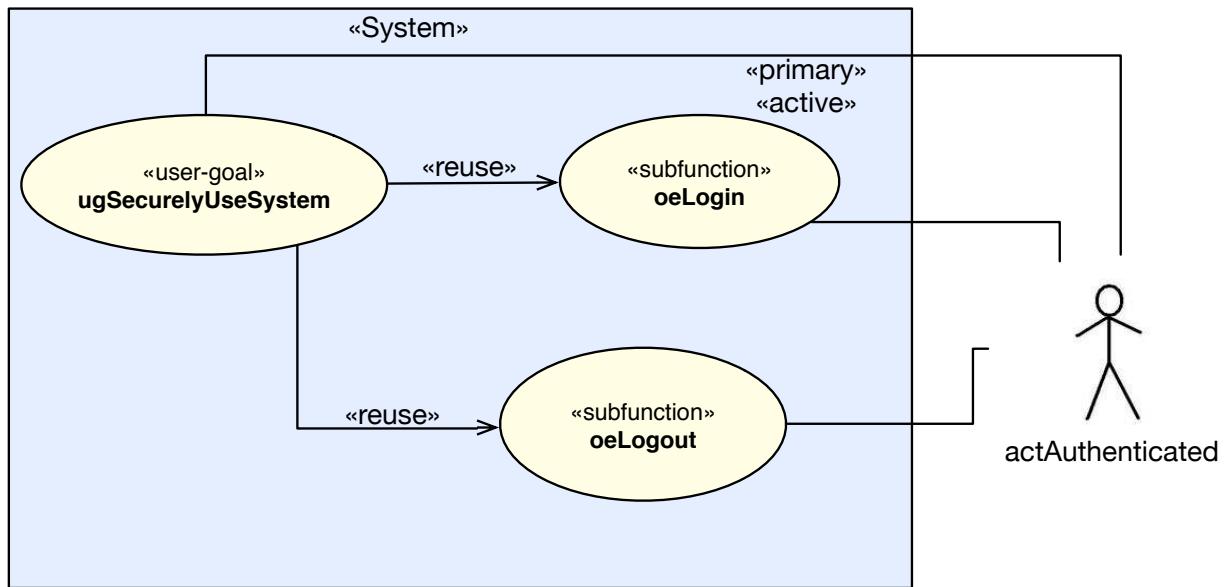


Fig. A.4 *iCrash* Use Case Diagram: ugSecurelyUseSystem

USE-CASE DESCRIPTION	
<i>Name</i>	ugManageCrisis
<i>Scope</i>	System
<i>Altitude</i>	Summary
Parameters	
1	none
Primary actor(s)	
1	actCoordinator[active]
Secondary actor(s)	
1	none.
Goal(s) description	
The goal is to use all operations available to the coordinator for the successful handling of a crisis or an alert received by our system.	
Reuse	
1	<u>oeSetCrisisStatus</u>
2	<u>oeSetCrisisHandler</u>
3	oeRequestEMSAssistance
4	<u>oeReportOnCrisis</u>
5	<u>oeCloseCrisis</u>
Protocol condition(s)	
1	the <i>iCrash</i> system has been started
2	the actCoordinator has been added to the system.
3	the actCoordinator has safely logged on to the system.
4	the actDomainExpert has validated the alert.
5	the actCoordinator has the right domains of expertise to manage this alert.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	there is a crisis whose related information has been changed.
Main success steps	
a	the actor actCoordinator executes the <u>oeSetCrisisHandler</u> use case.
b	the actor actCoordinator executes the <u>oeSetCrisisStatus</u> use case.
c	the actor actCoordinator executes the <u>oeReportOnCrisis</u> use case.
d	the actor actCoordinator executes the <u>oeCloseCrisis</u> use case.
Step Constraints Ordering and Extensions	
1	Step a) must be executed first.
2	Step d) must be executed last.
3	Steps b)to c) can be executed multiple times in order to manage a crisis.

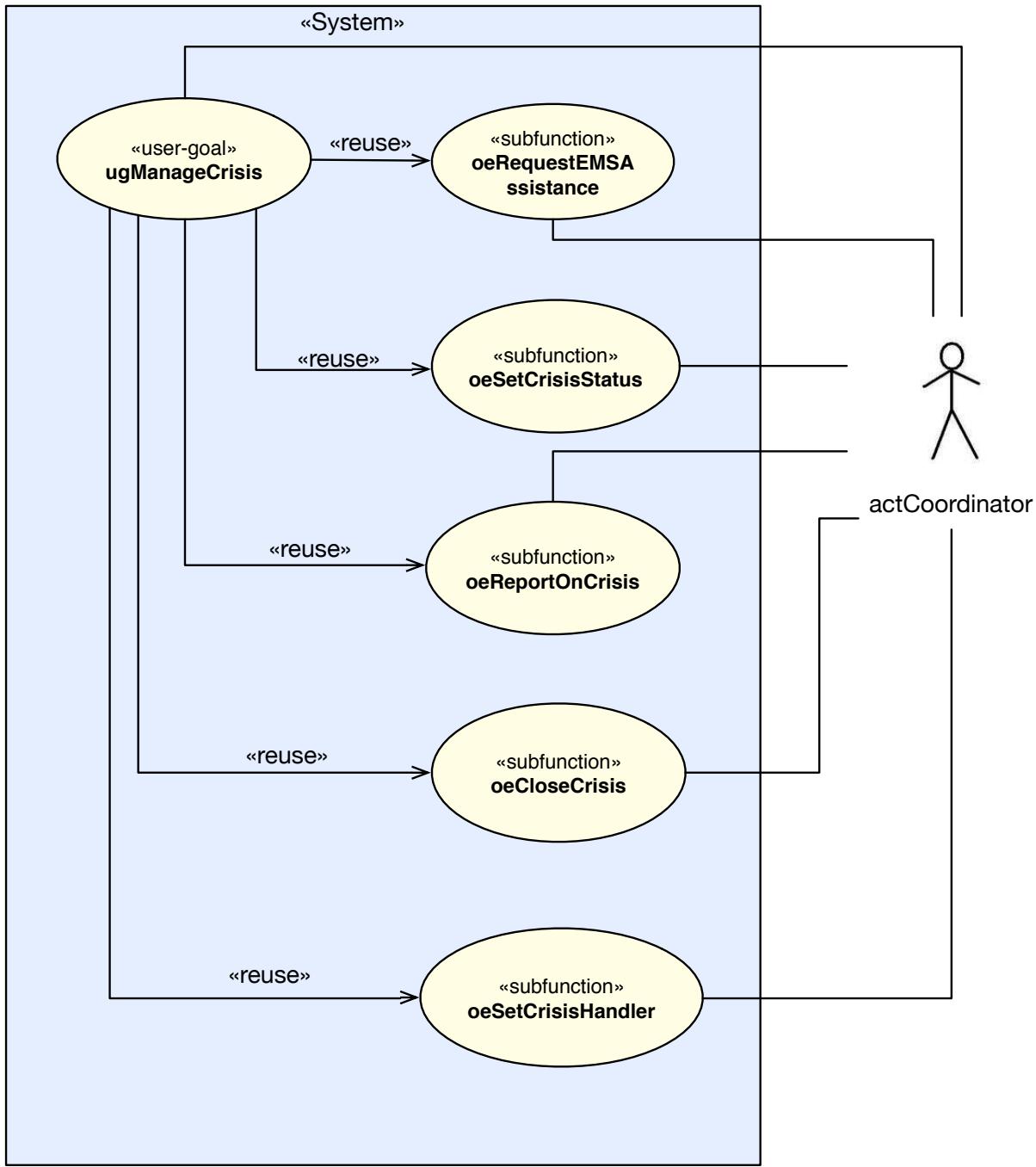


Fig. A.5 *iCrash* Use Case Diagram: UgManageCrisis

USE-CASE DESCRIPTION	
<i>Name</i>	ugMonitor
<i>Scope</i>	System
<i>Altitude</i>	Summary
<i>Parameters</i>	
1	none
<i>Primary actor(s)</i>	
1	actCoordinator [active]
2	actDomainExpert [active]
<i>Secondary actor(s)</i>	
1	None.
<i>Goal(s) description</i>	
The goal is to use all operations available to the coordinator for the successful handling of a crisis or an alert received by our system.	
<i>Reuse</i>	
1	<u>oeGetAlertSet</u>
2	<u>oeValidateAlert</u>
3	<u>oeSollicitateCrisisHandling</u>
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been started
2	the actCoordinator has been added to the system.
3	the actCoordinator has safely logged on to the system.
4	the actDomainExpert has safely logged on to system.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	the system is being monitored by the actCoordinator and actDomainExpert.
<i>Main success steps</i>	
a	the actor actDomainExpert executes the <u>oeValidateAlert</u> use case.
b	the actor actCoordinator executes the <u>oeGetAlertSet</u> use case.
c	the actor actDomainExpert executes the <u>oeSollicitateCrisisHandling</u> use case.
<i>Step Constraints Ordering and Extensions</i>	
1	Step a) must be executed first.
2	Step b) can be executed as a reaction to c) or to monitor the crisis by itself.
3	Steps a),b) and c) can be executed multiple times in order to monitor a crisis.

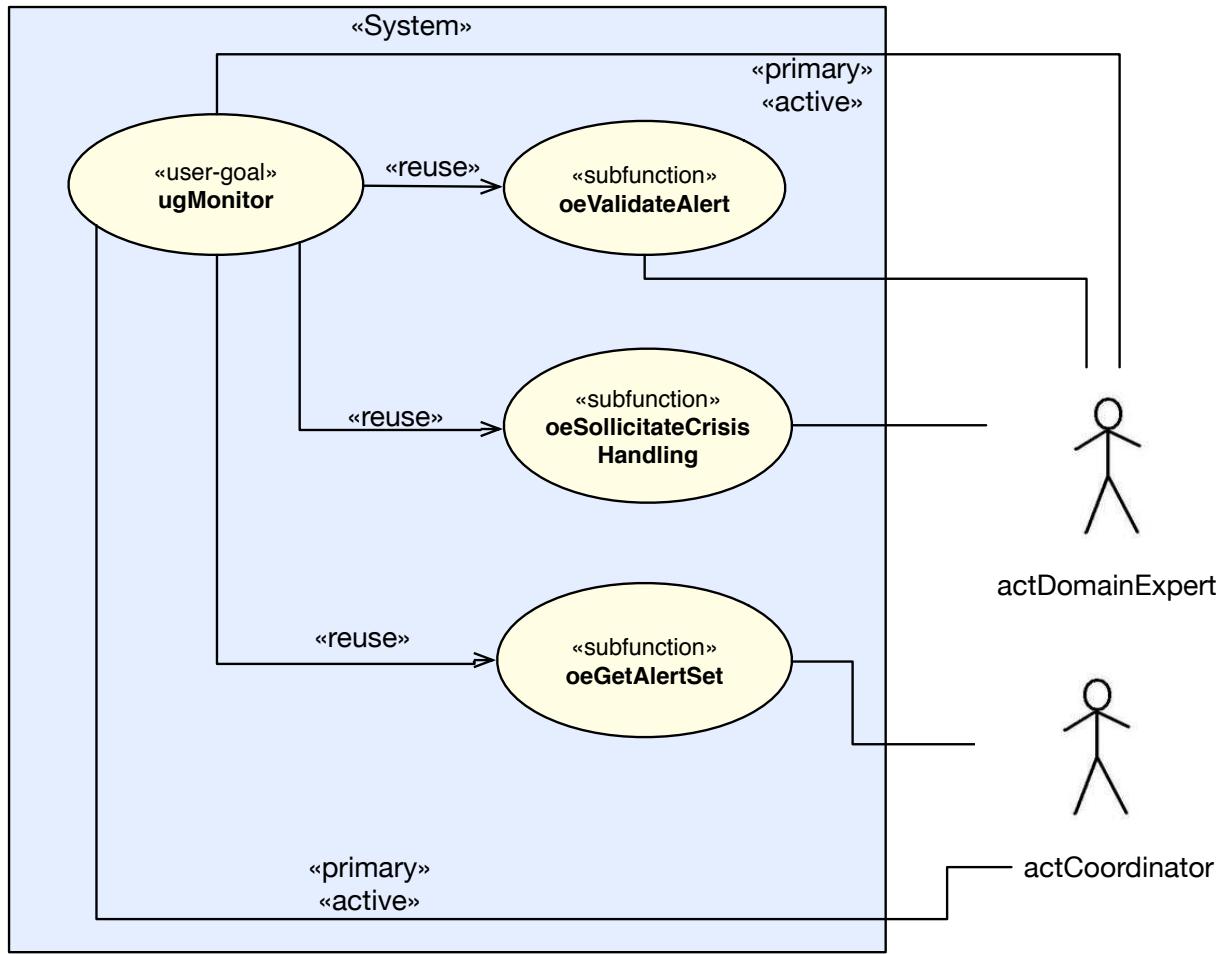


Fig. A.6 *iCrash* Use Case Diagram: UgMonitor

A.2.3 Subfunction

USE-CASE DESCRIPTION	
<i>Name</i>	oeSollicitateCrisisHandling
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	none
Primary actor(s)	
1	actDomainExpert [active]
Secondary actor(s)	
1	actAdministrator [passive]
2	actCoordinator [passive, multiple]
Goal(s) description	
the actDomainExpert's goal is to point out that there are crises still not being handled by any coordinator. And he wants to change that.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	there are some crisis still pending and for which demand has been sent to the administrator and the coordinators for a certain amount of time.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	a simple text message ieMessage('Some alert have been untreated for more than the 10 min. Please REACT !') is sent to the system administrator and to all the coordinators of the environment for each alert that is unhandled and for which no demand has been sent to the administrator and the coordinators for more than a certain amount of time.
2	the reminder period for the concerned crisis is initialized.
Main success steps	
a	the actor actDomainExpert sends the message <u>oeSollicitateCrisisHandling()</u> to the system.
Step Constraints Ordering and Extensions	
1	none.
Additional Information	
none	

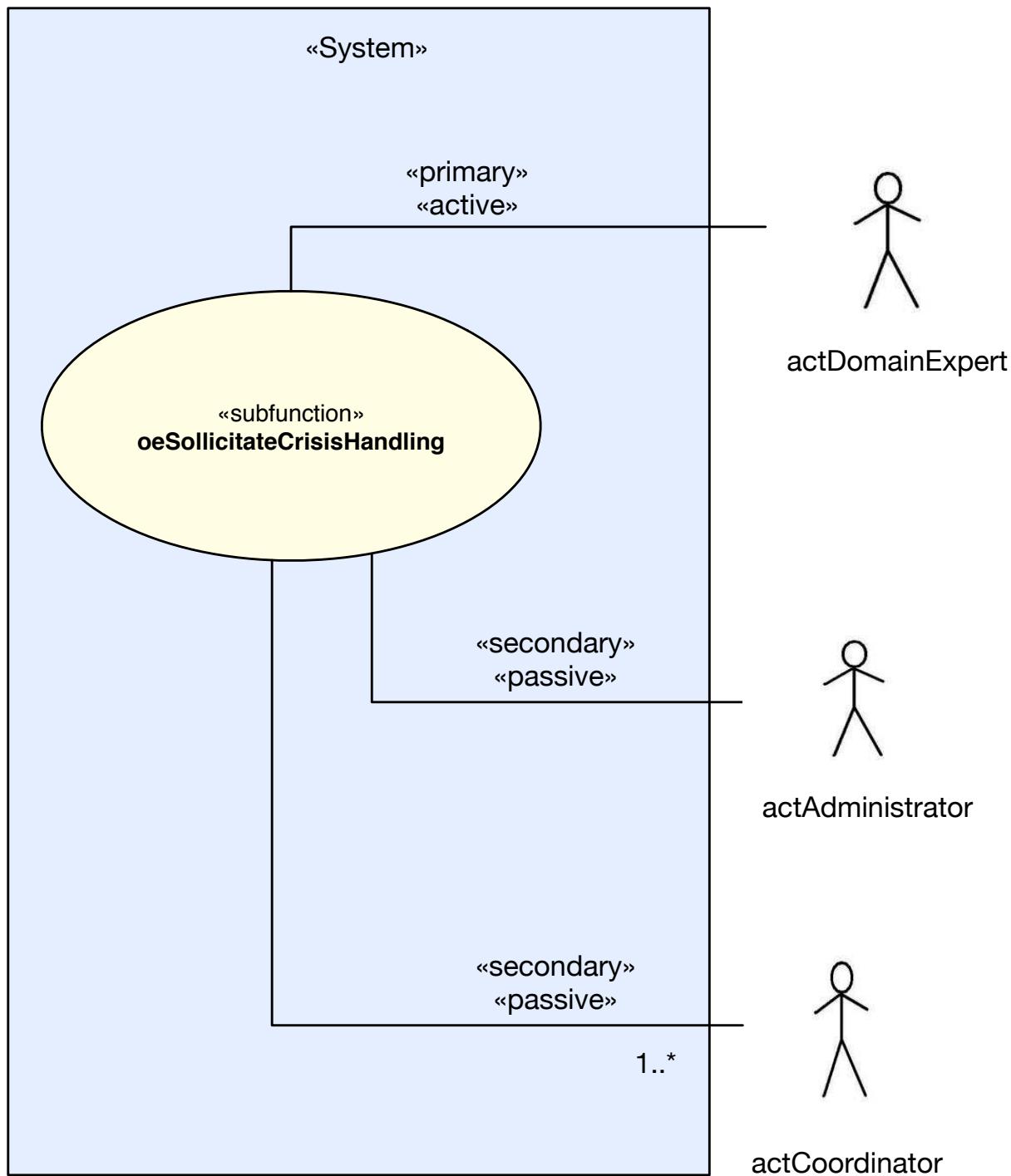


Fig. A.7 *iCrash* Use Case Diagram: **oeSollicitateCrisisHandling**

USE-CASE DESCRIPTION	
<i>Name</i>	oeAddUser
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	UserID:dtUserID - the user's identification.
2	Login:dtLogin - the username of the user.
3	Password:dtPassword - the password of the user.
4	TokenID:dtTokenID - the a unique identifier for the users token.
5	UserType:etUserType - the Type of user [Coordinator, DomainExpert or EMS]
6	CoordinatorDomain:ctDomain - the domains of expertise of a coordinator.
Primary actor(s)	
1	actAdministrator[active]
Secondary actor(s)	
1	actCoordinator[passive]
2	actDomainExpert[passive]
3	actEMS[passive]
Goal(s) description	
the actAdministrator's goal is to delete new user to the system so he/she can safely use the system.	
Reuse	
1	none
Protocol condition(s)	
1	the iCrash system has been deployed.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	A new user has been added to the system.
2	The new user can use the system safely.
Main success steps	
a	the actor actAdministrator sends the message oeAddUser (CoordinatorID, Login, Password, TokenID, UserType) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
When creating Coordinator you must specify his domains of expertise. Only users of the type Coordinator need to have specific Domains.	

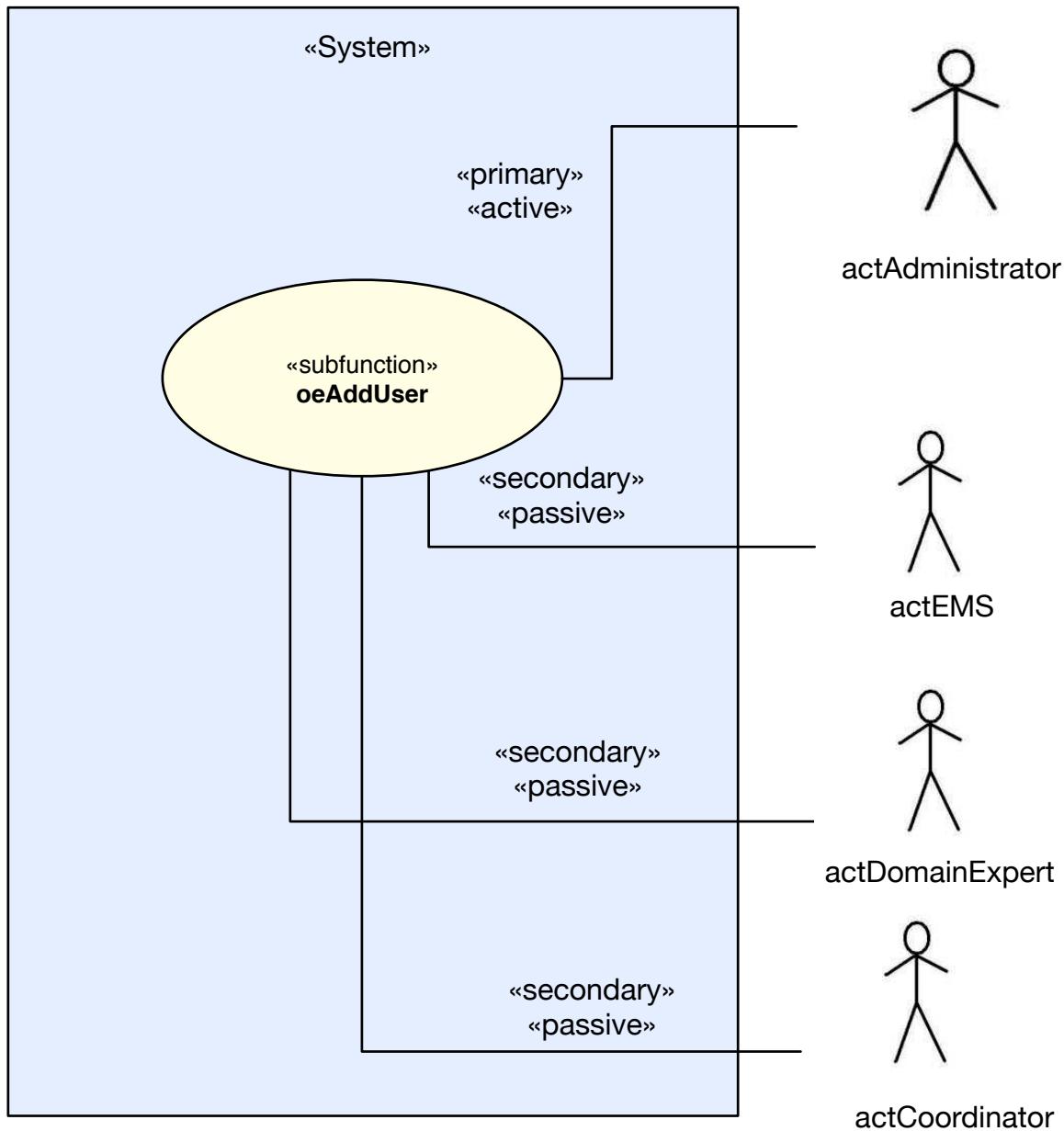


Fig. A.8 *iCrash* Use Case Diagram: `oeAddUser`

USE-CASE DESCRIPTION	
<i>Name</i>	oeDeleteUser
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	UserID:dtUserID - the user's identification
Primary actor(s)	
1	actAdministrator[active]
Secondary actor(s)	
1	actCoordinator[passive]
2	actDomainExpert[passive]
3	actEMS[passive]
Goal(s) description	
the actAdministrator's goal is to remove a user from the system.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	the user with the specified user ID already exists in the system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	A user has been removed from the system.
2	The specified user can no longer log on to the system.
Main success steps	
a	the actor actAdministrator sends the message oeDeleteUser(UserID) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	

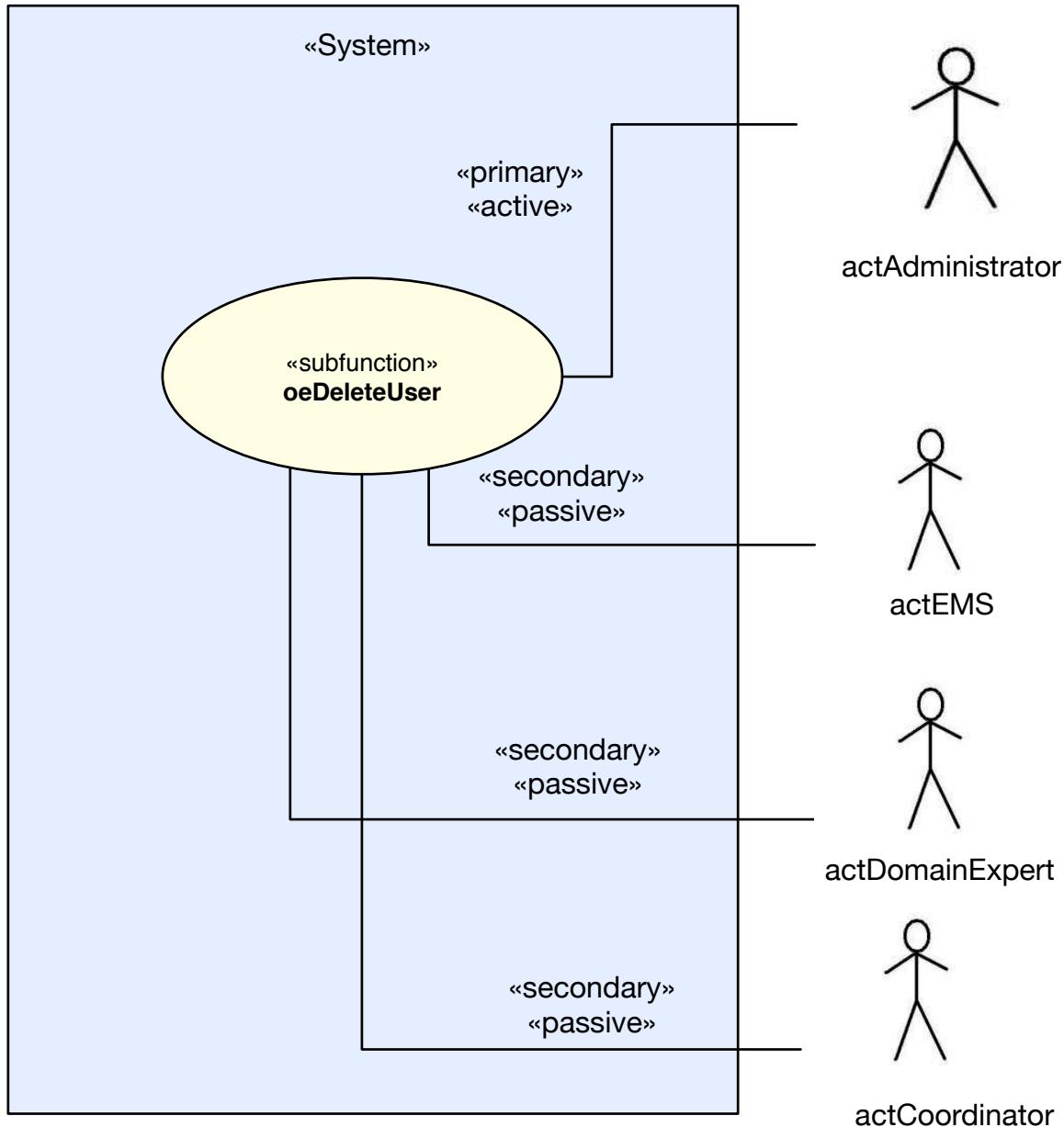


Fig. A.9 *iCrash* Use Case Diagram: `oeDeleteUser`

USE-CASE DESCRIPTION	
<i>Name</i>	oeRequestEMSService
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	Message: dtComment - A message from the actCoordinator to the actEMS
2	UniqueIDoftheReq: dtRequestID - a request identification enabling actCoordinators and actEMSS to differentiate requests.
3	NumberofVehiclesin accident: dtNbrofVehiclesInAccident - the number of vehicles involved in the accident.
4	NumberofVictimsinaccident: dtNbrofVictims - the number of victims involved in the accident.
5	TypeOfreqEMSService: ctEMSType - the type of EMS unit requested.(RequestFireFighter, RequestPolice and RequestAmbulance) You can either request one of each or just two or all.
Primary actor(s)	
1	Coordinator [active]
Secondary actor(s)	
1	None.
Goal(s) description	
the actCoordinator's goal is to request a EMS unit to provide assistance to the victims of a car accident.	
Reuse	
1	none
Protocol condition(s)	
1	the iCrash system has been deployed.
2	an Alert has been received by our system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	A message is sent to the actEMS requesting assistance.
Main success steps	
a	the actor actCoordinator sends the message oeRequestEMSService (dtComment AdtComment, dtRequestID AdtRequest, dtGPSLocation AdtGPSLocation, dtNbrofVehiclesInAccident AdtVehiclesInAccident, dtNbrOfVictims AdtNbrVictims, ctEMSType ActEMSType) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
The coordinator decided ultimately what information is put into the message to the EMS actor the shown data types are only meant as an example of what can be added to such a message.	

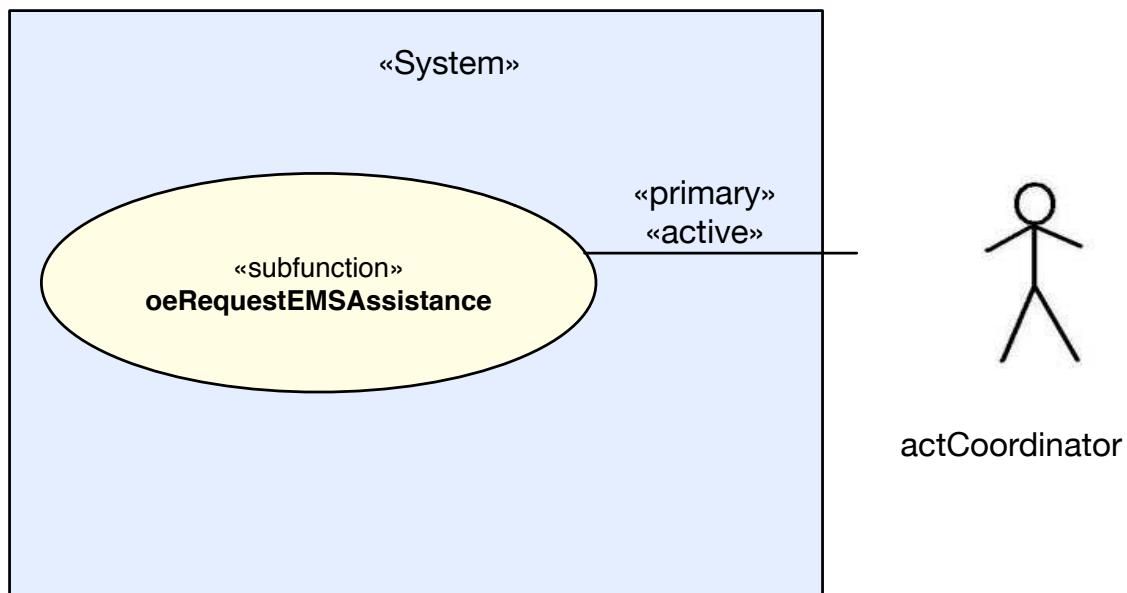


Fig. A.10 *iCrash* Use Case Diagram: oeRequestEMSAssistance

USE-CASE DESCRIPTION	
<i>Name</i>	oeSollicitateCrisisHandling
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	none
Primary actor(s)	
1	actDomainExpert [active]
Secondary actor(s)	
1	actAdministrator [passive]
2	actCoordinator [passive, multiple]
Goal(s) description	
the actDomainExpert's goal is to point out that there are crises still not being handled by any coordinator. And he wants to change that.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	there are some crisis still pending and for which demand has been sent to the administrator and the coordinators for a certain amount of time.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	a simple text message ieMessage('Some alert have been untreated for more than the 10 min. Please REACT !') is sent to the system administrator and to all the coordinators of the environment for each alert that is unhandled and for which no demand has been sent to the administrator and the coordinators for more than a certain amount of time.
2	the reminder period for the concerned crisis is initialized.
Main success steps	
a	the actor actDomainExpert sends the message <u>oeSollicitateCrisisHandling()</u> to the system.
Step Constraints Ordering and Extensions	
1	none.
Additional Information	
none	

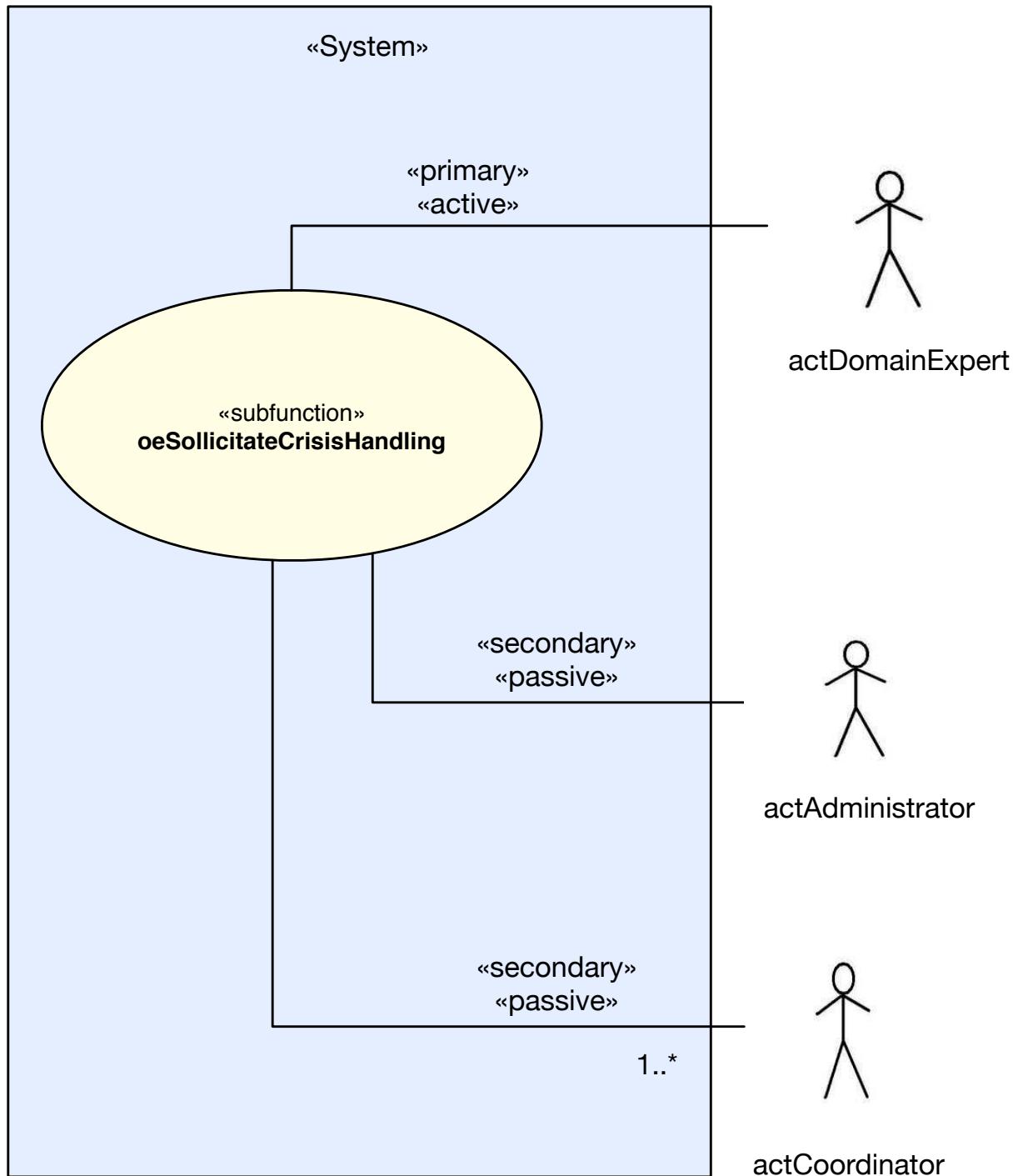


Fig. A.11 *iCrash* Use Case Diagram: oeSollicitateCrisisHandling

USE-CASE DESCRIPTION	
<i>Name</i>	oeReportOnCrisis
<i>Scope</i>	System
<i>Altitude</i>	subfunction
<i>Parameters</i>	
1	AlertID:dtAlertID - the identification of the Alert.
2	AText:dtComment - a text written by the actCoordinator to better describe the crisis, in his own words, in the report.
3	UserID:dtUserID - a unique identifier identifying the actCoordinator writing the report.
4	TypeofCrisis:etCrisisType - the type of the crisis differentiated here by (Small, Medium or Huge) thus describing the crisis type.
5	CrisisStatus:etCrisisStatus - the status of the crisis differentiated here by (pending, handled or solved).
6	RecievedMessages:dtRecievedMessage - This is a record of the alert messages received from the actor actComCompany.
7	NumberofVehiclesin accident:dtNbrofVehiclesInAccident - the number of vehicles involved in the accident.
8	NumberofVictimsinaccident:dtNbrofVictims - the number of victims involved in the accident.
<i>Primary actor(s)</i>	
1	Coordinator [active]
<i>Secondary actor(s)</i>	
1	ComCompany [passive]
<i>Goal(s) description</i>	
the actCoordinator's goal is to write a report about the current ongoing crisis or to write a final report of the crisis.	
<i>Reuse</i>	
1	none
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been deployed.
2	an Alert has been received by our system.
3	the actDomainExpert has validated the alert.
4	the actCoordinator actor has the right domain to access the Crisis and is thus allowed to write a report about it.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	A report of a certain crisis exists in the system.
<i>Main success steps</i>	
a	the actor actCoordinator sends the messageoeReportOnCrisis(dtAlertID AdtCrisisID, dtComment AdtComment, dtUserID AdtUserID, etCrisisType AdtCrisisType, etCrisisStatus AdtCrisisStatus, dtRecievedMessage AdtReceivedMessage, dtNbrofVehiclesInAccident AdtVehiclesInAccident, dtNbrOfVictims AdtNbrVictims) to the system.
<i>Step Constraints Ordering and Extensions</i>	
1	none
<i>Additional Information</i>	
The coordinator decided ultimately what information is put into the report the shown data types are only meant as an example of what can be added to such a report.	

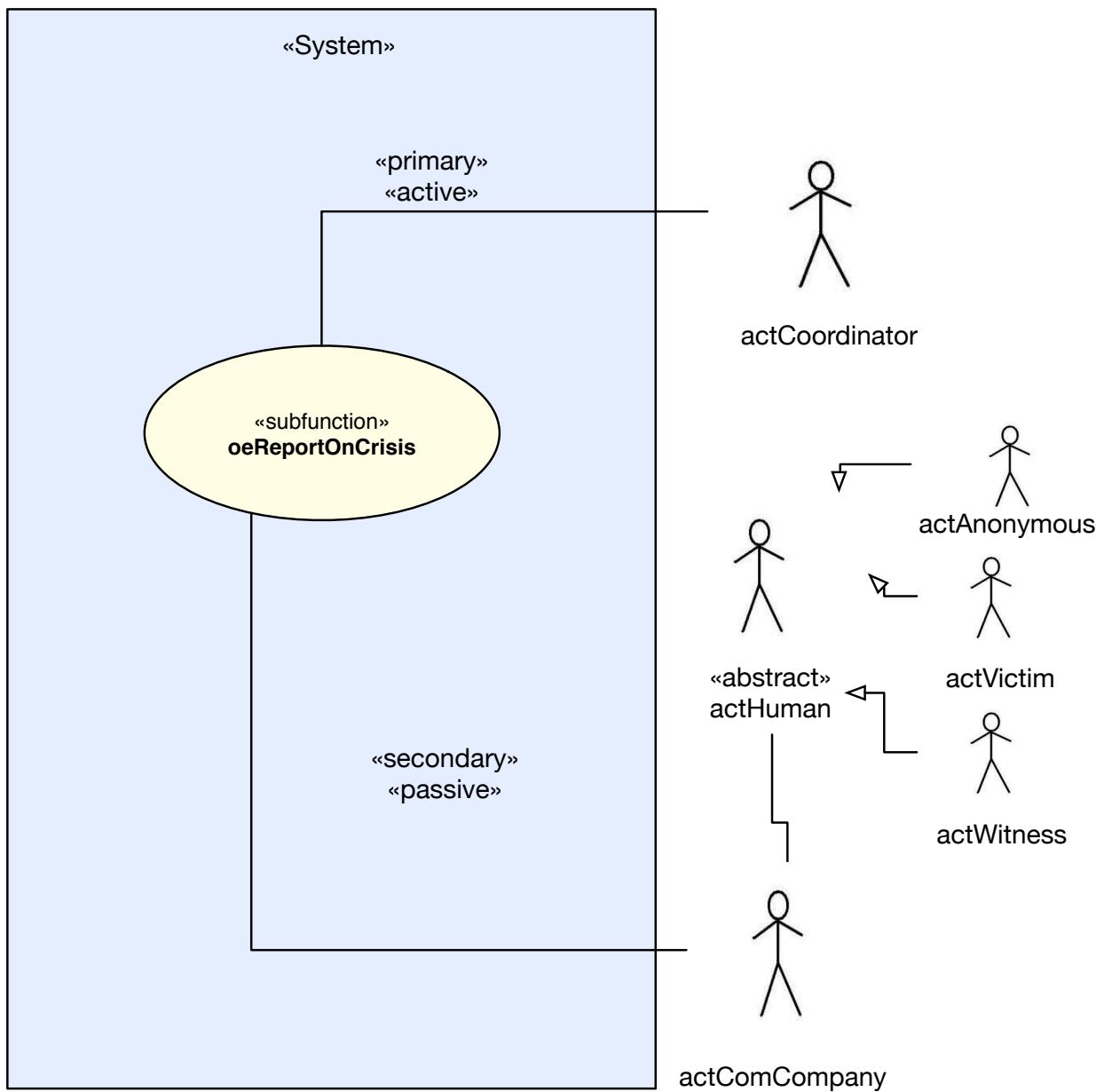


Fig. A.12 *iCrash* Use Case Diagram: **oeReportOnCrisis**

USE-CASE DESCRIPTION	
<i>Name</i>	oeInfoFam
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	Message:dtComment - the message that will be sent to the family members.
2	VictimDescription:ctVictim - included in this data are all the information about the victim. This includes the victims Name, phone number as well as the ICE contacts phone number and name.
3	NameofHospital:dtHospitalName - the name of the hospital that the victim was brought to.
Primary actor(s)	
1	ComCompany [active]
Secondary actor(s)	
1	actCoordinator
Goal(s) description	
the actComcompany's goal is send a message edited by the actCoordinator.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	the actEMS reports that a victim was delivered to the hospital.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	A message is sent to a victims ICE contact.
Main success steps	
a	the actor actCoordinator sends the message oeInfoFam(dtComment AdtComment, ctVictim ActVictim dtHospitalName AdtHospitalName) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
The coordinator ultimately decides what information is put into the message to the family members the shown data types are only meant as an example of what can be added to such a message.	

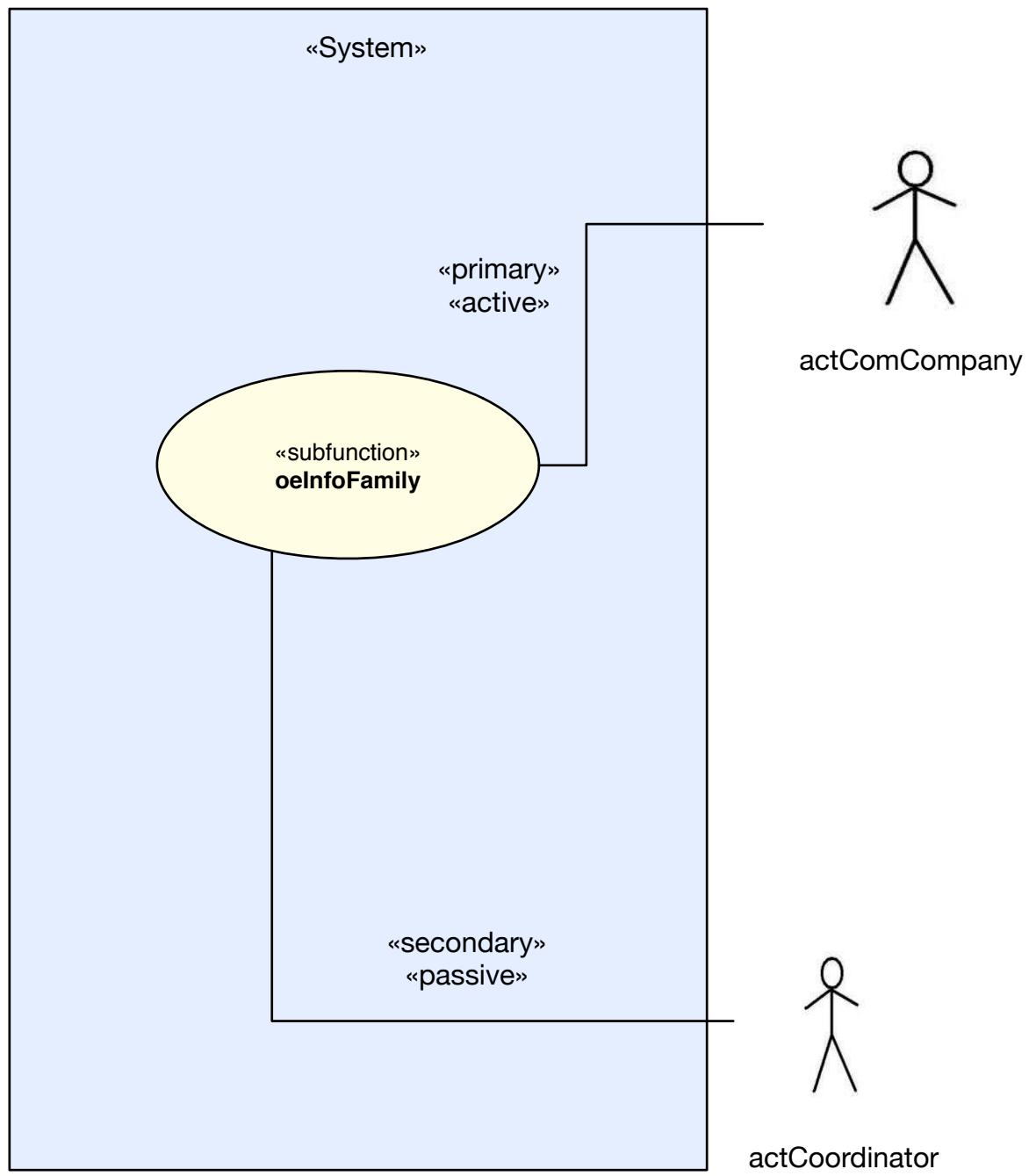


Fig. A.13 *iCrash* Use Case Diagram: oeInfoFamily

USE-CASE DESCRIPTION	
<i>Name</i>	oeCloseCrisis
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	crisisID:dtCrisisID - the crisis identification
Primary actor(s)	
1	actCoordinator[active]
Secondary actor(s)	
1	actComCompany[passive]
Goal(s) description	
the actCoordinator's goal is to declare a crisis as closed.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	a crisis with the given ID exists in the system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the crisis is closed.
2	a message ieSmsSend(AdtPhoneNumber, AdtComment) is sent to the actComCompany actor for each registered alert declared by a SMS coming from phone number AdtPhoneNumber and related to the crisis crisisID. All the messages have the same textual message AdtComment giving them the minimal information on the crisis they alerted.
Main success steps	
a	the actor actCoordinator sends the message oeCloseCrisis(crisisID) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
none	

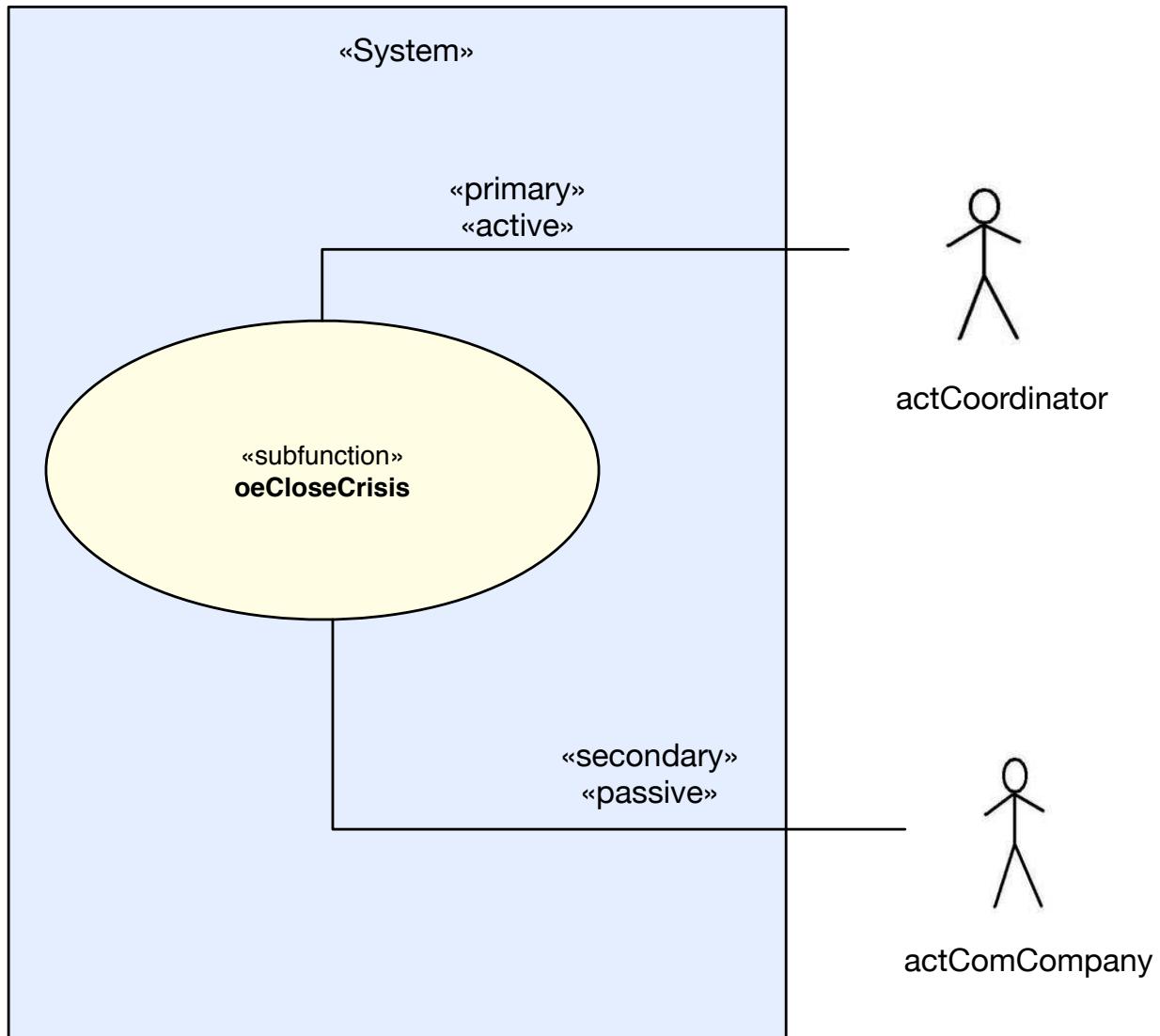


Fig. A.14 *iCrash* Use Case Diagram: oeCloseCrisis

USE-CASE DESCRIPTION	
<i>Name</i>	oeAlert
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	HumanKind:etHumanKind - the type of individual trying to use the system.(victim, witness or anonymous)
2	Date:dtDate - the date the Alert was sent.
3	Time:dtTime - the time the Alert was sent.
4	PhoneNumber:dtPhoneNumber - the phone number from which the Alert was sent.
5	GPSLocation:dtGPSLocation - the GPS location the Alert originated from.
6	Comment:dtComment - a text meant for the human to describe his situation.
Primary actor(s)	
1	actComCompany[active]
Secondary actor(s)	
1	actDomainExpert [passive]
Goal(s) description	
the actComCompany's goal is to send an Alert from a human to the system.	
Reuse	
1	none
Protocol condition(s)	
1	the iCrash system has been deployed.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	An Alert is received by the system.
Main success steps	
a	the actor actComCompany sends the message oeAlert (etHumanKind, dtDate, dtTime, dtPhoneNumber) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	

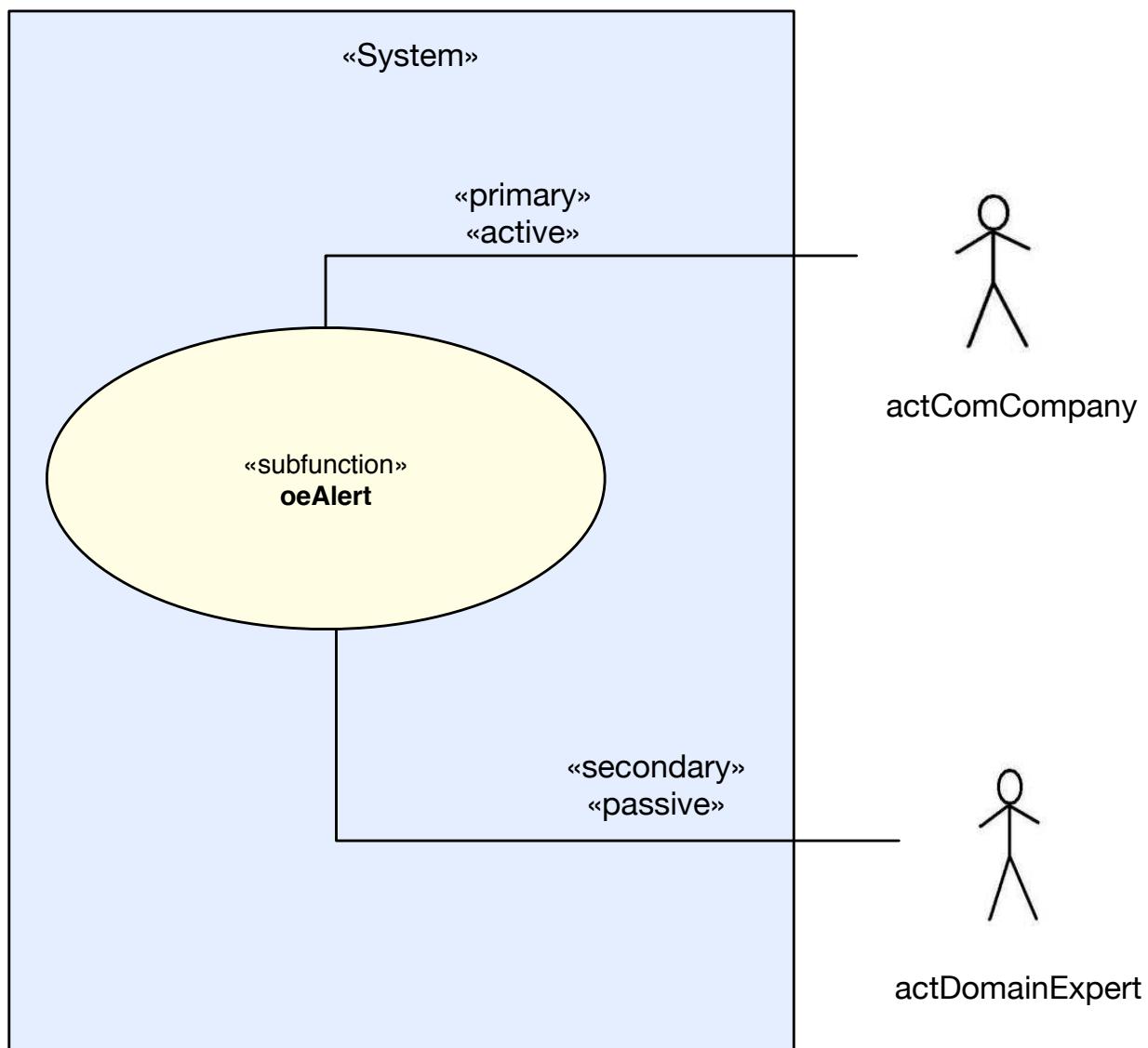


Fig. A.15 *iCrash* Use Case Diagram: oeAlert

USE-CASE DESCRIPTION	
<i>Name</i>	oeGetAlertSet
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	AlertStatus:etAlertStatus - the Alert status.
Primary actor(s)	
1	actCoordinator [active]
Secondary actor(s)	
1	actDomainExpert [passive]
Goal(s) description	
the actCoordinator's goal is to get a list/set of all the Alerts of a certain alert-status.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	the actCoordinator has logged on to the system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the actCoordinator is now aware of all the Alerts of a specified status.
Main success steps	
a	the actor actCoordinator sends the message oeCloseCrisis(crisisID) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
none	

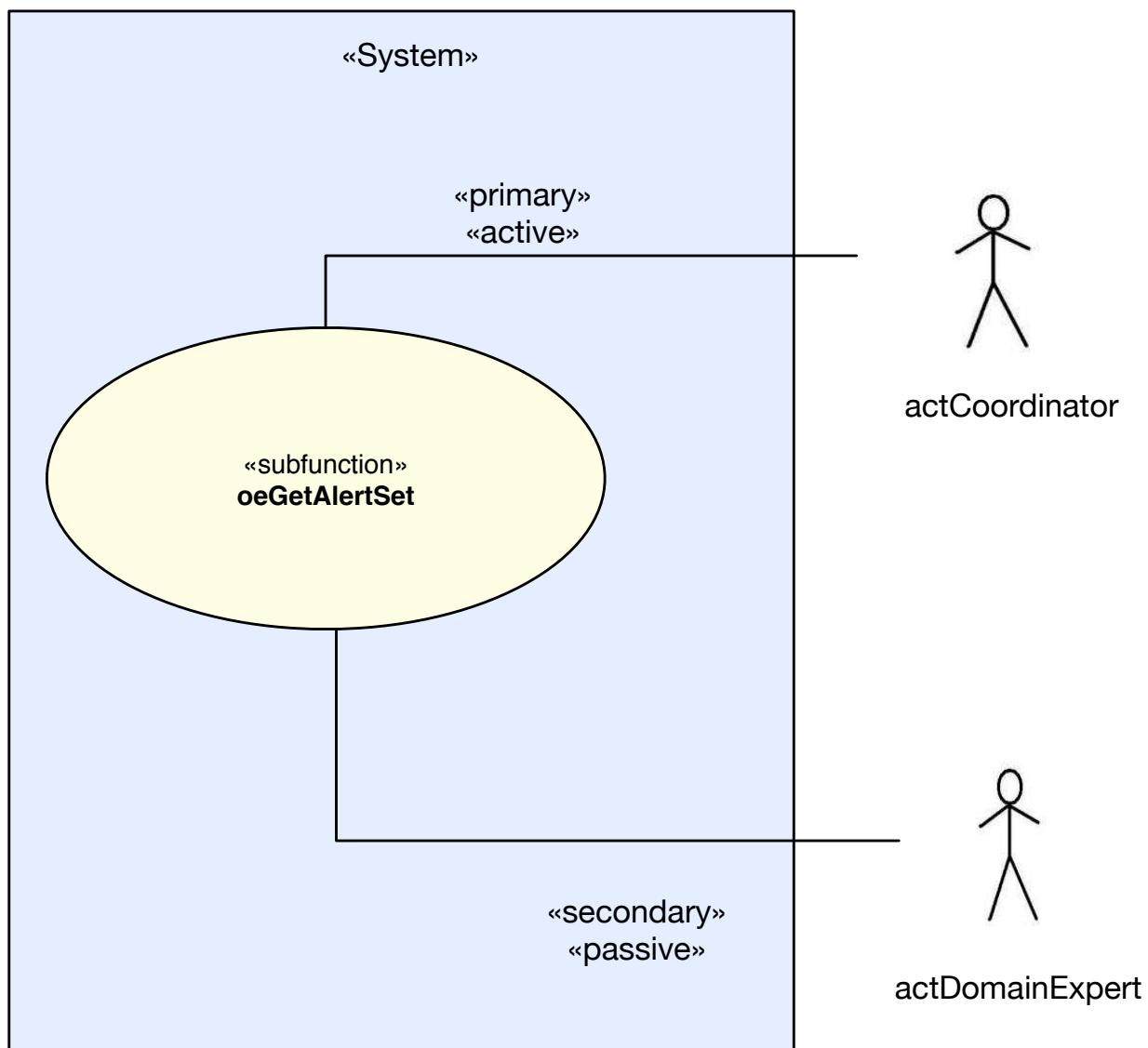


Fig. A.16 *iCrash* Use Case Diagram: **oeGetAlertSet**

USE-CASE DESCRIPTION	
<i>Name</i>	oeSetCrisisHandler
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	AlertID:dtAlertID - the Alert's unique identification.
2	Crisisdata:ctCrisis - the data required to initiate a crisis(CrisisType, CrisisStatus, GPSLocation, Date and Time).
Primary actor(s)	
1	actCoordinator[active]
Secondary actor(s)	
1	None.
Goal(s) description	
the actCoordinator's goal is to indicate to the system that he will take care of a certain Alert and thus create a crisis in the system.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	the actCoordinator has logged on to the system.
3	the actCoordinator has to have the right domains of expertise to be allowed access to the crisis.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	A crisis has been created for a certain Alert.
2	the actCoordinator has indicated to the system that he will now take care of the Alert specified.
Main success steps	
a	the actor actCoordinator sends the message <u>oeSetCrisisHandler(AlertID, ctCrisis)</u> to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
none	

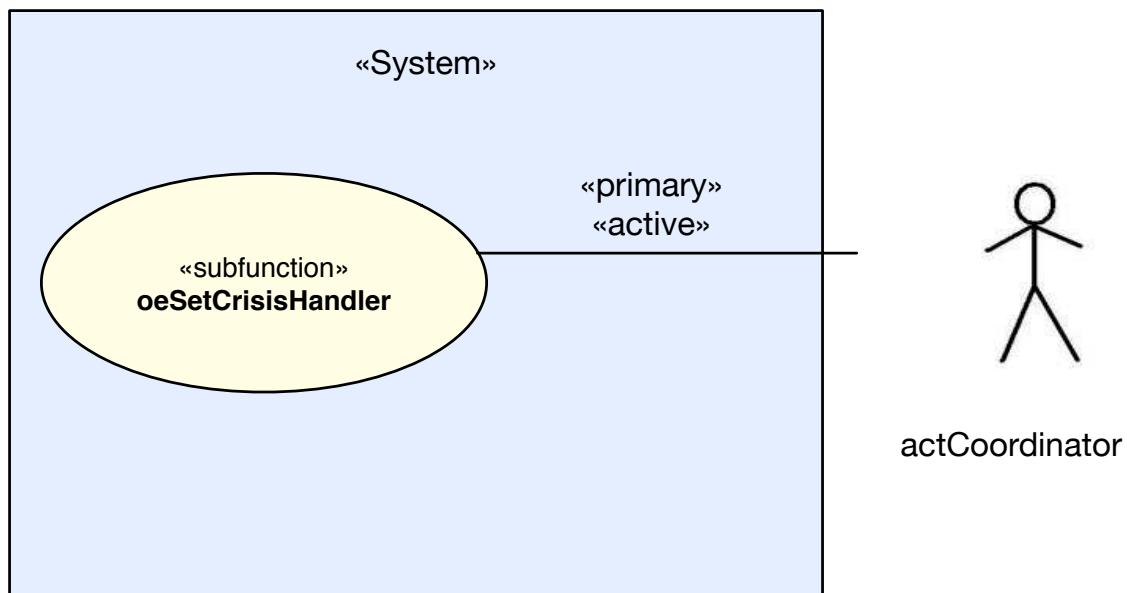


Fig. A.17 *iCrash* Use Case Diagram: oeSetCrisisHandler

USE-CASE DESCRIPTION	
<i>Name</i>	oeLogin
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	Username:dtLogin - the user's login/username.
2	Password:dtPassword - the user's password.
3	SerialKey:dtKey - the user's serial key generated by his token device.
Primary actor(s)	
1	actAuthenticated[active]
Secondary actor(s)	
1	actAdministrator[passive]
2	actCoordinator[passive]
3	actDomainExpert[passive]
4	actEMS[passive]
Goal(s) description	
the actAuthenticated's to log on to the system in order to securely use it.	
Reuse	
1	none
Protocol condition(s)	
1	the iCrash system has been deployed.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the actAuthenticated has securely logged on to the system.
Main success steps	
a	the actor actAuthenticatedActor sends the message oeLogin(dtLogin, dtPassword, dtKey) to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	

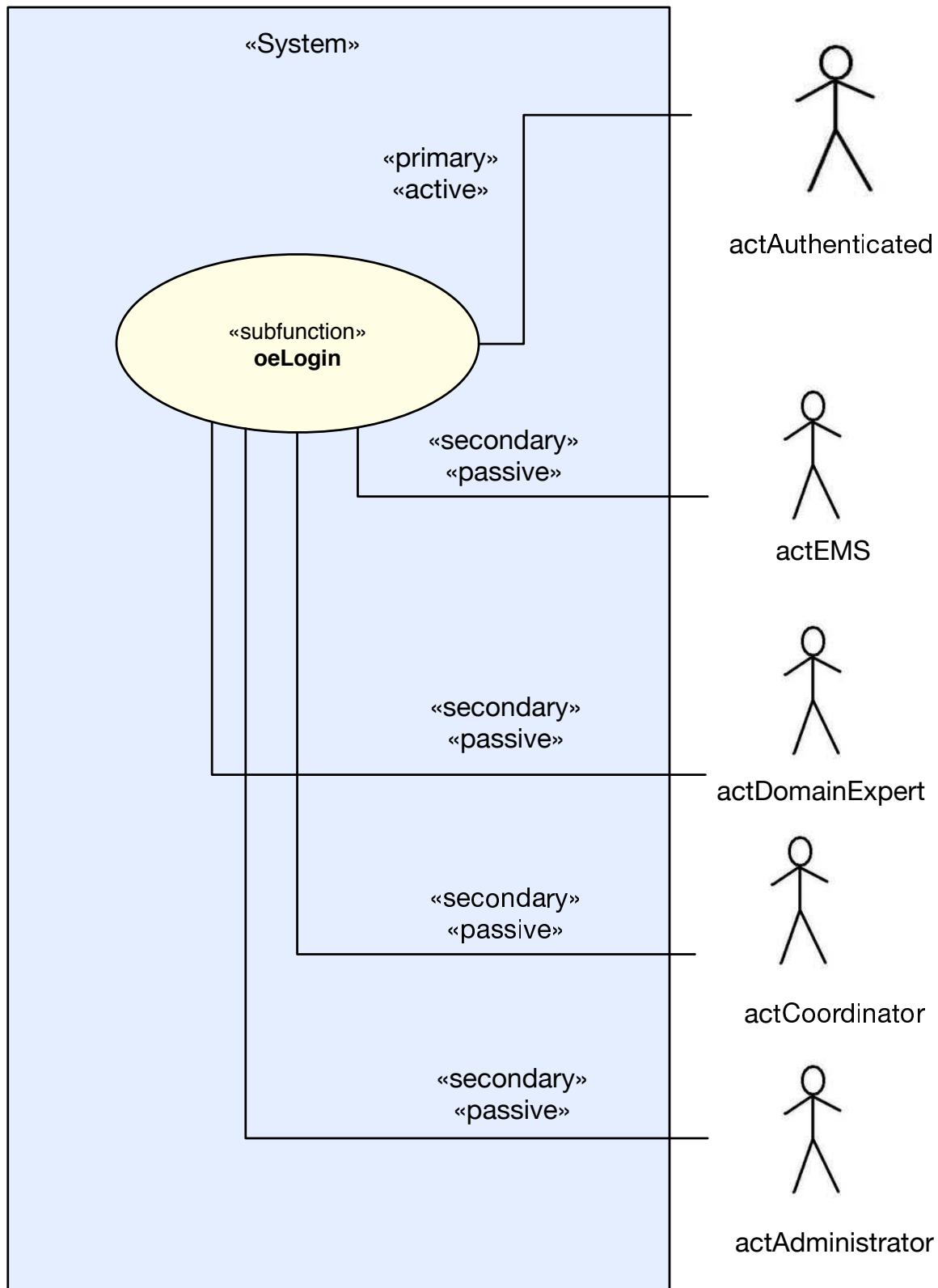


Fig. A.18 *iCrash* Use Case Diagram: oeLogin

USE-CASE DESCRIPTION	
<i>Name</i>	oeLogout
<i>Scope</i>	System
<i>Altitude</i>	subfunction
<i>Parameters</i>	
1	None.
<i>Primary actor(s)</i>	
1	actAuthenticated[active]
<i>Secondary actor(s)</i>	
1	actAdministrator[passive]
2	actCoordinator[passive]
3	actDomainExpert[passive]
4	actEMS[passive]
<i>Goal(s) description</i>	
the actAuthenticated's to logoff from the system.	
<i>Reuse</i>	
1	none
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been deployed.
2	the actAuthenticated performing the operation is logged in.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	the actAuthenticated has securely logged off from the system.
<i>Main success steps</i>	
a	the actor actAuthenticatedActor sends the message oeLogoff() to the system.
<i>Step Constraints Ordering and Extensions</i>	
1	none
<i>Additional Information</i>	

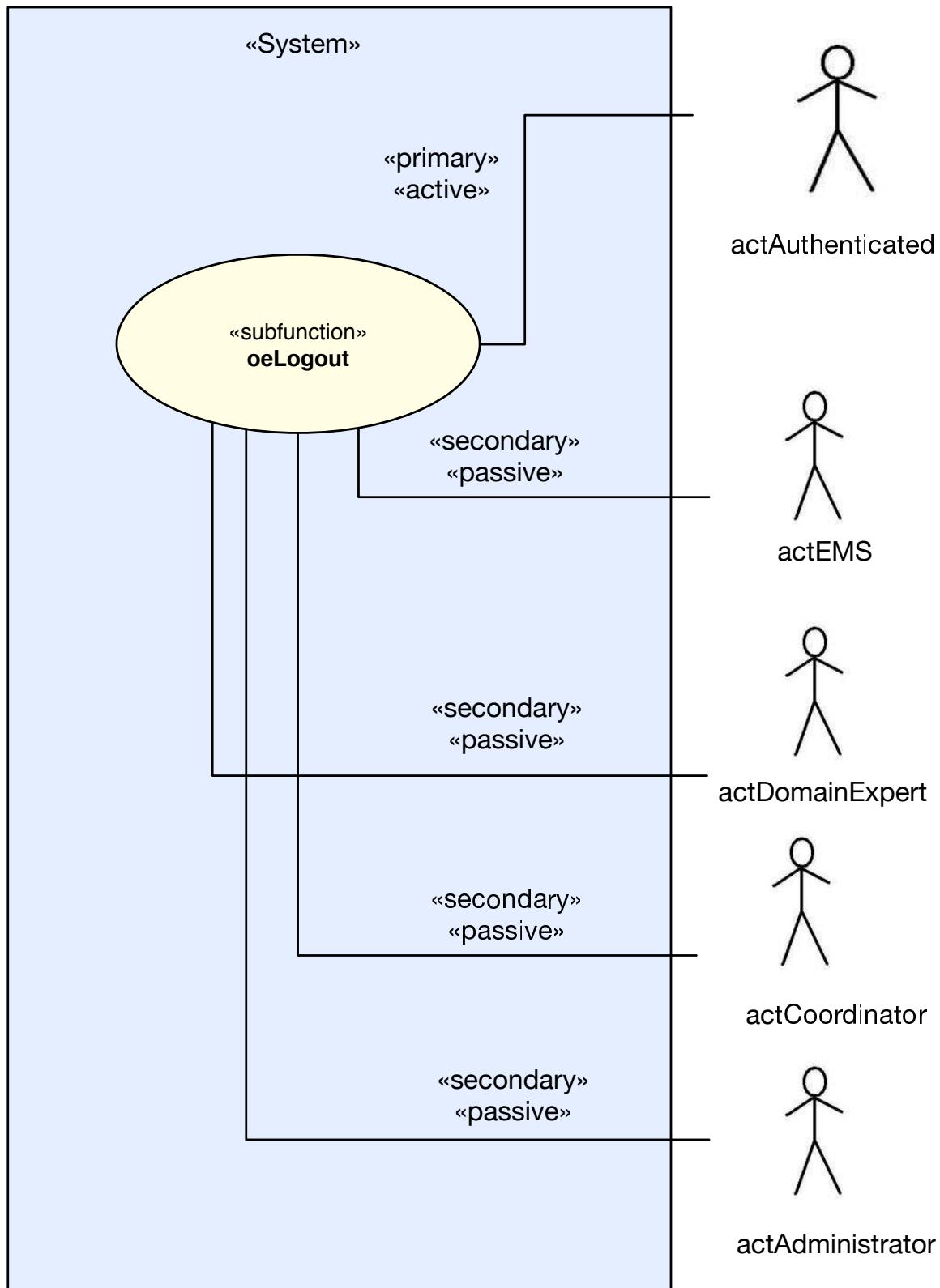


Fig. A.19 *iCrash* Use Case Diagram: oeLogout

USE-CASE DESCRIPTION	
<i>Name</i>	oeReportEMSCrisisStatus
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	crisisID:dtCrisisID - the crisis identification.
2	Comment:dtComment - a comment field for the EMS actor to describe the situation.
3	EMSCrisisStatus:etEMSCrisisStatus - the status of the situation(here crisis) as seen from their perspective.
4	HospitalName:dtHospitalName - the name of the hospital.
5	Victim information:ctVictim - information about the victim.(VictimName,VictimPhonnumber,HumanKind, VictimICEContactName and VictimICEPhoneNumber)
Primary actor(s)	
1	actEMS[active]
Secondary actor(s)	
1	actCoordinator[passive]
Goal(s) description	
the actEMS's goal is to describe the status of the situation they were asked to provide assistance to.	
Reuse	
1	none
Protocol condition(s)	
1	the iCrash system has been deployed.
2	a crisis with the given ID exists in the system.
3	an actCoordinator has requested EMS assistance for the specified crisis.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the report of the current status of the crisis from the actEMS's point of view has been sent to the system.
Main success steps	
a	the actor actEMS sends the message <u>oeoeReportEMSCrisisStatus ((dtCrisisID, dtComment, dtEMSCrisisStatus, ctVictim))</u> to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
This is just an example of data types that might be used in a message to report the ems crisis status, ultimately the actEMS decides what he puts in to such a message.	

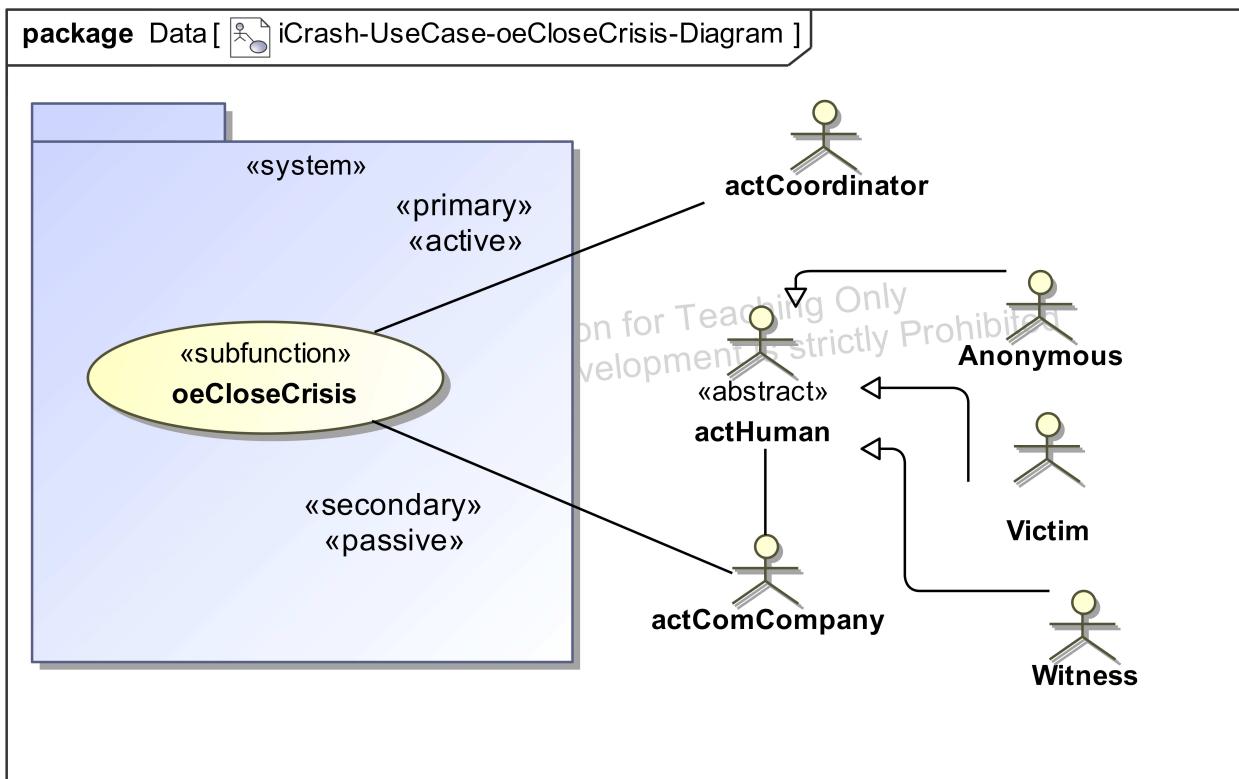


Fig. A.20 *iCrash* Use Case Diagram: oeCloseCrisis

USE-CASE DESCRIPTION	
<i>Name</i>	oeReplyToRequest
<i>Scope</i>	System
<i>Altitude</i>	subfunction
<i>Parameters</i>	
1	crisisID:dtCrisisID - the crisis identification.
2	Comment:dtComment - a comment field for the EMS actor to describe the situation.
3	RequestID:dtRequestID - a unique identifier for a request.
4	EMSCrisisStatus:etEMSCrisisStatus - the status of the situation(here crisis) as seen from their perspective.
<i>Primary actor(s)</i>	
1	actEMS[active]
<i>Secondary actor(s)</i>	
1	actCoordinator[passive]
<i>Goal(s) description</i>	
the actEMS's goal is to reply to a request previously sent by a coordinator.	
<i>Reuse</i>	
1	none
<i>Protocol condition(s)</i>	
1	the <i>iCrash</i> system has been deployed.
2	a crisis with the given ID exists in the system.
3	an actCoordinator has requested EMS assistance for the specified crisis.
<i>Pre-condition(s)</i>	
1	none
<i>Main post-condition(s)</i>	
1	the actEMS has sent a reply to a received request to the system.
<i>Main success steps</i>	
a	the actor actEMS sends the message <u>oeReplyToRequest</u> (dtCrisisID, dtComment, dtEMSCrisisStatus) to the system.
<i>Step Constraints Ordering and Extensions</i>	
1	none
<i>Additional Information</i>	
This is just an example of data types that might be used in a message to reply to a request ultimately the actEMS decides what he puts in to such a message.	

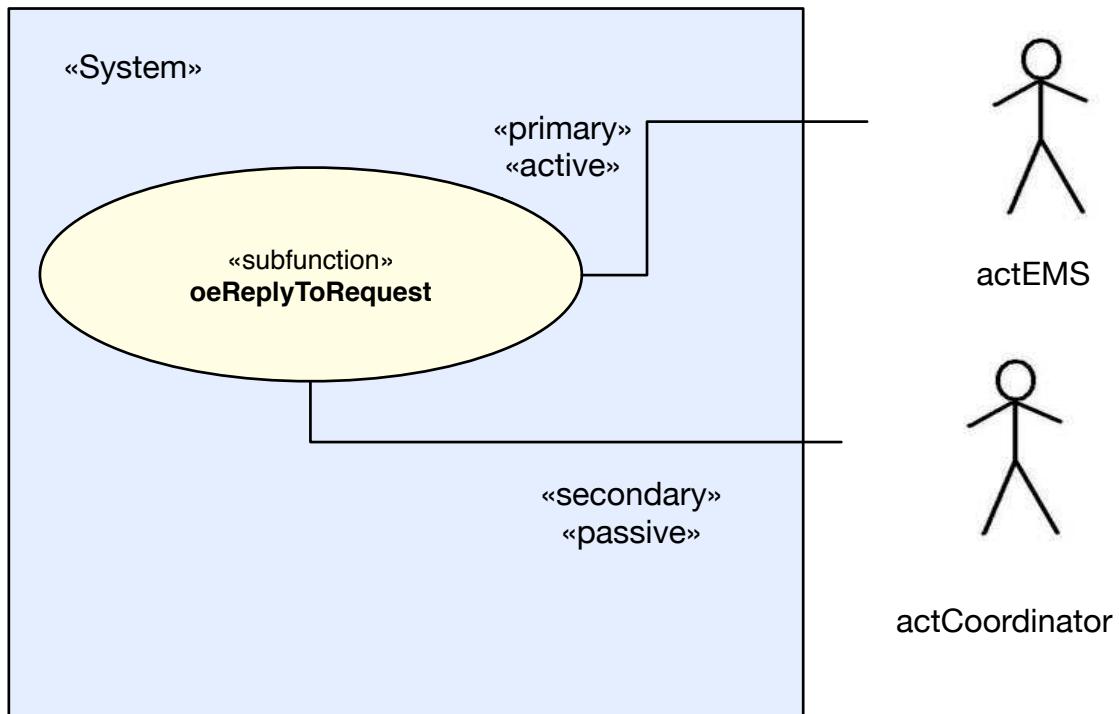


Fig. A.21 *iCrash* Use Case Diagram: oeReplyToRequest

USE-CASE DESCRIPTION	
<i>Name</i>	oeSetCrisisStatus
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	crisisID:dtCrisisID - the crisis identification.
2	CrisisStatus:etCrisisStatus - the status of the crisis.(in-handling, handled and solved)
Primary actor(s)	
1	actCoordinator[active]
Secondary actor(s)	
1	actComCompany[passive]
Goal(s) description	
the actCoordinator's goal is to change the status of a crisis.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	a crisis with the given ID exists in the system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	the status of the crisis has changed.
Main success steps	
a	the actor actCoordinator sends the message <u>oeSetCrisisStatus (dtCrisisID, etCrisisStatus)</u> to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
none	

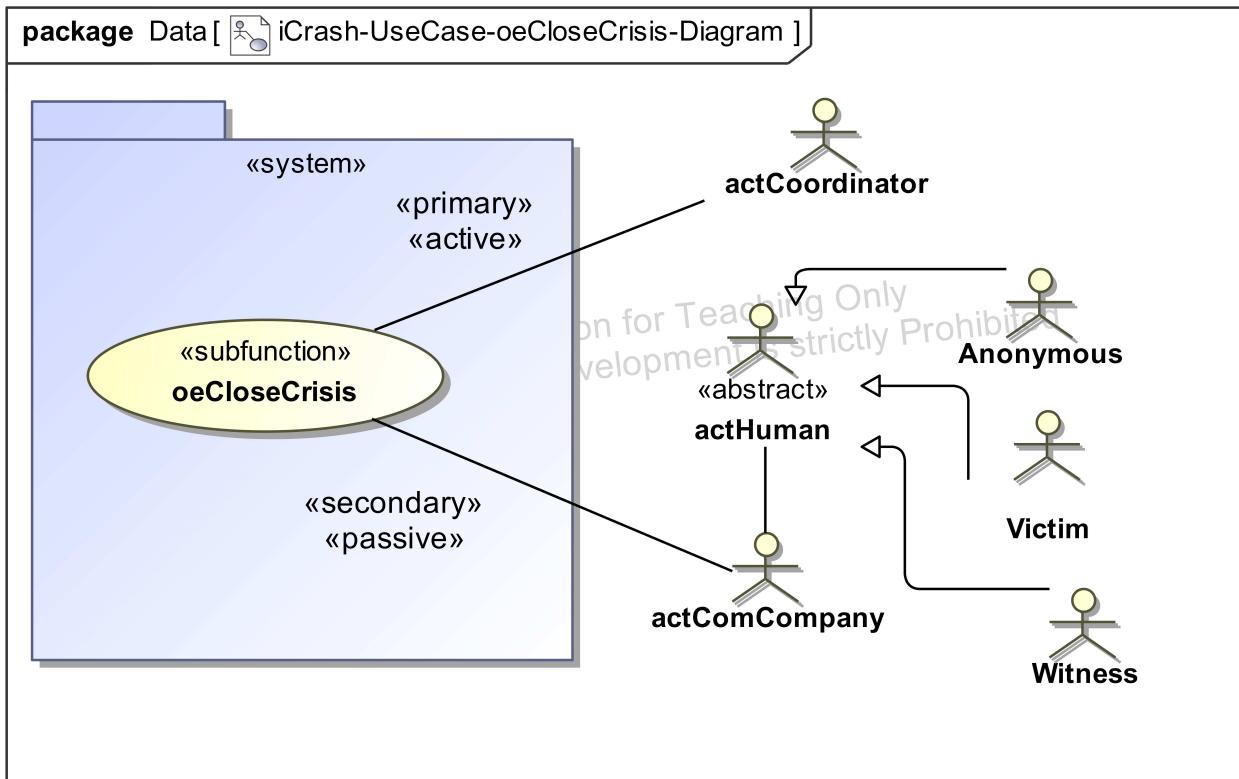


Fig. A.22 *iCrash* Use Case Diagram: `oeCloseCrisis`

USE-CASE DESCRIPTION	
<i>Name</i>	oeValidateAlert
<i>Scope</i>	System
<i>Altitude</i>	subfunction
Parameters	
1	AlertID:dtAlertID - the Alert unique identification.
2	GPSLocation:dtGPSLocation the GPS location of the Alert.
3	DateAndTime:dtDateAndTime - the date and time of when the alert was sent.
4	AlertStatus:etAlertStatus - the Alert status.
5	AlertDomains:dtDomain Domains of expertise required to handle this Alert.
Primary actor(s)	
1	actDomainExpert [active]
Secondary actor(s)	
1	None.
Goal(s) description	
the actDomainExpert's goal is to verify that an Alert is valid and to assign Domains to that Alert in order select the right Coordinator handle it.	
Reuse	
1	none
Protocol condition(s)	
1	the <i>iCrash</i> system has been deployed.
2	an Alert with the given ID exists in the system.
Pre-condition(s)	
1	none
Main post-condition(s)	
1	an Alert's status has been changed to verified and given Domains of expertise required to handle it.
Main success steps	
a	the actor actDomainExpert sends the message oeVerifyAlert(dtAlertID, dtGPSLocation, dtDat to the system.
Step Constraints Ordering and Extensions	
1	none
Additional Information	
none	

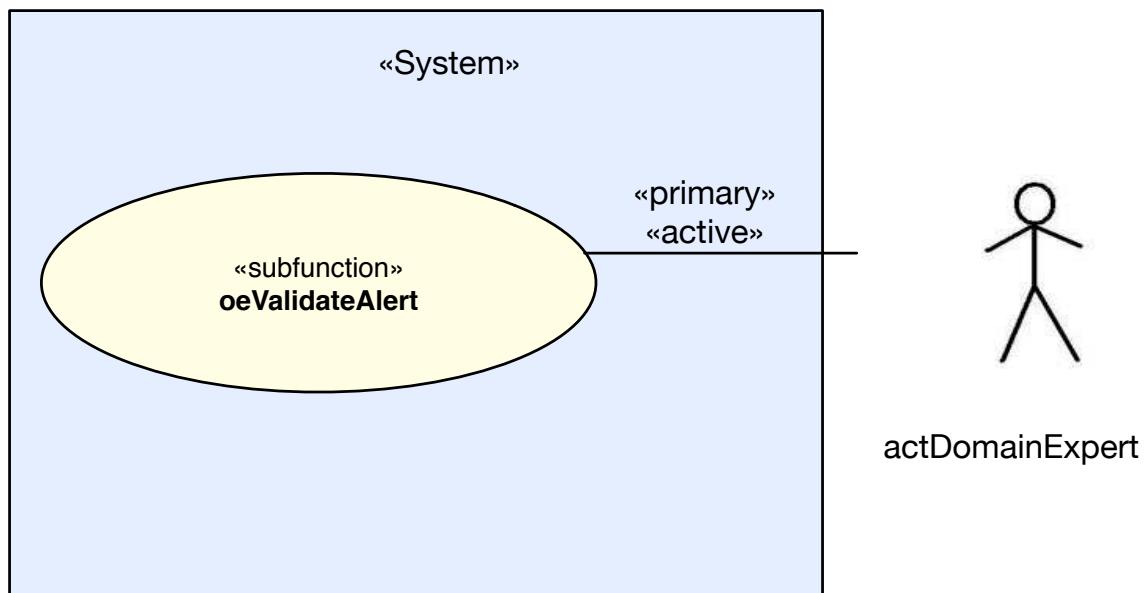


Fig. A.23 *iCrash* Use Case Diagram: oeValidateAlert

A.3 Use case instance Table(s)

USE-CASE INSTANCE	
Name	Simple and Complete DeployAndRun Instance
Instance ID	01
Environmental conditions and assumptions	The necessary IT infrastructure exists to allow for deployment of the <i>iCrash</i> system.
Inputs	inputs are sequences of characters interpreted as string or numbers.
Instance flow description	<p>1 the actMsrCreator actor theCreator sends the message oeCreateSystemAndEnvironment (1) requesting the initialization of the system and its environment (consisting of one administrator identified here by bill1) and indicating that the number of communication company actor instances for the system's environment is 1 (identified here by tango).</p> <p>2 the actAdministrator actor bill sends the message oeLogin(icrashadmin, 7WXC1359) to securely connect to the system.</p> <p>3 the actAdministrator actor bill sends the message oeAddUser(1, Steve, pwdMessirExcalibur2017, 1635242A75, 'Vehicular, Pedestrian, Wildlife, Property, Injury', Coordinator) to set up a coordinator (i.e. identified here by steve) and indicating his identification information, ID (i.e. 1) and a password (i.e. pwdMessirExcalibur2017), id from a token(i.e.1635242A75), specifying his domain of expertise(i.e. 'Vehicular, Pedestrian, Wildlife, Property, Injury') and finally he designates the user as a Coordinator(i.e.Coordinatator).</p> <p>4 the actAdministrator actor bill sends the message oeAddUser(2, franklin, pwdMessirExcalibur, 1456872B82, Null, DomainExpert) create a domain expert who is one person(i.e. identified here by franklin) and indicating his user identification his identification, ID(i.e.2), password(i.e.pwdMessirExcalibur123), id from a token(i.e.14566872B82), specifying the users domain of expertise as(Null meaning that the user has no specific domain and can therefore only be an EMS user or Domain expert) and finally he designates the user as a domain expert(i.e.DomainExpert).</p> <p>5 the actAdministrator actor bill sends the message oeAddUser(3, barry, pwdEMSExcalibur321, 2436787C82, Null, EMS) to set up a EMS user who is any user in the EMS headquarters(i.e. identified here by barry) and indicating his user identification as ID(i.e.3), password (i.e.pwdEMSExcalibur421), id from a token(i.e.2436787C82), specifying the users domain of expertise as(i.e.Null meaning that the user has no specific domain and can therefore only be an EMS user or DomainExpert) and finally he designates the user as an EMS user(i.e.EMS).</p> <p>6 the actAdministrator actor bill sends the message oeLogout() to disconnect from the system.</p> <p>7 the actComCompany actor tango sends the message oeAlert(witness, 2017-11-26-at-10-10-16AM, +3524666445252, 49.627675-6.159590, '3 cars involved in an accident.') to transfer a declaration of a car crash by a witness indicating specific phone number, the date and time, the GPS coordinates of the witnessed car crash and a short message giving additional details.</p> <p>8 the actDomainExpertactor franklin sends the message oeLogin(franklin, pwdMessirExcalibur123, 687594FAD9) to securely connect to the system entering his login(i.e.franklin), his password(i.e.pwdMessirExcalibur123)and entering his serial key that he reads form a token device given to him by the administrator(i.e.687594FAD9).</p> <p>9 the actEMSActor barry sends the message oeLogin(barry, pwdEMSExcalibur321, 24367872C282) to securely connect to the system entering his login(i.e.barry), his password(i.e.pwdEMSExcalibur321)and entering his serial key that he reads form a token device given to him by the administrator(i.e.24367872C282).</p> <p>10 the actDomainExpert actor franklin sends the message oeValidateAlert(1, 49.627675-6.159590, 2017-11-26-at-10-10-16AM, valid, 'Vehicular, Pedestrian') to validate the crisis and to set a Domain to it.</p> <p>11 the actDomainExpert actor franklin sends the message oeSollicitateCrisisHandling() indicating that there is a declared alert that is still not handled by any coordinator.</p>

continues in next page ...

...Use-Case Instance table continuation

- 12 the actCoordinator actor steve sends the message oeLogin(steve, pwdMessirExcalibur2017, 63524275D9) to securely connect to the system, entering his login(i.e.steve), his password(i.e.pwdMessirExcalibur2017) and his serial Key that he reads form a token device he is given by the administrator(i.e.63524275D9)
- 13 the actCoordinator actor steve sends the message oeGetAlertsSet (valid) to receive information about all the pending crisis.
- 14 the actCoordinator actor steve sends the message oeSetCrisisHandler(1,Medium, in-handling, 49.627095-6.160251, 2017-11-26-at-10-10-16AM) to declare that he is taking care of the alert with the ID equal to 1 which becomes the Crisis Id, sets the crisis type to(i.e.Medium, the crisis type to(i.e.in-handling), enters the GPS coordinates and the date and time.
- 15 the actComCompany actor tango sends the message oeAlert (witness, 2017-11-26-at-10-20-18AM, +3524666445314, 49.627095-6.160251,Please send rescue services.) to transfer a declaration of a car crash by a witness indicating specific the phone number, the date and time, the GPS coordinates of the witnessed car crash and a short message giving additional details. This alert's GPS coordinates match the previous alert sent coordinates in step 7 and the time elapsed between the two alerts is 10 minutes therefore the alert is considered to be originating from the same location.
- 16 the actDomainExpert actor franklin sends the message oeValidateAlert(1,49.627675-6.159590,2017-11-26-at-10-10-16AM, valid, 'Vehicular, Pedestrian')to validate the crisis and to set a Domain to it. Because the alert originated from the same location and only 10 min later it an automated message informs the Domain Expert to validate it to the same alert ID as the previous alert.
- 17 the actCoordinator actor steve send the message oeReportOnCrisis(1, ' 2 Alerts received about a car accident at the same location apparently involving 3 vehicles', 2, Medium, Handled,'witness, 2017-11-26-at-10-10-16AM,+3524666445252, 49.627675-6.159590,'3 cars involved in an accident', witness, 2017-11-26-at-10-2-18AM,+3524666445314, 49.627095-6.160251,'Please send rescue services.',3, 3) indicating the crisis ID(i.e.1), entering a specific comment in the comment area(i.e.'2Alertsrecived about a car accident at the same location apparently involving 3 vehicles'), specifying his user ID(i.e.2), setting the CrisisType(i.e.Medium), indicating the current crisisStatus(handled), entering the previous received alerts, indicating the number of vehicles involved in the accident(i.e.3) and entering the number of victims(i.e.3) because he suspects that there may be 3 victims due to the amount of cars involved in the accident. Entering a preliminary report on the crisis with the information that he presumes are right.
- 18 the actCoordinator actor steve send the message oeRequestEMSAssistance('We have received an Alert of an accident involving 3 cars from 1 victim and 1 witness at the same location please send police and ambulance',1, 49.627095-6.160251, 3,3, Ambulance Police), entering a comment(i.e. ' We have received an Alert of an accident involving 3 cars from 1 victim and 1 witness at the same location please send Police assistance'), indicating the RequestID(i.e.1), entering the GPS location of the accident(i.e.49.627095-6.160251), informing them of the number of cars involved(i.e.3) and informing them of the presumed number of victims(i.e.3) requesting emergency services assistance(i.e.Ambulance, Police).
- 19 the actCoordinator actor steve sends the message oeSetCrisisStatus(1, handled) to change the status of the crisis identified by the ID(i.e.1) to the status(i.e.handled) thus indicating that he has handled the situation for now but it's not solved yet.
- 20 the actEMS actor barry sends the message oeReplyToRequest(1,'Message received, dispatching police and ambulance units.', 1, Handled) indicating the CrisisID(i.e.1), sending the comment(i.e.'Message received, dispatching police and ambulance units.'), to confirm that the request with the ID(i.e.1) has been received and is being handled, indicated by the EMS crisis status(i.e. Handled).

continues in next page ...

... Use-Case Instance table continuation

- 21 the actEMS actor barry sends the message oeReportEMSCrisisStatus(1, 'Units arrived on scene report 4 victims, 3 cars involved in accident. Situation under control 1 victim brought Mercy hospital', solved, Mercy Hospital, Jeremy Springer, John Snow, +32168432) indicating the crisis ID(i.e.1), adding a comment(i.e. 'Units arrived on scene report 4 victims, 3 cars involved in accident. Situation under control 1 brought to Mercy hospital'), specifying the EMS crisis status as solved (i.e.solved), indicating the hospital name as (i.e.Mercy Hospital), indicating the victim name(i.e.Jeremy Springer), indicating the victims ICE contact's name as(i.e.John Snow) and the contacts phone number as (+32168432) to report on the EMS crisis Status.
- 22 the actComCompany actor tango sends message oeInfoFam('Dear John Snow, I regret to inform you that Jeremy Springer was in a car accident involving 3 other cars. Jeremy Springer was brought to the Mercy hospital for examination. Regretfully yours..', Jeremy Springer, +32168432, Mercy Hospital) to inform the victims family members about the state of the victim and in which hospital they reside.
- 23 the actCoordinator actor steve sends the message oeReportOnCrisis(1, ' 2 Alerts received about a car accident at the same location involving 3 vehicles and 4 victims one victim Jeremy Springer was brought to the Mercy hospital.', 2, Medium, Handled,'witness, 2017-11-26-at-10-10-16AM,+3524666445252, 49.627675-6.159590,'3 cars involved in an accident 'witness, 2017-11-26-at-10-2-18AM,+3524666445314, 49.627095-6.160251,'Please send rescue services.",3, 4) to set the report for the crisis with ID equal to 1 that he is handling.
- 24 the actCoordinator actor steve sends the message oeReportOnCrisis(1,'3 victims sent to hospital, 2 cars evacuated and 4 rescue unit mobilized') to set the report for the crisis with ID equal to 1 that he is handling.
- 25 the actCoordinator actor steve sends the message oeCloseCrisis(1) to declare the crisis with ID equal to 1 as closed.

Outputs

at the end of this instance flow the system should have its environment made of one communication company actor instance, one coordinator actor instance, one administrator instance and the creator instance. The system state should contain two alerts defined according to the information received from the communication company, only one crisis that should be considered as solved and being alerted by the two alerts received.



13: oeGetAlertSet(Valid)

Appendix B

Messir Specification Files Listing

B.1 File /concepts/primarytypes-associations.msr

```
1 package icrash.concepts.primarytypes.associations {
2
3 import icrash.concepts.primarytypes.datatypes
4 import icrash.concepts.primarytypes.classes
5 import icrash.environment
6
7 Concept Model {
8
9     Primary Types{
10
11 // Internal
12
13 association assctAlertctCrisis
14     ctAlert(rnAlerts) [1..*]
15     ctCrisis (rnTheCrisis) [1..1]
16
17 association assctAlertctHuman
18     ctAlert(rnSignaled) [1..*]
19     ctHuman (rnSignaler) [1..*]
20
21 association assctCrisisctCoordinator
22     ctCrisis(rnHandled) [0..*]
23     ctCoordinator(rnHandler) [0..1]
24
25 association assctCrisisctDomains
26     ctCrisis(rnTheCrisis) [1..1]
27     ctDomain(rnCrisisDomains) [1..*]
28
29 association assctCoordinatorctDomains
30     ctCoordinator(rnHandler) [1..1]
31     ctDomain(rnCrisisDomains) [1..*]
32 association assctDomainExpertDomainsctDomains
33     ctDomainExpert(rnCrisisDomains) [1..*]
34     ctDomainExpert(rnDomainExpertDomains) [1..*]
35
36 association assDomainExpertctDomains
37     ctDomainExpert(rnAlertDomainSetter) [1..1]
38     ctDomain(rnCrisisDomains) [1..*]
39
40 association assctAlertctDomains
41     ctAlert(rnAlerts) [1..1]
42     ctDomain(rnCrisisDomains) [1..*]
43
44 association assctAuthenticatedctToken
45     ctAuthenticated(rnctAuthenticated) [1..1]
46     ctToken(rnToken) [1..1]
47
48 // With Actors
49
50     association assctHumanactComCompany
51         ctHuman(rnctHuman) [0..*]
52         actComCompany(rnactComCompany) [1..1]
53
54     association assctCoordinatoractCoordinator
55         ctCoordinator(rnctCoordinator) [0..*]
56         actCoordinator(rnactCoordinator) [1..1]
57
58     association assctAuthenticatedactAuthenticated
```

```

59      ctAuthenticated(rnctAuthenticated) [1..1]
60      actAuthenticated(rnctactAuthenticated) [1..1]
61
62  association assctDomainExpertDomainExpert
63      ctDomainExpert(rnctDomainExpert) [0..*]
64      actDomainExpert(rnactDomainExpert) [1..1]
65
66  association assctEMSEMS
67      ctHuman(rnctHuman) [0..*]
68      actEMS(rnactEMS) [1..1]
69
70  }
71 }
72 }
```

Listing B.1 Messir Spec. file primarytypes-associations.msr.

B.2 File /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 initialCrisisReminderPeriod:dtSecond
19      attribute maxCrisisReminderPeriod:dtSecond
20      attribute vpStarted:ptBoolean
21
22      operation init( dtInteger AnextValueForAlertID,
23                      dtInteger AnextValueForCrisisID,
24                      dtDateAndTime Aclock,
25                      dtSecond AcrisisReminderPeriod ,
26                      dtSecond AinitialCrisisReminderPeriod ,
27                      dtSecond AmaxCrisisReminderPeriod ,
28                      ptBoolean AvpStarted): 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(
39      dtAlertID Aid,
40      etAlertStatus Astatus,
41      dtGPSLocation Alocation,
42      dtDateAndTime Ainstant,
43      dtComment Acomment):ptBoolean
44  }
45
46  class ctCrisis role rnctCrisis cardinality [0..*]{
47    attribute id:dtCrisisID
48    attribute type:etCrisisType
49    attribute status: etCrisisStatus
50    attribute location:dtGPSLocation
51    attribute instant:dtDateAndTime
52    attribute comment:dtComment
53
54    operation init(
55      dtCrisisID Aid,
56      etCrisisType Atype,
57      etCrisisStatus Astatus,
58      dtGPSLocation Alocation,
59      dtDateAndTime Ainstant
60      ):ptBoolean
61  }
62 }
```

```

63   class ctHuman role rnctHuman cardinality [0..*] {
64     attribute id:dtPhoneNumber
65     attribute name:dtName
66     attribute kind:etHumanKind
67
68     operation init(      dtName AName,
69                         dtPhoneNumber Aid,
70                         etHumanKind Akind):ptBoolean
71   }
72
73   class ctAuthenticated
74     role rnctAuthenticated
75     cardinality [0..*]{
76
77     attribute login:dtLogin
78     attribute pwd: dtPassword
79     attribute tokenID: dtTokenID
80     attribute tokenkey: dtKey
81     attribute vpIsLogged:ptBoolean
82
83     operation init(
84                   dtLogin Alogin ,
85                   dtPassword Apwd,
86                   dtTokenID Atokenid):ptBoolean
87   }
88
89   class ctCoordinator
90     role rnctCoordinator
91     cardinality [0..*]
92     extends ctAuthenticated{
93
94     attribute id:dtUserID
95
96     operation init(
97                   dtUserID Aid ,
98                   dtLogin Alogin ,
99                   dtPassword Apwd,
100                  dtTokenID Atokenid
101                  ):ptBoolean
102   }
103
104  class ctAdministrator
105    role rnctAdministrator
106    cardinality [1..1]
107    extends ctAuthenticated{
108
109    operation init(
110                  dtLogin Alogin ,
111                  dtPassword Apwd,
112                  dtTokenID Atokenid
113                  ):ptBoolean
114  }
115  class ctDomain role rnctDomains cardinality [1..*] {
116
117    attribute vpIsPedestrian :ptBoolean
118    attribute vpIsVehicular :ptBoolean
119    attribute vpIsWildlife :ptBoolean
120    attribute vpIsProperty :ptBoolean
121    attribute vpIsInjury :ptBoolean
122
123    operation init(
124
125      ptBoolean AvpIsPedestrian ,
126      ptBoolean AvpIsVehicular ,
127      ptBoolean AvpIsWildlife ,
128      ptBoolean AvpIsProperty ,
129      ptBoolean AvpIsInjury): ptBoolean
130
131  }
132
133  class ctEMStype role rnctEMStypesrequested cardinality [1..*] {
134
135    attribute vpRequestFireFighter :ptBoolean
136    attribute vpRequestPolice :ptBoolean
137    attribute vpRequestAmbulance :ptBoolean
138
139    operation init(
140
141      ptBoolean AvpRequestFireFighter ,
142      ptBoolean AvpRequestPolice ,
143      ptBoolean AvpRequestAmublance): ptBoolean
144
145  }
146

```

```

147     class ctVictim role victimofaCarCrash cardinality [1..*] extends ctHuman {
148
149     attribute VictimICEPhoneNumber:dtPhoneNumber
150     attribute VictimICEContactName:dtName
151
152     operation init(
153
154         ptBoolean VictimICEPhoneNumber ,
155         ptBoolean VictimICEContactName ):ptBoolean
156
157 }
158     class ctToken role rnctSysmetricKey cardinality [1..*] {
159
160     attribute symmetricKey:dtKey
161     attribute tokenID:dtTokenID
162
163     operation init(
164
165         dtTokenID AsymmetricKey ,
166         dtKey Akey):ptBoolean
167
168 }
169     class ctDomainExpert role rnctDomainExpert cardinality [1..*] extends ctAuthenticated {
170
171     attribute id:dtUserID
172
173     operation init(
174         dtUserID Aid,
175         dtLogin Alogin ,
176         dtPassword Apwd,
177         dtTokenID Atokenid
178     ):ptBoolean
179
180 }
181 }
182 }
183 }
```

Listing B.2 Messir Spec. file primarytypes-classes.msr.

B.3 File /concepts/primarytypes-datatypes.msr

```

1 package icrash.concepts.primarytypes.datatypes {
2
3
4 import lu.uni.lassy.messir.libraries.primitives
5 import lu.uni.lassy.messir.libraries.string
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         datatype dtAlertID extends dtString {
14             operation is():ptBoolean
15         }
16
17         datatype dtCrisisID extends dtString {
18             operation is():ptBoolean
19             operation isSameCrisis(dtDate aDate , dtTime aTimeandDate , dtDate bDate , dtTime bTime):
20                 ptBoolean
21         }
22         datatype dtLogin extends dtString {
23             operation is():ptBoolean
24         }
25         datatype dtPassword extends dtString {
26             operation is():ptBoolean
27         }
28         datatype dtUserID extends dtString {
29             operation is():ptBoolean
30         }
31         datatype dtHumanName extends dtString {
32             operation is():ptBoolean
33         }
34         datatype dtPhoneNumber extends dtString {
35             operation is():ptBoolean
36         }
37         datatype dtComment extends dtString {
38             operation is():ptBoolean
39         }
40     }
41 }
```

```

39      datatype dtLatitude extends dtReal {
40          operation is():ptBoolean
41      }
42
43      datatype dtLongitude extends dtReal {
44          operation is():ptBoolean
45      }
46      datatype dtKey extends dtString{
47          operation is():ptBoolean
48      }
49      datatype dtTokenID extends dtString{
50          operation is():ptBoolean
51      }
52
53      datatype dtGPSLocation {
54          attribute latitude: dtLatitude
55          attribute longitude: dtLongitude
56          operation is():ptBoolean
57          operation isNearTo(dtGPSLocation aGPSLocation, dtGPSLocation bGPSLocation):ptBoolean
58      }
59
60      datatype dtRecievedMessage extends dtString {
61          operation is():ptBoolean
62      }
63      datatype dtHospitalName extends dtString {
64          operation is():ptBoolean
65      }
66      datatype dtNbrofVehiculesInAccident extends dtReal {
67          operation is():ptBoolean
68      }
69      datatype dtNbrOfVictims extends dtReal{
70          operation is():ptBoolean
71      }
72      datatype dtRequestID extends dtString{
73          operation is():ptBoolean
74      }
75      datatype dtName extends dtString{
76          operation is():ptBoolean
77      }
78      datatype dtVictimICEContactName extends dtString{
79          operation is():ptBoolean
80      }
81      datatype dtVictimICEContactPhoneNumber extends dtString{
82          operation is():ptBoolean
83      }
84      enum etEMSCrisisStatus {
85          constants["pending", "handled", "solved"]
86          operation is():ptBoolean
87      }
88      enum etCrisisStatus {
89          constants["in-handling", "handled", "solved"]
90          operation is():ptBoolean
91      }
92      enum etAlertStatus {
93          constants["pending", "valid", "invalid"]
94          operation is():ptBoolean
95      }
96
97      enum etCrisisType {
98          constants["Small", "Medium", "Huge"]
99          operation is():ptBoolean
100     }
101     enum etHumanKind {
102         constants["witness", "victim", "anonymous"]
103         operation is():ptBoolean
104     }
105     enum etUserType {
106         constants["Coordinator", "DomainExpert", "EMS"]
107         operation is():ptBoolean
108     }
109
110   }
111 }
112 }
```

Listing B.3 Messir Spec. file primarytypes-datatatypes.msr.**B.4 File /concepts/secondarytypes-associations.msr**

```

1 package icrash.concepts.secondarytypes.associations {
2
```

```

3   Concept Model {
4     Secondary Types{
5       }
6     }
7   }
8 }
```

Listing B.4 Messir Spec. file secondarytypes-associations.msr.

B.5 File /concepts/secondarytypes-classes.msr

```

1 package icrash.concepts.secondarytypes.classes {
2
3   Concept Model {
4     Secondary Types{
5       }
6     }
7   }
8 }
```

Listing B.5 Messir Spec. file secondarytypes-classes.msr.

B.6 File /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 B.6 Messir Spec. file secondarytypes-datatypes.msr.

B.7 File /environment/environment.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon May 06 18:10:54 CEST 2013
4 */
5
6 package icrash.environment{
7
8 import icrash.concepts.primarytypes.datatypes
9 import icrash.concepts.primarytypes.classes
10 import icrash.concepts.secondarytypes.datatypes
11 import lu.uni.lassy.messir.libraries.primitives
12 import lu.uni.lassy.messir.libraries.math
13 import lu.uni.lassy.messir.libraries.calendar
14
15 Environment Model {
16
17   actor actMsrCreator role rnactMsrCreator cardinality [0..1] {
18     input interface inactMsrCreator {
19       }
20     output interface outactMsrCreator {
21       operation oeCreateSystemAndEnvironment(ptInteger AqtyComCompanies):ptBoolean
22     }
23   }
24
25   actor actAdministrator
26     role rnactAdministrator
27     cardinality [1..1]
```

```

28         extends actAuthenticated {
29
30     operation init():ptBoolean
31
32     output interface outactAdministrator{
33
34         operation oeAddUser(
35             dtUserID AdtUserID,
36             dtLogin AdtLogin,
37             dtPassword AdtPassword,
38             dtTokenID AdtTokenID,
39             etUserType AdtUserType,
40             ctDomain ActDomain
41         ):ptBoolean
42
43         operation oeDeleteUser(
44             dtUserID AdtUserID):ptBoolean
45     }
46
47     input interface inactAdministrator{
48
49         operation ieUserAdded():ptBoolean
50         operation ieUserDeleted():ptBoolean
51     }
52 }
53
54 actor actCoordinator
55     role rnactCoordinator
56     cardinality [0..*]
57     extends actAuthenticated{
58
59     operation init():ptBoolean
60
61     output interface outactCoordinator{
62         operation oeCloseAlert(dtAlertID AdtAlertID):ptBoolean //Never used
63         operation oeCloseCrisis(dtCrisisID AdtCrisisID):ptBoolean
64         operation oeGetAlertsSet(etAlertStatus AetAlertStatus):ptBoolean
65         operation oeGetCrisisSet(etCrisisStatus AetCrisisStatus):ptBoolean //never used
66         operation oeSetCrisisHandler(dtAlertID AdtAlertID, ctCrisis ActCrisis):ptBoolean
67         operation oeReportOnCrisis(
68             dtCrisisID AdtCrisisID,
69             dtComment AdtComment,
70             dtUserID AdtUserID,
71             etCrisisType AdtCrisisType,
72             etCrisisStatus AdtCrisisStatus,
73             dtReceivedMessage AdtReceivedMessage,
74             dtNbrOfVehiclesInAccident AdtVehiclesInAccident,
75             dtNbrOfVictims AdtNbrVictims
76         ):ptBoolean
77         operation oeSetCrisisStatus(
78             dtCrisisID AdtCrisisID,
79             etCrisisStatus AetCrisisStatus
80         ):ptBoolean
81         operation oeRequestEMSService(
82             dtComment AdtComment,
83             dtRequestID AdtRequest,
84             dtGPSLocation AdtGPSLocation,
85             dtNbrOfVehiclesInAccident AdtVehiclesInAccident,
86             dtNbrOfVictims AdtNbrVictims,
87             ctEMSType ActEMSType
88         ):ptBoolean
89     }
90
91     input interface inactCoordinator{
92         operation ieSendAnAlert(ctAlert AclAlert):ptBoolean
93         operation ieSendACrisis(ctCrisis ActCrisis):ptBoolean
94         operation ieMessage(ptString AMessage):ptBoolean
95     }
96 }
97
98 actor actComCompany role rnactComCompany cardinality [0..*]{
99
100    operation init():ptBoolean
101
102    output interface outactComCompany{
103        operation oeAlert(
104            etHumanKind AetKind,
105            dtDate AdtMyDate,
106            dtTime AdtTime,
107            dtPhoneNumber AdtPhoneNumber,
108            dtGPSLocation AdtGPSLocation,
109            dtComment AdtComment
110        ):ptBoolean
111

```

```

112
113     }
114
115     input interface inactComCompany{
116         operation ieSMSSend(dtPhoneNumber AdtPhoneNumber,
117             dtSMS AdtSMS
118             ) : ptBoolean
119     }
120 }
121
122 actor actAuthenticated role rnactAuthenticated cardinality [0..*]{
123
124     operation init():ptBoolean
125
126     output interface outactAuthenticated{
127         operation oeLogin(dtLogin AdtLogin, dtPassword AdtPassword, dtKey Adtkey):ptBoolean
128         operation oeLogout():ptBoolean
129     }
130
131     input interface inactAuthenticated{
132         operation ieMessage(ptString AMessage):ptBoolean
133     }
134 }
135
136 actor actDomainExpert role rnactDomainExpert cardinality [1..*] {
137
138     operation init():ptBoolean
139
140     input interface inDomainExpert {
141         operation ieAlertRecieved(): ptBoolean
142     }
143     output interface outactDomainExpert {
144         operation oeValidateAlert(dtAlertID AdtAlertID,
145             dtGPSLocation AdtGPSLocation,
146             dtDateAndTime AdtDateandTime,
147             etAlertStatus edtAlertStatus,
148             ctDomain ActDomain
149             ): ptBoolean
150         operation oeSollicitateCrisisHandling() : ptBoolean
151     }
152 }
153 actor actEMS role rnactEMS cardinality [1..*] {
154
155     input interface inactEMS {
156
157     }
158     output interface outactEMS {
159         operation oeReportEMSCrisisStatus(
160             dtCrisisID AdtCrisisID,
161             dtComment AdtComment,
162             etEMSCrisisStatus AdtEMSCrisisStatus,
163             dtHospitalName AdtHospitalName,
164             ctVictim ActVictim
165             ): ptBoolean
166         operation oeReplyToRequest(
167             dtCrisisID AdtCrisisID,
168             dtComment AdtComment,
169             dtRequestID AdtRequest,
170             etEMSCrisisStatus AdtEMsCrisisStatus
171             ): ptBoolean
172     }
173 }
174 }
175 }
```

Listing B.7 Messir Spec. file environment.msr.

B.8 File /operations/concepts/primarytypes-classes/primarytypes-classes-ctAdministrator.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.classes.ctAdministrator{
7
8 import lu.uni.lassy.messir.libraries.primitives
9 import lu.uni.lassy.messir.libraries.calendar
10 import lu.uni.lassy.messir.libraries.math
11
```

```

12 import icrash.concepts.primarytypes.datatypes
13 import icrash.concepts.primarytypes.classes
14 import icrash.concepts.secondarytypes.datatypes
15 import icrash.concepts.secondarytypes.classes
16
17 Operation Model {
18
19     operation: icrash.concepts.primarytypes.classes.ctAdministrator.init(
20         dtLogin Alogin,
21         dtPassword Apwd,
22         dtTokenID Awptoken
23     ):ptBoolean
24     preF: true
25     postF: true
26
27 }
28 }
```

Listing B.8 Messir Spec. file primarytypes-classes-ctAdministrator.msr.

B.9 File

/operations/concepts/primarytypes-classes/primarytypes-classes-ctAlert.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.classes.ctAlert{
7
8 import lu.uni.lassy.messir.libraries.primitives
9 import lu.uni.lassy.messir.libraries.calendar
10
11 import icrash.concepts.primarytypes.datatypes
12 import icrash.concepts.primarytypes.classes
13 import icrash.concepts.secondarytypes.datatypes
14 import icrash.concepts.secondarytypes.classes
15
16 Operation Model {
17
18     operation: icrash.concepts.primarytypes.classes.ctAlert.init(dtAlertID Aid, etAlertStatus Astatus,
19                     dtGPSLocation Alocation, dtDateAndTime Ainstant, dtComment Acomment):ptBoolean
20     preF: true
21     postF: true
22
23 }
```

Listing B.9 Messir Spec. file primarytypes-classes-ctAlert.msr.

B.10 File /operations/concepts/primarytypes-classes/primarytypes-classes-ctState.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.classes.ctState{
7
8 import lu.uni.lassy.messir.libraries.primitives
9 import lu.uni.lassy.messir.libraries.calendar
10 import lu.uni.lassy.messir.libraries.math
11
12 import icrash.concepts.primarytypes.datatypes
13 import icrash.concepts.primarytypes.classes
14 import icrash.concepts.secondarytypes.datatypes
15 import icrash.concepts.secondarytypes.classes
16
17 Operation Model {
18
19     operation: icrash.concepts.primarytypes.classes.ctState.init(
20         dtInteger AnextValueForAlertID,
21         dtInteger AnextValueForCrisisID,
22         dtDateAndTime Aclock,
23         dtSecond AcrisisReminderPeriod,
24         dtSecond AinitialCrisisReminderPeriod,
25         dtSecond AmaxCrisisReminderPeriod,
```

```

26     ptBoolean AvpStarted
27   ):ptBoolean
28   preF: true
29   postF: true
30
31 }
32 }
```

Listing B.10 Messir Spec. file primarytypes-classes-ctState.msr.

B.11 File /operations/concepts/primarytypes-datatatypes/primarytypes-datatypes-dtAlertID.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.datatypes.dtAlertID {
7
8 import lu.uni.lassy.messir.libraries.primitives
9
10 import icrash.concepts.primarytypes.datatypes
11 import icrash.concepts.primarytypes.classes
12 import icrash.concepts.secondarytypes.datatypes
13 import icrash.concepts.secondarytypes.classes
14
15 Operation Model {
16
17   operation: icrash.concepts.primarytypes.datatypes.dtAlertID.is():ptBoolean
18   ocl: "
19 context PrimaryTypesDatatypes::dtAlertID ::"
20 is():Boolean
21 /**
22 pre:
23 /* Pre Functional:*/
24 /* Pre F01 */
25 true
26 post:
27 let TheResult:Boolean in
28 if
29 (
30 /* Post Functional:*/
31 /* Post F01 */
32 self.value > 0
33 )
34 then TheResult = true
35 else TheResult = false
36 endif
37 and result = TheResult
38
39 /**
40 "
41 }
42 }
```

Listing B.11 Messir Spec. file primarytypes-datatypes-dtAlertID.msr.

B.12 File /operations/concepts/primarytypes-datatatypes/primarytypes-datatypes-dtComment.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.datatypes.dtComment {
7
8 import lu.uni.lassy.messir.libraries.primitives
9
10 import icrash.concepts.primarytypes.datatypes
11 import icrash.concepts.primarytypes.classes
12 import icrash.concepts.secondarytypes.datatypes
13 import icrash.concepts.secondarytypes.classes
14
15 Operation Model {
16
17   operation: icrash.concepts.primarytypes.datatypes.dtComment.is():ptBoolean
```

```

18     ocl:"
19 context PrimaryTypesDatatypes::dtComment :: 
20 is () : Boolean
21 /*-----*/
22 pre:
23 /* Pre Functional:*/
24 /* Pre F01 */
25 true
26 post:
27 let TheResult:Boolean in
28 let MaxLength:Integer in
29 if
30 (
31 /* Post Functional:*/
32 /* Post F01 */
33 MaxLength = 160
34 and self.value.size() <= MaxLength
35 )
36 then TheResult = true
37 else TheResult = false
38 endif
39 and result = TheResult
40 /*-----*/
41 "
42 }
43 }
```

Listing B.12 Messir Spec. file primarytypes-datatatypes-dtComment.msr.

B.13 File /operations/concepts/primarytypes-datatypes/primarytypes-datatypes-dtPhoneNumber.msr

```

1 /*
2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.primarytypes.datatypes.dtPhoneNumber{
7
8 import lu.uni.lassy.messir.libraries.primitives
9
10 import icrash.concepts.primarytypes.datatypes
11 import icrash.concepts.primarytypes.classes
12 import icrash.concepts.secondarytypes.datatypes
13 import icrash.concepts.secondarytypes.classes
14
15 Operation Model {
16
17     operation: icrash.concepts.primarytypes.datatypes.dtPhoneNumber.is(): ptBoolean
18     ocl:"
19 context PrimaryTypesDatatypes::dtPhoneNumber :: 
20 is () : Boolean
21 /*-----*/
22 pre:
23 /* Pre Functional:*/
24 /* Pre F01 */
25 true
26 post:
27 let TheResult:Boolean in
28 if
29 (
30 /* Post Functional:*/
31 /* Post F01 */
32 self.value.size() = 10
33 )
34 then TheResult = true
35 else TheResult = false
36 endif
37 and result = TheResult
38 /*-----*/
39 "
40 }
41 }
```

Listing B.13 Messir Spec. file primarytypes-datatypes-dtPhoneNumber.msr.

B.14 File /operations/concepts/secondarytypes-datatypes/dtSMS.msr

```
1 /*
```

```

2 * @author nicolas.guelfi
3 * @date Mon Sep 30 17:11:03 CEST 2013
4 */
5
6 package icrash.operations.concepts.secondarytypes.datatypes.dtSMS {
7
8 import lu.uni.lassy.messir.libraries.primitives
9 import lu.uni.lassy.messir.libraries.calendar
10 import lu.uni.lassy.messir.libraries.math
11
12 import icrash.concepts.primarytypes.datatypes
13 import icrash.concepts.primarytypes.classes
14 import icrash.concepts.secondarytypes.datatypes
15 import icrash.concepts.secondarytypes.classes
16
17 Operation Model {
18
19 operation: icrash.concepts.secondarytypes.datatypes.dtSMS.is () : ptBoolean
20   preF: true
21   postF: true
22
23 }
24 }
```

Listing B.14 Messir Spec. file dtSMS.msr.

B.15 File /operations/environment/environment-actAuthenticated.msr

```

1 package icrash.operations.environment.actAuthenticated {
2
3 import icrash.environment
4
5 import lu.uni.lassy.messir.libraries.primitives
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
12 Operation Model {
13
14 operation: icrash.environment.actAuthenticated.outactAuthenticated.oeLogin(dtLogin AdtLogin ,
15   dtPassword AdtPassword , dtKey Adtkey) : ptBoolean
16 //-----
16 preF: true
17 //-----
18 preP:
19 // PreP 01
20 let theSystemPre: ctState in
21 theSystem
22
23 operation: icrash.environment.actAuthenticated.outactAuthenticated.oeLogout() : ptBoolean
24   preF: true
25   postF: true
26
27 }
28 }
```

Listing B.15 Messir Spec. file environment-actAuthenticated.msr.

B.16 File /operations/environment/environment-actComCompany.msr

```

1 package icrash.operations.environment.actComCompany {
2
3 import icrash.environment
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 import icrash.concepts.secondarytypes.classes
13
14 Operation Model {
15
16 operation: icrash.environment.actComCompany.outactComCompany.oeAlert(etHumanKind AetKind , dtDate
17   AdtMyDate , dtTime AdtTime , dtPhoneNumber AdtPhoneNumber , dtGPSLocation AdtGPSLocation , dtComment
18   AdtComment) : ptBoolean
19 }
```

```

17  preF: true
18  preP: true
19  postF: true
20  postP: true
21  ocl: "iyyiyuuouuhuhuhulhhu
22 hii
23 preF: if coordinator"
24
25 }
26 }
```

Listing B.16 Messir Spec. file environment-actComCompany.msr.

B.17 File /operations/environment/environment-actMsrCreator.msr

```

1 package icrash.operations.environment.actMsrCreator{
2
3 import icrash.environment
4
5 import lu.uni.lassy.messir.libraries.primitives
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
12 Operation Model {
13
14 operation: icrash.environment.actMsrCreator.outactMsrCreator.oeCreateSystemAndEnvironment ( ptInteger
15   AqtyComCompanies ) : ptBoolean
16
17 ocl:
18 "
19 context Environment::outactMsrCreator::
20 oeCreateSystemAndEnvironment(qtyComCompanies: Integer)
21 /*-----*/
22 pre:
23 /* Pre Protocol:*/
24 /* PreF01 */
25 true
26 /* Pre Functional:*/
27 /* PreP01 */
28 and true
29 /*-----*/
30 post:
31 let TheState: PrimaryTypesClasses::ctState in
32 let AactMsrCreator: Environment::actMsrCreator in
33 /* Post Functional:*/
34 /* PostF01 */
35 TheState._init(1, true)
36
37 /* PostF02 */
38 and TheState.rnactComCompany->size() = qtyComCompanies
39 and TheState.rnactComCompany
40 ->forAll(cca: Environment::actComCompany | cca._init())
41
42 /* PostF03 */
43 and AactMsrCreator._init()
44
45 /* Post Protocol:*/
46 /* PostP01 */
47 and TheState.vpmStarted = true
48 "
49
50 operation: icrash.environment.actComCompany.init () : ptBoolean
51
52 ocl:
53 "
54 context Environment::actComCompany:: _init () : Boolean
55 /*-----*/
56 pre: true
57 /*-----*/
58 post:
59 let pIN: Environment::inactComCompany in
60 let pOUT: Environment::outactComCompany in
61 if (
62 /* Post Functional:*/
63 /* Post F01 */
64 self.oclIsNew()
65 /* Post F02 */
```

```
66     and pIN.oclIsNew()
67     and pOUT.oclIsNew()
68     and pIN = self.InterfaceIN
69     and pOUT= self.InterfaceOUT
70     )
71     then result = true
72     else result = false
73   endif
74   /*-----*/
75 }
76 }
```

Listing B.17 Messir Spec. file environment-actMsrCreator.msr.

