Application of the AORE4PF Method on the Crisis Mangement Systems Case Study

Keywords: Early Aspect, Problem Frames, Requirements Engineering

Abstract: This document provides the details of an experiment that we performed for the evaluation for our method AORE4PF (Aspect-Oriented Requirements Engineering with Problem Frames). In this experiment, we applied the AORE4PF method on crisis mangement systems case study.

1 Input Documents

In this section, we present the input documents, which we used as starting point for the application of our method. This documents are the informal description shown in subsection 1.1 and the use cases shown in subsection 1.2. Both inputs were taken from (Kienzle et al., 2010) where Kienzle et al. propose a crisis management system (CMS) as a case study for aspect-oriented modeling.

1.1 Informal Description

A crisis management scenario is usually triggered by a crisis report from a witness at the scene. A coordinator, who is in charge of organizing all required resources and tasks, initiates the crisis management process. The coordinator has access to the camera surveillance system. The surveillance system is an external system used to monitor traffic on highways or other busy routes. The cameras are installed only in specific locations. If a crisis occurs in locations under surveillance, the crisis management system can request video feed that allows the coordinator to verify the witness information. A super observer, an expert in the field (depending on the kind of crisis), is assigned to the scene to observe the emergency situation and identify the tasks necessary to cope with the situation. The tasks are crisis missions defined by the observer. The coordinator is then required to process the missions by allocating suitable resources to each task. Depending on the type of crisis, human resources could include firemen, doctors, nurses, policemen, and technicians, and hardware resources could include transportation systems, computing resources, communication means (such as PDAs or mobile phones), or other necessities like food or clothes. Animals, for instance police dogs, are also used as resources in some situations. The human and animal resources act as first-aid workers. Each first-aid worker is assigned a specific task which needs to be executed to recover from the abnormal situation. The workers are expected to report on the success or failure in carrying out the missions. The completion of all missions would allow the crisis to be concluded.

A crisis management system (CMS) should include the following functionalities:

- initiating a crisis based on an external input from a witness
- processing a crisis by executing the missions defined by a super observer and then assigning internal and/or external resources
- wrapping-up and archiving crisis
- authenticating users
- handling communication between coordinator/system and resources

Initiating a crisis based on an external input from a witness. The system shall support at least 1000 witnesses calling in at a time and it shall support management of at least 100 crises at a time. Maintenance shall be postponed or interrupted if a crisis is imminent without affecting the systems capabilities. The system shall monitor criminal activity to ensure safety of rescue resources, civilians and casualties.The safety of rescue personnel shall take top priority for the system. The system shall provide support for storing, updating and accessing the following information on crisis resolution strategies: type of crisis; step-by-step guide to resolve crisis; con- figuration of missions required; links to alternate strategies; applications to previous crises; success rate.

Processing a crisis by executing the missions defined by a super observer and then assigning internal and/or external resources. The system shall provide up-to-date information to rescue resources. The system shall support management of at least 200 missions per crisis at a time. The system shall recommend or enlist alternate resources in case of unavailability or shortage of suitable resources. The resources, which are mobile, shall be able to communicate with other resources on the crisis site and the control center regardless of location, terrain and weather conditions. The system shall have access to detailed maps, terrain data and weather conditions for the crisis location and the routes leading to it.The information about maps, terrain and weather data shall be accessible with a 99% accuracy. The system shall monitor weather and terrain conditions at crisis site to ensure safe operation and withdrawal of rescue resources, and removal of civilians and casualties. The system shall determine a perimeter for the crisis site to ensure safety of civilians and removal of casualties to a safe distance. The system shall recommend alternate strategies for dealing with a crisis as the crisis conditions (e.g., weather conditions, terrain conditions, civilian or criminal activity) change. Rescue resources shall be able to access information on the move. The system shall authenticate users and resources on the basis of the access policies when they first access any components or information. If a user remains idle for 30 minutes or longer, the system shall require them to re-authenticate. The system shall provide support for storing, updating and accessing the following information before and during a crisis: information on available and deployed resources (both internal and external): type of resource (human or equipment); capability; rescue team; location; estimated time of arrival (ETA) on crisis site. Any retrieved information shall be information with a maximum delay of 500 milliseconds. The system shall additionally record the following statistical information on both on-going and resolved crises: rate of progression; average response time of rescue teams; individual response time of each rescue team; success rate of each rescue team; rate of casualties; success rate of missions. The system shall monitor emissions from crisis site to determine safe operating distances for rescue resources.

Wrapping-up and archiving crisis. The system shall provide support for storing, updating and accessing the following information on both resolved and on-going crises: type of crisis; location of crisis; witness report; witness location; witness data; time reported; duration of resolution; resources deployed; civilian casualties; crisis management personnel casualties; strategies used; missions used; location of super observer; crisis perimeter; location of rescue teams on crisis site; level of emissions from crisis site; log of communications; log of decisions; log of problems encountered. The system shall provide statistical analysis tools to analyse individual crisis data and data on multiple crises.

Authenticating users. The system shall define access policies for various classes of users. The access policy shall describe the components and information each class may add, access and update.

Handling communication between coordinator/system and resources. The system shall support communication, coordination and information access for at least 20000 rescue resources in deployment at a time. The system shall record data upon receipt without modifications. The delay in communication of information between control center and rescue personnel as well as amongst rescue personnel shall not exceed 500 milliseconds. All communications in the system shall use secure channels compliant with AES-128 standard encryption. The system shall provide location-sensitive information to rescue resources. Rescue resources shall communicate their location to the control center. The system shall be able to use alternate communication channels in case of unavailability or shortage of existing channels. The system shall be able to maintain effective communication in areas of high disruption or noise at the crisis site. The communication between the system and rescue resources shall have a maximum deterioration factor of 0.0001 per 1000 kilometers. Whenever an failure occurs, the system shall recover in a maximum of 30 seconds. The probability of a failure shall not exceed a maximum failure rate of 0.001%. The control center shall receive and update the following information on an on-going crisis at intervals not exceeding 30 seconds: resources deployed; civilian casualties; crisis management personnel casualties; location of super observer; crisis perimeter; location of rescue teams on crisis site; level of emissions from crisis site; estimated time of arrival (ETA) of rescue teams on crisis site.

CMS replace existing crisis management systems that a) still manually keep track of important crisis-related information and that b) operate largely without automated support for crisis resolution strategies in order to respond to a crisis. The system shall be in operation 24 hours a day, everyday, without break, throughout the year except for a maximum downtime of 2 hours every 30 days for maintenance.

In some of the models presented in this technical report, we have focused on one particular CMS: the car crash crisis management system. The car crash CMS includes all the functionalities of general crisis management systems, and some additional features specific to car crashes such as facilitating the rescuing of victims at the crisis scene and the use of tow trucks to remove damaged vehicles.

A car crash is defined in Wikipedia as following:

car crash A car accident or car crash is an incident in which an automobile collides with anything that causes damage to the automobile, including other automobiles, telephone poles, buildings or trees, or in which the driver loses control of the vehicle and damages it in some other way, such as driving into a ditch or rolling over. Sometimes a car accident may also refer to an automobile striking a human or animal.

Our Car Crash CMS addresses car crashes involving single or multiple vehicles, humans, or other objects. This case study is however limited to management of human victims only and does not provide rescue missions specifically for animals. First-aid animal workers are not included in the scope of this case study either.

Car crash specific functionalities include the following:

- facilitating the rescue mission carried out by the police by providing them with detailed information on the location of the crash
- managing the dispatch of ambulances or other alternate emergency vehicles to transport victims from the crisis scene to hospitals
- facilitating the first-aid missions by providing relevant medical history of identified victims to the first-aid workers by querying data bases of local hospitals
- facilitating the medical treatment process of victims by providing important information about the crash to the concerned workers, i.e. paramedics, doctors, upon arrival at the hospital
- managing the use of tow trucks to remove obstacles and damaged vehicles from the crisis scene.

1.2 Use Cases

The use case model includes a summary use case diagram (presented in subsection 1.2.1, and individual use cases presented in subsection 1.2.2).

1.2.1 Use Case Diagram

Fig. 1 shows the use cases related to the summary-level goal Resolve Crisis in the Car Crash Crisis Management System, by means of a use case diagram.

Details of all the use cases that directly relate to the summary level use case Resolve Crisis are given in section 1.2.2. The listed use cases are: Resolve Crisis, Capture Witness Report, Assign Internal Resource, Assign External Resource, Execute Mission, Execute SuperObserver Mission, Execute Rescue Mission, and Authenticate User.

Use cases describing other missions, such as the Execute Helicopter Transport Mission, or Execute Remove Obstacle Mission are not shown for space reasons. Likewise, details of use cases related to the management of the resource database are not included for space reasons. Such use cases would, for instance, include:

- Creating records for CMSEmployees
- Managing access rights of CMSEmployees
- Updating the availability of CMSEmployees due to sickness or vacation
- Dealing with problems of the CMS-controlled vehicles that are not related to a crisis

Finally, following a dependability-focussed requirements engineering process such as DREP [1], exceptional situations that a CMS might be exposed to should also be considered. For this case study, several exceptional situations were discovered that affect the context in which the system operates, and that require the system to react in a certain way to continue to provide reliable and safe service. The situations are:

- Severe Weather Conditions: Bad weather makes helicopter transportation impossible.
- Strike: A strike affects the availability of CMS employees and external workers.
- Risk of Explosion: Leaking gas and open fire threatens the safety of workers.
- VIP Victim: One of the crash victims is a VIP (such as for instance, the president). Handling of the crisis should therefore be coordinated by the appropriate office.
- Criminal Case: The reason for the crash is of criminal nature, and therefore the rescue missions have to be carried out accordingly.

To detect and to handle the above situations, we added the following exceptional actors: Weather- InformationSystem, NationalCrisisCenter. The detailed handler use cases that describe the functionality that such a reliable car crash CMS is to provide are not described in this document for space reasons.

1.2.2 Textual Use Cases

Use cases are a widely used formalism for discovering and recording behavioral requirements of software systems, since they can be effectively used as a communication means between technical as well as

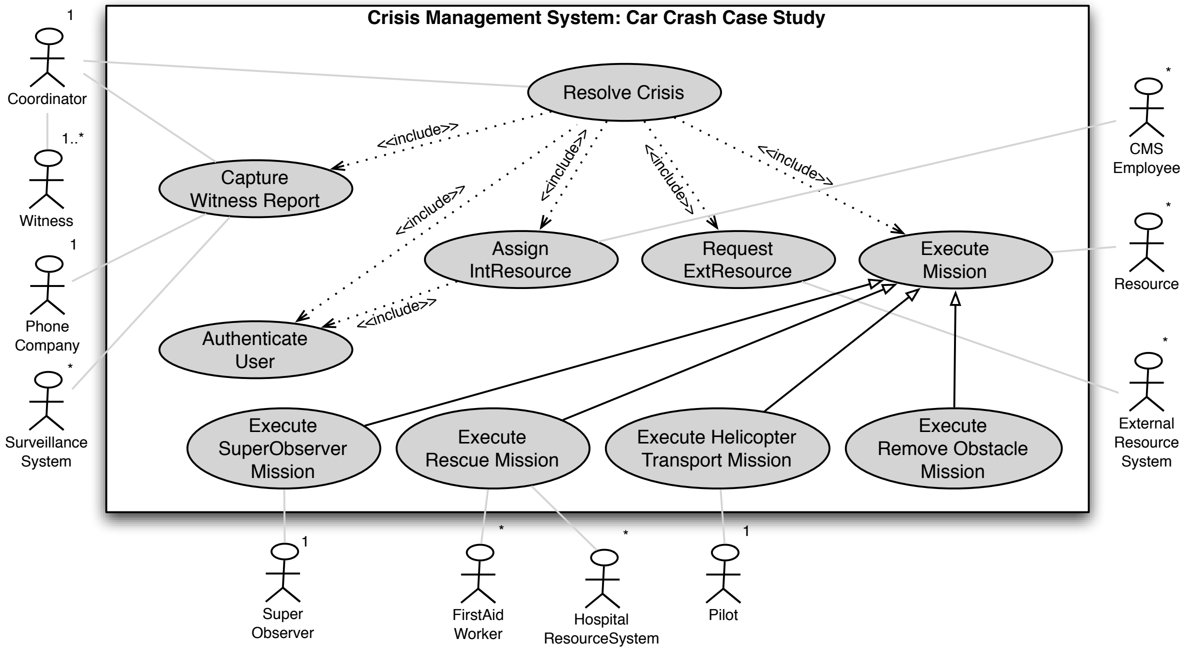


Figure 1: Car Crash Case: Standard Use Case Diagram

non- technical stakeholders of the software under development. In short, use cases are stories of using a system to meet goals. They are in general text-based, but their strength is that they both scale up or scale down in terms of sophistication and formality, depending on the need and context.

The use cases presented here follow a textual template. The main success scenario is a numbered list of lines of text (subsequently named steps) that describes the possible interactions between the primary actor, potential secondary actors and the Car Crash CMS (subsequently named System) that occur to reach a particular goal. Alternate ways of achieving a goal, or situations in which the goal can not be reached, are described in the extension part of the template.

1.2.3 Resolve Crisis

Use Case 1: Resolve Crisis
Scope: Car Crash Crisis Management System
Primary Actor: Coordinator
Secondary Actor: Resource
Intention: The intention of the Coordinator is to resolve a car crash crisis by asking employees and external workers to execute appropriate missions.
Main Success Scenario:
Witness places a call to the crisis centre, where it is answered by a Coordinator.
1. *Coordinator* captures witness report (UC 2).
2. *System* recommends to *Coordinator* the missions that are to be executed based on the current information about the crisis and resources.
3. *Coordinator* selects one or more missions recommended by the system.
4. For each internal resource required by a selected mission, *System* assigns an internal resource (UC 3).
5. For each external resource required by a selected mission, *System* requests an external resource (UC 4).
6. *Resource* notifies *System* of arrival at mission location.
7. *Resource* executes the mission (UC 5).
8. *Resource* notifies *System* of departure from mission location.
9. In parallel to steps 6-8, *Coordinator* receives updates on the mission status from *System*.
10. In parallel to steps 6-8, *System* informs *Resource* of relevant changes to mission / crisis information.
11. *Resource* submits the final mission report to *System*.
12. In parallel to steps 4-8, *Coordinator* receives new information about the crisis from *System*.
13. *Coordinator* closes the file for the crisis resolution.
Use case ends in success.

Extensions:
1a. *Coordinator* is not logged in.
1a.1 *Coordinator* authenticates with System (UC 10).
1a.2 Use case continues with step 1.
4a. Internal resource is not available after step 4.
4a.1 *System* requests an external resource instead (i.e., use case continues in parallel with step 5).
5a. External resource is not available after step 5.
5a.1 Use case continues in parallel with step 2.
6a. *System* determines that the crisis location is unreachable by standard transportation means, but reachable by helicopter.
6a.1 *System* informs the *Coordinator* about the problem.
6a.2 *Coordinator* instructs *System* to execute a helicopter transport mission (UC 09).
6a.3 Use case continues with step 6.
6b. *Resource* is unable to contact *System*.
6b.1 *SuperObserver* notifies *System* that resource arrived at the mission location.
6c. Although *Resource* should be at mission location by now, *Resource* has not yet notified *System*.
6c.1 *System* requests *Resource* to provide an update of its location.
6c.2 Use case continues at step 6.
7a. One or more further missions are required in step 6.
7a.1 Use case continues in parallel with step 2.
7b. The mission failed.
7b.1 Use case continues with step 2.
8a. *Resource* is unable to contact *System*.
8a.1 *SuperObserver* notifies *System* that resource is leaving the mission location.
8b. Although mission should be completed by now, *Resource* has not left mission location.
8b.1 *System* requests *Resource* to provide the reason for the delay.
8b.2 Use case continues at step 7.
9a. Changes to mission are required.
9a.1 Use case continues in parallel with step 2.
11a. *Resource* never files a mission report.
11a.1 Mission use case ends without mission report.
12a. Changes to mission are required.
12a.1 Use case continues in parallel with step 2.

1.2.4 Capture Witness Report

Use Case 2: Capture Witness Report
Scope: Car Crash Crisis Management System
Primary Actor: Coordinator
Secondary Actor: PhoneCompany, SurveillanceSystem
Intention: The Coordinator intends to create a crisis record based on the information obtained from witness.
Main Success Scenario:
Coordinator requests Witness to provide his identification.
1. *Coordinator* provides witness information[1] to *System* as reported by the witness.
2. *Coordinator* informs *System* of location and type of crisis as reported by the witness.
In parallel to steps 2-4:
2a.1 *System* contacts *PhoneCompany* to verify witness information.
2a.2 *PhoneCompany* sends address/phone information to *System*.
2a.3 *System* validates information received from the PhoneCompany.
3. *System* provides *Coordinator* with a crisis-focused checklist.
4. *Coordinator* provides crisis information[2] to *System* as reported by the witness.
5. *System* assigns an initial emergency level to the crisis and sets the crisis status to *active*.
Use case ends in success.
Extensions:
1a,2a. The call is disconnected. The base use case terminates.
In parallel to steps 3-4, if the crisis location is covered by camera surveillance:
3a.1 *System* requests video feed from *SurveillanceSystem*.
3a.2 *SurveillanceSystem* starts sending video feed to *System*.
3a.3 *System* starts displaying video feed for *Coordinator*.
4a. The call is disconnected.
4a.1 Use case continues at step 5 without crisis information.
5a. PhoneCompany information does not match information received from Witness.
5a.1 The base use case is terminated.
5b. Camera vision of the location is perfect, but *Coordinator* cannot confirm the situation that the witness describes *or* the *Coordinator* determines that the witness is calling in a fake crisis.
5b.1 The base use case is terminated.

[1]Witness information includes the first name, last name, phone number, and address.
[2]Crisis information includes the details about the crisis, the time witnessed, etc.

1.2.5 Assign Internal Resource

Use Case 3: Assign Internal Resource
Scope: Car Crash Crisis Management System
Primary Actor: None
Secondary Actor: CMSEmployee
Intention: The intention of System is to find, contact, and assign a mission to the most appropriate available CMSEmployee.
Main Success Scenario:
System selects an appropriate CMSEmployee based on the mission type, the emergency level, location and requested expertise. In very urgent cases, steps 1 and 2 can be performed for several CMSEmployees concurrently, until one of the contacted employees accepts the mission.
1. *System* sends *CMSEmployee* mission information.
2. *CMSEmployee* informs *System* that he accepts the mission.
Use case ends in success.
Extensions:
1a. *CMSEmployee* is not logged in.
1a.1 *System* requests the *CMSEmployee* to login.
1a.2 *CMSEmployee* authenticates with System (UC 10).
1a.3 Use case continues at step 1.
1b. *CMSEmployee* is unavailable or unresponsive.
1b.1 *System* selects the next appropriate *CMSEmployee*.
1b.2 Use case continues at step 1.
1b.1a No other *CMSEmployee* is available. Use case ends in failure.
2a. *CMSEmployee* informs *System* that he cannot accept the mission.
2a.1 *System* selects the next appropriate *CMSEmployee*.
2a.2 Use case continues at step 1.
2a.2a No other *CMSEmployee* is available. Use case ends in failure.

1.2.6 Request External Resource

Use Case 4: Request External Resource
Scope: Car Crash Crisis Management System
Primary Actor: Coordinator
Secondary Actor: ExternalResourceSystem (ERS)
Intention: The System requests a mission from an external resource, such as a fire station, police station or external ambulance service.
Main Success Scenario:
1. *System* sends mission request to *ERS*, along with mission-specific information[1].
2. *ERS* informs *System* that request can be processed.
Use case ends in success.
Extensions:
2a. *ERS* notifies *System* that it partially approves request for resources. Use case ends in degraded success.
2b. *ERS* notifies *System* that it can not service the request. Use case ends in failure.

[1]Mission-specific information includes things such as the location and emergency level of the mission, the quantity of vehicles requested, special characteristics of the aid worker or vehicle, etc.

1.2.7 Execute Mission

Use Case 5: Execute Mission
Intention: The Resource executes a mission in order to help resolve a crisis. ExecuteMission is an abstract use case. The details of the interaction for specific missions are presented in child use cases such as ExecuteSuperObserverMission (UC 6), or ExecuteRescueMission (UC 7).

1.2.8 Execute SuperObserver Mission

Use Case 6: Execute SuperObserver Mission
Scope: Car Crash Crisis Management System
Primary Actor: SuperObserver
Secondary Actor: None
Intention: The intention of the SuperObserver is to observe the situation at the crisis site to be able to order appropriate missions.
Main Success Scenario:
SuperObserver is at the crisis location.
1. *System* sends a crisis-specific checklist to *SuperObserver*.
2. *SuperObserver* feeds *System* with crisis information.
3. *System* suggests crisis-specific missions to *SuperObserver*.
Steps 4-8 is repeated as many times as needed.
4. *SuperObserver* notifies *System* of the type of mission he wants to create.
5. *System* sends a mission-specific information request to *SuperObserver*.
6. *SuperObserver* sends mission-specific information[1] to *System*.
7. *System* acknowledges the mission creation to *SuperObserver*.
8. *System* informs *SuperObserver* that mission was completed successfully.
9. *SuperObserver judges that his presence is no longer needed at the crisis location*.
Use case ends in success.
Extensions:
7a. Mission cannot be created and replacement missions are possible.
7a.1 *System* suggests replacement missions to *SuperObserver*.
7a.2 Use case continues with step 4.
7b. Mission cannot be created and no replacement missions are possible.
7b.1 *System* suggests notifying the *NationalCrisisCenter*.
7b.2 Use case continues with step 4.
8a. Mission failed.
8a.1 *System* informs *SuperObserver* and *Coordinator* about mission failure.
8a.2 Use case continues with step 4.

[1]Mission-specific information includes things such as the quantity of vehicles requested, special characteristics of the aid worker or vehicle, etc.

1.2.9 Execute Rescue Mission

Use Case 7: Execute Rescue Mission
Scope: Car Crash Crisis Management System
Primary Actor: FirstAidWorker
Secondary Actor: HospitalRS
Intention: The intention of the FirstAidWorker is to accept and then execute a rescue mission that involves transporting a victim to the most appropriate hospital.
Main Success Scenario:
FirstAidWorker is at the crisis location.
1. *FirstAidWorker* transmits injury information of victim to *System*.
Steps 2 and 3 are optional.
2. *FirstAidWorker* determines victim's identity and communicates it to *System*.
3. *System* requests victim's medical history information from all connected *HospitalResourceSystems*.
FirstAidWorker administers first aid procedures to victim.
4. *System* instructs *FirstAidWorker* to bring the victim to the most appropriate hospital.
5. *FirstAidWorker* notifies *System* that he is leaving the crisis site.
6. *FirstAidWorker* notifies *System* that he has dropped off the victim at the hospital.
7. *FirstAidWorker* informs *System* that he has completed his mission.
Use case ends in success.
Extensions:
4a. *HospitalResourceSystem* transmits victim's medical history information to *System*.
4a.1 *System* notifies *FirstAidWorker* of medical history of the victim relevant to his injury.
4a.2 Use case continues at step 4.

1.2.10 Execute Helicopter Transport Mission

Use Case 8: Execute Helicopter Transport Mission
Scope: Car Crash Crisis Management System
Primary Actor: Pilot
Secondary Actor: None
Intention: The intention of the Pilot is to accept and then execute a transport mission that involves transporting a CMSEmployee to and from a mission location.
Main Success Scenario: To be defined.

1.2.11 Execute Remove Obstacle Mission

Use Case 9: Execute Remove Obstacle Mission
Scope: Car Crash Crisis Management System
Primary Actor: TowTruckDriver
Secondary Actor: None
Intention: The intention of the TowTruckDriver is to accept and then execute a remove obstacle mission that involves removing a crashed car from a mission location.
Main Success Scenario: To be defined.

1.2.12 Authenticate User

Use Case 10: AuthenticateUser
Scope: Car Crash Crisis Management System
Primary Actor: None
Secondary Actor: CMSEmployee
Intention: The intention of the System is to authenticate the CMSEmployee to allow access.
Main Success Scenario:
1. *System* prompts *CMSEmployee* for login id and password.
2. *CMSEmployee* enters login id and password into *System*.
3. *System* validates the login information.
Use case ends in success.
Extensions:
2a. *CMSEmployee* cancels the authentication process. Use case ends in failure.
3a. *System* fails to authenticate the *CMSEmployee*.
3a.1 Use case continues at step 1.
3a.1a *CMSEmployee* performed three consecutive failed attempts.
3a.1a.1 Use case ends in failure.

2 Application of AORE4PF

In this section, we show the results of the application of AORE4PF on the CMS case study. Figure 2 shows the steps of our method and each of the following subsections presents the results of one of the method steps. Note that we only used the information given in the first section of this document as external input.

2.1 Classify Requirements

2.1.1 Preliminary Functional Requirements

We identified the following functional requirements:

R1 Coordinators can capture witness reports

R2 System recommends missions to coordinators

R3 Coordinators can assign or request resources for the missions

R4 System shall register resource notifications

R5 Coordinators shall be informed about the mission status

R6 Provide relevant information to resources

R7 Store final resource report

R8 Inform coordinator about new crisis information

R9 Coordinator can close file for crisis

R10 Authentication with system

R11 Provide access to camera surveillance to coordinator

R12 Send/process crisis-specific checklist to super observer

R13 Super observers can create missions and add information

R14 Super observers are informed about mission success

R15 Provide up-to date information to rescue resources

R16 Monitor weather and terrain conditions at crisis site

R17 Request resources status during crisis

R18 Record data upon receipt

R19 Compute statistical information

R20 Encrypt all communications with AES 128-Bit

R21 Use alternate communication channels if needed

The distinction of these requirements into base requirements and aspect requirements is shown in Table 1.

2.1.2 Preliminary Quality Requirements

And the following software qualities:

Q1 Performance
- Scalability (1000 witness calling, 100 crises, 200 missions per crisis, 20000 rescue resources)
- Response time (500 ms for all requests)

Q2 Maintenance
- interruptable maintenance
- only 2h of maintenance in 30 days

Q3 Safety
- monitor criminal activity
- safety of rescue personal
- Reliability (recover in 30 sec, probability of failure 0,001%)

Q4 Security
- Authentication and authorization
- Confidentiality of communications
- Availability of communication
- Availability of system (24/7) with max 2h per 30 days of maintenance

2.1.3 Preliminary Cross-Cut Relation

The crosses in normal font (X) in Table 1 show the preliminary cross-cut relations between the base requirements, aspect requirements, and quality requirements.

2.1.4 Effort Spent

Effort spent in this step: 5h including 1h reading the document and 1h for preparation of the documentation.

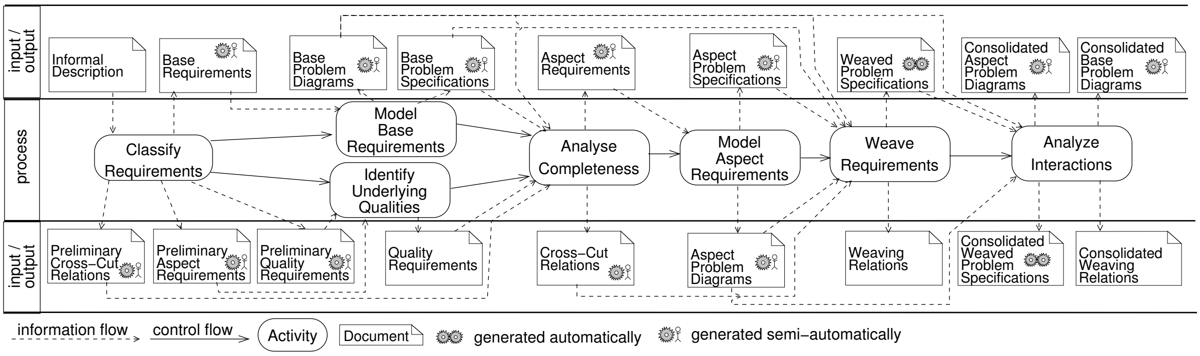


Figure 2: The AORE4PF method

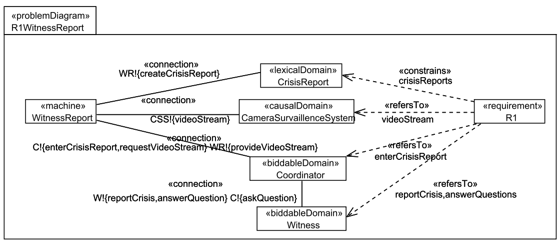
Table 1: Cross-cut relation

	R5,R8	R6,R15	R18	R10	R20	R21	Q1	Q2	Q3	Q4Au	Q4Co	Q4Av
R1,R11			X	X	**X**[4]	**X**[4]	**X**[4]	**X**[4]	**X**[4]	X	**X**[4]	**X**[4]
R2			X	X	**X**[4]		**X**[4]	**X**[4]	**X**[4]	X	**X**[4]	
R3			X	X	**X**[4]		**X**[4]	**X**[4]	**X**[4]	X	**X**[4]	
R4,R7	X	X	X	X	**X**[4]	**X**[4]	**X**[4]		**X**[4]	X	**X**[4]	**X**[4]
R9			X	X	**X**[4]		**X**[4]			X	**X**[4]	
R12,R13			X	X	**X**[4]	**X**[4]	**X**[4]		**X**[4]	X	**X**[4]	**X**[4]
R14			X	X	**X**[4]	**X**[4]	**X**[4]		**X**[4]	X	**X**[4]	**X**[4]
R16	X	X	X		**X**[4]	**X**[4]	**X**[4]		**X**[4]		**X**[4]	**X**[4]
R17	X	X	X	**X**[4]	**X**[4]	**X**[4]	**X**[4]		**X**[4]	**X**[4]	**X**[4]	**X**[4]
R19												
Q1												
Q2			**X**[3]									
Q3	**X**[3]	**X**[3]	**X**[3]			**X**[3]						
Q4Au				X								
Q4Co				**X**[3]	X							
Q4Av						X						

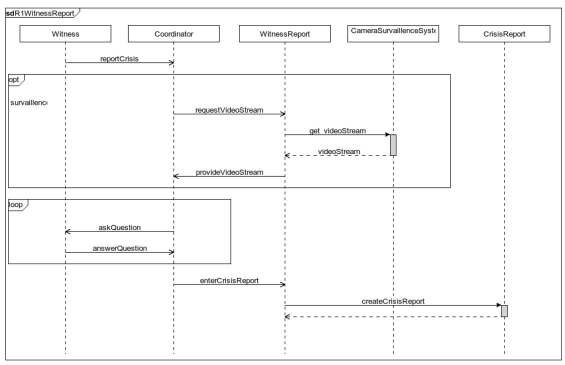
2.2 Model Base Requirements

As already visible in Table 1, we merged several functional requirements, because the problems they describe are similar or heavily depend on each other. You find the problem diagrams and problem specifications as sequence diagrams for the base requirements in the following Figures:

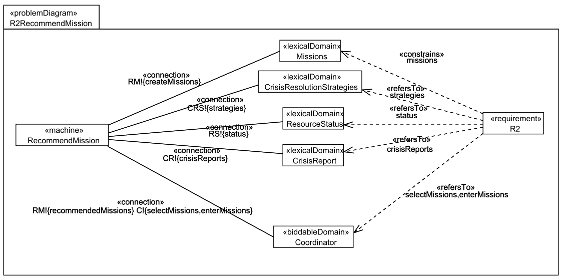
2.2.1 R1 Problem Diagram



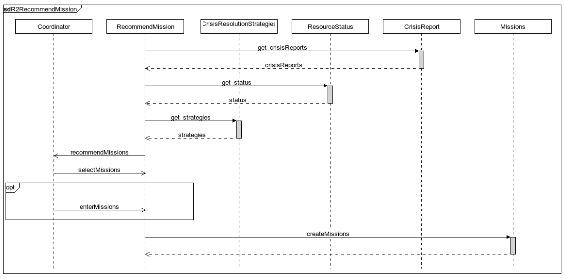
2.2.2 R1 Problem Specification



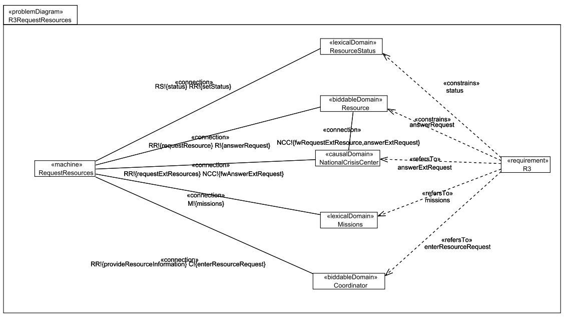
2.2.3 R2 Problem Diagram



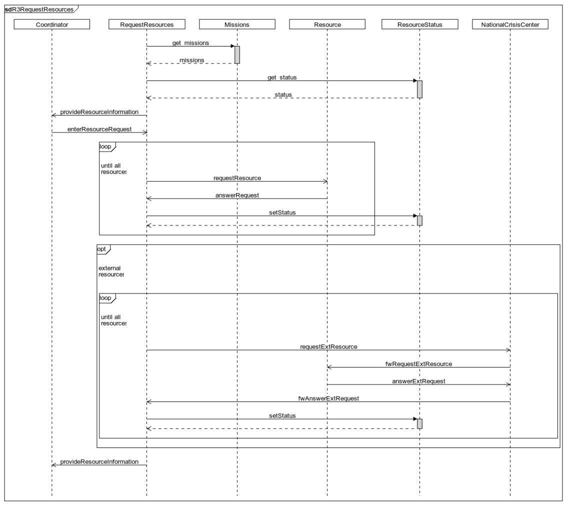
2.2.4 R2 Problem Specification



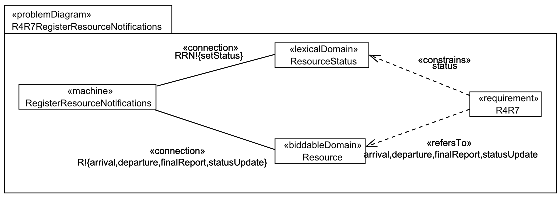
2.2.5 R3 Problem Diagram



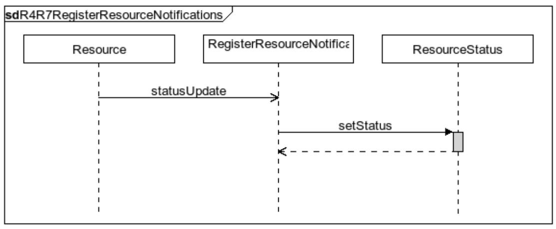
2.2.6 R3 Problem Specification



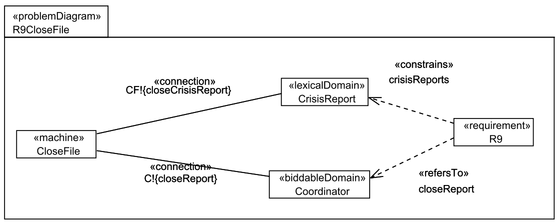
2.2.7 R4R7 Problem Diagram



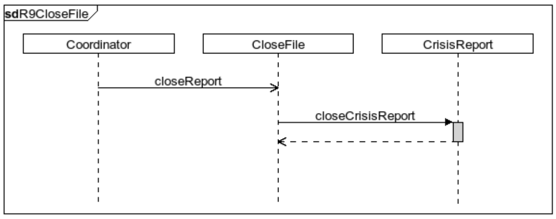
2.2.8 R4R7 Problem Specification



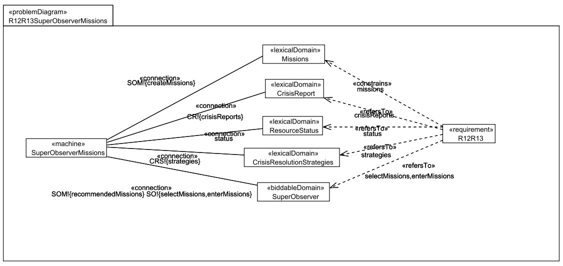
2.2.9 R9 Problem Diagram



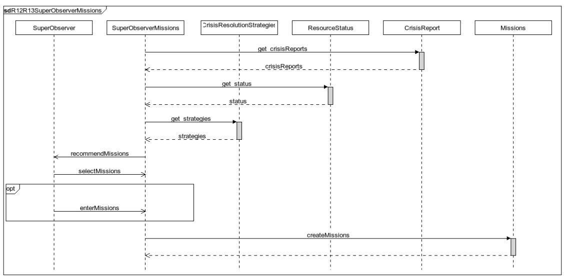
2.2.10 R9 Problem Specification



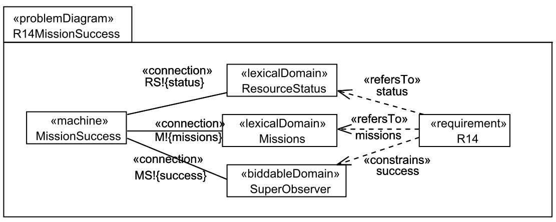
2.2.11 R12R13 Problem Diagram



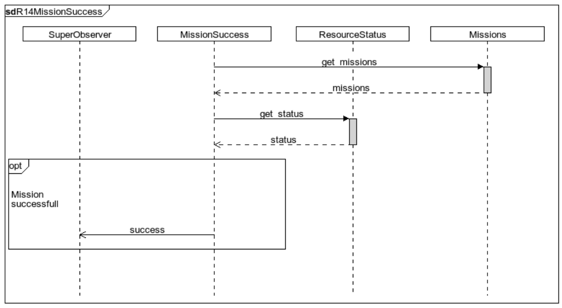
2.2.12 R1213 Problem Specification



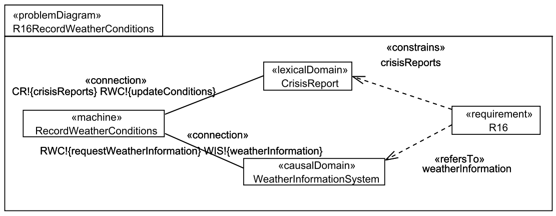
2.2.13 R14 Problem Diagram



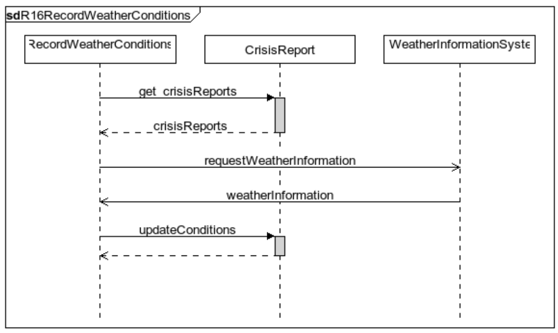
2.2.14 R14 Problem Specification



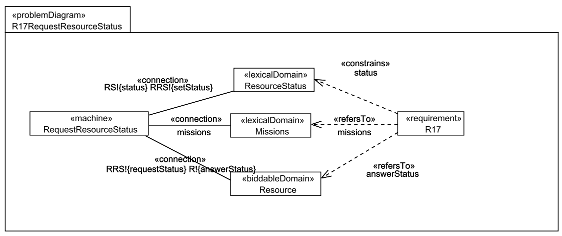
2.2.15 R16 Problem Diagram



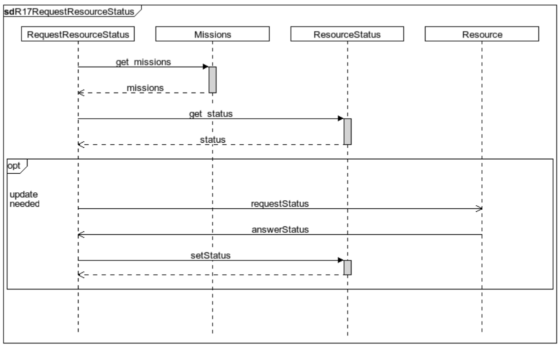
2.2.16 R16 Problem Specification



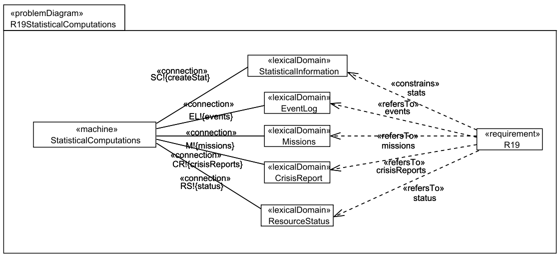
2.2.17 R17 Problem Diagram



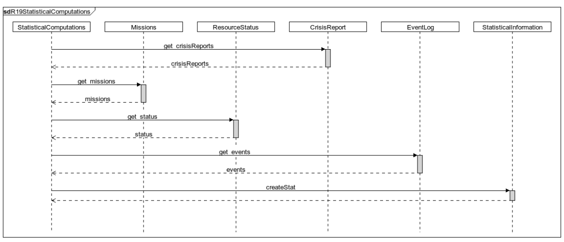
2.2.18 R17 Problem Specification



2.2.19 R19 Problem Diagram



2.2.20 R19 Problem Specification



2.2.21 Effort Spent

The effort for modeling the problem and sequence diagrams for the base requirements is summarized in the following table. In total, we spent in this step an effort of 6h 3min including documentation and tool setup.

	Problem Diagram	Specification
R1	20 min	20 min
R2	12 min	11 min
R3	16 min	14 min
R4R7	10 min	3 min
R9	17 min	2 min
R12R13	11 min	6 min
R14	9 min	4 min
R16	15 min	4 min
R17	11 min	4 min
R19	12 min	5 min
Sum	2h 13 min	1h 13 min

2.3 Identify Underlying Qualities

For each aspect requirement, we investigated whether it was introduced to address a specific software quality. The identified relations between quality requirements and aspect requirements are visualized in Table 1 using bold crosses annotated with a three ($\mathbf{X}^3$). Effort spent: 45min

2.4 Analyze Completeness

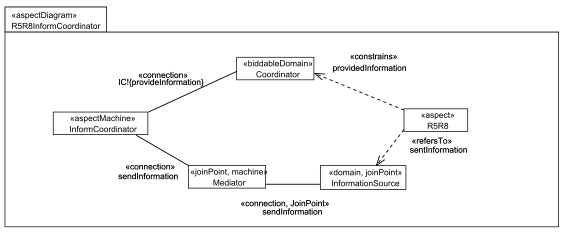
During the analysis of the completeness of the crosscut relation, we checked for each base requirement whether a specific software quality, such as authentication, confidentiality, or safety has to be considered. If this was the case, then we added a bold cross annotated with a four ($\mathbf{X}^4$) into the corresponding cell of Table 1.

Effort spent: 1h 15min

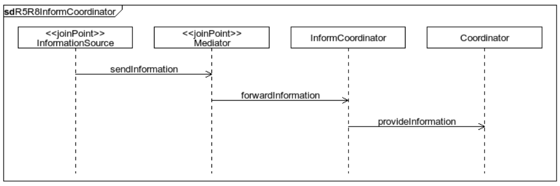
2.5 Model Aspect Requirements

We modeled the following aspect requirements:

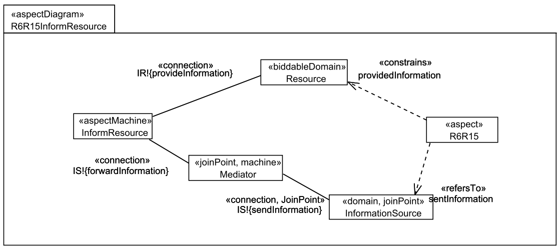
2.5.1 R5R8 Aspect Diagram



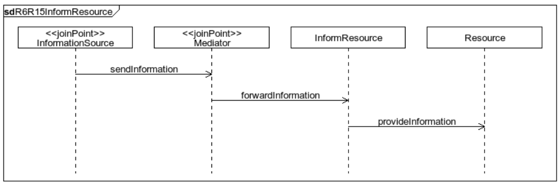
2.5.2 R5R8 Aspect Specification



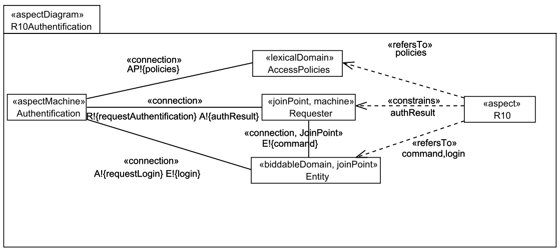
2.5.3 R6R15 Aspect Diagram



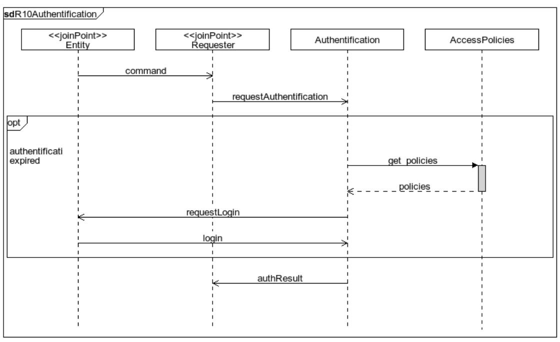
2.5.4 R6R15 Aspect Specification



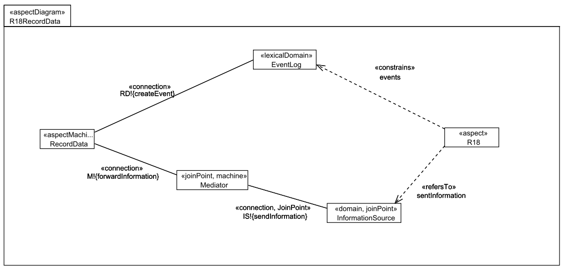
2.5.5 R10 Aspect Diagram



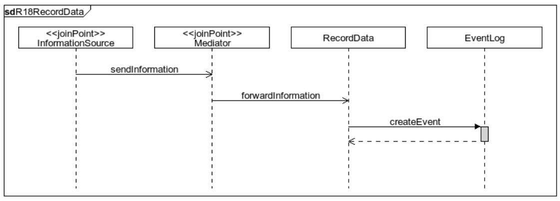
2.5.6 R10 Aspect Specification



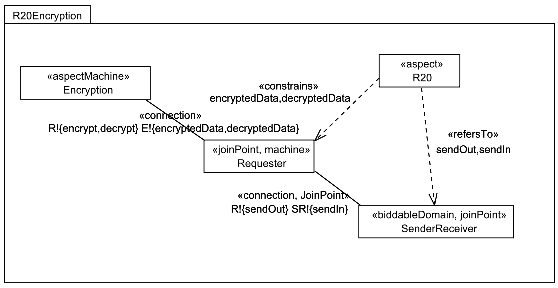
2.5.7 R18 Aspect Diagram



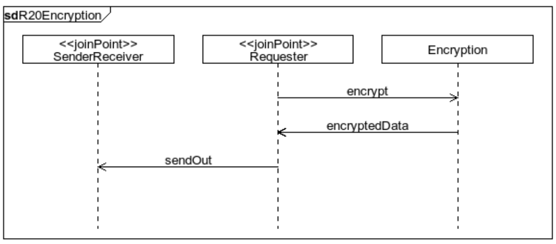
2.5.8 R18 Aspect Specification

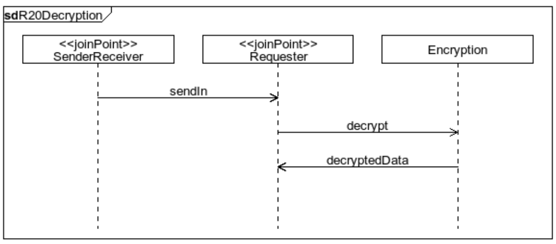


2.5.9 R20 Aspect Diagram



2.5.10 R20 Aspect Specification





2.5.11 Effort Spent

Effort spent in this step: 2h 51min including documentation and tool setup.

	Aspect Diagram	Specification
R5R8	13 min	5 min
R6R15	11 min	6 min
R18	10 min	6 min
R10	15 min	12 min
R20	11 min	7 min
R21	- min	- min
Sum	1h 0 min	36 min

2.6 Weave Requirements

On the basis of the cross-cut relation shown in Figure 1, we define for each base requirement the weaving relation and build the corresponding weaved problem specification.

2.6.1 R1 RecordWitnessreport

We integrate the aspect requirements R10, R18, and R20 into the base requirement R1 as specified in Table 2. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 3.

2.6.2 R2 RecommendMissions

We integrate the aspect requirements R10, R18, and R20 into the base requirement R2 as specified in Table 3. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 4.

2.6.3 R3 RequestResources

We integrate the aspect requirements R10, R18, and R20 into the base requirement R3 as specified in Table 4. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 5.

2.6.4 R4R7 RegisterResourceNotifications

We integrate the aspect requirements R5R8, R6R15, R18, R10, and R20 into the base requirement R4R7 as specified in Table 5. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 6.

2.6.5 R9 CloseFile

We integrate the aspect requirements R18, R10, and R20 into the base requirement R9 as specified in Table 6. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 7.

2.6.6 R1213 SuperObserverMission

We integrate the aspect requirements R18, R10, and R20 into the base requirement R12R13 as specified in Table 7. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 8.

2.6.7 R14 MissionSuccess

We integrate the aspect requirements R18, R10, and R20 into the base requirement R14 as specified in Table 6. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 7.

2.6.8 R16 RecordWeatherConditions

We integrate the aspect requirements R5R8, R6R15, R18, and R20 into the base requirement R16 as specified in Table 9. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 10.

2.6.9 R17 RequestResourceStatus

We integrate the aspect requirements R5R8, R6R15, R18, R10, and R20 into the base requirement R16 as specified in Table 10. From this relation, we built the weaved problem specification as sequence diagram shown in Figure 11.

Message	**Key**	**AspectSequenceDiagram**
requestVideostream	After	R10Authenfification[Entity/Coordinator, Requester/WitnessReport, command/requestVideoStream]
enterCrisisReport	After	R10Authenfification[Entity/Coordinator, Requester/WitnessReport, command/enterCrisisReport]
requestVideostream	After	R18RecordData[InformationSource/Coordinator, Mediator/WitnessReport, command/requestVideoStream]
enterCrisisReport	After	R18RecordData[InformationSource/Coordinator, Mediator/WitnessReport, command/enterCrisisReport]
requestVideostream	After	R20Decryption[SenderReceiver/Coordinator, Requester/WitnessReport, sendIn/requestVideoStream]
enterCrisisReport	After	R20Decryption[SenderReceiver/Coordinator, Requester/WitnessReport, sendIn/enterCrisisReport]
videoStream	After	R20Decryption[SenderReceiver/CameraSurvaillenceSystem, Requester/WitnessReport, sendIn/videoStream]
provideVideoStream	Before	R20Encryption[SenderReceiver/Coordinator, Requester/WitnessReport, sendOut/provideVideoStream]
get_videoStream	Before	R20Encryption[SenderReceiver/CameraSurvaillenceSystem, Requester/WitnessReport, sendOut/get_videoStream]

Table 2: Weaving relation for R1

Message	**Key**	**AspectSequenceDiagram**
recommendMissions	Before	R20Encryption[SenderReceiver/Coordinator, Requester/RecommendMission, sendOut/recommendMissions]
selectMissions	After	R20Decryption[SenderReceiver/Coordinator, Requester/RecommendMission, sendIn/selectMissions]
enterMissions	After	R20Decryption[SenderReceiver/Coordinator, Requester/RecommendMission, sendIn/enterMissions]
selectMissions	After	R10Authenfification[Entity/Coordinator, Requester/RecommendMission, command/selectMission]
enterMissions	After	R10Authenfification[Entity/Coordinator, Requester/RecommendMission, command/enterMissions]
selectMissions	After	R18RecordData[InformationSource/Coordinator, Mediator/RecommendMission, command/selectMissions]
enterMissions	After	R18RecordData[InformationSource/Coordinator, Mediator/RecommendMission, command/enterMissions]

Table 3: Weaving relation for R2

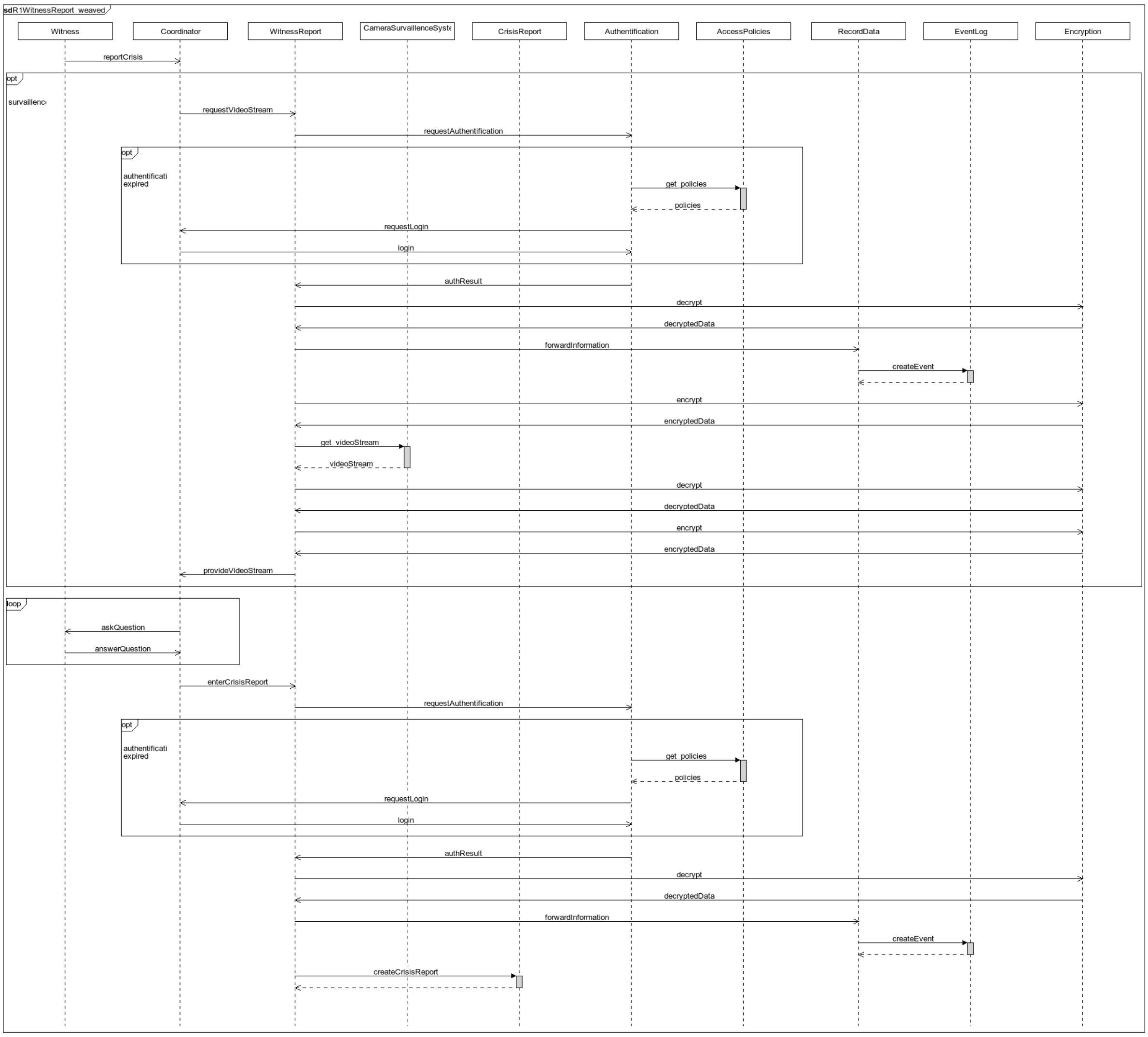


Figure 3: Weaved problem specification for R1

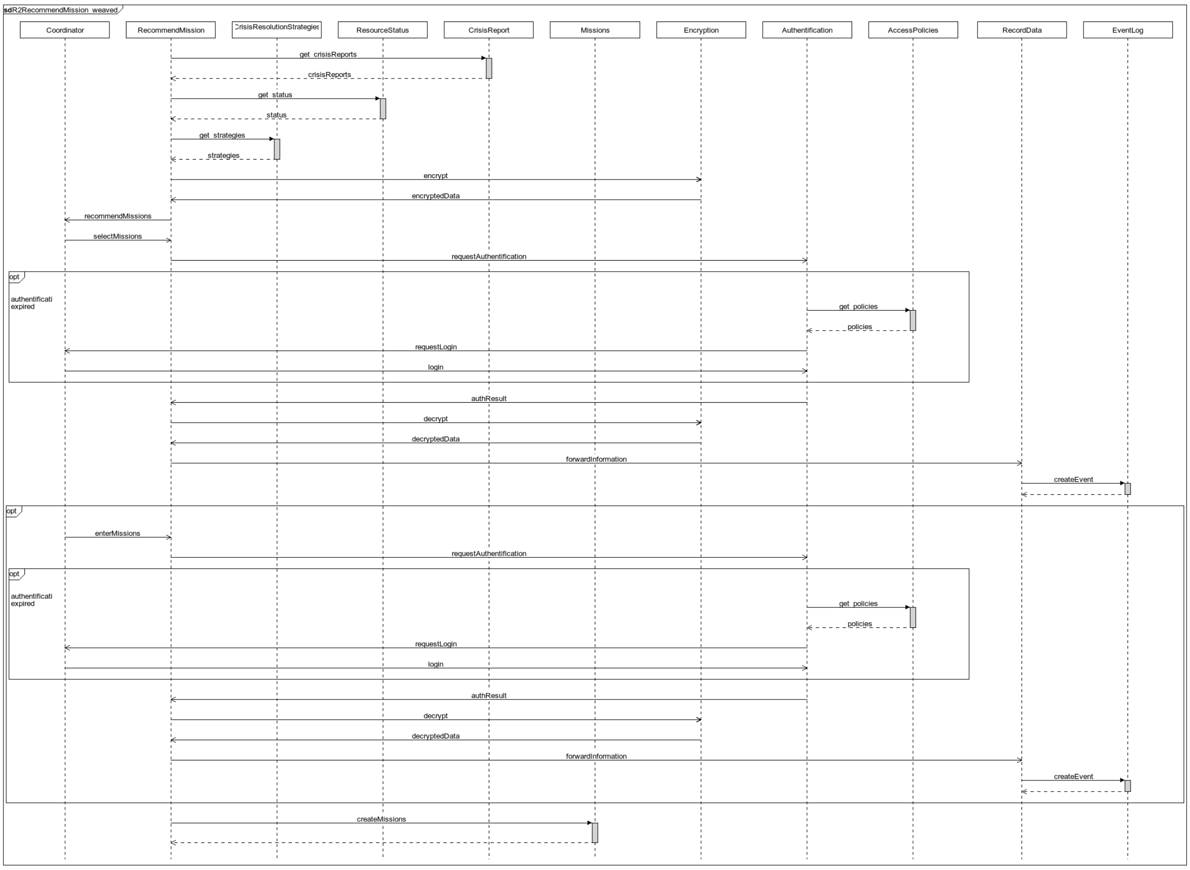


Figure 4: Weaved problem specification for R2

Message	Key	AspectSequenceDiagram
provideResourceInformation	Before	R20Encryption[SenderReceiver/Coordinator, Requester/RequestResources, sendOut/provideResourceInformation]
enterResourceRequest	After	R10Authenfification[Entity/Coordinator, Requester/RequestResources, command/enterResourceRequest]
enterResourceRequest	After	R20Decryption[SenderReceiver/Coordinator, Requester/RequestResources, sendIn/enterResourceRequest]
enterResourceRequest	After	R18RecordData[InformationSource/Coordinator, Mediator/RequestResources, command/enterResourceRequest]
requestResource	Before	R20Encryption[SenderReceiver/Resource, Requester/RequestResources, sendOut/requestResource]
answerRequest	After	R10Authenfification[Entity/Resource, Requester/RequestResources, command/answerRequest]
answerRequest	After	R20Decryption[SenderReceiver/Resource, Requester/RequestResources, sendIn/answerRequest]
answerRequest	After	R18RecordData[InformationSource/Resource, Mediator/RequestResources, command/answerRequest]
requestExtResource	Before	R20Encryption[SenderReceiver/NationalCrisisCenter, Requester/RequestResources, sendOut/requestExtResource]
fwAnswerRequest	After	R10Authenfification[Entity/NationalCrisisCenter, Requester/RequestResources, command/fwAnswerRequest]
fwAnswerRequest	After	R20Decryption[SenderReceiver/NationalCrisisCenter, Requester/RequestResources, sendIn/fwAnswerRequest]
fwAnswerRequest	After	R18RecordData[InformationSource/NationalCrisisCenter, Mediator/RequestResources, command/fwAnswerRequest]

Table 4: Weaving relation for R3

Message	Key	AspectSequenceDiagram
statusUpdate	After	R10Authenfification[Entity/Resource, Requester/RegisterResourceNotifications, command/statusUpdate]
statusUpdate	After	R20Decryption[SenderReceiver/Resource, Requester/RegisterResourceNotifications, sendIn/statusUpdate]
statusUpdate	After	R18RecordData[InformationSource/Resource, Mediator/RegisterResourceNotifications, command/statusUpdate]
statusUpdate	After	R5R8InformCoordinator[InformationSource/Resource, Mediator/RegisterResourceNotifications, sendInformation/statusUpdate]
statusUpdate	After	R6R15InformResource[InformationSource/Resource, Mediator/RegisterResourceNotifications, sendInformation/statusUpdate]

Table 5: Weaving relation for R4R7

Message	Key	AspectSequenceDiagram
closeReport	After	R10Authenfification[Entity/Coordinator, Requester/CloseFile, command/closeReport]
closeReport	After	R20Decryption[SenderReceiver/Coordinator, Requester/CloseFile, sendIn/closeReport]
closeReport	After	R18RecordData[InformationSource/Coordinator, Mediator/CloseFile, command/closeReport]

Table 6: Weaving relation for R9

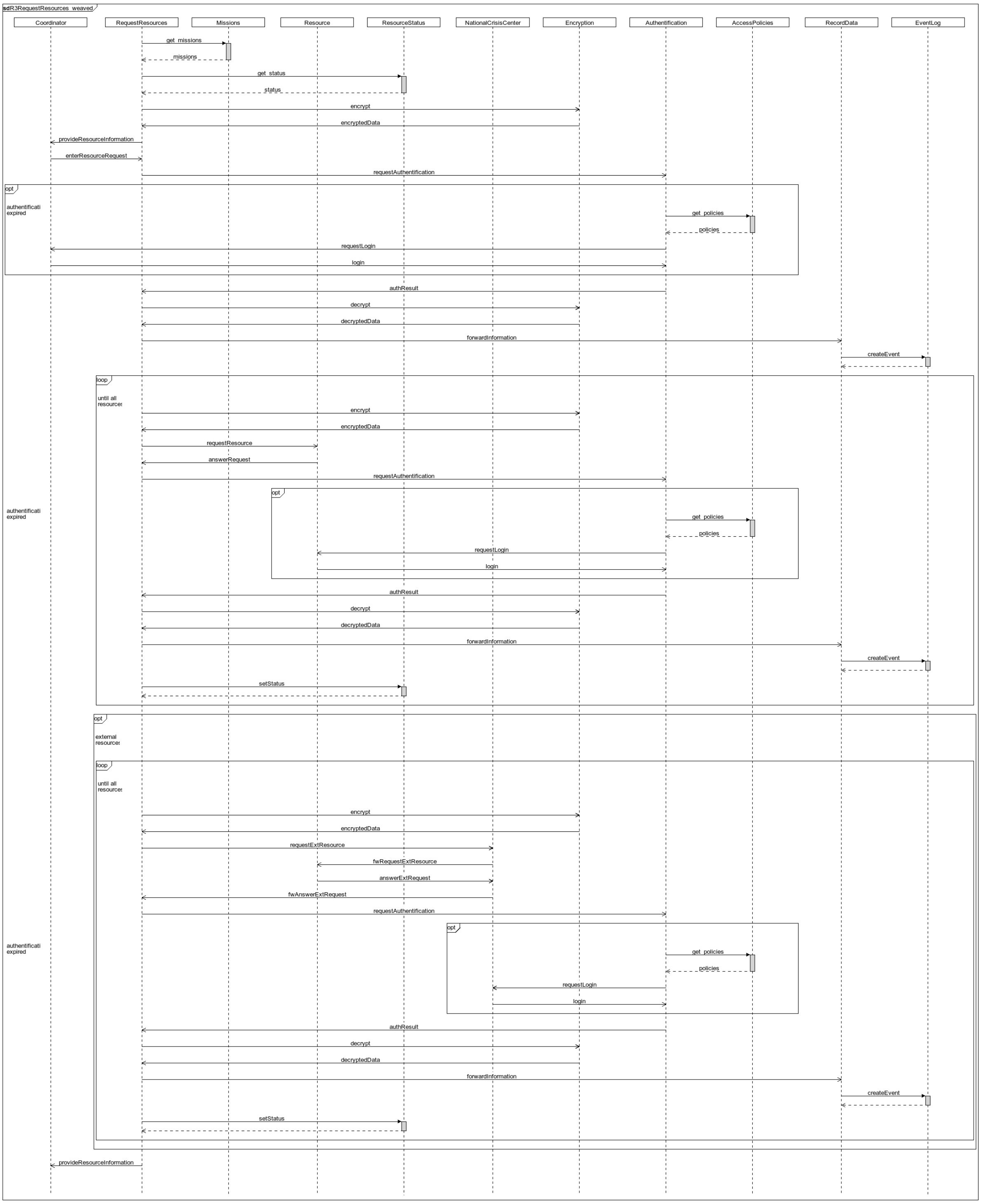


Figure 5: Weaved problem specification for R3

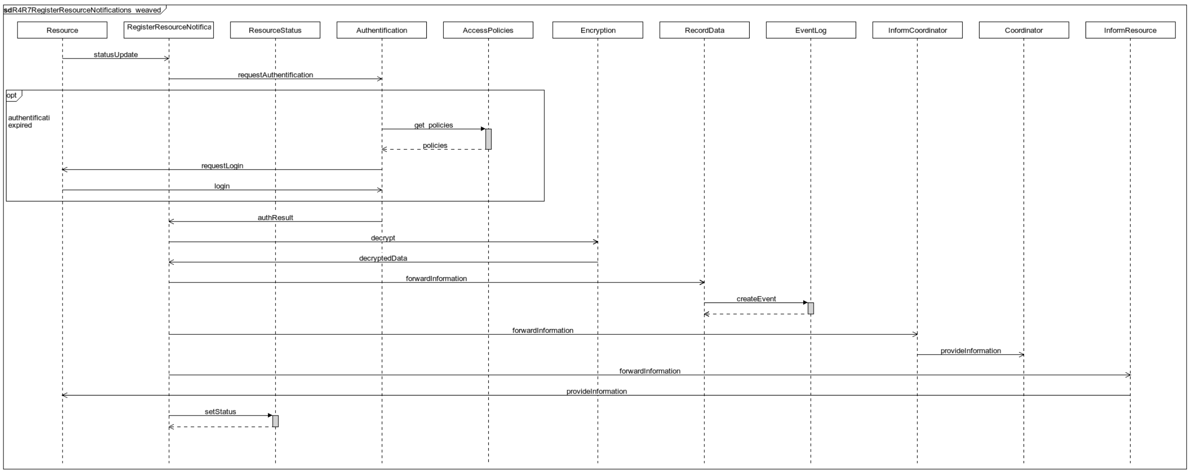


Figure 6: Weaved problem specification for R4R7

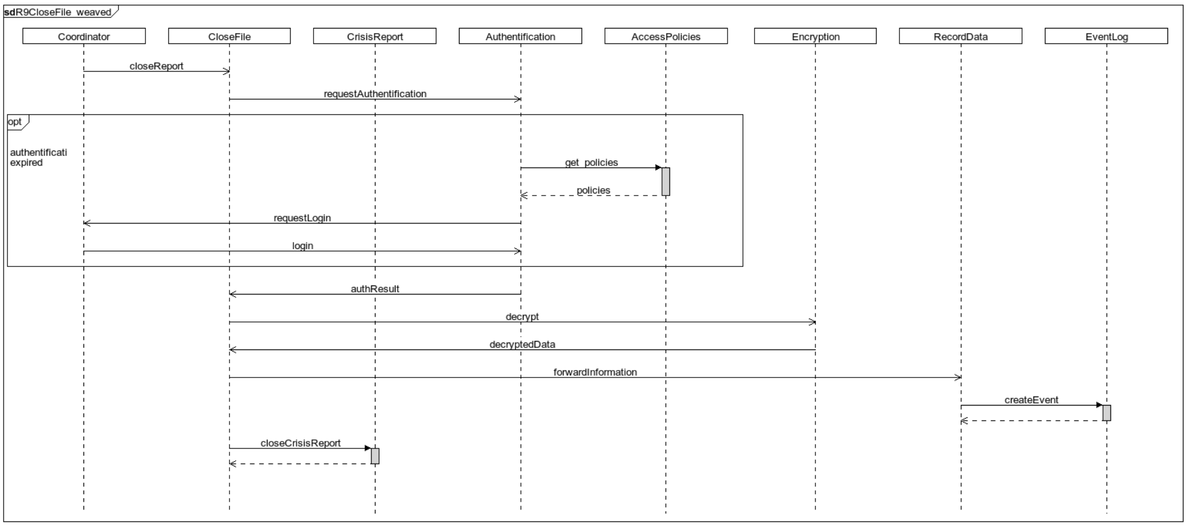


Figure 7: Weaved problem specification for R9

Message	**Key**	**AspectSequenceDiagram**
recommendMissions	Before	R20Encryption[SenderReceiver/SuperObserver, Requester/SuperObserverMissions, sendOut/recommendMissions]
selectMissions	After	R10Authenfification[Entity/SuperObserver, Requester/SuperObserverMissions, command/selectMissions]
selectMissions	After	R20Decryption[SenderReceiver/SuperObserver, Requester/SuperObserverMissions, sendIn/selectMissions]
selectMissions	After	R18RecordData[InformationSource/SuperObserver, Mediator/SuperObserverMissions, command/selectMissions]
enterMissions	After	R10Authenfification[Entity/SuperObserver, Requester/SuperObserverMissions, command/enterMissions]
enterMissions	After	R20Decryption[SenderReceiver/SuperObserver, Requester/SuperObserverMissions, sendIn/enterMissions]
enterMissions	After	R18RecordData[InformationSource/SuperObserver, Mediator/SuperObserverMissions, command/enterMissions]

Table 7: Weaving relation for R1213

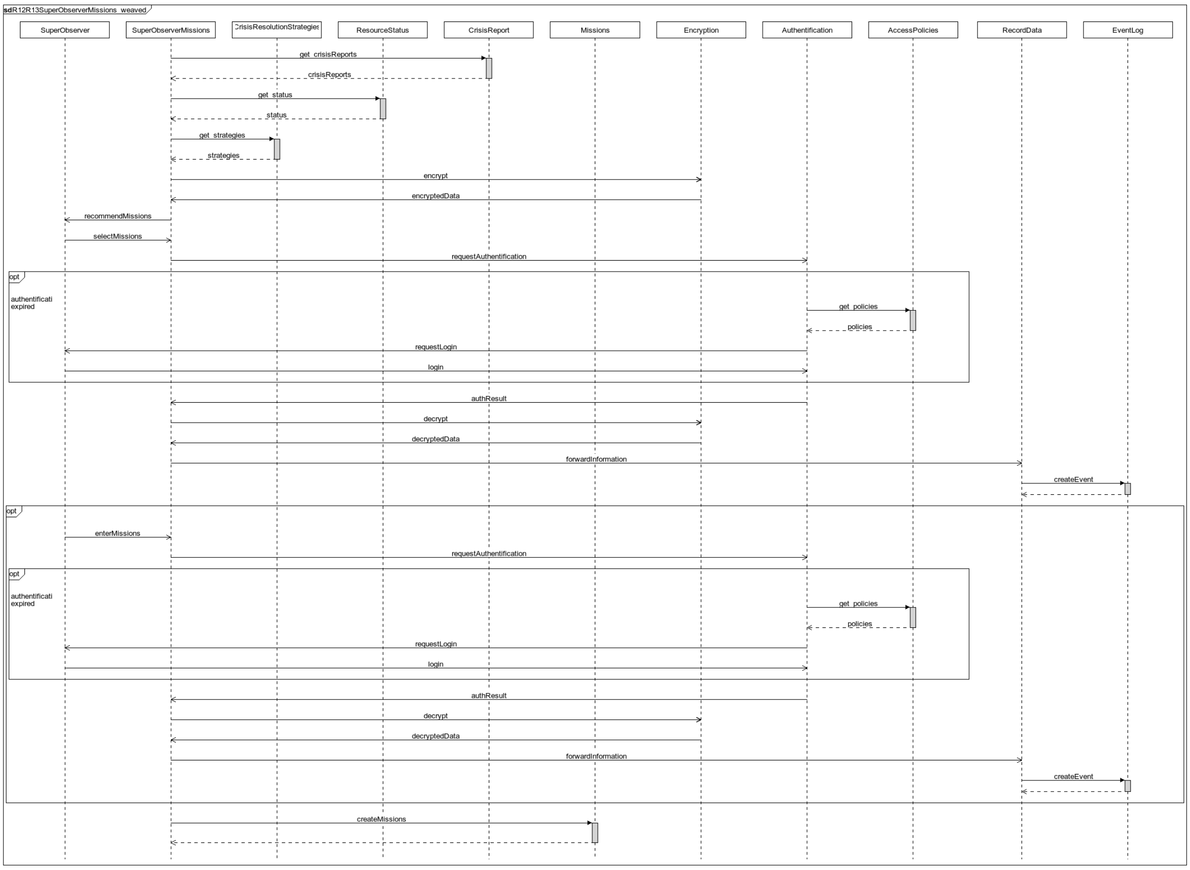


Figure 8: Weaved problem specification for R12R13

Message	Key	AspectSequenceDiagram
success	Before	R20Encryption[SenderReceiver/SuperObserver, Requester/MissionSuccess, sendOut/success]
success	After	R18RecordData[InformationSource/SuperObserver, Mediator/SuperObserver, command/computesuccess]

Table 8: Weaving relation for R14

Message	Key	AspectSequenceDiagram
requestWeatherInformation	Before	R20Encryption[SenderReceiver/WeatherInformationSystem, Requester/RecordWeatherConditions, sendOut/requestWeatherInformation]
weatherInformation	After	R20Decryption[SenderReceiver/WeatherInformationSystem, Requester/RecordWeatherConditions, sendIn/weatherInformation]
weatherInformation	After	R18RecordData[InformationSource/WeatherInformationSystem, Mediator/RecordWeatherConditions, command/weatherInformation]
weatherInformation	After	R5R8InformCoordinator[InformationSource/WeatherInformationSystem, Mediator/RecordWeatherConditions, sendInformation/weatherInformation]
weatherInformation	After	R6R15InformResource[InformationSource/WeatherInformationSystem, Mediator/RecordWeatherConditions, sendInformation/weatherInformation]

Table 9: Weaving relation for R16

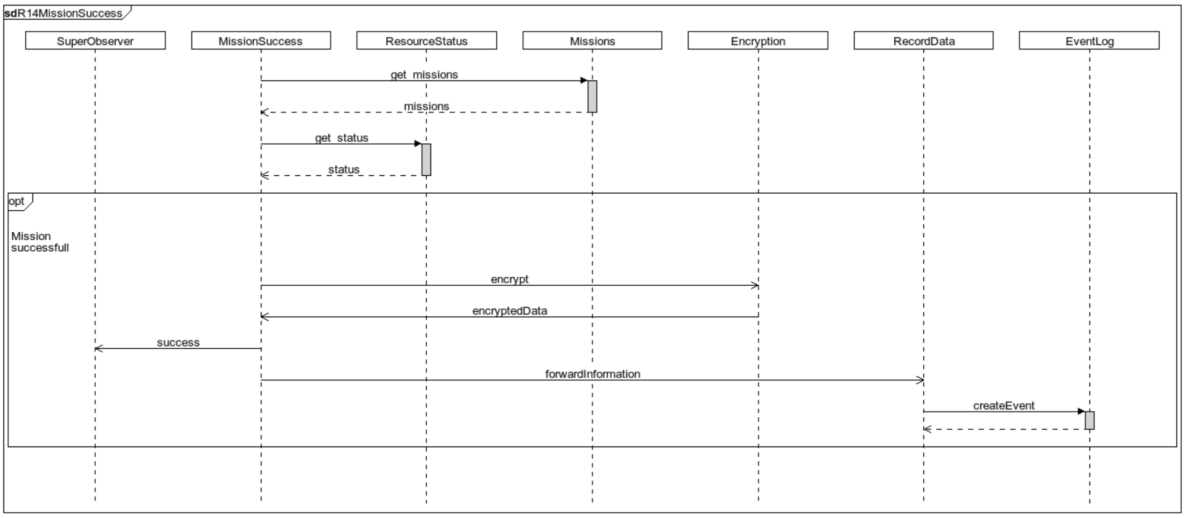


Figure 9: Weaved problem specification for R14

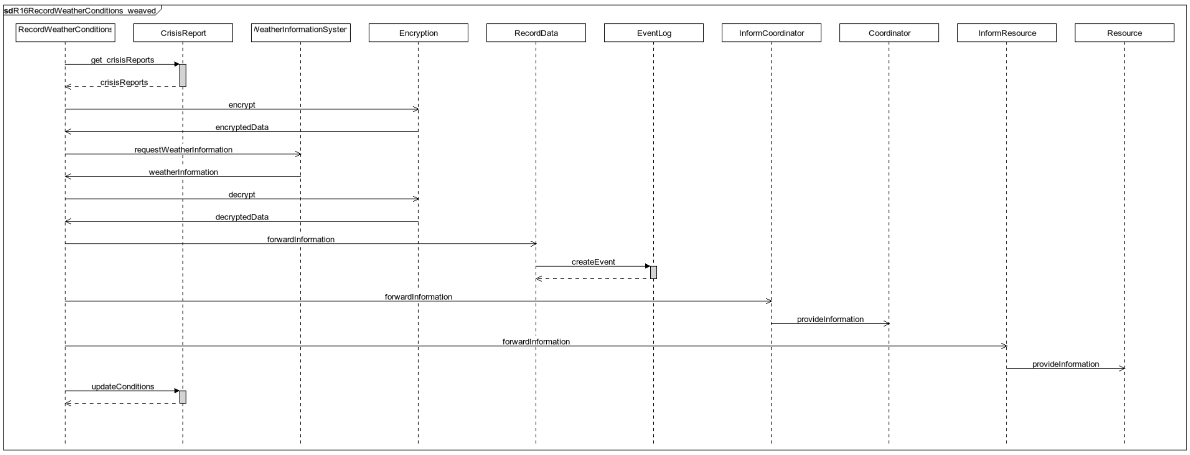


Figure 10: Weaved problem specification for R16

Message	**Key**	**AspectSequenceDiagram**
requestStatus	Before	R20Encryption[SenderReceiver/Resource, Requester/RequestResourceStatus, sendOut/requestStatus]
answerStatus	After	R10Authenfification[Entity/Resource, Requester/RequestResourceStatus, command/answerStatus]
answerStatus	After	R20Decryption[SenderReceiver/Resource, Requester/RequestResourceStatus, sendIn/answerStatus]
answerStatus	After	R18RecordData[InformationSource/Resource, Mediator/RequestResourceStatus, command/answerStatus]
answerStatus	After	R5R8InformCoordinator[InformationSource/Resource, Mediator/RequestResourceStatus, sendInformation/answerStatus]
answerStatus	After	R6R15InformResource[InformationSource/Resource, Mediator/RequestResourceStatus, sendInformation/answerStatus]

Table 10: Weaving relation for R17

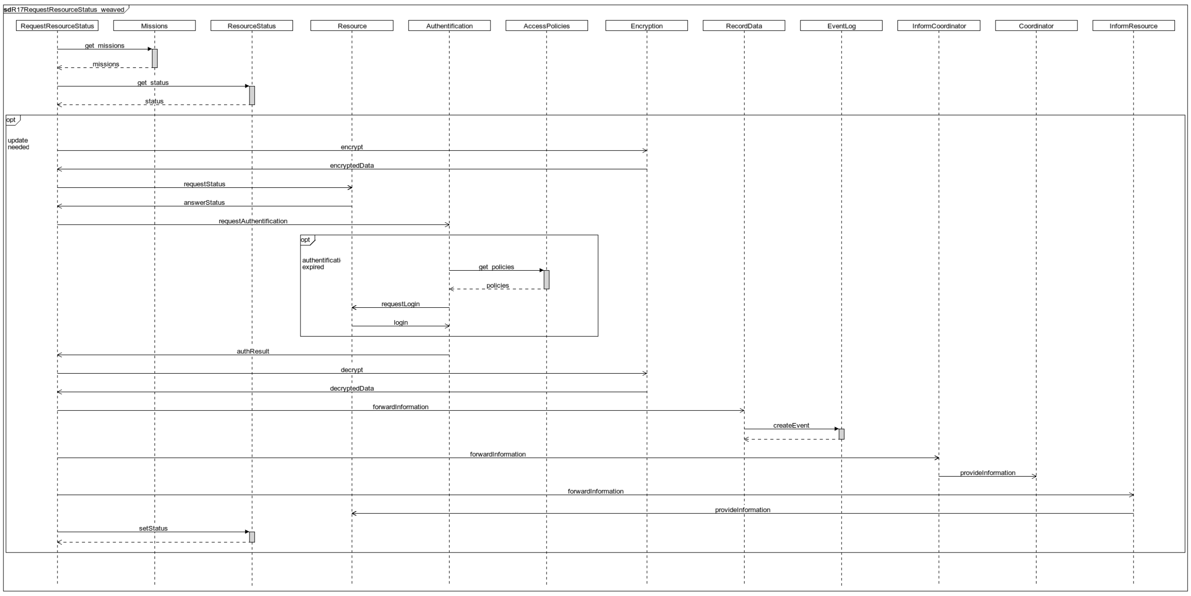


Figure 11: Weaved problem specification for R17

2.6.10 Effort Spent

The effort spent for step 6 is presented in the following table:

	Duration
R1	51 min
R2	29 min
R3	31 min
R4R7	22 min
R9	10 min
R12R13	10 min
R14	8 min
R16	15 min
R17	12 min
R19	- min
Doc	45 min
Sum	3h 53 min

2.7 Interaction Analysis

For the analysis of functional interactions, we computed the interaction table shown in Table 11.

We used the following life cycle for the interaction analysis:

$$
\begin{aligned}
LC_{Coordinator} =& (R1;R2;R3;R9)^* \\
LC_{Resource} =& R4R7^* \parallel R17^* \\
LC_{SuperObserver} =& (R12R13;R14)^* \\
LC_{CrisisSystem} =& (\|_{i=1}^{n} LC_{Coordinator}) \parallel (\|_{i=1}^{n} LC_{Resource}) \\
& \parallel (\|_{i=1}^{n} LC_{SuperObserver}) \parallel R16 \parallel R19
\end{aligned}
$$

2.7.1 Interactions on CrisisReport

Requirements R1, R9, and R16 all constrain the crisisReports of the domain CrisisReport and are hence candidates for pairwise interactions. But R1 and R9 do not have to be satisfied in parallel and hence only R1 and R16, and R9 and R16 are interaction candidates. After closer examination, we assume that a proper implementation of the update of weather and terrain information does not effect the consistence of the crisisReports.

2.7.2 Interactions on Missions

Requirements R2, and R12R13 both constrain the missions of the domain Missions and possibly interact with each other. For example, coordinators and super observers are able to define contradicting missions. It has to be discussed how such situations are handled.

2.7.3 Interactions on Resource

Requirements R3, and R6R15 both constrain Resources to behave in some way. As resources are often human resources, we have to carefully evaluate how these humans react under pressure (in the case of a crisis) and if they are then able to behave in a way the requirements want them to.

2.7.4 Interactions on ResourceStatus

Requirements R3, R4R7, and R17 all constrain the status of the domain ResourceStatus and are hence candidates for pairwise interactions. After closer examination, we see that R4R7 and R17 do not have to be considered as interacting, because the status is only updated in R17 if there were no incoming notifications due to R4R7 for a specific duration. Furthermore, R3 only updates only the status if the resource is free, and R4R7 and R17 only if the resource is in duty. Thus we do not assume any interactions on the domain ResourceStatus.

2.7.5 Effort Spent

Performing and documenting this step took 1h 34min.

REFERENCES

Kienzle, J., Guelfi, N., and Mustafiz, S. (2010). Transactions on aspect-oriented software development vii. chapter Crisis Management Systems: A Case Study for Aspect-oriented Modeling, pages 1–22. Springer-Verlag, Berlin, Heidelberg.

	R1	R2	R3	R4R7	R9	R12R13	R14	R16	R17	R19	R5R8	R6R15	R18	R10	R20
CrisisReport	X				X			X							
Missions		X				X									
Resource			X									X			
ResourceStatus			X	X					X						
SuperObserver							X								
StatisticalInformation							X								
Coordinator											X				
EventLog													X		

Table 11: Interaction Table