Requirements Elicitation

Homework Session 2

2007-09-14

Requirement

Client acceptability of System

1. Features system must have
2. Constraints system must satisfy

Now what does client understand about system?

Scenarios, Use cases (requirement elicitation) for this purpose

Requirement Elicitation

- Describe purpose of System

- Identify users/system that act with system (identify actors)
- Examples of systems in use (identify scenarios)
- All possible cases (identify use cases)
- (refine use cases)
- (identify non-functional requirements)

Identifying Actors

- Actor: External entities interacting with System
- Also refered to as Role Abstraction.

  Different Roles = Different Actors

- Actors are external (out of system boundary)
- How to define system boundary

Identifying Actors

- Which user groups supported by System ?
- Which user groups use System's main functions ?
- Which user group use System's secondary functions ? (maintenance, administration)
- Which external hardware/software interacts with system ?

Identifying Scenarios

- An informal description of single system usage from view point of a single actor.

  (Specific instance of system usage, Doesn't describe all possible situations)

Identifying Scenarios

- Which tasks actor want to perform ?
- Which information actors needs to access ? Who creates/deletes/modifies that information ?
- Which external events actors need to inform to System ?
- Which events System needs to inform to Actors ?

Example Scenario

Scenario name	warehouseOnFire
Participating actor instances	bob, alice:FieldOfficer john:Dispatcher
Flow of events	1. Bob, driving down main street in his patrol car, notices smoke coming out of a warehouse. His partner, Alice, activates the "Report Emergency" function from her FRIEND laptop. 2. Alice enters the address of the building, a brief description of its location (i.e., northwest corner), and an emergency level. In addition to a fire unit, she requests several paramedic units on the scene, given that the area appears to be relatively busy. She confirms her input and waits for an acknowledgment. 3. John, the Dispatcher, is alerted to the emergency by a beep of his workstation. He reviews the information submitted by Alice and acknowledges the report. He allocates a fire unit and two paramedic units to the Incident site and sends their estimated arrival time (ETA) to Alice. 4. Alice receives the acknowledgment and the ETA.

Figure 4-6 warehouseOnFire scenario for the ReportEmergency use case.

Identifying Use cases

- Use cases abstract all possible scenarios for a given functionality
- A use case is initiated by an actor and describes all interrelated interactions (flow of events) resulting from such initiations.

Clearly states which part Actor and which part System is supposed to do

Use case name	`ReportEmergency`
Participating actors	Initiated by `FieldOfficer` Communicates with `Dispatcher`
Flow of events	1. The `FieldOfficer` activates the "Report Emergency" function of her terminal. 2. `FRIEND` responds by presenting a form to the `FieldOfficer`. 3. The `FieldOfficer` completes the form by selecting the emergency level, type, location, and brief description of the situation. The `FieldOfficer` also describes possible responses to the emergency situation. Once the form is completed, the `FieldOfficer` submits the form. 4. `FRIEND` receives the form and notifies the `Dispatcher`. 5. The `Dispatcher` reviews the submitted information and creates an `Incident` in the database by invoking the `OpenIncident` use case. The `Dispatcher` selects a response and acknowledges the report. 6. `FRIEND` displays the acknowledgment and the selected response to the `FieldOfficer`.
Entry condition	• The `FieldOfficer` is logged into `FRIEND`.
Exit conditions	• The `FieldOfficer` has received an acknowledgment and the selected response from the `Dispatcher`, OR • The `FieldOfficer` has received an explanation indicating why the transaction could not be processed.
Quality requirements	• The `FieldOfficer`'s report is acknowledged within 30 seconds. • The selected response arrives no later than 30 seconds after it is sent by the `Dispatcher`.

Use cases Writing Guidelines

- Name usecases such that name identifies what actor wants to accomplish. (ReportEmergency, PerformTransaction etc)
- Clearly mark boundary of system (What system does, & what actors do)
- Use proper nonambigous names for Actors and System
- Clearly define relationship between steps
- Pharase use case tasks in Active voice.

Refining Use cases

- Incorporate further detail (Requirements) in Usecases and get those validated by user.

Flow of events

1. The `FieldOfficer` activates the "Report Emergency" function of her terminal.
2. `FRIEND` responds by presenting a form to the `FieldOfficer`.
3. The `FieldOfficer` completes the form by selecting the emergency level, type, location, and brief description of the situation. The `FieldOfficer` also describes possible responses to the emergency situation. Once the form is completed, the `FieldOfficer` submits the form.
4. `FRIEND` receives the form and notifies the `Dispatcher`.
5. The `Dispatcher` reviews the submitted information and creates an `Incident` in the database by invoking the `OpenIncident` use case. The `Dispatcher` selects a response and acknowledges the report.
6. `FRIEND` displays the acknowledgment and the selected response to the `FieldOfficer`.

Flow of events

1. The `FieldOfficer` activates the "Report Emergency" function of her terminal.
2. `FRIEND` responds by presenting a form to the officer. *The form includes an emergency type menu (general emergency, fire, transportation) and location, incident description, resource request, and hazardous material fields.*
3. The `FieldOfficer` completes the form by *specifying minimally the emergency type and description fields.* The `FieldOfficer` may also describe possible responses to the emergency situation *and request specific resources.* Once the form is completed, the `FieldOfficer` submits the form.
4. `FRIEND` receives the form and notifies the `Dispatcher` *by a pop-up dialog.*

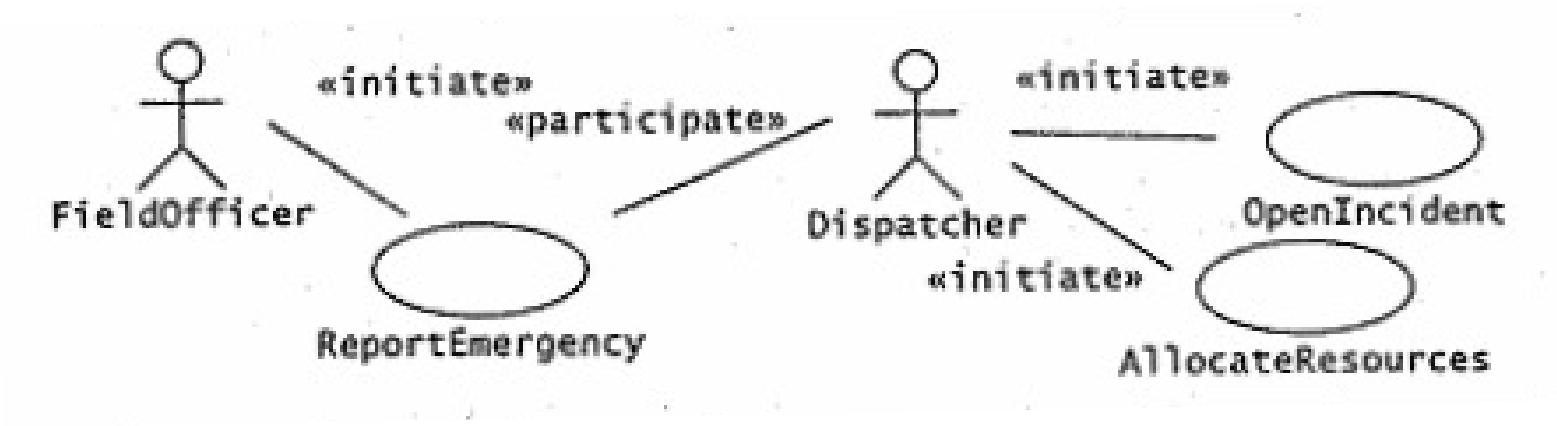
Refining Use cases

- Incorporate attributes needed for some interactoin.
- Define Access rights
- Define exceptions and handling procedure.
- Factor out common functionality, and clearly state if some other use case is needed to be invoked in some interaction

Communication relation b/w Actor & Use cases

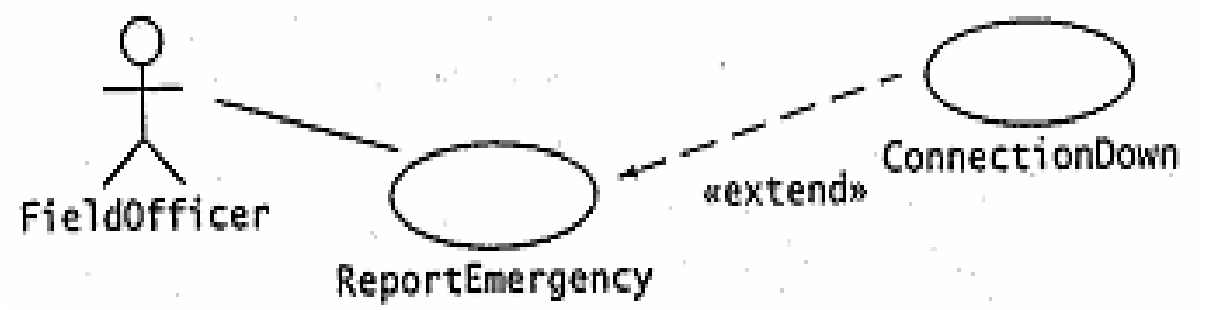
- Relation b/w Actor & Use cases represent “Flow of information”

  Clearly mark initiator actor for a use case, and mark participant actor for further clarity



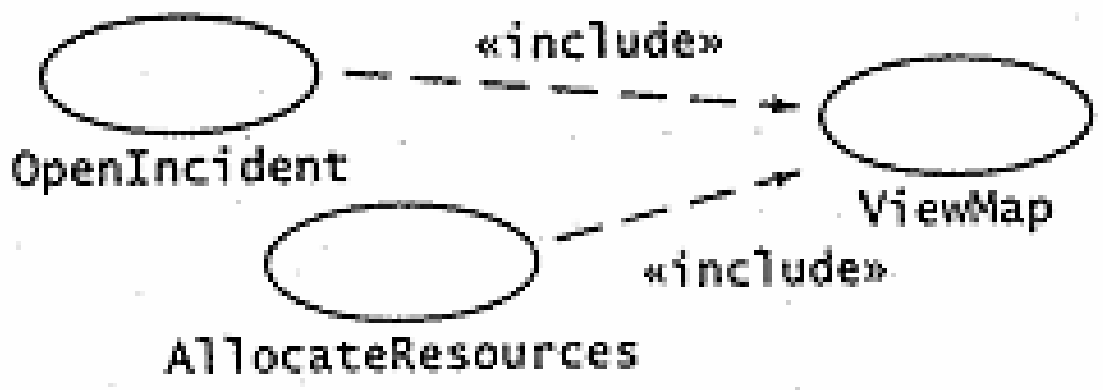
Extending a use case

- Extend a use case if it has some exceptional, Optional behavior under certain conditions.



Including a use case

- For behavior shared across number of use cases, make a separate use case and include it where needed.



Creating a Glossary

- Identify all participating Objects &
- Establish a clear terminology by describing them unambiguously in glossary.

FieldOfficer	Police Officer on duty. A FieldOfficer can only be assigned to one Incident. FieldOfficer are identified by a badge number.

Identify non functional requirements

Usability	User expertise, Interface Standards, Documentation
Reliability	Robustness issues, Restart Acceptable ?, safety requirements, security requirements
Performance	Response time, time-critical tasks, number of concurrent users, worst latency
Packaging	Constraints on installation, number of installations
Legal	Software license, failure liability issues, licenses for used components

Home Work 2 Deadline

- Submission deadline: 2007-09-22

- Email your report with subject “MMSE07 Homework 2” to ahaseeb@kth.se