Large Scale Systems Design
G52LSS

Lecture 13 – Use Case Analysis

- Refine Requirements
- Use Case Diagrams and Use Cases
- Steps of Use Case Analysis
- Example: University Registration System

**Learning outcomes:** understand the importance of continuously refine requirements; identify some tools for use case analysis; apply the steps for use case analysis; demonstrate use case analysis for a small case.

---

**Refine Requirements**

To better **understand processes and data** within the system and to help refining system requirements the following can help:

- Summary of business activities
- Use-case analysis: user stories, use case diagrams, use cases

The requirements specification phase in the SDLC has a clear purpose:

**to define requirements that match the real users’ needs.**

---

**Summary of Business Activities**

Short and easy to understand description of all the **business activities** that should be incorporated into the system to be developed.

The key difference between the summary of business activities and a user story is in their scope.

The **summary of business** activities describes all business processes within the system.

A **user story** refers to an specific business process or function within the system.

---

**Use Case Analysis**

*Event-driven modelling* technique – everything in the system is a response to some triggering event.

A use case diagram represents a **complete functionality**, not part of an overall function.

A **use-case** shows a single function of the system, describes how the system reacts to an event that triggers the system, is a set of activities that produce some output result.

Simple use-cases may have only one path and complex use cases may have several possible paths.

The information in a use case is organised in **three main parts:** basic information, inputs/outputs, and details.
**User Stories**

Helpful technique from extreme programming (XP) to identify valuable business user requirements.

*Narrates from the users’ perspective,* the way in which business processes are performed.

*Short and easy to understand* so that developers get an overall picture that is clear enough to estimate what it takes to complete the project.

*Several user stories may be needed* to describe the various processes in the system to be developed.

*User stories help developers* to understand what happens between the users and the system.

Users and developers have different backgrounds and points of view. Therefore, user stories is a *good way of communication* between developers and users.

---

**Reading good stories is easy but writing them is not straightforward.**

### Purpose and Characteristics of Good User Stories

<table>
<thead>
<tr>
<th>User</th>
<th>Developer</th>
</tr>
</thead>
<tbody>
<tr>
<td>(how the system should work)</td>
<td>(which units of code should be developed)</td>
</tr>
<tr>
<td>- Common language</td>
<td>- Interactions between roles and system</td>
</tr>
<tr>
<td>- Simple sentences</td>
<td>- Notions are transformable to code and constructs</td>
</tr>
<tr>
<td>- Starting even and ending</td>
<td>- Changes in story are easy to trace in the code</td>
</tr>
<tr>
<td>- Notions from known vocabulary</td>
<td></td>
</tr>
</tbody>
</table>

---

Good user stories can follow the SVO (subject-verb-object) format plus some explanation of the terms used in the story.

For example:

- Student enters the ID
- System verifies the ID is valid (where ID is a 7-digit number starting with 4 and unique for each student)
- System shows list of available modules
- Student selects modules to register
- System validates that chosen modules are allowed

---

**The development cycle** of user stories

1. Users write **initial stories**
2. Users meet with developers and together write **clarified user stories** with notions
3. Developers translate the clarified stories into code. Stories are further clarified with users when required. **Test stories** are also written
4. Developers make sure that the **test stories are fulfilled** and hand system to users
5. Users **verify system** and possibly write corrected stories and new stories
Use Case Diagrams and Use Cases

The use case diagram shows the system’s behavior together with the key actors for a specific scenario.

The elements of use case diagrams are:

- Actors
- Use cases
- System boundary
- Connections
- Extend relationships
- Include relationships

Example 13.1 The following is the use case diagram for an appointments system in a surgery.

Example 13.2 The following is one use case for the appointments system in a surgery in which the main actor is the patient.
Steps of Use Case Analysis

Following the creation of a use case diagram, then cycle through the steps below in an iterative manner.

1. Identify the major use cases
   · Use one use case form for each use case
   · If more than nine, group into packages
   · Ask who, what, and where about the tasks and their inputs and outputs

2. Identify the major steps within each use case
   · For each use case, fill in the major steps needed to process the inputs and produce the outputs
   · Ask how about each use case

3. Identify elements within steps
   · For each step, identify its triggers, its inputs and outputs
   · Ask how about each step

4. Confirm the use case.
   · For each use case, validate that it is correct and complete
   · Ask the user to execute the process using the written steps in the use case, that is, have the user role-play the use case

Example: University Registration System

Exercise D. (Dennis et al. 2006, chapter 5). A University Registration System should enable staff of each academic department to examine the modules offered by their department, add and remove modules, and change the information about them (e.g. the maximum number of students permitted). It should permit students to examine currently available modules, add and drop modules to and from their schedules, and examine the modules for which they are enrolled. Department staff should be able to print a variety of reports about the modules and the students enrolled in them. The system should ensure that no student takes too many modules and that students who have any unpaid fees are not permitted to register (Note: assume that a fees data store is maintained by the university’s financial office and this data store is accessed by the registration system but the fees data store is not modified by the registration system).
**Use Case Diagram**

**Use Case Name**: Enrol student in module

**ID Number**: 2

**Short Description**: Describes how students review listing of modules available for enrolment, add and remove modules from their schedules, and review their schedules.

**Trigger**: Student requests to enrol in modules.

**Type**: External / Temporal

**Major Inputs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available module request</td>
<td>Student____________________________</td>
</tr>
<tr>
<td>Available modules</td>
<td>Module offerings file________________</td>
</tr>
<tr>
<td>Module enrolment request</td>
<td>Student____________________________</td>
</tr>
<tr>
<td>Fee payment status</td>
<td>Fees file_________________________</td>
</tr>
</tbody>
</table>

**Major Outputs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Destination</th>
</tr>
</thead>
</table>
| List of available modules   | Student_____
| Student enrolment file      | Enrolment file
| Student schedule            | Student___

**Major Steps Performed**

1. Student requests list of available modules. List of available modules is generated.
2. Student adds module to current schedule. Fee payment status is checked and total hours is checked. If OK, module is added to student schedule.
3. Student removes module from schedule.
4. Student reviews current scheduled modules.

**Information for Steps**

- Available module request
- Available modules list
- Module ID
- Fee payment status
- Total hours
- Student schedule

---

**Use Case Name**: Maintain information about available modules

**ID Number**: 1

**Short Description**: Describes how department staff reviews module offerings, adds new modules, deletes existing modules or changes existing module information.

**Trigger**: Departments must prepare upcoming module offerings.

**Type**: External / Temporal

**Major Inputs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module offering changes</td>
<td>Department staff____________________</td>
</tr>
<tr>
<td>Department identifier</td>
<td></td>
</tr>
<tr>
<td>New module information</td>
<td></td>
</tr>
<tr>
<td>Module offering file</td>
<td></td>
</tr>
</tbody>
</table>

**Major Outputs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updates module offerings</td>
<td>Module offering file________________</td>
</tr>
<tr>
<td>Module offerings list</td>
<td>Department staff____________________</td>
</tr>
</tbody>
</table>

**Major Steps Performed**

1. Department staff requests module offering list for the department.
2. New module information is entered.
3. Modules to delete are entered.
4. Module modifications are entered.

**Information for Steps**

- Module offering list request
- Department identifier
- New module information
- Module offering file
- Module number to delete
- Module offering file
- Module number to modify
- Module changes
- Module offering file
Use Case Name: Module enrolment reports  
ID Number: 3  

Short Description: Describes how department staff prints various reports on modules and module enrolments.  
Trigger: Department staff needs information on modules and module enrolments.  
Type: External / Temporal  

<table>
<thead>
<tr>
<th>Major Inputs</th>
<th>Source</th>
<th>Major Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report request</td>
<td>Staff</td>
<td>Report requested</td>
<td>Staff</td>
</tr>
<tr>
<td>Module information</td>
<td>Module offerings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enrolment information</td>
<td>Module information</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Information for Steps         |                |                                |             |
| Report type                   |                |                                |             |
| Module offerings              |                |                                |             |
| Enrolments                    |                |                                |             |

| Major Steps Performed         |                |                                |             |
| 1. Staff enters report request.|                |                                |             |
| 2. Requested report is generated. |            |                                |             |

A simple user story for the case in which a student enrols in a module:

**Student access the system.** If the student has a valid username and password, allow the student to sign in, otherwise display an explanatory message.

**Student requests list of available modules.** Search the fees file to determine whether the student is permitted to register or not.

**System shows list of available modules.** Displays a list of available modules according to the fee paying status of the student.

**Student request enrolment in selected module.** Check if the enrolment is permitted according to the number of modules in which the student is already registered.

**System confirms enrolment.** Ask the student to confirm the enrolment. Display confirmation of enrolment and update the enrolment file. Provide the student with a confirmation code for the enrolment.

**Student requests dropping a module.** Display updated information and ask the student to confirm the drop. Update the enrolment file. Provide the student with a confirmation code.

**Student continues with more enrolments and drops.** If the number of modules in which the student is enrolled has not reached the maximum permitted, give the student the option to enrol in other modules. If there are at least one module in which the student is enrolled, give the student the option to drop modules.

**System shows current student’s schedule.** Display student’s schedule for modules enrolment. Give student the options of printing, saving and emailing the current schedule.

A summary of business activities for the University Registration System:

Staff and students access the system providing a valid username and password.

Staff access information for the available modules in their department.

Staff adds, removes, and change information about the modules for which they are responsible.

For each academic department, authorised staff modifies the maximum number of modules that students are permitted to take.

Staff prints reports of various kinds about the available modules and their detailed information and reports about student enrolment.
Students are permitted to access the list of available modules if they do not have unpaid fees.

Students make request for enrolment in a module of their choice. If the student has not reached the maximum number of modules permitted then the enrolment is processed.
Students are notified of the enrolment.

Students make request for dropping a module of their choice. Students are notified of the change in their enrolment record.
Students can access their schedule and they can print, email or save it in a file.

**Additional Reading**

Chapter 5 of (Dennis et al., 2006)

Chapter 7 of (Hoffer et al., 2005) pages 225-233.