IT4490 - SOFTWARE DESIGN AND CONSTRUCTION

3. REQUIREMENT MODELING WITH UC

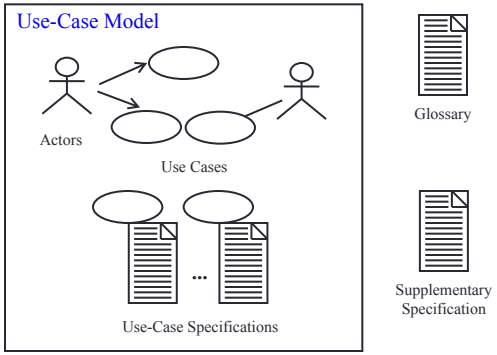
Some slides extracted from IBM coursewares

Content

1.1. Purpose of Requirement

- Establish and maintain agreement with the customers and other stakeholders on what the software should do.
- Give software developers a better understanding of the requirements of the software.
- Delimit the software.
- Provide a basis for planning the technical contents of the iterations.
- Provide a basis for estimating cost and time to develop the software.
- Define a user interface of the software.

1.2. Relevant Requirements Artifacts

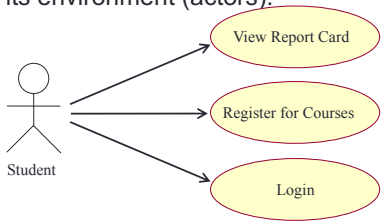


Content

1. Requirements
2. Use case diagram
3. Use case specification/scenario
4. Glossary
5. Supplementary Specification

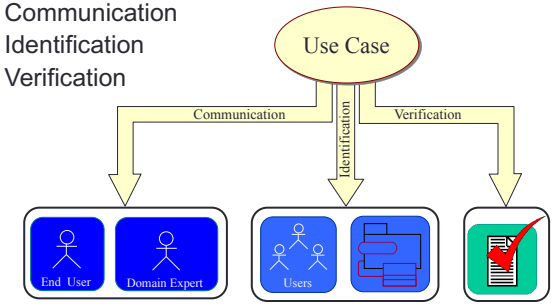
2.1. Overview of Use-Case Diagram

- A diagram modeling the dynamic aspects of softwares that describes a software's functional requirements in terms of use cases.
- A model of the software's intended functions (use cases) and its environment (actors).



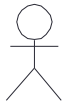
Benefits of a Use-Case Model

- Communication
- Identification
- Verification



Major Concepts in Use-Case Modeling

- An actor represents anything that interacts with the software.

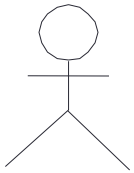


- A use case describes a sequence of events, performed by the software, that yields an observable result of value to a particular actor.

9

2.2. Actors

- Actors represent roles a user of the software can play
  - They can represent a human, a machine, or another software
  - They can be a peripheral device or even database
- They can actively interchange information with the software
  - They can be a giver of information
  - They can be a passive recipient of information
- Actors are not part of the software
  - Actors are EXTERNAL



Actor

Actors and Roles

Charlie: Is employed as a math professor and is an economics undergraduate.

Jodie: Is a science undergraduate.

Charlie and Jodie both act as a Student.

Student

Register for courses

Charlie also acts as a Professor.

Professor

Submit grades

11

Some guideline to extract actors

- Pay attention to a noun in the problem description, and then extract a subject of action as a Actor.
- Ensure that there are no any excesses and deficiencies between the problem description and Actors extracted.
- Actor names
  - should clearly convey the actor's role
  - good actor names describe their responsibilities

12

Put some questions to find actors

- Who or what uses the software?
- Who or what gets information from this software?
- Who or what provides information to the software?
- Where in the company is the software used?
- Who or what supports and maintains the software?
- What other softwares use this software?

13

ATM software

- The ATM software, allowing interbank network, communicates with bank customers via a terminal. The ATM first accepts a bank cards and asks the customer to enter a personal identification number (PIN) - both of which will be sent to the bank for validation as part of each transaction.
- ATM customers can select any of the four transaction types: deposits, withdrawals, transfer and balance inquiries.
- In any transaction, after receiving enough information from the customer, the ATM asks the bank consortium to process the request.
- The bank consortium forwards the request to the appropriate bank. The bank then processes and responses to the bank consortium which in turn notifies the result to the ATM software.
- The cashier can access to the ATM software to collect deposits. The maintenance crew may use the ATM to print logs, start, stop or backup the software.

14

Find actors in the ATM software

An Automated Teller Machine (ATM)

16

2.3. Use Cases

- Define a set of use-case instances, where each instance is a sequence of actions a software performs that yields an observable result of value to a particular actor.
  - A use case models a dialogue between one or more actors and the software
  - A use case describes the actions the software takes to deliver something of value to the actor

17

Some guidelines to extract use cases

- Pay attention to a verb in the problem description, and then extract a series of Actions as a UC.
- Ensure that there are no any excesses and deficiencies between the problem description and Use cases extracted.
- Check the consistency between Use Cases and related Actors.
- Conduct a survey to learn whether customers, business representatives, analysts, and developers all understand the names and descriptions of the use cases

18

Use case name

- Be unique, intuitive, and self-explanatory
- Define clearly and unambiguously the observable result of value gained from the use case
- Be from the perspective of the actor that triggers the use case
- Describe the behavior that the use case supports
- Start with a verb and use a simple verb-noun combination

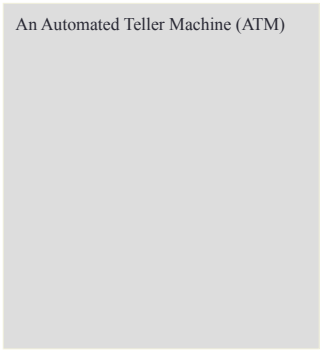
19

Put some questions to find use cases

- What are the goals of each actor?
- Why does the actor want to use the software?
- Will the actor create, store, change, remove, or read data in the software? If so, why?
- Will the actor need to inform the software about external events or changes?
- Will the actor need to be informed about certain occurrences in the software?

20

Find use cases in the ATM software



21

ATM software

- The ATM software, allowing interbank network, communicates with bank customers via a terminal. The ATM first accepts a bank cards and asks the customer to enter a personal identification number (PIN) - both of which will be sent to the bank for validation as part of each transaction.
- ATM customers can select any of the four transaction types: deposits, withdrawals, transfer and balance inquiries.
- In any transaction, after receiving enough information from the customer, the ATM asks the bank consortium to process the request.
- The bank consortium forwards the request to the appropriate bank. The bank then processes and responses to the bank consortium which in turn notifies the result to the ATM software.
- The cashier can access to the ATM software to collect deposits. The maintenance crew may use the ATM to print logs, start, stop or backup the software.

23

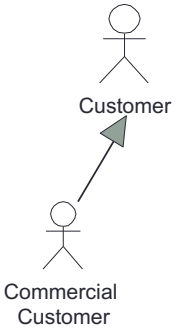
2.4. Relationships

- Not recommended to use many times
- Three kinds
  - Between actors: generalization, association
  - Between actor and use cases: association
  - Between use cases: generalization, include, extend

24

2.4.1. Between actors

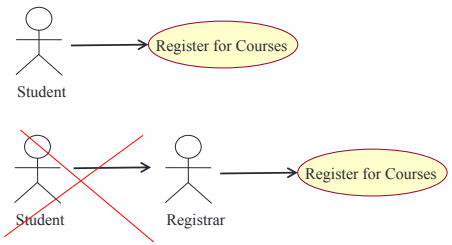
- Generalization
  - The child actor inherits parent's characteristics and behaviors.



25

2.4.1. Between actors (2)

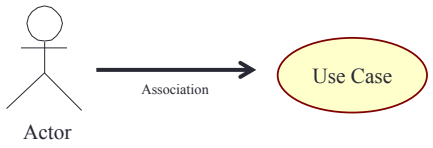
- Association
  - Consider the difference between two diagrams



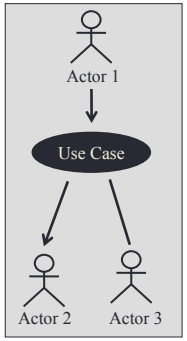
26

2.4.2. Between actor and use case

- Establish the actors that interact with related use cases → Use associations
  - Associations clarify the communication between the actor and use case.
  - Association indicate that the actor and the use case instance of the software communicate with one another, each one able to send and receive messages.
- The arrow head is optional but it's commonly used to denote the initiator.

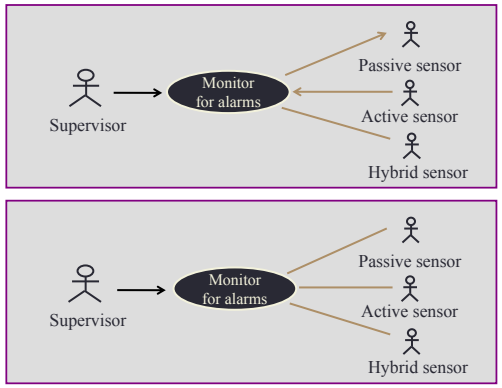


Communicates-Association



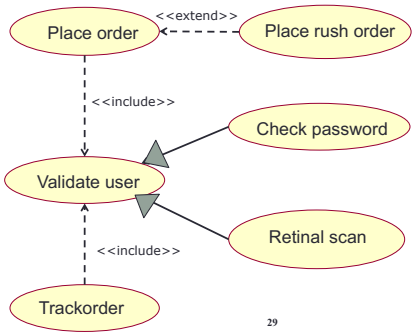
- A channel of communication between an actor and a use case.
- A line is used to represent a communicates-association.
  - An arrowhead indicates who initiates each interaction.
  - No arrowhead indicates either end **can** initiate each interaction.

Arrowhead Conventions



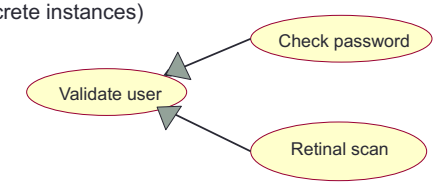
2.4.3. Between use cases

- Generalization
- <<include>>
  - always use
- <<extend>>
  - sometime use



Between use cases - Generalization

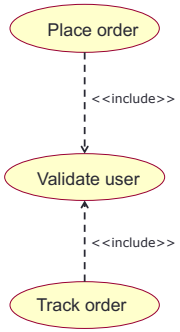
- The child use case inherits the behavior and meaning of the parent use case;
  - the child may add to or override the behavior of its parent;
  - the child may be substituted any place the parent appears (both the parent and the child may have concrete instances)



31

Between use cases - Include

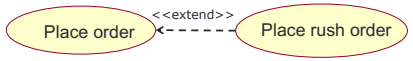
- The base use case explicitly incorporates the behavior of another use case at a location specified in the base.
- The included use case never stands alone, but is only instantiated as part of some larger base that includes it



32

Between use cases - Extend

- The base use case implicitly incorporates the behavior of another use case at a location specified indirectly by the extending use case.
- The base use case may stand alone, but under certain conditions its behavior may be extended by the behavior of another use case.



33

Structuring Use Cases

- You should try to describe your system's behavior with just a few major use cases. Each large use case defines a major goal that an actor achieves
- When you have made these goals clear, you can go into more detail about how the each goal is achieved, and about variations in the basic goals.
- Avoid decomposing the use cases into too much detail. Use cases are about the users' experience of your system, instead of its internal workings. Additionally, you will generally find it more productive to create early working versions of the code, instead of spending time structuring use cases into fine detail.

38

2.5. Use case diagram

- The Use case diagram shows a set of use cases and actors and their relationships.
- The Use case diagram serves as a contract between the customer and the developers.
- Because it is a very powerful planning instrument, the Use case diagram is generally used in all phases of the development cycle

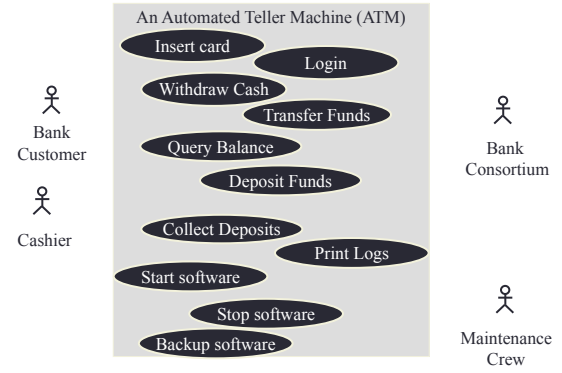
39

Notes

- Should not use two many relationships between use case in the Use case diagram
  - Tangle and make the diagram difficult to observe
  - Only use the relationship if necessary
  - In the Use case diagram, the sequence do use cases are not specified

40

Find use cases in the ATM software



42

Case Study: Course Registration

Problem Statement

- Review the problem statement provided in the Course Registration Requirements Document.

Course Registration Requirements Document

43

Problem Statement (1/4)

- As the head of information systems for a university, you are tasked with developing a new student registration system. The college would like a new client-server system to replace its much older system developed around mainframe technology. The new system will allow students to register for courses and view report cards from personal computers attached to the campus LAN. Professors will be able to access the system to sign up to teach courses as well as record grades.

44

Problem Statement (2/4)

- Due to a decrease in federal funding, the college cannot afford to replace the entire system at once. The college will keep the existing course catalog database where all course information is maintained. This database in an Ingres relational database running on a DEC VAX. Fortunately the college has invested in an open SQL interface that allows access to this database from college's Unix servers. The legacy system performance is rather poor, so the new system must ensure that access to the data on the legacy system occurs in a timely manner. The new system will access course information from the legacy database but will not update it. The registrar's office will continue to maintain course information through another system.

45

Problem Statement (3/4)

- At the beginning of each semester, students may request a course catalogue containing a list of course offerings for the semester. Information about each course, such as professor, department, and prerequisites, will be included to help students make informed decisions.
- The new system will allow students to select four course offerings for the coming semester. In addition, each student will indicate two alternative choices in case the student cannot be assigned to a primary selection. Course offerings will have a maximum of ten students and a minimum of three students. A course offering with fewer than three students will be canceled. For each semester, there is a period of time that students can change their schedule. Students must be able to access the system during this time to add or drop courses. Once the registration process is completed for a student, the registration system sends information to the billing system so the student can be billed for the semester. If a course fills up during the actual registration process, the student must be notified of the change before submitting the schedule for processing.

46

Problem Statement (4/4)

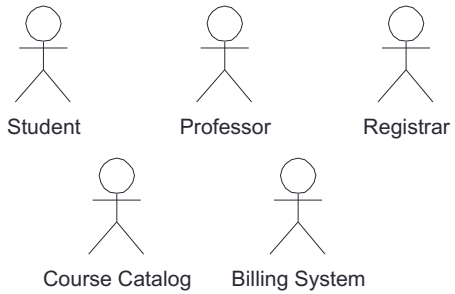
- At the end of the semester, the student will be able to access the system to view an electronic report card. Since student grades are sensitive information, the system must employ extra security measures to prevent unauthorized access.
- Professors must be able to access the on-line system to indicate which courses they will be teaching. They will also need to see which students signed up for their course offerings. In addition, the professors will be able to record the grades for the students in each class

47

Case Study: Course Registration

Actors

- Find actors of Course Registration software?



50

Case study

- Draw Use case diagram for the Course Registration software?

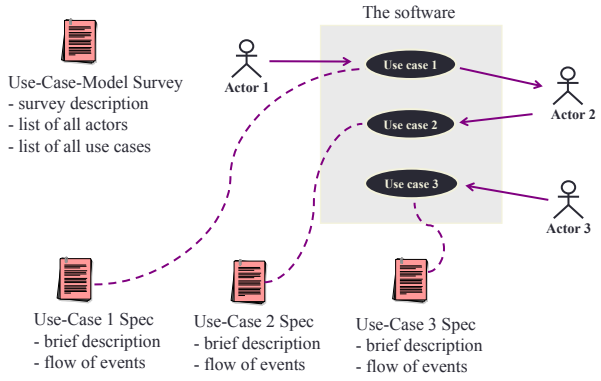
Content

1. Requirements
2. Use case diagram
3. Use case specification/scenario
4. Glossary
5. Supplementary Specification

51

52

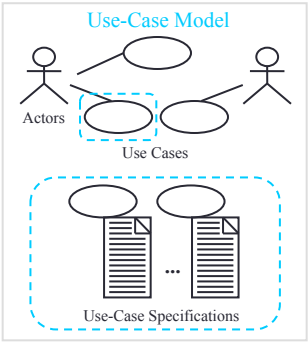
Use-Case Specification



53

Use-Case Specification

- Code
- Name
- Brief description
- Flow of Events
- Relationships
- Activity diagrams
- Use-Case diagrams
- Special requirements
- Pre-conditions
- Post-conditions
- Other diagrams



55

Brief description of UC

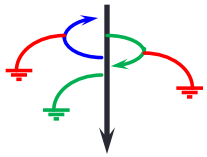
- *Describe briefly the purpose of UC*
- Example: Use case “Log in” in the ATM software:

“This use case describes the interaction between bank customers and the ATM machine when the customer wishes to log in to the software to perform transactions”

56

Use-Case Flow of Events

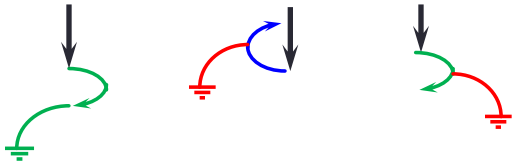
- Has one normal, basic flow
- Several alternative flows
  - Regular variants
  - Odd cases
  - Exceptional flows for handling error situations



57

What Is a Scenario?

- A scenario is an instance of a use case.



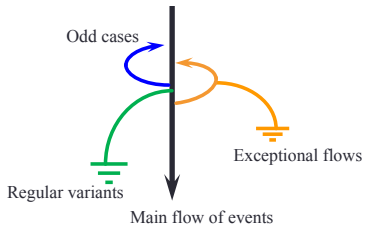
58

UC Login in the ATM software

- Main flows of events: The use case starts when software prompts the Customer for a PIN Number. The Customer can now enter a PIN number. The Customer commits the entry. The software then checks this PIN number to see if it is valid. If valid, the software acknowledges the entry, thus ending the use case
- Regular variants: The Customer cancel a transaction at any time, thus restart the UC. No changes are made to the Customer’s account.
- Odd case: The Customer clear a PIN number anytime before committing it and re-enter a new PIN number
- Exceptional flow of events: If the Customer enter an invalid PIN number, the UC restarts. If this happens 3 times in a row, the software cancel the entire transaction, and keep the ATM card.

59

Example



60

Flow of events

Success / Main flow of event

#	Doer	Action
1	Customer	requests to log in
2	software	prompts a Log in screen
3	Customer	enters a PIN number
4	Customer	submit to login
5	software	checks if the PIN number is valid
6	software	displays the main menu if the PIN number is valid

Alternative flow of event

#	Doer	Action
	Customer	cancels a transation **at any time**
4a	Customer	clears PIN number **before submitting**
6a	software	notifies Invalid PIN number, goes to Step 2 **if the PIN number is not valid less than 3 times**
6b	software	notifies invalid PIN number 3 times, keep the ATM card **if the PIN number is not valid 3 times**

What Is an Activity Diagram?

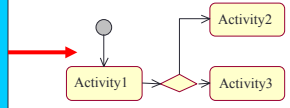
- An activity diagram in the Use-Case Model can be used to capture the activities in a use case.
- It is essentially a flow chart, showing flow of control from one activity or action to another.

Flow of Events

This use case starts when the Registrar requests that the software close registration.

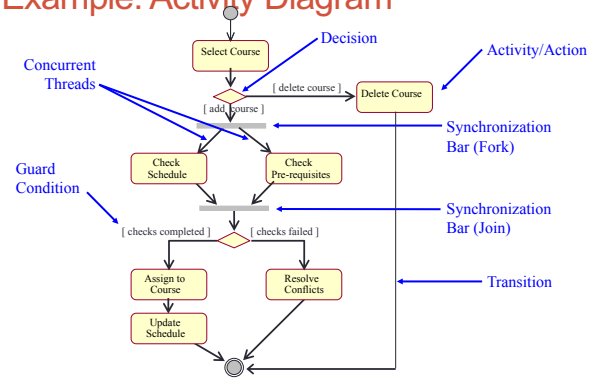
1. The software checks to see if registration is in progress. If it is, then a message is displayed to the Registrar and the use case terminates. The Close Registration processing cannot be performed if registration is in progress.

2. For each course offering, the software checks if a professor has signed up to teach the course offering and at least three students have registered. If so, the software commits the course offering for each schedule that contains it.



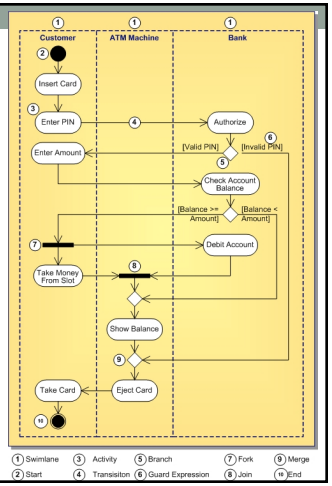
62

Example: Activity Diagram



Partitions

- Each partition should represent a responsibility for part of the overall workflow, carried by a part of the organization.
- A partition may eventually be implemented by an organization unit or a set of classes in the business object model.



Content

1. Requirements
2. Use case diagram
3. Use case specification/scenario
4. Glossary
5. Supplementary Specification

65

66

4. Glossary

- The **Glossary** defines important terms used in the project for all models.
- There is only one Glossary for the software.
- This document is important to many developers, especially when they need to understand and use the terms that are specific to the project.
- The **Glossary** is used to facilitate communications between domain experts and developers

67

4. Glossary (2)

- **Introduction**: Provides a brief description of the Glossary and its purpose.
- **Terms**: Define the term in as much detail as necessary to completely and unambiguously characterize it.

4. Glossary (3)

Glossary

Course Registration software Glossary

1. Introduction

This document is used to define terminology specific to the problem domain, explaining terms, which may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal *data dictionary*, capturing data definitions so that use-case descriptions and other project documents can focus on what the software must do with the information.

2. Definitions

The glossary contains the working definitions for the key concepts in the Course Registration software.

2.1 Course: A class offered by the university.

2.2 Course Offering: A specific delivery of the course for a specific semester – you could run the same course in parallel sessions in the semester. Includes the days of the week and times it is offered.

2.3 Course Catalog: The unabridged catalog of all courses offered by the university.

Case Study: Glossary

- Make the Glossary of the Course Registration software

Glossary

Content

1. Requirements
2. Use case diagram
3. Use case specification/scenario
4. Glossary
5. Supplementary Specification

5. Supplementary Specification

- Includes the nonfunctional requirements and functional requirements not captured by the use cases
- Contains those requirements that do not map to a specific use case: Functionality, Usability, Reliability, Performance, Supportability

Supplementary Specification

72

5. Supplementary Specification (2)

- **Functionality**: List of the functional requirements that are general to many use cases.

This section lists functional requirements that are common to more than one use case.

1. System Error Logging

All system errors shall be logged. Fatal system errors shall result in an orderly shutdown of the system.
The system error messages shall include a text description of the error, the operating system error code (if applicable), the module detecting the error condition, a data stamp, and a time stamp. All system errors shall be retained in the Error Log Database.

1. Remote Access

All functionality shall be available remotely through an internet connection. This may require applications or controllers running on the remote computers.

73

5. Supplementary Specification (2)

- **Usability**: Requirements that relate to, or affect, the usability of the software. Examples include ease-of-use requirements or training requirements that specify how readily the software can be used by its actors.

This section lists all of those requirements that relate to, or affect, the usability of the system
1 Windows Compliance
The desktop user-interface shall be Windows 95/98 compliant
2 Design for Ease-of-Use
The user interface of the C-Registration System shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System
3 Online Help
Each feature of the C-Registration System shall have built-in online help for the user. Online Help shall include step by step instructions on using the System. Online Help shall include definitions for terms and acronymns

74

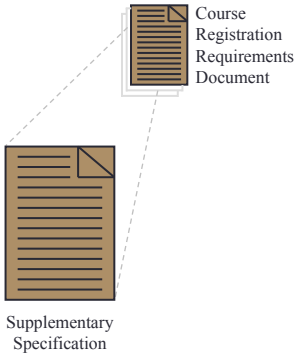
5. Supplementary Specification (3)

- **Reliability**: Any requirements concerning the reliability of the software. Quantitative measures such as mean time between failure or defects per thousand lines of code should be stated.
- **Performance**: The performance characteristics of the software. Include specific response times. Reference related use cases by name.
- **Supportability**: Any requirements that will enhance the supportability or maintainability of the software being built.

75

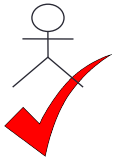
Case study: Supplementary Specification

- Make the Supplementary Specification for the Course Registration software



Checkpoints: Actors

- Have all the actors been identified?
- Is each actor involved with at least one use case?
- Is each actor really a role? Should any be merged or split?
- Do two actors play the same role in relation to a use case?
- Do the actors have intuitive and descriptive names? Can both users and customers understand the names?

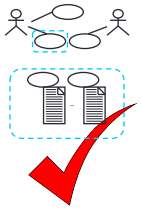


Checkpoints: Use-Cases

- Is each use case involved with at least one actor?
- Is each use case independent of the others?
- Do any use cases have very similar behaviors or flows of events?
- Do the use cases have unique, intuitive, and explanatory names so that they cannot be mixed up at a later stage?
- Do customers and users alike understand the names and descriptions of the use cases?

Checkpoints: Use-Case Model

- Is the Use-Case Model understandable?
- By studying the Use-Case Model, can you form a clear idea of the software's functions and how they are related?
- Have all functional requirements been met?
- Does the Use-Case Model contain any superfluous behavior?
- Is the division of the model into use-case packages appropriate?



Checkpoints: Use-Case Specifications

- Is it clear who wants to perform a use case?
- Is the purpose of the use case also clear?
- Does the brief description give a true picture of the use case?
- Is it clear how and when the use case's flow of events starts and ends?
- Are the actor interactions and exchanged information clear?
- Are any use cases overly complex?

80

Checkpoints: Glossary

- Does each term have a clear and concise definition?
- Is each glossary term included somewhere in the use-case descriptions?
- Are terms used consistently in the brief descriptions of actors and use cases?

81

Review

- What are the main artifacts of Requirements?
- What are the Requirements artifacts used for?
- What is a Use-Case Model?
- What is an actor?
- What is a use case? List examples of use case properties.
- What is the difference between a use case and a scenario?
- What is a Supplementary Specification and what does it include?
- What is a Glossary and what does it include?

82

Question?