CSE 403

Software Engineering

Spring 2023

#6: Use cases

Logistics

WEEK 2		
04/03	L: Dev. Cycle	DUE: PP_1.1!!!
04/04	T: Proposals	DUE: PP_1.2!!!
04/05	L: Requirements	Project Requirements (PR)
04/06	P: Requirements	
04/07	L: Use-Cases	
WEEK 3		
04/10	L: SCRUM	
04/11	T:	DUE: PR!!!
04/12	L: Version Control	GitHub Project Setup (GPS)
04/13	P:	
04/14	LX: GIT	

Logistics: What is up?

ASMR: A Smart Music Recommender	**Idea Factory**	**RateMyUWPlan**	**UW Roomie**
BeatBuddy	**cumulative.do**	**DuoCode**	**MyCalendar**
RCRS: Resilient Course Registration System	**PathKit**	**WorkoutHolic**	**DubSpot**
Region Attack	**DormWiki**	**Games In One**	**ReWrite**

Recap: Life-cycle stages

Virtually all SDLC models have the following stages:

- **Requirements ← Our focus this week**
- Design
- Implementation
- Testing
- Maintenance

Requirements engineering

The process of eliciting, analyzing, documenting, and maintaining requirements.

One way to classify requirements

- Functional requirements
- Non-functional requirements
- Additional constraints

Requirements engineering

The process of eliciting, analyzing, documenting, and maintaining requirements.

One way to classify requirements

- Functional requirements
- Non-functional requirements
- Additional constraints

Examples from your projects?

Cockburn's requirements template

1. Purpose and scope
2. Terms (glossary)
3. **Use cases (the central artifact of requirements)**
4. Technology used
5. Other
    a. Development process: participants, values (fast-good-cheap), visibility, competition, dependencies
    b. Business rules (constraints)
    c. Performance demands
    d. Security, documentation
    e. Usability
    f. Portability
    g. Unresolved (deferred)
6. Human factors (legal, political, organizational, training)

See file on Canvas for comprehensive write up and examples.

Use cases

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

- It is an **example behavior** of the system
- Written from an **actor's point of view**, not the system's
- **3-9 clearly written steps** lead to a "main success scenario"

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

- It is an **example behavior** of the system
- Written from an **actor's point of view**, not the system's
- **3-9 clearly written steps** lead to a "main success scenario"

Terminology

- **Actor**: someone (or another system) interacting with the system
- **Primary actor**: person who initiates the action
- **Goal**: desired outcome of the primary actor

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

- It is an **example behavior** of the system
- Written from an **actor's point of view**, not the system's
- **3-9 clearly written steps** lead to a "main success scenario"

Terminology

- **Actor**: someone (or another system) interacting with the system
- **Primary actor**: person who initiates the action
- **Goal**: desired outcome of the primary actor

Use cases capture **functional requirements** of a system!

Use Case: simple example

UC1	Search for a student in the groups' page
Actor	Course Instructor
Flow	- Go to the Groups page - Type in the student name in the search box - The system presents the groups (and its membres) with the matching student names highlighted

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

- It is an **example behavior** of the system
- Written from an **actor's point of view**, not the system's
- **3-9 clearly written steps** lead to a "main success scenario"

Terminology

- **Actor**: someone (or another system) interacting with the system
- **Primary actor**: person who initiates the action
- **Goal**: desired outcome of the primary actor

Let's try this out!!!

What is a use case?

A **use case** is a **written description** of a **user's interaction with** the software **system to accomplish** a **goal**.

- It is an **example behavior** of the system
- Written from an **actor's point of view**, not the system's
- **3-9 clearly written steps** lead to a "main success scenario"

Terminology

- **Actor**: someone (or another system) interacting with the system
- **Primary actor**: person who initiates the action
- **Goal**: desired outcome of the primary actor

How did it go?

Benefits of use cases

- Establish an understanding between the customer and the developers of the requirements (**success scenarios**)
- Alert developers of special cases (alternatives) and error cases (exceptions) to test (**extension scenarios**)
- Capture a level of functionality (**list of goals**)

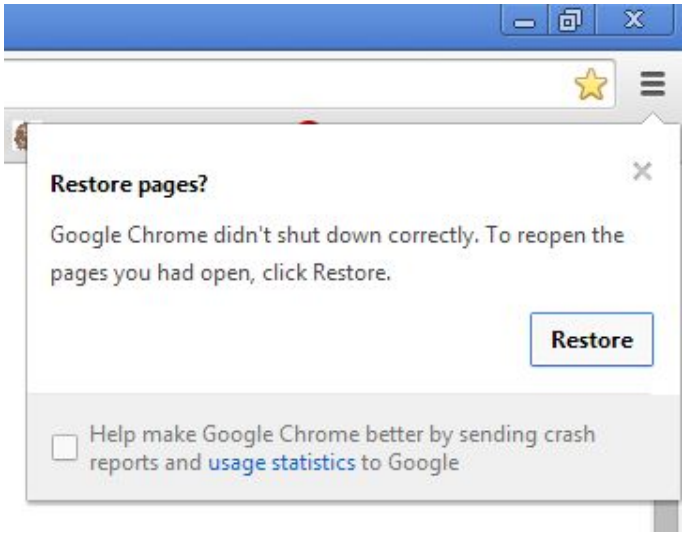
Alternative Flow (or Extension Condition)

Alternative Flow

A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**

Alternative Flow

A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**



Alternative Flow

A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**

Do

- Think about how every step of the use case could fail
- Give a plausible response to each extension from the system
- Response should either jump to another step of the case, or end it

Alternative Flow

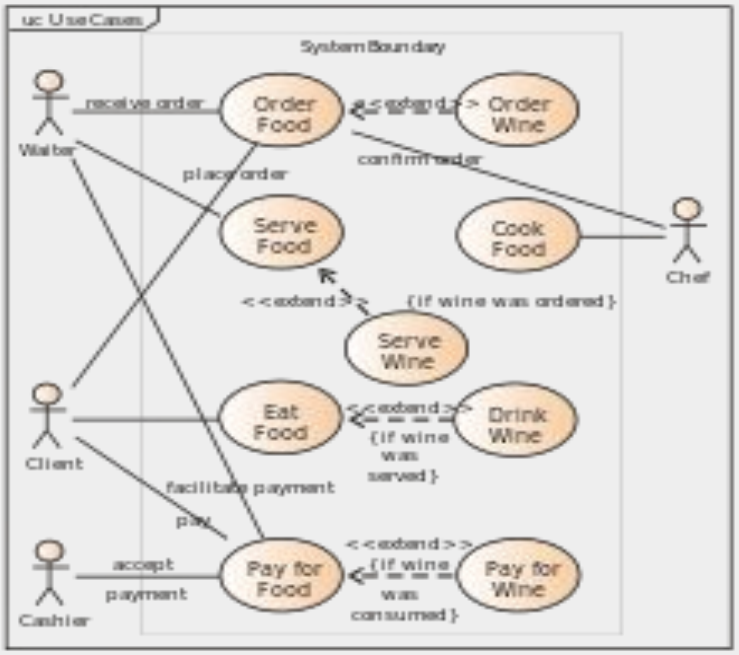
A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**

Do

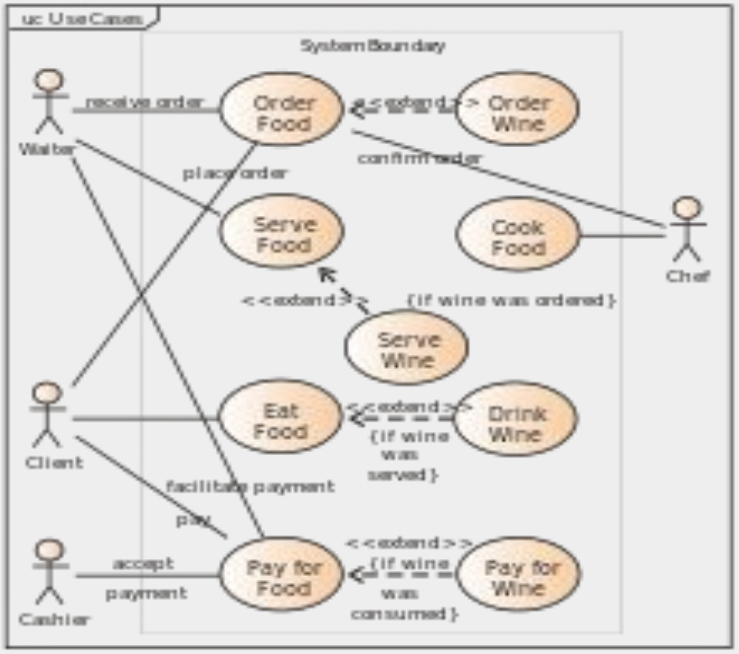
- Think about how every step of the use case could fail
- Give a plausible response to each extension from the system
- **Response should either jump to another step of the case, or end it**

If that is not enough → Use Case Relations?

(UML) Use case diagram



Use case diagram



“For reasons that remain a mystery to me, many people have focused on the stick figures and ellipses in use case writing since Jacobson's first book came out, and neglected to notice that use cases are fundamentally a text form.”
[Writing Effective Use Cases, Alistair Cockburn, 2000]

Alternative Flow

A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**

Do

- Think about how every step of the use case could fail
- Give a plausible response to each extension from the system
- Response should either jump to another step of the case, or end it

Don’t

- List things outside the use case ("User's power goes out")
- Make unreasonable assumptions ("DB will never fail")
- List a remedy that your system can't actually implement

Use Case: simple example (reloaded)

UC1	Search for a student in the groups' page
Actor	Course Instructor
Flow	- Go to the Groups page - Type in the student name in the search box - The system presents the groups (and its membres) with a matching student name
Alternative Flow A	- If the system does not find a match it presents an empty results section with a message signaling that no students were found.

Alternative Flow

A possible **branch** in a use case, e.g., **triggered by an error**; useful for identifying what **edge cases** need to be **handled/tested**

Do

- Think about how every step of the use case could fail
- Give a plausible response to each extension from the system
- Response should either jump to another step of the case, or end it

Don’t

- List things outside the use case ("User's power goes out")
- Make unreasonable assumptions ("DB will never fail")
- List a remedy that your system can't actually implement

Let's try this out!!!

3 steps for creating a use case

3 steps for creating a use case

1. **Identify actors and goals**

- Actors: What users and (sub)systems interact with our system?
- Goals: What does each actor need our system to do?

3 steps for creating a use case

1. **Identify actors and goals**
2. **Write the main success scenario**

- Main success scenario is the preferred "happy path”
  - Easiest to read and understand
  - Everything else is a complication on this
- Capture each actor's intent and responsibility, from trigger to goal
  - State what information passes between actors
  - Number each step (line)

3 steps for creating a use case

1. **Identify actors and goals**
2. **Write the main success scenario**
3. **List the alternative flows (failure extensions)**

- Many steps can fail (e.g., denied credit card, out of stock)
  - Note each failure condition separately, after the main success scenario
- Describe failure-handling
  - recoverable: back to main scenario (low stock + reduce quantity)
  - non-recoverable: fails (out of stock)
  - each scenario goes from trigger to completion
- Label with step number (success scenario line) and letter
  - 5a <failure condition>; 5a.1 <fail with error message>
  - 5b <failure condition>; 5b.1 <action>; 5b.2 <continue at failure step 7>

Qualities of a good use case

- **Focuses on interaction**
  - Starts with a request from an actor to the system
  - Ends with the production of all the answers to the request

Qualities of a good use case

- **Focuses on interaction**
  - Starts with a request from an actor to the system
  - Ends with the production of all the answers to the request
- **Focuses on essential behaviors, from actor's point of view**
  - Does not describe internal system activities
  - Does not describe the GUI in detail

Qualities of a good use case

- **Focuses on interaction**
  - Starts with a request from an actor to the system
  - Ends with the production of all the answers to the request
- **Focuses on essential behaviors, from actor's point of view**
  - Does not describe internal system activities
  - Does not describe the GUI in detail
- **Concise, clear, and accessible to non-programmers**
  - Easy to read
  - Summary fits on a page
  - Main success scenario and extensions

Use cases vs. other requirements

Which of the following requirements should be directly represented as a use case?

- Special deals may not run longer than 6 months.
- Customers only become preferred after 1 year.
- A customer has one and only one sales contact.
- Database response time is less than 2 seconds.
- Web site uptime requirement is 99.8%.
- Number of simultaneous users will be 200 max.

Formal use case

Name	The Use Case name. Typically the name is of the format <action> + <object>.
ID	An identifier that is unique to each Use Case.
Description	A brief sentence that states what the user wants to be able to do and what benefit he will derive.
Actors	The type of user who interacts with the system to accomplish the task. Actors are identified by role name.
Organizational Benefits	The value the organization expects to receive from having the functionality described. Ideally this is a link directly to a Business Objective.
Frequency of Use	How often the Use Case is executed.
Triggers	Concrete actions made by the user within the system to start the Use Case.
Preconditions	Any states that the system must be in or conditions that must be met before the Use Case is started.
Postconditions	Any states that the system must be in or conditions that must be met after the Use Case is completed successfully. These will be met if the Main Course or any Alternate Courses are followed. Some Exceptions may result in failure to meet the Postconditions.
Main Course	The most common path of interactions between the user and the system. 1. Step 1 2. Step 2
Alternate Courses	Alternate paths through the system. AC1: <condition for the alternate to be called> 1. Step 1 2. Step 2 AC2: <condition for the alternate to be called> 1. Step 1
Exceptions	Exception handling by the system. EX1: <condition for the exception to be called> 1. Step 1 2. Step 2 EX2 <condition for the exception to be called> 1. Step 1

Informal use case

Patron loses a book

The **library patron** reports to the librarian that she has lost a book. The **librarian** prints out the library record and asks patron to speak with the head librarian, who will arrange for the patron to pay a fee. The **system** will be updated to reflect lost book, and patron's record is updated as well. The **head librarian** may authorize purchase of a replacement book.

Nice quick reference

https://www.usability.gov/how-to-and-tools/methods/use-cases.html

What’s next?

WEEK 3		
04/10	L: SCRUM	
04/11	T:	DUE: PR!!!
04/12	L: Version Control	GitHub Project Setup (GPS)
04/13	P:	
04/14	LX: GIT	

What’s next?

WEEK 3		
04/10	L: SCRUM	
04/11	T:	DUE: PR!!!
04/12	L: Version Control	GitHub Project Setup (GPS)
04/13	P:	
04/14	LX: GIT	

Question, please!