Requirements and Specification, ESOF 328, Spring 2020
"Understanding user requirements" (Chapter 8)
Jan. 29

Understanding user requirements, Chapter 8

User requirements lie between business requirements and functional/non-functional requirements

2 techniques for exploring user requirements:

- Use cases
- User stories (came about with agile development)

Both are "user-centric", focus on what *users* want to accomplish not what the *system* should do.

Use cases:

- Goal-oriented set of interactions between an actor and the system that results in an outcome that provides value to the actor
- Name is a verb followed by an object
- Provide a list of steps needed to achieve the goal
- The actor can be a human or an external system
- Can encompass multiple scenarios (alternative flow)

User story:

- One or two sentences that articulate a user need, or desired functionality, and the benefit gained
- Format is:

  As a <type of user>, I want <some goal> so that <some reason>
- Just-in-time information – can fill the story in as information is needed
- User stores can be refined into more focused user story (large user stories called "epics)

Use cases and user stories are used in different ways:

- Use cases – typically go on and define requirements, and maybe tests
- User stories – typically go on to define acceptance cases, requirements aren't developed

Use Case diagram

- Use stick figure for actor
- Ovals are use cases
- Arrows show the connection between an actor and a use case

Context-Diagram versus Use Case Diagram

- Both define boundary between objects and the system
- Context diagram provides no visibility into the system, whereas the use case diagram shows some internal aspects of the system
- Arrows on context diagrams show flow of data, control signals, or physical materials; arrows on use case diagram use show connection between an actor and use case (according to Wiegers). Others show action on arrow. (See slide, not from text)

Use cases:

- Unique identifier
- Short descriptive name
- Short textual description
- List of preconditions - activities that must take place, or any conditions that must be true, before the use case can be started
- List of postconditions - the state of the system at the conclusion of the use case execution
- Normal flow – list and number the user actions and system responses that will take place during execution of the use case under normal, expected conditions
- Alternate flow – list and number the user actions and system responses of other legitimate usage scenarios that can take place within this use case. Number steps to show where these could branch off from the normal flow
- Exceptions – list and number (use E to show an exception) any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions. As with alternate flow, number these to show where they could occur

Write essential use cases rather than concrete

- Essential – devoid of implementation specifics and constraints. An essential model depicts information at a conceptual level, independent of how it might be implemented in a system (page 485).

“Extends” versus “include” stereotypes

- Extends – the extended use case is an alternate course that occurs at a location specified in the base use case (the location is called the extension point)
- Includes – the included use case is always used. Also the included use case never stands alone