Hat Racks for Understanding

Lecture 6

Nadia Noormohamed

[Year]

How to convey or represent the acquired information?

Informal Methods: Involve natural language. This could be forms, letters, memos and interviews. Requirements elicitation can be done by analyzing this language. Nouns and verbs represent actions and functionality. Identify constraints, targets, owners and actors.

**Informal Methods:**

User Stories

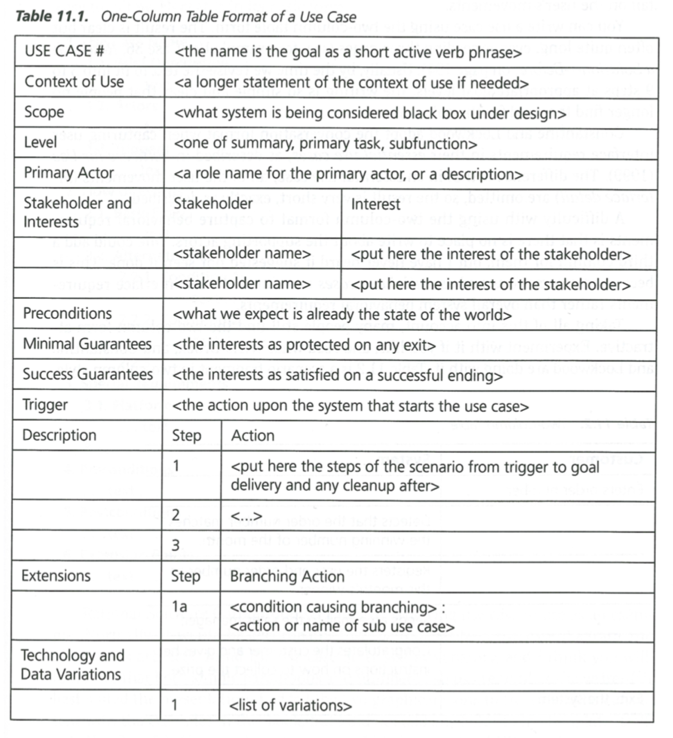
* Describes functionality that will be available to the users of the system.
* Can be of varying length
* Comprise of written description, conversations and details of the story.
* Stories should be independent.
* Can be altered or modified based on new information.
* Story should be estimable in terms of the time it should take to create the functionality.
* Should be testable

Use Cases

* More complex than user stories.
* Describes a system under various conditions
* Creates a unified language so that the UC can be understood by Software engineers, regardless of who created them.
* May take many different forms.
* Consists of:

1. The primary initiator, for example, the purchases, claimant or user.
2. Level of importance
3. List of stakeholders who may be interested

* In the case of the UX analyst UCs serve a perfectly adequate purpose for the necessity of specifying user-facing aspects of the system.
* Normally created as a series of steps or bullet points.



Personas

* It is ambiguous as to what a persona method encompasses.
* Common understanding: description of a fictitious person.
* Description based on assumption or data?
* What should the description cover?
* Related to scenarios and user stories.
* Can be very short.

Scenarios

* Detailed description of a user’s interaction with the computer.
* Larger than use cases
* Written in plain English as opposed to bullet points.
* Many actors and many interrelated stories crisscrossing throughout the scenario.
* Stories are not independent
* Detailed definition of a specific instance of interactivity.
* Easier to create than user stories and use cases.

Wireframes

* Hand drawn sketches of proposed UIs
* Can be created at the same time as requirements capture is happening.
* Can make fast and direct corrections and developments to the interface based on immediate feedback from the user.
* Fast to design
* Can design a computerized wireframe for better presentation.

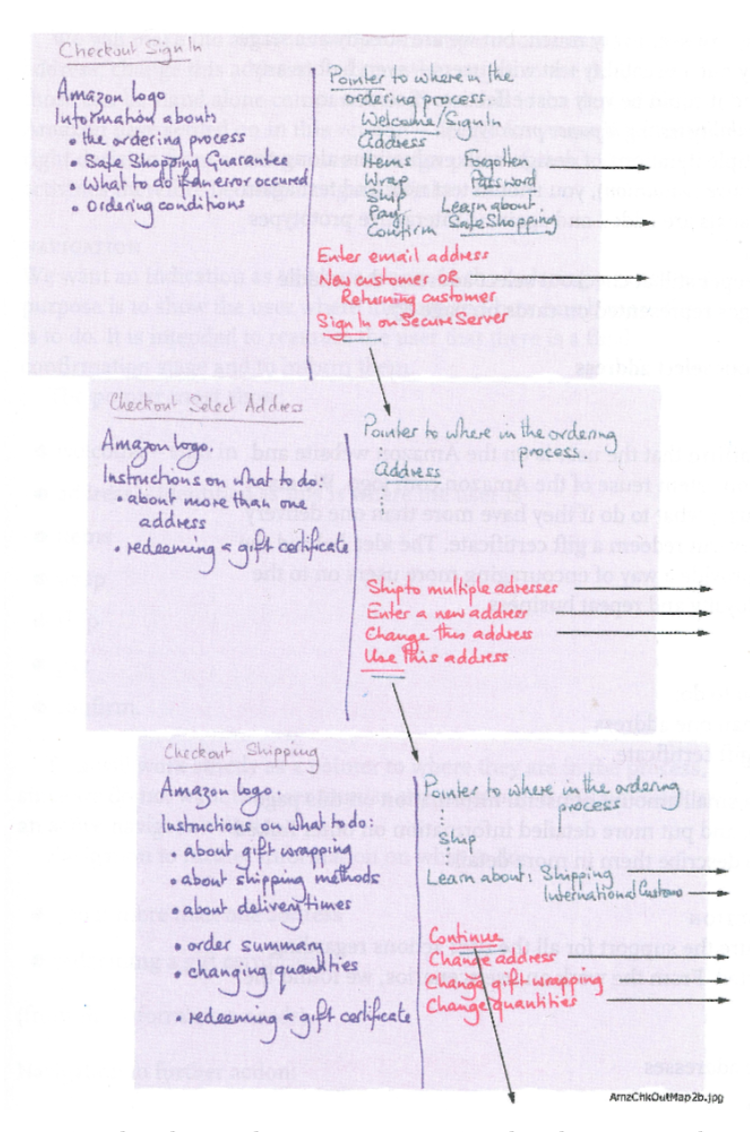


Mock-ups

* Extensions of wireframes
* Can occur as hand drawn sketches or as computer generated interface elements
* Usually contain ‘more’ than wireframes

Storyboards

* Adds interactive elements and interaction design to wireframes or mockups.
* Interactions are usually simplistic



Semi-Formal Methods:

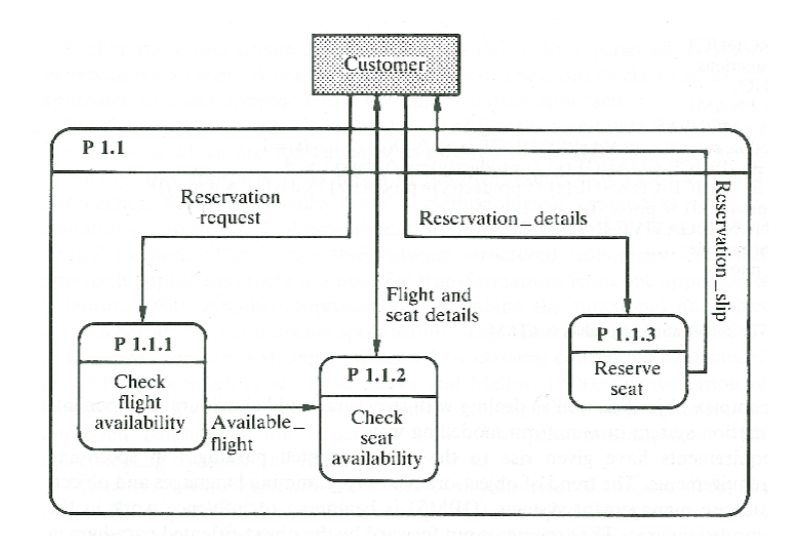
* Software requirements specification: description of the behavior of the system to be developed. Includes a set of use cases that describe all of the interactions the user will have with the system. Use cases AKA functional requirements.
* Also contains non-functional requirements such as performance requirements, quality standards or design constraints.
* Some requirements elicitation processes, other than use cases and user stories, need to be translated to a form that can be understood by software engineers.
* Methods based on modelling languages: Include, UML, flow charts, data flow diagrams and control flow diagrams.

Flow charts

* Describes the flow of actions, control and information
* Each box has a text written within it which describes either a process, choice or artefact.
* Flow is represented by adjoining arrows.

Data flow diagram

* Represents the flow of data through an information system in a graphical format.
* The data processing aspects of the system can be better understood, and detailed analysis of the way information moves around the system, including its storage, can be derived.
* Important for the UX’er because it shows how information moves, what information will be required by the system which could indicate the type of inputs that will be required from the interface and thus implies a certain interaction and behavior that will be exhibited by the user.
* No information if processes will operate serially or in parallel.



Control flow diagram

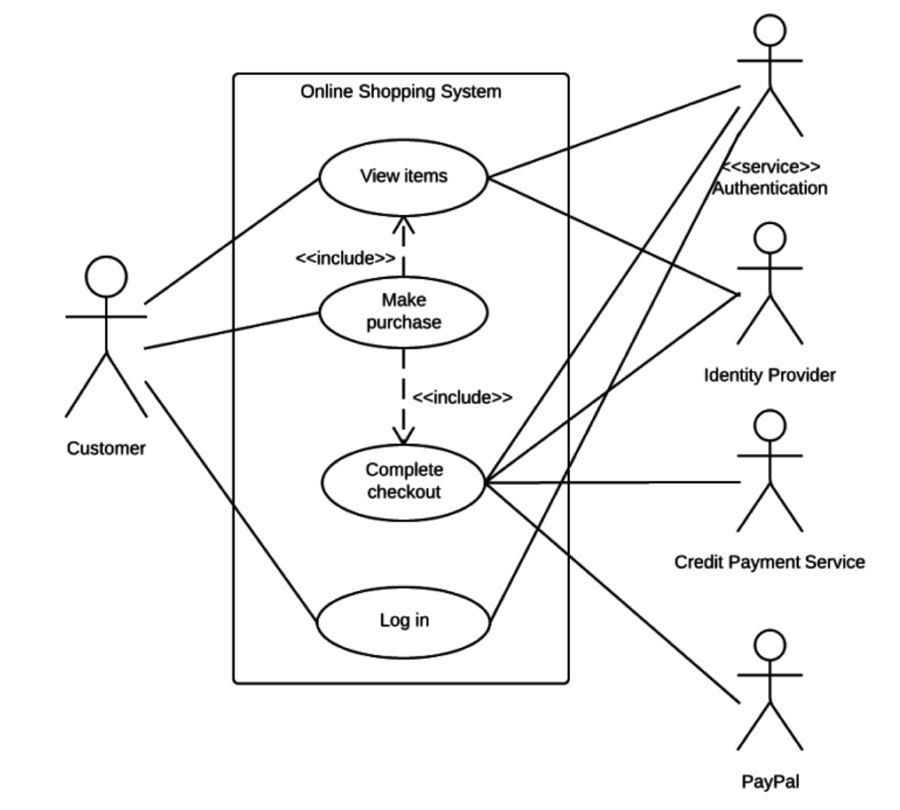
* Represents a sequence of subdivided steps that often represent well-known programming concepts such as if-then-else conditions, repetition, and/or case conditions.
* Annotated boxes and shapes are used to represent operations, data, or equipment.
* Arrows are used to represent sequential flow as with normal flow charts.
* Several types: configuration decision control flow diagram are used in configuration management; quality control flow diagram are used in quality control; change control flow diagram are used in project management; and process control flow diagrams are used in the software engineering practice.
* Used for control flow analysis, data flow analysis, algorithm analysis and simulation.
* Graphical flows easier to analyze.

State transition diagram

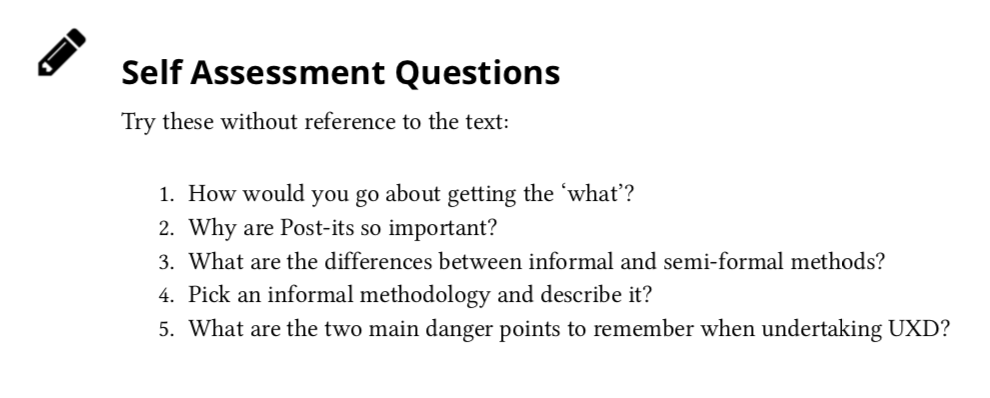
* Each node represents a specific system state
* Used to describe behavior of the system
* Finite number of states
* States could be tangible or abstract
* Directed arrow: transition between states. Labelled with transition states.
* Each node is divided in half with the state of being written in bold within the top half and the entry action written in the bottom half (state that currently must exist for the new state to be entered)
* Many possible representations. A possible representation is a state table.
* …

The Unified Modelling Language (UML)

* The most common element is called the class diagram. Rectangular box divided into 3 separate sections. Name of the class, variables, methods.

Use case diagrams

* Show relationship between actors and use cases within a system.
* Provide an overview of usage requirements



1. Need to do requirements capturing. This can be done through doing focus groups, interviews. If there are not enough users, can use archives and similar developments to do this.
2. Post-it notes are used in the requirements elicitation process. Used to convey design information to Software engineers. Computerizing the process requirements elicitation is inflexible relative to post-it notes. Because once the information is added it is difficult to change. Post it notes are flexible because anything can be written or designed on them, they can be moved around easily, moved to one side If the idea is not definite at the moment. They can be used comfortably by users as they are familiar to almost anyone.
3. Need to read page 80 to 87 to answer this q
4. Wireframes are usually drawn. They depict the rough user interface based on the users requirements. They are fast to form and can be added to and changed during the requirements capture process.
5. Look at page 69 and 70