Users and Use Cases

- User Characterization
- Use Cases
  - what are they, why they are interesting
  - how they differ from specifications
- Use Case Development
  - identifying and characterizing classes of users
  - identifying and elaborating usage scenarios
- Representing Use Cases
  - XP user stories
  - UML (use case, activity, state) diagrams

User Characterization

- User Characterization
  - identifying distinct sub-classes of users
- Common Criteria
  - product use roles (or intentions)
  - knowledge (domain or technology)
- Results in a profile for each user class
  - goals and expectations
  - knowledge and experience
  - typical usage scenarios

Use Case

A stylized story about how an end user (in a specified role) interacts with the system under a specific set of circumstances.

In requirements, it captures a contract that describes the way the system should behave in response to a specified user request.

Use Cases vs. Specifications

- Specifications are “abstract descriptions”
  - required behavior and other characteristics
- Use Cases are “stories”
  - scenarios the product must support
  - key instances of user/product interactions
- Use Cases are a better starting point
  - they are easier to gather and review
  - they more honestly capture requirements
  - they do not describe implementations
  - a good starting point for specifications

Scenarios, Tasks, & Steps

- Scenario … representative work session
  a sequence of related tasks to solve a problem
  (e.g. handle a customer phone call)
- Task … smallest interesting unit of work
  sequence of steps culminating in useful a result
  (e.g. scheduling an appointment)
- Step … smallest interesting unit of action
  one individual action in a sequence
  (e.g. entering a password)

Use Cases

- Are a really good form for requirements
  - they tend to be concise
  - they tend to be very concrete
  - they are tied to real-world problems
  - they are expressed from the users’ perspective
  - they describe functionality (vs. design)
  - they help us understand the user’s world view
- They are also easy to develop/validate
  - ask people what they do, or simply watch them

Developing Use Cases

- They can be based on existing processes
  - interview potential users
  - ask them to describe the things they do
  - identify the tasks within their scenarios
  - get descriptions of steps within those tasks
  - get descriptions of problems and exceptions
- Use these as a starting point
  - design to enable the same tasks & scenarios
  - design to deal with the problems & exceptions

Developing Use Cases

- They can be brain-stormed
  - who are the primary actors?
  - what tasks do they need to perform?
  - what information is needed to perform task?
  - what information is user interested in?
  - what could go wrong in performing task?
  - how would user want to be informed of these?
- These suppositions must be validated
  - by interviews
  - by usability testing with prototypes

Representing Use Cases

- Use cases capture product capabilities
  - describe what product should be able to do
  - may describe detailed user/system interactions
- They can be represented in many ways
  - XP user stories overview of an operation
  - UML Diagrams
    - use case actors, objects, operations among them
    - behavior detailed interactions among actors & objects
    - state observable states and transitions
  - user interface prototypes or mock-ups

User Story Cards

- Brief summary of a desired capability
  - name of story
  - brief general description of a useful behavior
  - user assigned priority
  - development estimated cost
- All on a single 3x5 card
  - just enough to capture the concept
  - specifically not a complete specification
    - this is established between user and developer

Sample User Story Card

Story #107: buy parking permit

Any registered student can buy a parking permit.
Can choose semester or annual.
Need to provide car license plate number.
Payment options:
credit card, PayPal, charge to student account

Priority: high Estimate: 3 days

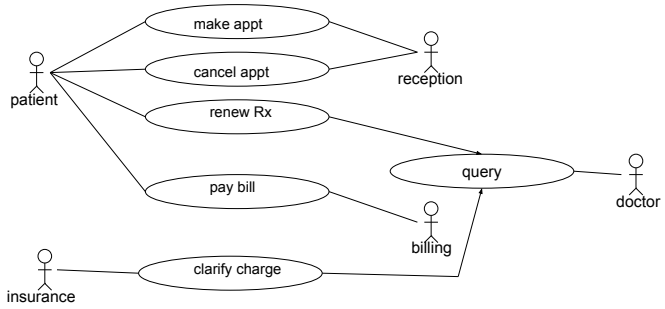
Using User Story Cards

- Product Descriptions
  - a product is a collection of features/capabilities
- Work planning tokens (the planning game)
  - each represents a requested development task
  - they can be laid out on a table
  - they can be laddered, grouped into releases
- Project Management metrics
  - completed cards record work accomplished
  - completion rate measures project velocity

UML Use Case Diagrams

- A graphical overview of capabilities
  - a list of defined actions (use cases)
  - shows boundary between system and actors
- Use Case diagrams identify
  - classes of actors who can initiate actions
  - actions (use cases) each can initiate
  - other people (or objects) that will be affected
- They do not describe the interactions
  - that is left to more detailed behavioral models

Sample UML Use Cases



Using UML Use Cases

- an overview of system functionality
  - what are the general classes of users
  - what are the operations each can perform
- a summary of system capabilities
  - what are the basic things it can do
- a table of contents for the use cases
  - enumerating the things to be defined
- May be an introduction to the solution
  - external object classes and methods

How they compare

- User Story Cards
  - summary of a specific functional capability
  - not a description, but a feature planning token
- UML Use Case Diagrams
  - enumerate the actors, objects and operations
  - a single snapshot of the major capabilities
- Use Case/Scenario Descriptions
  - detailed behavior descriptions/specifications
  - basis for designs and acceptance tests

Modeling more detailed behavior

- Use-cases and story cards are imprecise
  - they capture intent, but not details
- Activity Diagrams
  - steps in a (potentially distributed) process
  - sequences of events
  - key decisions
- State Models
  - can be user-visible or internal states
  - events that cause changes between them

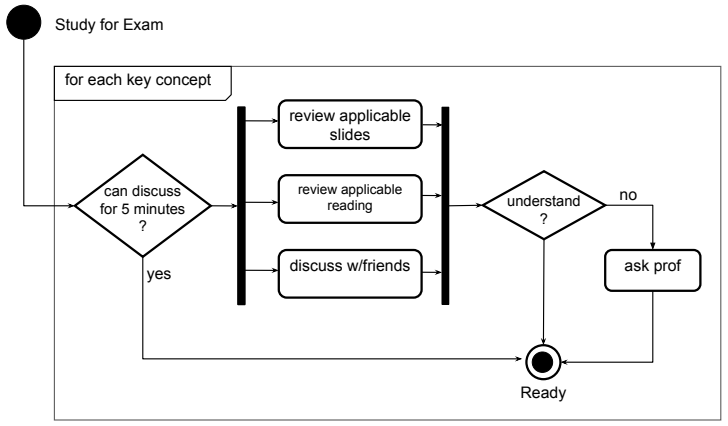
Universal Modeling Language

- a family of related graphical notations
  - for representing software system designs
  - particularly those built in object oriented style
- supports a wide range of uses
  - a design sketching language
  - a system specification language
  - a programming language
- also has a textual (XML) representation
  - enabling development of CAD tools

UML General Conventions

- square boxes represent things
  - classes, objects, packages, components
  - 3D boxes represent physical things
- round boxes represent processes/states
- circles represent class interfaces
  - entry-points
- arrows represent relationships
  - flow, communication, dependency
  - arrow heads determine type and direction
- UML diagrams can be nested/composed

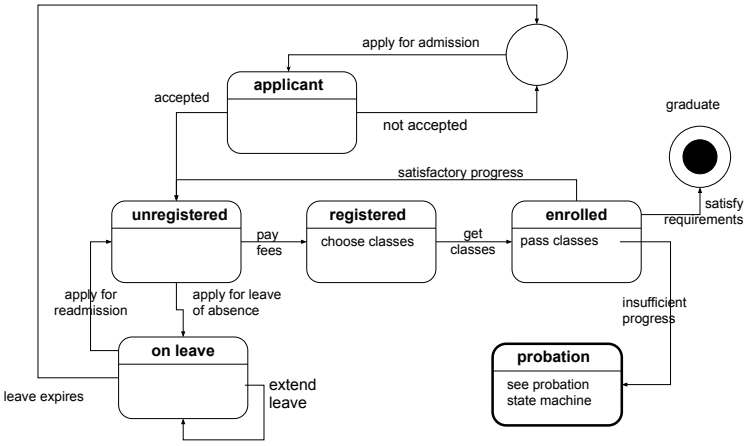
UML Activity Diagrams



(UML Activity Diagrams)

- Describe processes (not just algorithms)
  - in terms of steps (decisions and actions)
- a standard UML representation
  - rounded “capsules” represent activities
  - arrows represent temporal sequencing
  - diamonds represent decision & merge points
  - bars delimit parallel activities
- excellent for behavioral requirements
  - illustrative sample usage scenarios
  - additional detail for a use-case or story card

UML State Diagrams



(UML State Models)

- describe state/transition models
  - where events drive state changes
- a type of activity diagram
  - activity boxes have two compartments
    - state name in the top portion
    - processing steps in the bottom portion
  - arrows represent state transitions
    - from previous state, to next state
    - labels describe conditions triggering the transition
    - processing steps can also be placed on lines

Team Exercise

- identify distinct classes of your users
  - what usefully distinguishes each sub-class
- identify key operations involving each
  - and the participants in each
  - diagram these as UML use cases
- you will be asked to briefly discuss
  - why you chose those classification criteria?
  - how well this diagram captures your product?

Requirements Elicitation Session

- when
  - next lecture – in class
- goals
  - to demystify the process
  - to allow you to observe the process
  - provide an example we can discuss
  - to prepare you for your own sessions
- preparation
  - review chap 3 of my User Requirements paper

For Next Lecture

- McConnell Ch 20
  - overview of quality assurance techniques
- Spolsky – Five Worlds
  - different types of S/W, different problems
- Rakitin – What is S/W Quality Assurance
  - history and overview of roles
- Boehm – Software Defect Reduction
  - problems: locations, costs, and remedies
- Wiegers – Peer Reviews
  - where peer reviews fit in to the process