COMP 7012

Midterm Study Guide

Spring 2012

Development Problems and Processes

- Know/understand the problems of software development:
  - Requirements
  - Cost estimation
  - Bugs
  - Maintenance
- Know what the waterfall process model is and why it's considered a bad practice.
- Understand what iterative development is and why we do it.
  - Understand the difference between "defined" and "empirical" control models.
  - Know long development iterations should be.
- Understand what agile methods are and why we use them.
  - Know the agile values from the Agile Manifesto.
- Know the difference/relationship between the following activities:
  - Requirements
  - Analysis
  - Design
- Know the "secret of modeling".

Requirements

- Be able to define each of the FURPS+ requirement types.
- Be able to identify the type (from FURPS+) of a requirement.
- Be able to specify requirements of a given FURPS+ type.

Use Cases

- Be able to specify "brief", "casual", and/or "fully dressed" UCs.
  - Note: I won't ask you to recall what all the parts of a "fully dressed" UC are from memory, but if I give you a list of the parts, you should know what they are and how to fill them in.
- Understand the (UI-free) "essential style" for UCs.
  - Be able to apply the style.
  - Be able to critique UCs with respect to the style.
- Understand the "black-box" style for UCs.
  - Be able to apply the style.
  - Be able to critique UCs with respect to the style.
- Understand the actor and actor-goal perspective with respect to UCs.
  - Be able to apply the perspective.
  - Be able to critique UCs with respect to the perspective.

Domain Models

- Be able to represent a Domain Model as a UML class diagram based on a textual specification.
    - Know/apply the noun-phrase technique.
    - Know/apply thinking like a mapmaker.
- Be able to read/interpret/create UML class diagrams.
    - Know/apply classes, attributes, and associations.
    - Know/apply multiplicities.
    - Know/apply generalization relationships.
        - Know/apply the “100% Rule”.
        - Know/apply the “Is-a Rule”.
        - Know when to model a subclass.
    - Know/apply abstract classes.
    - Know/apply composition relationships.
        - Know the 3 implications of a composition relationship.

System Sequence Diagrams

- Be able to represent a scenario of interaction with the System as an SSD.
- Know/apply the UML Sequence Diagram notation.

Subversion (SVN)

- Know the basic SVN client operations and how they affect the repository and/or your working copy.
- Be able to choose the client operation(s) to apply in a given situation or to solve a particular problem.