SE2890 Software Engineering Practices

Dr. Walter Schilling

Spring, 2012-2013

On the midterm and final exam, you are permitted one 8.5 x 11 sheet of paper with notes.

The following paragraph describes a system which you have been asked to design and implement. This system will be used for several questions on the exam. Read through the system description so you are familiar with it and think about the potential design questions which can be asked.

> A software system, "MSOE ID Manager" is to be developed which will control the operation of a photo ID system. Essentially, the system is to be used by campus security to automatically issue photo ids to students. The photo id system communicates with an already existing student database as well as an existing credit card processing system.
>
> The system must be capable of dealing with new students. In the case of a new student, the student has never been issued an ID before and there will be no charge for the id. The student will sit down in front of an existing camera and their photo will be taken by a security officer. The officer will enter information about the student (name, height, weight, etc.) which will then be stored on the photo id as well as in the student data base. The photo id will then be thermally printed by an existing thermal printer.
>
> The system must also be capable of reporting lost ids. A student who has lost their id may report that it is missing online on a web page, at which point all access allowed via that id will be terminated. To do this, the student must authenticate with the system to indicate their credentials as well as perform appropriate steps to indicate that their id is lost.
>
> A student who has lost an id may come in to have a replacement id issued. In this case, the system will check to make certain the student does not have a hold placed on their account by the bursor's office which would prevent a new id from being issued. (A hold would be indicated in the existing system that the bursor's office uses.) The student must then present a valid credit card which will be billed for the replacement cost of the id ($25.00). The student will then have their photo taken and a new id will be printed.
>
> A student who has gotten married or divorced may request a new id at no charge. In this case, the student must submit legal proof of their name change in the form of a marriage certificate, divorce decree, or other court document. This information is verified by campus security, but is not entered into the system in any manner. The operator will then follow the procedure for issuing a replacement id.
>
> Campus security also can invalidate the student IDs of graduating senior after commencement. This will involve security logging on and uploading a text file of graduates to the system.
>
> In the event of a student being reported as missing to campus security, the system will allow the security officer to automatically generate a missing persons report, including the students name, height, weight, date of birth, and photo and submit it to the police.
>
> Authentication for the system is handled via a user name and password combination. This information is stored on an existing LDAP server, and is the same username and password used to access all other MSOE records.

1. Week 1 - Introduction
   (a) Lecture 1: Introduction
      i. Recognize the economic impacts of software failure
      ii. Discuss the ramifications of software failure
      iii. Understand that to improve software quality, we must learn from our past failures
      iv. Explain the concept of the CMM and CMMI process initiatives.
   (b) Lecture 2: Software Development Processes
      i. Explain the difference between general software and real-time software
      ii. List the three tracks of software engineering lifecycles
      iii. Define the term software process and software methodology

iv. Compare and contrast the aspects of the Waterfall model, the V model, spiral models, and other software development models.
v. Explain the ROPES Software Development Process (Rapid Object Oriented Process for Embedded Systems)
vi. Explain the relationship between Systems Engineering and Software Engineering
vii. Define the concept of an Architecture

2. Week 2 - Requirements and Use Cases

(a) Lecture 1- Requirements
i. Recognize the relationship between different types of requirements within the realm of software engineering
ii. Compare and contrast functional requirements, non-0functional requirements, and constraints.
iii. Critique the wording of a requirement and constraints.

(b) Lecture 2 - Use Cases
i. Define a use case
ii. Interpret the meaning of a use case diagram.
iii. Define Actor
iv. Explain the relationship between Use Case Diagrams and Use Case Scenarios
v. List the items present in a use case scenario
vi. Construct a use case scenario for a given problem
vii. Explain how the level of detail in use cases may change throughout the phases of the software development process.

3. Week 3

(a) Lecture 1 - Introduction to Version Control and Configuration Management
i. Explain the concept of a version control system
ii. Recognize the importance of disciplined configuration management to the successful completion of a software development project
iii. Understand the model of a client server configuration management system.
iv. Define repository
v. Define local working copy
vi. Explain the concepts of checking out, committing, and updating.
vii. Explain the concept of merging code files

(b) Lecture 2 Object Domain Analysis
i. Explain the purpose for Object Domain Analysis.
ii. Explain the relationship between the use case model and the Object Model
iii. Explain the mechanism used to connect object domain models with use case models
iv. Apply key strategies for identifying objects within a problem domain
- Noun Strategy
- Services
- Physical Devices
- Persistent Information

4. Week 4 Object Domain Analysis

(a) Lecture 1 Domain Modeling - Part 2
i. Define aggregation and inheritance in terms of UML.
ii. Critique a model for correct usage of UML.
iii. Explain the difference between compact and expanded versions of UML.
iv. Review Aggregation and composition in UML (Lab Review)
v. Construct an object domain model for a given problem based upon a use case

(b) Lecture 2 Defining Object Behavior
i. Explain the relationship between simple, state, and continuous behaviors.
ii. Define entry actions, exit actions, and activities.
iii. Explain the transaction syntax for transitions.

iv. Define guard condition.
v. Define object state behavior using a UML state charts
vi. Represent concurrent state machine behavior using Harel state machines.

5. Week 5 Defining Object Behavior

   (a) Lecture 1 Defining Object Behavior

      i. Define the 4 types of systems and their behaviors.
      ii. Categorize systems into the appropriate classifications.
      iii. Explain the steps in object interaction modeling.
      iv. Critique examples of notation for correctness.
      v. Construct interaction diagrams to define the interactions between multiple classes.

   (b) Lecture 2- Introduction to Software Reviews

      i. Explain the concept of a software review
      ii. Illustrate the flow for a typical software review
      iii. Explain the differences between an inspection and a walkthrough.
      iv. Recognize the applicability of checklists to improving the review process
      v. Identify the risks of reviewing at too fast or too slow of a rate.

6. Week 6

   (a) Lecture 1 Midterm Exam

      i. Pass the exam with an appropriate score.

   (b) Lecture 2 Design and Design Patterns

      i. Define design.
      ii. Define the architectural, mechanistic, and detailed phases of design.
      iii. List the aspects of a design pattern.
      iv. List existing design patterns.
      v. Explain the Observer pattern.
      vi. Draw the observer pattern structure.
      vii. Implement an observer pattern in Java.
      viii. Justify the need for using an observer pattern in implementing Java programs.

7. Week 7

   (a) Java Multithreading

      i. Define thread.
      ii. Explain the purpose for the Runnable interface.
      iii. Draw a representation for the stack when multiple threads are executing.
      iv. Implement a simple multithreaded application in Java.
      v. Draw the thread lifecycle.
      vi. Explain how threads are indicated by the design of a state machine.

   (b) Lecture 2 - Implementing State Charts in Source Code

      i. Explain the decisions made during detailed design, including data structure, class refactoring, associations, operations, visibility, algorithms, and exception handling.
      ii. Explain the ramifications of data structure during detailed design, including but not limited to numeric types, precision, and range.
      iii. Compare and contrast absolute error and relative error.
      iv. Explain the usage of constraints in detailed design.
      v. Construct Java Interfaces matching the detailed design
      vi. Define a generic state interface in Java including the capability to define entry action, exit action, and do action.
      vii. Translate state charts into Java using interface patterns and state charts.

8. Week 8

(a) Lecture 1 Software Testing

i. Explain the relationships between unit testing, integration testing, and system testing.
ii. Explain the testing process.
iii. Explain the relationship between test plans, test cases, and test procedures
iv. Develop rudimentary test cases for a software package from requirements and use case scenarios.

9. Week 9

(a) Lecture 1 Software Testing

i. Explain the purpose for a testing framework.
ii. Explain the purpose for the JUnit framework.
iii. Implement rudimentary test cases using an automated test framework.
iv. Explain the key aspects of a quality bug report
v. Explain how bugs can be reported using a bug tracking system.

(b) Lecture 2 Code Reviews

i. Provide examples of common faults programmers make when implementing source code
ii. Discuss the appropriate rates for code reviews
iii. Explain the importance of checklists when reviewing code
iv. Introduce the concept of code beautifiers and code formatting to ease code reviews

10. Week 10

(a) Lecture 1 Introduction to Project Tracking and Analysis

i. Explain the concept of Earned value analysis
ii. Explain the concept of "ahead of schedule", "behind schedule", and "on schedule"
iii. Explain how earned value tracking can aid in keeping a project on schedule.

(b) Lecture 2 Final Class Review and Course Evaluations

i. Perform final review
ii. Assess course effectiveness.