SWEN 383-01 Exam 1 Topics

Format

- 50 Minutes
- Closed book, reference sheet provided with the class diagram of each pattern covered
- Optionally, you may bring one 8 1/2 x 11 notes sheet, both sides OK
- Format is short answer and small design problems

Topics

(Based on class room discussion and resources provided via the course website and myCourses.)

- Design's role in the software development lifecycle - cost of change, functional vs non-functional requirements
- Design Principles
    - Coupling & Cohesion
    - Separation of Concerns
    - Refactoring
    - DRY - Don't Repeat Yourself
    - DIP - Dependency Inversion Principle
    - P2I - Program to an Interface, not an Implementation
    - Object Creation - factory methods
    - Dependency Injection
    - Open-Close Principle
- Design Patterns (only those we have covered in class)
    - Observer
    - Adapter
    - Factory Method
- Version Control Systems (concepts only, no specific GIT commands)
- UML Notation
    - Class Diagrams
    - Sequence Diagrams

Sample Questions

Below are three 10-point question sets similar to those you will see on the first exam. It is guaranteed that *at least* one of these question sets will be among the ones that comprise the exam. Feel free to work with anyone in the class or to consult any material on the Internet in developing your answers. You may also ask questions of your instructor, but he or she will only answer those which serve simply to clarify what is being requested.

Remember: You are allowed to bring one **8-1/2 inch by 11 inch sheet** of paper to the exam; you may put whatever you want on **both sides** of the paper. Your instructor may ask you to turn in your sheet at the end of the exam. You will also be given a handout with pattern descriptions (including some that have not been covered and thus are not applicable to this exam).

If you need any special accommodations for the exam, please make me aware of this via email and provide me with all the materials I need before end of work this week!

Q1 Design Principles

What is meant by the term *cohesion* in the context of design? Do we want cohesion to be low or high? Describe why.
(3 points)

What is meant by the term *coupling* in the context of design? Do we want coupling to be high or low? Describe why this is so.
(3 points)

It's often claimed that cohesion and coupling are in tension; for example, improving one might worsen the other. If you had to favor improving one over the other, which would you choose? Describe positive and a negative way in which your decision might impact the overall qua**lity** of your software design?
(4 points)

Q2 Version Control

A team of developers are collaborating on a common set of files. One developer suggests that the team use a file "locking" technique to prevent concurrent edits to the same file from occurring. Another team member suggests the use of a version control system, like GIT.

Compare and contrast two advantages to using a version control system as opposed to manually tracking multiple versions of files.
(4 points)

Is “locking” a file a reasonable solution to prevent concurrent editing (Yes/No)? State an advantage or disadvantage for this approach based on your answer.
(3 points)

What version control mechanism is used to support the maintenance of a software product, such as adding new features or correcting a defect? Support your answer with a brief example.

(3 points)

Q3 Refactoring & Technical Debt

(hint: see Week 2, Software Design Glossary)

What is technical debt? What causes technical debt to increase?
(3 points)

How is progress in maintaining and enhancing a software product affected when the product is carrying a high technical debt load? Briefly justify your answer.
(4 points)

A product's code may change due to (a) continuing development (b) refactoring. What differs between these two types of change? That is, what is different between the goals of (a) and (b)?
(3 points)