**Final Exam – Software Engineering Advanced Topcis**

1. Describe the three key assumptions regarding software projects that every agile software process must address.

Answer (Section 5.3):

It is difficult to predict in advance which software requirements and customer priorities will change and which will not.

For many types of software design and construction must be interleaved, it is difficult to predict how much design is needed before construction can be used to prove the design.

Analysis, design, construction, and testing are not always predictable processes and this makes planning difficult.

1. List the characteristics that should be considered when considering potential classes for inclusion in an analysis model

Answer (Section 10.1):

Contains information that should be retained

Provides needed services

Contains multiple attributes

Has common set of attributes that apply to all class instances

Has common set of operations that apply to all object instances

Represents external entity that produces or consumes information

1. Under what circumstances should requirements modeling be utilized for Web or mobile apps?

Answer (Section 11.5)

Large or complex app to be built

Large number of stakeholders

Large number developers onapp team

Development team members have not worked together before

App success will have strong bearing on success of company

1. Describe practices that enable designers to think about using patterns

Answer (Section 16.2.2):

Be sure you understand the big picture (context in which the software will reside)

Extract the patterns that are present at that level of abstraction in the big picture.

Begin your design with ‘big picture’ patterns that establish a context or skeleton for further design work.

“Work inward from the context” looking for patterns at lower levels of abstraction that contribute to the design solution.

Repeat steps 1 to 4 until the complete design is fleshed out.

Refine the design by adapting each pattern to the specifics of the software you’re trying to build.

1. List four content architectures found in WebApps?

Answer (Section 17.7.1):

Linear structures

Grid structures

Hierarchical structures

Networked or pure web structures

1. What are the steps in the engineering process model for developing MobileApps?

Answer (Section 18.2):

Formulation

Planning

Analysis

Engineering

Implementation and testing

User evaluation

1. Discuss how poor management decisions can impact software quality?

Answer (Section 19.3.6):

Estimation decisions – irrational delivery date estimates cause teams to take short-cuts that can lead to reduced product quality

Scheduling decisions – failing to pay attention to task dependencies when creating the project schedule may force the project team to test modules without their subcomponents and quality may suffer

Risk-oriented decisions – reacting to each crisis as it arises rather than building in mechanisms to monitor risks and having established contingency plans may result in products having reduced quality

1. Describe statistical quality assurance?

Answer (Section 21.6):

Information about

software defects is collected and categorized

Each defect is traced back to its cause

Using the Pareto principle (80% of the defects can be traced to 20% of the causes) isolate the "vital few" defect causes

Move to correct the problems that caused the defects in the “vital few”

1. Describe three partitioning strategies that can be used when performing class level testing for OO systems.

Answer (Section 24.5.2):

State-based partitioning - tests designed so that operations that cause state changes are tested separately from those that do not

Attribute-based partitioning - for each class attribute, operations are classified according to those that use the attribute, those that modify it, and those that do not use or modify the attribute

Category-based partitioning - operations are categorized according to the function performed: initialization, computation, query, or termination

1. Describe the WebApp interface testing strategy.

Answer (Section 25.4.1):

Interface features are tested to ensure that design rules, aesthetics, and related visual content is available for user without error.

Individual interface mechanisms are tested using unit testing strategies.

Each interface mechanism is tested in the context of a use-case of navigation semantic unit (e.g. thread) for a specific user category

Complete interface is tested against selected use-cases and navigation semantic unit to uncover interface semantic errors

Interface is tested in a variety of environments to ensure compatibility

1. What types of testing should be followed in a comprehensive MobileApp testing strategy.

Answer (Section 26.2):

Conceptual Testing

Unit and System Testing

User Experience Testing

Stability Testing

Connectivity Testing

Performance Testing

Compatibility Testing

Security Testing

Certification Testing

1. Describe the process of writing a formal specification for some system function.

Answer (Section 28.6):

Define a system state in terms of the objects manipulated by the function

Using a simple mathematical function as an example, define the data invariant by writing data relations that will not change during the execution of the function

Write the precondition and postcondition for the function using mathematical notation to show the system state before and after the

1. Describe the five activities associated with the software measurement process.

Answer (Section 30.1.3):

Formulation - derivation of metrics and measures that are appropriate for the software representation being considered

Collection - mechanism used to gather the data used to derive the metrics

Analysis - metric computation and use of mathematical tools

Interpretation - evaluation of metrics results to gain insight into the quality of the software representation

Feedback - recommendations derived from the interpretation are transmitted to the software team

1. Describe the role of class-oriented metrics in assessing the quality of an OO system.

Answer (Section 30.3.3):

The class is the fundamental unit of an OO system. The number of methods and their complexity are directly related to the effort required to test a class. The depth of the inheritance tree can be used to estimate the complexity of the class hierarchy. It is important to strive to have low coupling between classes and high cohesion within each class. If large class sizes are detected during OOA review, the classes should partitioned to improve the modularity of the system and make it easier to maintain.

1. Explain how size-oriented metrics differ from function-oriented metrics. Discuss the pros and cons of each.

Answer (Section 32.2.3)

Size-oriented measures are computed by normalizing direct measures of the software engineering process (e.g. effort or defects) over the product size, measured in lines of code. Function-oriented measures are indirect measures that are computed from measures of the information domain of a business application and an assessment of its complexity. Size-oriented metrics are relatively easy to collect, but can present problems when component-based or visual programming methods are applied. Function-oriented metrics can be determined much earlier in the software cycle, but are an abstraction that is open to interpretation.

1. Describe all activities that must occur in order to produce a Risk Mitigation, Monitoring, and Management Plan.

Answer (Chapter 35.7):

Risk Identification - determine the risks that are appropriate

Risk Projection - determine the likelihood that each risk will occur and the damage likely to occur

Risk Mitigation - figuring out strategies to avoid the risks

Risk Management and Contingency Planning - assuming each risk becomes a reality determine ways to limit their impact

1. How does business process reengineering (BPR) differ from software reengineering?

Answer (Section 36.4):

BPR defines business goals, identifies and evaluates existing business processes, and creates revised business processes that better meet current goals. The software reengineering process activities have as their intent the creation of new versions of existing programs that exhibit higher quality and higher maintainability.

1. What characteristics need to be exhibited by organization to improve its software process?

Answer (Section 37.1):

Management commitment and support for SPI, staff involvement throughout SPI process, process integration into organization culture, customizable SPI strategy, solid management of the SPI project

1. Explain the differences between model-driven and test-driven software development?

Answer (Section 38.5.5 and 38.5.7):

Model-Drive Software Development

Couples domain-specific modeling languages with transformation engines and generators in a way that facilitates the representation at a high level and transforms it to lower levels

Domain-specific languages (DSML) represent application structure, behavior, and requirements within particular application domains and are described with meta models that define key semantics and constraints associated with domain concepts

Test Driven Development

Requirements for a software component serve as the basis for the creation of a series of test cases that exercise the interface and attempt to find errors

Test cases are created before source code

Code is developed in very small increments (one sub-function at a time)

Each new increment requires its own set of test case that must be passed and regression is performed as well