**Name : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_                                                                                                 ID : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**COMP 2920 : Software Architecture & Design**

**Quiz #1**

Date: 13th October 2017                                                                                                                weight : 3%

Q1. Find the correct alternative

1. Enterprise and system architecture may
2. provide an environment in which a software may operate
3. provides the requirements and constraints to which software architecture must adhere.
4. are likely to be associated with one or more software architectures.
5. **all of the above**
6. none of the above

1. Software architecture is concerned with:
2. **Software elements of the system**
3. Human elements of the system
4. Hardware elements of the system
5. All of the above
6. None of the above

1. Goal of the Software architecture is NOT to:
2. **Hides the details of what software elements do internally.**
3. Try to address the requirements of various stakeholders
4. Describes the algorithms, data structures and all tools used by the software programmers.
5. Handle both functional and quality requirements

1. What are the notations for the Use case Diagrams ?   
   a) Use case   
   b) Actor   
   c) Prototype   
   d) **a and b**e) a,b and c

1. Find the ODD one out:
2. The software architecture reveals early design decisions about a system.
3. The software architecture enables communication among stakeholders regarding the system.
4. **The software architecture defines the test procedures**
5. The software architecture is a transferable, reusable abstraction of a system.

1. The least influence on the design of a software architecture is of:
2. The architect's background and experience
3. The technical environment in which the system is developed
4. **The way in which the architecture is represented**
5. The structure of the development organization

1. Why is Requirements Elicitation a difficult task?

a) Problem of scope   
b) Problem of understanding   
c) Problem of volatility   
d**) All of the mentioned**

1. What a good Software Requirements Specification (SRS) should NOT be ?
2. Verifiable
3. **Ambiguous**
4. Complete
5. Traceable

1. What is the diagram that represents the structural elements and their interfaces by which the system is composed?
2. **Use case diagram**
3. Activity diagram
4. Sequence diagram
5. Class diagram

1. Which quality attribute would you consider most important for the development of Google type search query system?
2. Modifiability
3. Testability
4. **Performance**
5. Security

Q2. State true or false

1. UML sequence diagram is a good way to represent the interaction between the classes for a specific scenario.        **T**
2. Every software system has an architecture.     **T**
3. Informal language is a typical and valid way to write down use cases.   **F**
4. Maintenance is one of the stages of software development.  **T**
5. Design and architecture are often referred to as the \what" of the system and requirements as the \how".    **F**
6. The output of architectural design process is SRS.   **F**
7. Static analysis aims at recovering the structure of a software system, while dynamic analysis focuses on its run time behavior.  **T**
8. MVC pattern has the benefit of separation of concerns and modularization.  **T**
9. Composition allows more code reuse as compared to inheritance as the two objects don't fit into a single class taxonomy, whereas inheritance is more useful to represent a domain problem.  **T**
10. The software architects should involve stakeholders early in the life cycle of a system so that they may manage the stakeholder expectations.  **T**

 Q3.  study the following case study

Name: Place Order

Actors:

* Shopper
* Fulfillment System
* Billing System

Use Case Description: After the user has selected items to purchase and then order the items. The user will provide payment and shipping information. The system will respond with confirmation of the order and a tracking number that the user can use to check on order status in the future. The system will also provide the user with an estimated delivery date for the order, which will include all selected items. The user may already have an account with the company with billing and shipping information.

1. Design a use case diagram

1. Sketch the basic class diagram

Q4. **Separation of concerns** (**SoC**) is a design principle. Explain it with an example.

**Separation of Concerns (SoC)** – is the process of breaking a computer program into distinct features that overlap in functionality as little as possible. A concern is any piece of interest or focus in a program. Typically, concerns are synonymous with features or behaviors.

It leads to decoupling of distinct Logical Units with well defined Boundaries (responsibilities).

Ex: Model-View-Controller (MVC) design pattern is an excellent example of separating these concerns for better software maintainability.

Q5. What is the fundamental characteristic of a repository architecture or MVC architecture? Explain it with a diagram

 A repository architecture is a system that will allow several interfacing components to share the same data. Each component interfaces the same dataset that is utilized system wide. Data manipulation taking place in one component will reflect an identical representation of data in another component. Components can be interchanged and are independent of other system components. A good example of a repository architecture would be a database management system. Such a system would provide both a console and graphical user interface to update both the structure and dataset of any particular database.

 Model-View-Controller (MVC) architectural design pattern is about different layers having different concerns for better software maintainability. Each service may be hosted on different or same server. Several clients may run concurrent instances of any given service simultaneously. One server can be replaced or undergo maintenance without affecting the entire system. A great advantage of this architecture is it provides a possibility of integrating geologically separated components all into one system. A good example of a MVC architecture is a library system that may provide multiple services to multiple students such as an article database, a book finder and an order placer.