ERIC WEBB

*Nova Southeastern University*

*CISC 680 - Software Engineering - CRN – 21741*

Due Oct 13 by 11:59pm

Assignment 1 (Question set 1)

1. How does software differ from the artifacts produced by other engineering disciplines? Give examples.

The artifact produced in software engineering that is different from other engineering principals is the most important product of our time, information. Typically, other engineering disciplines are thought to be more physical in nature, where software engineering provides a product that is more digital in nature. Software offers a dual role of being a product and the vehicle for delivering a product. As a product, it provides computing potential and is an information transformer. Transforming data that could be binary or multimedia in nature. As a vehicle it acts as a control, creation, and communication of a computer.

1. Explain what is wrong with the notion that computer software does not need to evolve over time?

In software engineering change is natural so we should not try to fight it. Not only should software evolve to meet the needs of new computing environments, but it should also be enhanced to implement new business requirements. It is said that the goal of modern software engineering is to devise methodologies that are founded on the notion of evolution. Since technology is always growing and changing, software must stay up with needs of these new demands. Without the ability to evolve software becomes stagnant and stale.

1. Why has the Personal Software Process not been widely adopted by industry?

The personal software process has not been widely adopted by the industry because the process for one project might be significantly different than a process adopted for another project. This could be such topics as the overall flow of activities, actions, and tasks. Such as how project tracking and control activities are applied. As well as the degree in which quality assurance activities are applied. Since every project does not have a one size fits all business process the personal software process has not been widely adopted.

1. Why are evolutionary models considered by many to be the best approach to software development in a modern context?

Evolutionary models provide high-quality software in an iterative and incremental manner. This provides a balance between the actual critical project, the product parameters, and customer satisfaction. One concept of this is prototyping, this allows for a basic working product to be developed from and built upon effectively evolving over time. Another evolutionary model would be the Spiral model which effectively designs, builds, deploys, communicates, and repeats for constant communication, feedback, and improvements sent from stakeholders and end users. Another evolutionary model is concurrent modeling. This allows for the modeling to be in a number different of states such as under development or awaiting changes. Allowing different process to evolve into different states. Ultimately evolutionary models are beneficial because they allow for software to improve over time.

1. What are the benefits of using analysis patterns during the analysis modeling process?

One benefit of analysis patterns is that they provide reusable analysis models that capture the main requirements of concrete problems. This reusability speeds up the development of abstract analysis models. Another example is how analysis patterns facilitate transformations of the analysis models into design models by suggesting design patterns and reliable solutions for common problems. Analysis patterns are stored in analysis designs pattern repositories, so requirement engineers can search them to be reused.

1. Describe the contents of the WebApp content, functional, interaction, and configuration models?

The WebApp requirements models vary depending on the nature of the application. The Content model identifies all the content to be provided by the application such as media, text, and audio. The Interaction model describes how the user will interact with the application and in what manner. The Functional model is what defines the operations that will be used to describe processing functions and manipulate content. It also provides the functions that are necessary for the end user. Lastly, the configuration model will describe the infrastructure and environment where the application resides. Each model contains a common schema for a multitude of topics and should be dealt with accordingly.

7. Explain how a process specification (PSPEC) differs from a control specification

(CSPEC)?

A control specification (CSPEC) is used to indicate how software behaves when a specific event occurs. This invokes the process trigged from this event. When a control signal is sent a specific functional process will be initiated. This is different from a process specification (PSPEC) because A PSPEC will describe the process and inner workings of the input. So instead of just getting a straight control you are setting parameters of what can potentially be the process for the input. Essentially the PSPEC describes the input algorithm, restrictions, limitations, and performance characteristics of a process.

8. How does the object-oriented view of component-level design differ from the traditional view?

With object-oriented view of a component-level design, details of a component can be modeled at different levels of abstraction. A UML activity diagram can be represented using pseudocode or a diagrammatic form like a flow chart. This is in contrast the traditional view because the traditional view will fully describe each internal detail of each software component. To do this each component defines its own data structures for its local data objects. So, in contrast traditional views require objects to be defined locally where with object oriented objects can be abstracted at any level.

9. List four interface design issues present in the development of most user interfaces, explain how and when they are used?

The first design issue in user interfaces is response time, if system response time is to long then user frustration occurs. So, taking into account length of response time is a factor. Variability is another factor and designed to keep the user off kilter. For example, if a response is different but always between slots of times then the user will think something new is being done behind scenes. Another issue is help facilities, users should be able to get adequate help without having to leave the application interface. The next issue would be error handling, errors should describe what the issue is, any constructive advice on how to resolve it, and what can happen if it is not resolved. You should not place the blame on the user, but errors should have some type of visual or audio cue to draw attention. Lastly, listing command and menus properly should be accounted for. Making sure the user knows what little icons mean in the GUI vs what commands can be issued in the CLI should be clear and concise and be able to be learnt without leaving the application interface. All of these are issues that should be accounted for and dealt with in user interface design.

10. Describe practices that enable designers to think about using patterns? (355)

Practices that enable designers to think about using patters are understanding the bigger picture. Once example of this is reviewing the requirements model. When examining the bigger picture, you should extract the patterns that are present at that level of abstraction. You should start your design with big picture patterns to establish a skeleton for future work. From the context derived you should work inward from the context to look for patterns at lower levels of abstraction. Once that is done it is recommended you repeat the steps above until a complete design is flushed out.

11. What are three dimensions of software quality, how and when are they used?

David Garvin defines eight quality dimensions, although they do not specifically relate to software they can be used as considerations for software development. We will discuss three of these eight. To begin, Reliability is one quality dimension. This ensures that software is available when it is needed. This is used when code needs to be error free. Another software quality is durability. This allows for software to be changed without errors. Durability is used when software needs to be maintained over time. Lastly for our third software quality dimension we will discuss performance quality. This ensures that all content and features are useful to the user. This is used when something of value needs to be modeled to the user.

12. Why is regression testing an important part of any integration testing procedure?

Regression testing is important because when a new change is made you want to make sure that any previous functionality is not effected and still works accordingly. After a change is made you run regression tests to make sure your previous functionality is still working, if it passes then you move on to integration testing to see if the new change works as expected. You should always apply regression testing in your procedures, so you can have predefined test sets built to see if any changes to future code does not affect previous functions in a negative manner. Having these tests built save time and effort when it comes to testing.

13. Describe object-oriented unit testing, why and when is it used and give an example?

The idea of unit testing is to write tests that can be re-used for other tests. This becomes more difficult in object-oriented scenarios due to the nature of encapsulation hiding the ability to test and abstract data. Basically, if there is a Parent class let’s say “animal” and you have two sub-classes “dog” and “snake”. In your unit testing you would like to test the parent class “animal” method called “eats” functionality. You cannot test the “eats” method until it is instantiated in either the “dog” or “snake” class. Even then you have to test the methods differently for how a “snake” eats compared to how a “dog” eats. Both eats methods are inherited from animal but are executed differently and thus need a different test for each. So with object-oriented unit testing it is important to keep note that you need to write more specific tests for test case scenarios compared to a test written for global exposure.

14. List four types of systems tests, the purpose of each and how they can be applied to real world software development process?

Besides running tests on software, it is also important to run tests on the actual physical systems as well that this software is deployed on. We will now discuss four types of system tests. First is the concept of Recovery testing. In this testing, software is forced to fail in a variety of fashions to test and gauge recovery time. The purpose is to recover from faults and resume processing with little or no downtime. Next, we have security testing, this tests that security measures built into the system do indeed work. Engineers will constantly test security measures to verify things are secure. The goal is to be protected from both internal and external attacks. Another testing concept is stress testing. Stress testing is designed to confront programs with abnormal situations. This is done by running the system in a manner that demands resources in an abnormal quality, frequency, and size. The purpose is to find a threshold of at which point the system fails. Lastly, we have deployment testing. Deployment testing executes software on a number of different types of environments while documenting the installation process. This is used to better suited the end user and get the product to market more effectively.

15. What are the key differences between validation testing goals and acceptance testing goals, give an example of when one would be used over the other?

Validation testing refers to a set of tasks that ensure software is built to customer requirements. It verifies the output is even important and meaningful to the end user. Any functional requirements that are not satisfied will be added to deficiency list to be brought up for review. Acceptance testing verifies that the software are works correctly for the user in their normal environment. This will test typical user work conditions and inputs. Validation is more about whether the software is correct for the customer where acceptance is more if the software is working correctly.