A Homework

Of

Software Engineering

Course no: CSE-3203

Course title: Software Engineering and Information System Design

Submitted To:

Sk. Alamgir Hossain

Assistant Professor

CSE Discipline

Khulna University, Khulna.

Submitted By:

Md. Mostafizur Rahman

Id no: 120234

Third Year Second Term

CSE Discipline

Khulna University, Khulna.

Date of Submission: 06.08.2015

Question 3.1. Provide three examples of software projects that would be amenable to the incremental model. Be specific.

Answer:

Incremental model: The whole requirement of incremental model is divided into various builds. Multiple development cycles can take place there, making the life cycle. Cycles are divided up into smaller, vary easily managed modules of this model. Each module of this model passes through the requirements, design, implementation and testing phases. A working version of a software is produced during the first module, so you have been working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved.

There are several possible three examples of software projects that would be amenable to the incremental model.

* An Operating System: The various parts of the Operating System could be developed as the customer wants them. For example, the customer might want to specify the GUI first and try it out before providing further specifications for the remaining parts of the Operating System. The GUI could then be developed; once a user approved, some of the most important functions of the Hardware Abstraction Layer portion could be included. The process could be continued until the entire system is being completed, with all customers are getting continuing updates to test and approve.
* An Internet Browser Application: The internet base application could be developed and distributed, followed by any of a number of plug-ins to increase functionality. The plug-ins might be included JavaScript interpretation, XML parsing, and so on. Many browsers available have which could follow this model.
* Satellite Telemetry and Control software: There have a number of parts to a system of this nature, all of which can be developed independently, and then later integrated. The customer can take delivery of parts of the system, such as the telemetry denomination command encoder, and telemetry history gainer.

3.2. Provide three examples of software projects that would be amenable to the prototyping model. Be specific.

Answer:

Prototyping Model: This model is good to use when the customer has legitimate needs, but is not able to articulate the details at the start of the project. A small mock-up of a working system is developed and presented to the customer. Sometimes this first system is discarded and sometimes it is extended based on the customer's feedback.

Provide three example of software projects that would be amenable to the incremental model. Be specific. Software applications that are relatively easy to prototype almost always involve human-machine interaction and/or heavy computer graphics. Other applications that are sometimes amenable to prototyping are certain classes of mathematical algorithms, subset of command driven systems and other applications where results can be easily examined without real-time interaction. Applications that are more difficult to prototype include control and process control functions, many classes of Real-time applications and embedded software. Software applications that are relatively easy to prototype almost always involve Human-machine interaction and/or heavy computer graphics. Other applications that are sometimes amenable to prototyping are certain classes of mathematical algorithms, subset of command driven systems and other applications where results can be easily examined without real-time interaction. Applications that are more difficult to prototype include control and process control functions, many classes of real-time applications and embedded software

Question 3.3. What process adaptations are required if the prototype will evolve into a deliverables system or product?

Answer:

There are three process adaptations that are required if the prototype will evolve into a deliverable system or product. The processes are more rigorous design rules and SQA procedures must be applied from the beginning, the prototype must be designed with extensibility in mind, and then it becomes the framework for extensions that will cause it to evolve into a production system.

Question 3.4. To achieve rapid development, the RAD model assumes the existence of one thing. What is it, and why is the assumption not always true?

Answer:

RAD Model: The rapid application deployment model is a high-speed adaptation of the linear sequential model. Project requirements must be well understood and the project scope tightly constrained. Developers are often able to use component-based construction techniques to build a fully functional system in a short time period.

Rapid Application Development (RAD) is an incremental software process model that emphasizes a short development cycle.

RAD is a high-speed adaptation of the waterfall model, in which rapid development is achieved by using a component based construction approach.

If requirements are well understood and project scope is constrained, the RAD process enables a development team to create a fully functional system within a short period of time.

The drawbacks of the RAD model are following here:

* For large, but scalable projects, RAD requires sufficient human resources to create the right number of RAD teams.
* If developers and customers are not committed to the rapid-fire activities necessary to complete the system in a much abbreviated time frame, RAD project will fail.
* If a system cannot properly be modularized, building the components necessary for RAD will be problematic.

Question 3.5. Provide three examples of software projects that would be amenable to the waterfall model. Be specific.

Answer:

The waterfall model is appropriate for projects with the following characteristics:

* The problem is well understood(requirements are well- defined)
* The delivery data is realistic
* It’s unlikely that major change in requirements will be requested as the project.

Three examples of software projects that would be amenable to the waterfall model might be:

* A well under stood modification to an existing program;
* A straight forward implementation of a numerical calculation or business rule, even if it’s complex;
* A constrained enhancement to an existing program.

Question 3.7. Is it possible to combine process models? If so, provide an example.

Answer:

Yes, it is possible to combine a process models easily. It is the spiral process model. This model of development application combines the some features the waterfall model and of the prototyping model.

Let us consider the case study of the baggage-handling system of this process model at the Hazrat Shah Jalal International Airport again. Initially, this airport had been intended that each individual airline would be responsible for building its own baggage-handling system of the process system. However, as the construction of the airport have progressed, a larger vision have emerged for the inclusion of an airport-wide integrated baggage-handling system that could be provided a main improvement in the efficiency of luggage delivery and nobody have given any thought to risk assessment. The essential concept of the Spiral Model is to minimize risks by the repeated can be used of prototypes and other means. Unlike other models, at every stage risk analysis is being performed. The Spiral Model works by building progressively the most complete versions of the application by starting at the center of the spiral and working outwards.

Question 3.10. What are the advantages and disadvantages of developing software in which quality is “good enough”? That is, what happens when we emphasize development speed over product quality?

Answer:

The advantages and disadvantages of developing software are following here:

* Advantages:
  + Develop management and leadership skills by increased self-development.
  + Opens a new atmosphere conscious of quality.
  + Function as a nucleus for companywide quality control at the workshop level.
  + Increases employee morale and sense of common goal.
  + Employees need to be well educated and have a good understanding of the organization beyond own work area.
  + Quality circles cost effective.
  + Frees management -shop floor workers best located to identify problems.
* Disadvantages:
* Intensity of work increases -as more problems are solved more is expected of workers.
* Can be introduced for incorrect reasons i.e. Attitude change.
* Management needs to be fully committed to quality systems -if solutions not implemented can be frustrating for participants.
* Can have a negative effect on industrial relations.
* Can focus on mundane problems.

When one emphasizes development over product quality, it is very likely that product quality will suffer greatly. There would be increased output but product quality could be poor.

Question 3.12. Provide three examples of software projects that would be amenable to the component-based model. Be specific.

Answer:

Object-based technologies provide the technical framework for component-based software engineering. The component-based development (CBD) model incorporates many of the iterative characteristics of the spiral model. The main difference is that in CBD the emphasis is on composing solutions from prepackaged software components or classes. This CBD emphasizes software reusability. The unified software development process is an example of CBD that has been proposed for industrial use. The unified modeling language (UML) is used to define the components and interfaces used in the unified software development process

There are several possible examples that might fit well in the component-based model.

A wireless PDA application in Java. This might incorporate any number of possible data applications such as address book, task list, meeting calendar, phone list, and many more. Components are available to perform these functions, so that the application would be developed using them. The development team could focus on all parts independently, checking each COTS part out individually before integrating it with the whole system. Parts could be swapped in and out as needed, based on the results of the integration testing process.

A distributed computer controller for industrial control in a factory. All of the machines could have their own real-time computer-based systems, which are networked together. The network could allow control by passing a "control token", much as in a "token ring" network. Each machine's individual controller firmware would be a component in the system.

An object-oriented event handling system as in Java. Components would include the requisite objects which have behaviors to handle the different types of events that the virtual machine can produce.

A real-time language translator. The translator could have components for several different languages, and could even include a thesaurus and/or a phrase dictionary for any or all of the language components.

Question 3.14. Are the Unified Process and UML the same thing? Explain your answer.

Answer:

The Unified Process and Unified modeling language (UML) are not the same thing. There have a various difference between them and using each one of them is the same.

Unified modeling language (UML) is a modelling language which is a set of rules and standards for drawing diagrams. Unified Process is a software development methodology or process which tells you step by step what you should do to develop software. Some of those steps may be required drawing UML diagrams.

* The difference between the UP and UML:
* Unified Process (UP) = a process framework in which UML may be applied as part of Software Engineering activities
* Unified modeling language (UML) = Unified modeling language is a modeling notation and language

Question 3.15. As you move outward along the spiral process flow, what can you say about the software that is being developed or maintained?

Answer:

When software engineering team moves around the spiral, the first circuit around the spiral results in development of product specification. The subsequent passes around the might be used to develop prototype in more subsequent manner. In each pass, through planning region, some adjustment to project plan are made. Cost and schedule adjustments can also be made according to customer feedback.

* What happens to the software as you move outwards along the spiral process model flow:
  + The product moves towards The most complete state
  + The level of abstraction at which work is being performed which is reduced

i.e. implementation specific work accelerates as we can move further from the origin