Advanced Systems Analysis and Development

Gevorderde Stelselontleding en Ontwikkeling

INF305F / SWE401I / SED401I

Tutorial Letter 202

Studiebrief 202

School of Computing

Skool vir Rekenaarkunde
1 Study material received

You should have received the following Tutorial Letters:

<table>
<thead>
<tr>
<th>Tutorial letter 101</th>
<th>First Tutorial Letter containing important administrative information and Assignments for 2010.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tutorial letter 102</td>
<td>Errata: Tutorial Letter 101; How to study for INF305F; Assignment 03: Self-evaluation; Solutions for Assignment 03</td>
</tr>
<tr>
<td>Tutorial 201</td>
<td>Solutions Assignment 1</td>
</tr>
<tr>
<td>Tutorial 202</td>
<td>Solutions Assignment 2: This tutorial letter</td>
</tr>
</tbody>
</table>

If you are missing any study material, please contact the UNISA Contact Centre at (+27 861 670 411 for RSA calls only or +27 11 670 9000 for international calls) or download it from the myUnisa website at http://www.unisa.ac.za. Do not contact us for missing or lost study material since we are unable to assist you with any missing or lost study material.

2 Lecturers responsible for INF305F/SED401I/SWE401I

The lecturers are Mrs M Loock and Mr Baldreck Chipangura. Please phone us on +27 12 429 6376 or send e-mails to INF305@osprey.unisa.ac.za
3 Solution Assignment 2

Chapter 2

Question 1

The concurrent process model defines a set of ‘states’. Describe in your own words what these states represent, and then indicate how they come into play within the concurrent process model. [5]

Stated simply, the concurrent process model assumes that different parts of a project will be different stages of completeness, and therefore, different software engineering activities are all being performed concurrently. The challenge is to manage the concurrency and be able to assess the status of the project. The stages include inactive, underdevelopment, awaiting changes, under review, under revision, baselined, done. These states represent the different stages of completion of software development project, and are performed concurrently. It provides an accurate situation of the present state of the software development project.

Question 2

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? [7]

The advantages of developing software in which quality is “good enough” is that the product or software will meet the deadline, it may however lead to the delivery of software that is low in quality and requires time to improve the quality. When speed is emphasized over the product quality it may lead to many flaws, the software may require more testing, design and implementation work then done. Requirements may be poorly defined and may need to continuously change. Half hearted and speed may cause the risk management to fail to detect major project risks. Too little quality may result in quality problems and later rework.

Chapter 3

Question 3

Describe the XP concepts of refactoring and pair programming in your words. [5]

XP encourages refactoring—a construction technique that is also a design technique. Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves the internal structure. A central notion in XP is that design occurs both before and after coding commences. Refactoring means that design occurs continuously as the system is constructed a key concept during the coding activity is pair programming. XP recommends that two people work together at one computer workstation to create code for a story. This provides a mechanism for real-time problem solving (two heads are often better than one) and real-time quality assurance.

Question 4

Why is Crystal called a family of agile methods? [5]

Like other agile methods, the Crystal approach emphasizes collaboration and communication among people who have varying interest in the software project. The method is also tolerant of varying team cultures and can accommodate both informal and relatively formal software engineering approaches hence it is called the
family of agile methods”. Crystal family is actually a set of example agile processes that have been proven effective for different types of projects. The intent is to allow agile teams to select the member of the crystal family that is most appropriate for their project and environment.

Chapter 4

Question 5

Describe separation of concerns in your own words. [5]

A large problem is easier to solve if it is subdivided into a collection of elements each of which delivers a distinct functionality that can be developed, and in some cases validated, independently of other concerns. (Divide and conquer)

Question 6

How does agile communication differ from traditional software engineering communication? How is it similar? [6]

The agile view of customer communication and collaboration is applicable to all software engineering practice and like all software engineering practices, communication and collaboration are iterative. Each iteration divulges important new information and enhances the overall understanding of the software to be built.

Agile communication is quicker than traditional. Traditional projects design and code up until the delivery only to discover problems at a time when it is very expensive to correct or rectify.

Chapter 5

Question 7

Why is it that many software developers don’t pay enough attention to requirements engineering? Are there ever circumstances where you can skip it? [6]

Understanding the requirements of a problem is among the most difficult tasks that a software engineer face since requirements change continuously, hence they tend to pay little attention to it. In some cases, an abbreviated approach may be chosen. In others, every task defined for comprehensive requirements engineering must be performed rigorously. Requirements engineering builds a bridge to design and construction and cannot be skipped.

Question 8

Why do we say that the requirements model represents a snapshot of a system in time? [5]

The intent of the requirements model is to provide a description of the required information, functional, and behavioral domains for a computer-based system. The model changes dynamically as software engineers learn more about the system to be built, and stakeholders understand more about what they really require. For that reason, the analysis model is a snapshot of requirements at any given time.
Chapter 8

Question 9

If a software design is not a program (and it isn’t), then what is it? [5]

The intent of software design is to apply a set of principles, concepts, and practices that lead to the development of a high quality system or product. The goal of design is to create a model of software that will implement all customer requirements correctly and bring delight to those who use it.

Question 10

Describe software architecture in your own words. [5]

Software Architecture is the structure or organization of program components (modules), the manner in which these components interact, and the structure of data that are used by the components. In a broader sense, however, components can be generalized to represent major system elements and their interactions.

Chapter 10

Question 11

Why are control components necessary in traditional software and generally not required in object-oriented software? [5]

Like object-oriented components, traditional software components are derived from the analysis model. In this case, however, the data flow-oriented element of the analysis model serves as the basis for the derivation. Each transform (bubble) represented at the lowest levels of the data flow diagram is mapped (Section 10.6) into a module hierarchy. Control components (modules) reside near the top of the hierarchy (architecture) and problem domain components tend to reside toward the bottom of the hierarchy. To achieve effective modularity, design concepts like functional independence are applied as components are elaborated.

Question 12

Why is chunking important during the component-level design review process? [5]

People can only keep track of a small number (5 to 9) of things at a time in short term memory. Chunking allows reviewers to combine related concepts into larger pieces or bigger chunks. The components can serve as chunks (if the components are highly cohesive and loosely coupled) making it easier for reviewers to keep track of the interactions of several components during a design review rather than a large number of individual classes and their methods.

Chapter 17

Question 13

Is it always possible to develop a strategy for testing software that uses the sequence of testing steps described in Section 17.1.3? What possible complications might arise for embedded systems? [6]

It is not always possible to conduct thorough unit testing in that the complexity of a test environment to accomplish unit testing (i.e., complicated drivers and stubs) may not justify the benefit. Integration testing is
complicated by the scheduled availability of unit-tested modules (especially when such modules fall behind schedule). In many cases (especially for embedded systems) validation testing for software cannot be adequately conducted outside of the target hardware configuration. Therefore, validation and system testing are combined.

**Question 14**

**Why is a highly coupled module difficult to unit-test?**

A highly coupled module interacts with other modules, data and other system elements. Therefore its function is often dependent of the operation of those coupled elements. In order to thoroughly unit test such a module, the function of the coupled elements must be simulated in some manner. This can be difficult and time consuming.

**Chapter 18**

**Question 15**

**Can you think of any additional testing objectives that are not discussed in Section 14.1.1?**

In addition to those objectives:

a) A successful test demonstrates compliance with function and performance;

b) A successful test uncovers documentation errors;

c) A successful test uncovers interfacing problems;

d) A successful test demonstrates an understanding of program architecture, data structure, interface design and procedural design;

e) A successful test establishes an entry into a test case database that may later be used for regression testing.

**Question 16**

**Give at least three examples in which black-box testing might give the impression that ‘everything is OK’, while white-box tests might uncover an error.**

Give at least three examples in which white-box testing might give the impression that “everything is OK, while black-box tests might uncover an error.

For specific input, an error occurs internally resulting in:

1) Improper data placed in a global data area;

2) Improper flags that will be tested in a subsequent series of tests;

3) Improper hardware control that can only be uncovered during system test; yet "correct" output is produced.

Error detection by white-box testing contradicting black-box testing:
• Testing to ensure that all independent paths within a module will be executed at least once.
• Testing to exercise all logical decisions on their true and false branches.
• Testing to ensure that all loops execute at their boundaries and within their operational bounds.

Error detection by black-box testing contradicting white-box testing:

• Testing for interface functionality.
• Testing system behavior and performance.
• Test for classes of input.

Chapter 19

Question 17

Which is more valuable to object-oriented testing, white-box or black-box testing? Why? [8]

During latter stages of their development, OO analysis and design models provide substantial information about the structure and behavior of the system. For this reason, these models should be subjected to rigorous review prior to the generation of code. All object-oriented models should be tested (in this context, the term testing incorporates technical reviews) for correctness, completeness, and consistency within the context of the model’s syntax, semantics and pragmatics. These reviews have the potential to eliminate much unnecessary work and rework (again the earlier errors are discovered the cheaper they are to fix).

OO System validation testing is black-box oriented and can be accomplished by applying the same black-box methods discussed for conventional software.

Question 18

What is the difference between thread-based and use-based strategies for integration testing? How does cluster testing fit in? [5]

Thread-based testing is used to integrate a set of classes that are required to respond to a single program input or a single event. Use based testing is an integration testing strategy that begins by integrating the classes that do not collaborate with “server” classes (these are called independent classes). Then, classes that collaborate with the independent classes are tested in a layered fashion.

Cluster testing is one step in the integration testing of OO software. Here a cluster of collaborating classes (determined by examining the CRC and object-relationship model) is exercised by designing test cases that attempt to uncover errors in the collaborations.

©

Unisa 2010