Course: CSC 316 – Software Engineering I (2 Credits – Compulsory)

Course Duration: Two hours per week for 15 weeks (30 hours), as taught in 2011/2012 session

Lecturer: Aremu, D. R.
Department of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Ilorin, Nigeria.
E-mail: aremu.dr@unilorin.edu.ng, draremu2006@gmail.com
Office Location: Block 10, Room G21
Faculty of Communication and Information Sciences, University of Ilorin, Ilorin, Nigeria.
Consultation Hours: 12 noon – 2.00pm (Tuesdays and Thursdays).

Course Content
Software and Software Engineering, Software life cycle, process models, project planning, project scheduling and tracking, Requirement analysis, software design principles, implementation, integration, testing, maintenance, quality assurance and software metric.

Course Description
This course covers the process of analysis, design, implementation, and testing of a software, using the conventional method for software engineering approach.

Course Justification
Computer software has become a driving force. It is the engine that drives business decision making. It serves as the basis for modern scientific investigation and engineering problem solving. It is a key factor that differentiates modern products and services. It is embedded in systems of all kinds: transportation, medical, telecommunications, military, industrial processes, entertainment, office products, the list is almost endless. Software is virtually inescapable in a modern world. And as we move into the twenty-first century, it will become the driver for new advances in everything from elementary education to genetic engineering. Today, software engineering is recognized as a subject worthy of serious research, conscientious study, and tumultuous debate. Throughout the industry, software engineer has replaced programmer as the job title of preference. Software process models, software engineering methods, and software tools have been adopted successfully across a broad spectrum of industry applications. Although managers and practitioners alike recognize the need for a more disciplined approach to software, they continue to debate the manner in which discipline is to be applied. Many individuals and companies still develop software haphazardly, even as they build systems to service the most advanced technologies of the day.
Many professionals and students are unaware of modern methods. And as a result, the quality of the software that we produce suffers and bad things happen. In addition, debate and controversy about the true nature of the software engineering approach continue. The status of software engineering is a study in contrasts. Attitudes have changed, progress has been made, but much remains to be done before the discipline reaches full maturity.

**Course Objectives**

The main objective of this course is to improve the skill of the students in software engineering. By the end of the course, students should be able to:

(i) explain the Software Engineering life cycle;

(ii) list the requirements engineering process steps;

(iii) explain the technical metrics available for assessing the quality of analysis and design models, source code, and test cases;

(iv) describe the basic concepts, principles, and strategies applicable to software testing; and

(v) demonstrate an understanding of the Software analysis, design, and testing approaches.

**Course Requirements**

The course requires that students attend 75% of the classes before they can sit for the examination and participate very actively in the class. Students are also required to read widely on the subject matter and be able to apply the knowledge gained to solve real life problems.

**Method of Grading**

<table>
<thead>
<tr>
<th>S/No.</th>
<th>Type of Grading</th>
<th>Score (10%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Attendance</td>
<td>05</td>
</tr>
<tr>
<td>2</td>
<td>Test</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>Assignment</td>
<td>05</td>
</tr>
<tr>
<td>4</td>
<td>Online discussion</td>
<td>05</td>
</tr>
<tr>
<td>5</td>
<td>Final Examination</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td><strong>TOTAL</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>

**Course Delivery Strategies**

The course delivery strategies will be by face-to-face lectures, and by assigning students into groups, and each group will be required to present group work in the class. Online discussion will also be available to all students of the course to exchange ideas.
Week 1: Introduction to Software and Software Engineering

Objectives

At the end of this week, students should be able to:
(i) describe the basic concepts of software engineering;
(ii) list the generic view of software engineering;
(iii) explain the software life cycle; and
(iv) list the importance of software engineering.

Description

First Hour
An overview of the course

Second Hour
Software life cycle.

Study Questions

1. How is software defined within the context of a larger system and how does system engineering play a role?
2. What basic concepts and principles are applicable to the analysis of software requirements?
3. What is structured analysis and how do its various models enable you to understand data, function, and behavior?
4. What basic concepts and principles are applied to the software design activity?
5. How are design models for data, architecture, interfaces, and components created?
6. What basic concepts, principles, and strategies are applicable to software testing?
7. How are black-box and white-box testing methods used to design effective test cases?
8. What technical metrics are available for assessing the quality of analysis and design models, source code, and test cases?

Reading List

Week 2: Introduction to Software product and process

Objectives
The objectives of this week are to enable the students to be able to:
(i) Describe the software process;
(ii) List some software engineering tools; and
(iii) Explain the software process models.

Description
First Hour
The Software Process
Second Hour
Software Process Models.

Study Questions
1. What is a “software process”?  
2. Is there a generic way to assess the quality of a process?  
3. What process models can be applied to software development?  
4. How do linear and iterative process models differ?  
5. What are their strengths and weaknesses?  
6. What advanced process models have been proposed for software engineering work?

Reading List
Week 3: Managing Software Projects

Objectives
At the end of this week, students should be able to:

(i) Explain the software management concepts;
(ii) List the software management spectrum; and
(iii) Describe each of the software management spectrum.

Description
First Hour
Project Management concept
Second Hour
The Software Management Spectrum.

Study Questions
1. Describe three real-life situations in which the customer and the end-user are the same. Describe three situations in which they are different.
2. The decisions made by senior management can have a significant impact on the effectiveness of a software engineering team. Provide five examples to illustrate that this is true.
3. You have been appointed a project manager within an information systems organization. Your job is to build an application that is quite similar to others your team has built, although this one is larger and more complex. Requirements have been thoroughly documented by the customer. What team structure would you choose and why? What software process model(s) would you choose and why?
4. You have been appointed a project manager for a small software products company. Your job is to build a breakthrough product that combines virtual reality hardware with state-of-the-art software. Because competition for the home entertainment market is intense, there is significant pressure to get the job done. What team structure would you choose and why? What software process model(s) would you choose and why?
5. You have been appointed a project manager for a major software products company. Your job is to manage the development of the next generation version of its widely used word-processing software. Because competition is intense, tight deadlines have been established and announced. What team structure would you choose and why? What software process model(s) would you choose and why?
6. You have been appointed a software project manager for a company that services the genetic engineering world. Your job is to manage the development of a new software product that will accelerate the pace of gene typing. The work is R&D oriented, but the goal to to produce a product within the next year. What team structure would you choose and why? What software process model(s) would you choose and why?
7. You have been asked to develop a small application that analyzes each course offered by a university and reports the average grade obtained in the course (for a given term). Write a statement of scope that bounds this problem.

Reading List
1. Airlie Council (March 8, 1999). Performance Based Management: The
Week 4: Project Scheduling and Tracking

Objectives
The objectives of this week are to:
(i) Explain the basic concepts of project scheduling and tracking;
(ii) List software engineering tasks; and
(iii) Describe a task set for the software project.

Description
First Hour
Basic Concepts

Second Hour
Task set for software engineering project.

Study Questions
1. “Unreasonable” deadlines are a fact of life in the software business. How should you proceed if you’re faced with one?
2. What is the difference between a macroscopic schedule and a detailed schedule? Is it possible to manage a project if only a macroscopic schedule is developed? Why?
3. Is there ever a case where a software project milestone is not tied to a review? If so, provide one or more examples.
4. Although adding people to a late software project can make it later, there are circumstances in which this is not true. Describe them.

Reading List

**Week 5: Requirement analysis**

**Objectives**
The objectives of this week are to enable the students to be able to:

(i) list the requirements engineering process;
(ii) describe requirements elicitation, analysis, specification, validation, and management; and
(iii) explain system modelling.

**Description**

*First Hour*
Analysis Concepts and Principles

*Second Hour*
Analysis modeling

**Study Questions**

1. Build a hierarchical "system of systems" for a system, product, or service with which you are familiar. Your hierarchy should extend down to simple system elements (hardware, software, etc.) along at least one branch of the "tree."

2. Select any large system or product with which you are familiar. Define the set of domains that describe the world view of the system or product. Describe the set of elements that make up one or two domains. For one element, identify the technical components that must be engineered.

3. Select any large system or product with which you are familiar. State the assumptions, simplifications, limitations, constraints, and preferences that would have to be made to build an effective (and realizable) system model.

4. Business process engineering strives to define data and application architecture as well as technology infrastructure. Describe what each of these terms means and provide an example.

5. Information strategy planning begins with the definitions of objectives and goals. Provide examples of each from the business domain.
Reading List

Study Questions
1. You have been asked to build one of the following systems:
   a. A network-based course registration system for your university.
   b. A Web-based order-processing system for a computer store.
   c. A simple invoicing system for a small business.
   d. Software that replaces a Rolodex and is built into a wireless phone.
e. An automated cookbook that is built into an electric range or microwave.

Select the system that is of interest to you and develop an entity/relationship diagram that describes data objects, relationships, and attributes.

2. What is the difference between cardinality and modality?

3. Draw a context-level model (level 0 DFD) for one of the five systems that are listed in Problem 2. Write a context-level processing narrative for the system.

4. Using the context-level DFD developed in Problem 3, develop level 1 and level 2 data flow diagrams. Use a "grammatical parse" on the context-level processing narrative to get yourself started. Remember to specify all information flow by labeling all arrows between bubbles. Use meaningful names for each transform.

5. Develop a CFDs, CSPECs, PSPECs, and a data dictionary for the system you selected in Problem 2. Try to make your model as complete as possible.

6. Does the information flow continuity concept mean that, if one flow arrow appears as input at level 0, then one flow arrow must appear as input at subsequent levels? Discuss your answer.

**Week 6: Design Concepts and Principles**

**Objectives**
At the end of this topic, students should be able to:

(i) list some basic design concepts and principles;
(ii) explain each of the design concepts; and
(iii) describe different types of software engineering design models.

**Description**
First Hour
The design process.
Second Hour
Design principles, concepts, heuristics, model, and documentation.

**Study Questions**
1. Do you design software when you "write" a program? What makes software design different from coding?

2. Apply a "stepwise refinement approach" to develop three different levels of procedural abstraction for one or more of the following programs:
   a. Develop a check writer that, given a numeric dollar amount, will print the amount in words normally required on a check.
   b. Iteratively solve for the roots of a transcendental equation.
   c. Develop a simple round-robin scheduling algorithm for an operating system.

3. Describe the concept of information hiding in your own words.

4. Discuss how structural partitioning can help to make software more maintainable.

5. Discuss the relationship between the concept of information hiding as an attribute of effective modularity and the concept of module independence.
**Week 7: Architectural Design**

**Objectives**
At the end of this topic, students should be able to:

(i) list some architectural styles;
(ii) explain the importance of software architecture; and
(iii) describe data modelling, data structures, databases and data warehouse.

**Description**

First Hour
Software architecture Design

Second Hour
Quantitative guidance for architectural design.

**Study Questions**

1. Using the architecture of a house or building as a metaphor, draw comparisons with software architecture. How are the disciplines of classical architecture and the software architecture similar? How do they differ?

2. Explain the difference between a database that services one or more conventional business applications and a data warehouse.

3. Write a three- to five-page paper that describes how data mining techniques are used in a business context and the current state of KDD techniques.

4. Present two or three examples of applications for each of the architectural styles discussed in this lesion.

5. Some of the architectural styles noted in this lesion are hierarchical in nature and others are not. Make a list of each type. How would the architectural styles that are not hierarchical be implemented?

**Reading List**

Week 8: User Interface design

Objectives

First Hour
Interface design models

Second Hour
Interface design process

Study Questions

1. Describe the worst interface that you have ever worked with and critique it relative to the concepts introduced in this lecture. Describe the best interface that you have ever worked with and critique it relative to the concepts introduced in this lecture.

2. Develop two additional design principles that “place the user in control.”

3. Develop two additional design principles that “reduce the user’s memory load.”

4. Consider one of the following interactive applications:
   a. A desktop publishing system.
   b. A computer-aided design system.
   c. An automated course registration system for a university.
   d. A library management system.
   e. An Internet-based polling booth for public elections.
   f. A home banking system.

   Develop a design model, a user model, a system image, and a system perception for any one of these systems.

5. Develop an interface evaluation questionnaire that contains 20 generic questions that would apply to most interfaces. Have ten classmates complete the questionnaire for an interactive system that you all use. Summarize the results and report them to your class.
Reading List

Week 9: Component –Level Design

Objectives
First Hour
Structured Programming
Second Hour
Comparison of Design Notation.

Study Questions
1. Select a small portion of an existing program (approximately 50–75 source lines). Isolate the structured programming constructs by drawing boxes around them in the source code. Does the program excerpt have constructs that violate the structured programming philosophy? If so, redesign the code to make it conform to structured programming constructs. If not, what do you notice about the boxes that you’ve drawn?
2. All modern programming languages implement the structured programming constructs. Provide examples from three programming languages.
3. Why is “chunking” important during the component-level design review process?
4. Develop a procedural design for components that implement the following sorts: Shell-Metzner sort; heapsort; BSST (tree) sort. Refer to a book on data structures if you are unfamiliar with these sorts.
5. Develop a procedural design for an interactive user interface that queries for basic income tax information. Derive your own requirements and assume that all tax computations are performed by other modules.
6. Develop a procedural design for a program that accepts an arbitrarily long text as input and produces a list of words and their frequency of occurrence as output.
7. Develop a procedural design of a program that will numerically integrate a function \( f \) in the bounds \( a \) to \( b \).
8. Develop a procedural design for a generalized Turing machine that will accept a set of quadruples as program input and produce output as specified.
9. Develop a procedural design for a program that will solve the Towers of Hanoi problem. Many books on artificial intelligence discuss this problem in some detail.
10. Develop a procedural design for all or major portions of an LR parser for a compiler. Refer to one or more books on compiler design.
11. Develop a procedural design for an encryption/decryption algorithm of your choosing.

Reading Lists

Week 10: Software Testing

Objectives
The objectives of this week are to enable the students to be able to:
(i) Describe some Software Testing Techniques;
(ii) Explain some Software Testing Strategies; and
(iii) List Software Testing Principles.

First Hour
Software Testing Fundamentals.

Second Hour
Software Testing Techniques/Strategies

Study Questions
1. Myers, G. (1979) uses the following program as a self-assessment for your ability to specify adequate testing: A program reads three integer values. The three values
are interpreted as representing the lengths of the sides of a triangle. The program prints a message that states whether the triangle is scalene, isosceles, or equilateral. Develop a set of test cases that you feel will adequately test this program.

2. Design and implement the program (with error handling where appropriate) specified in Problem 1. Derive a flow graph for the program and apply basis path testing to develop test cases that will guarantee that all statements in the program have been tested. Execute the cases and show your results.

3. Specify, design, and implement a software tool that will compute the cyclomatic complexity for the programming language of your choice. Use the graph matrix as the operative data structure in your design.

4. Give at least three examples in which black-box testing might give the impression that "everything's 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's OK," while black-box tests might uncover an error.

Reading List

Week 11: Software Quality Assurance

Objectives
The objectives of this week are to enable the students to be able to:
(i) Describe some Software Quality Assurance Concepts; and
(ii) Explain some Software Quality Assurance Approaches.

Descriptions
First Hour
Quality Concept.
Second Hour
Software Quality Assurance Approaches.

Study Questions
1. Is it possible to assess the quality of software if the customer keeps changing what it is supposed to do?
2. Quality and reliability are related concepts but are fundamentally different in a number of ways. Discuss them.
3. Can a program be correct and still not be reliable? Explain.
4. Can a program be correct and still not exhibit good quality? Explain.
5. You have been given the responsibility for improving the quality of software across your organization. What is the first thing that you should do? What's next?
6. Besides counting errors, are there other countable characteristics of software that imply quality? What are they and can they be measured directly?

Reading List
Week 12: Software Metrics

Objectives
The objectives of this week are to enable the students to:
   (i) Explain the challenge of technical metrics;
   (ii) List the measurement principles of software metrics; and
   (iii) Describe the attributes of effective software metrics.

Description
First Hour
Second Hour
Metrics for the Analysis Model

Study Questions

Reading List

Week 13: Metrics for the Design Model

Objectives
The objectives of this week is to discuss the various design metrics

Description
First Hour
Architectural Design Metrics
Second Hour
Component-Level Design Metrics, and Interface Design Metrics

Study Questions

Reading List


Week 14: Software Metrics

Objectives
The objective of this week is to expose the students to various software metrics.

Description
First Hour
Metrics for Source Code
Second Hour
Metrics for Testing and Metrics for Maintenance

Study Questions

1. Research the literature and write a paper on the relationship of Halstead's metric and McCabe's metric on software quality (as measured by error count). Are the data compelling? Recommend guidelines for the application of these metrics.
2. Research the literature for any recent papers on metrics specifically developed to assist in test case design. Present your findings to the class.
3. A legacy system has 940 modules. The latest release required that 90 of these modules be changed. In addition, 40 new modules were added and 12 old modules were removed. Compute the software maturity index for the system.

Reading List


**Week 15: Revision**

**Objectives**
All topics would be reviewed. Also questions will be asked from students to ascertain how well the course is understood by them.

**First Hour**
Test

**Second Hour**
Revision.

**Revision Questions**
(1) What is the purpose of modelling?
(2) A programming language is a notation for representing algorithms and data structures, List two advantages and two disadvantages of using a programming language as sole notation through the development process.
(3) What is meant by “knowledge acquisition is nonlinear”? Provide a concrete example of knowledge acquisition that illustrates this.
(4) What is the difference between a task and an activity?
(5) What are the four important attributes which all software products should have? Suggest four other attributes which may be significant.
(6) What is the difference between a software process model and software process? Suggest two ways in which a software process model might be helpful in identifying possible process improvements.
(7) Explain why system testing costs are particularly high for generic software products which are sold to a very wide market.
Give reasons for your answer based on the type of system being developed, suggest the most appropriate generic software process model which might be used as a basis for managing the development of the following systems:
(i) a system to control anti-lock braking in a car;
(ii) a virtual reality system to support software maintenance;
(iii) a university accounting system that replaces an existing system;
(iv) an interactive system for railway passengers that finds train times from terminals installed in stations.

Explain why programs which are developed using evolutionary development are likely to be difficult to maintain.

Explain how both the waterfall model of the software process and the prototyping model can be accommodated in the spiral process model.

Suggest why it is important to make a distinction between developing the user requirements and developing the system requirements in the requirements engineering process.

Describe the main activities in the software design process and the outputs of these activities. Using an entity – relation diagram, show possible relationships between the outputs of these activities.

What are the five components of a design method? Take any method which you know and describe its components. Assess the completeness of the method which you have chosen.

Design a process model for running system tests and recording their results.

Explain why a software system that is used in a real – world environment must change or become progressive less useful.

Explain why the intangibility of software system poses special problems for software project management.

Explain why the best programmer do not always make the best software managers.

Explain why the process of project planning is an iterative one and why a plan must be continually reviewed during a software project.

What is the critical distinction between a milestone and a deliverable?

Briefly explain the purpose of each of the section in a software project plan.

**General Reading Lists**