UNIVERSITY OF TECHNOLOGY, JAMAICA

SYLLABUS OUTLINE

FACULTY: Engineering and Computing (FENC)

SCHOOL/DEPT: School of Computing and Information Technology (SCIT)

COURSE OF STUDY: Master of Science in Software Engineering

YEAR/LEVEL: Five (5)

MODULE TITLE: **SOFTWARE QUALITY MANAGEMENT**

MODULE CODE: **ISM5022**

DURATION (Hours): Seventy-Five (75)

CREDIT VALUE: Three (3)

CO-REQUISITES:

PRE-REQUISITES:

1.0 MODULE DESCRIPTION

Offering enterprise level quality management concepts using modern technologies, frameworks, techniques and widely used industrial third party tools. At the end of the module, students will be able to integrate quality management at appropriate stages of the Software Development Life Cycle (SDLC) to improve the process using technologies, methods and techniques covered in the module.

2.0 MODULE OBJECTIVES/LEARNING OUTCOMES

Upon completion of this module, student should:

i. understand quality management as a fundamental building block in the Software Development Life Cycle (SDLC)
ii. understand pertinent topics such as maintenance testing, reviews, test development process, configuration management and test planning and estimation
iii. improve logical and problem-solving skills required to translate software system's requirements, structure and or architecture into test suites
iv. apply a modular approach in quality management integration in the SDLC

v. prepare testing and tracing scenarios for small to medium sized software projects
vi. demonstrate competence in executing test suites designed for small to medium sized software projects
vii. recognize the need for proper documentation to enhance software quality management for projects traceability
viii. work independently to build competence in the analysis of risks, writing of effective tests for software projects
ix. generate ethical code in accordance with the standards of the computing profession

3.0 MODULE CONTENT AND CONTEXT

UNIT 1 – Fundamentals of Testing
(3 hours: *2hrs Lecture/Tutorial, 1Hr Lab*)

Upon completion of Unit 1, the student should be able to:

1.1. recall the common objectives of testing
1.2. recall the five fundamental test activities and respective tasks from planning to closure
1.3. recall the psychological factors that influence the success of testing
1.4. describe, with examples, the way in which a defect in software can cause harm to a person, to the environment or to a company
1.5. distinguish between the root cause of a defect and its effects
1.6. give reasons why testing is necessary by giving examples
1.7. describe why testing is part of quality assurance
1.8. give examples of how testing contributes to higher quality
1.9. provide examples for the objectives of testing in different phases of the software development life cycle (SDLC)
1.10. differentiate testing from debugging
1.11. explain the seven principles in testing
1.12. contrast the mindset of a tester and of a developer

Content

- **Why is Testing necessary**
- **What is Testing**
- **Seven Testing Principles**
- **Fundamental Test Process**
- **Psychology of Testing**
- **Code of Ethics**

UNIT 2 – Testing throughout the Software Development Life Cycle
(6 hours: *4hrs Lecture/Tutorial, 2Hrs Lab*)

Upon completion of Unit 2, the student should be able to:

2.1. explain the relationship between development, test activities and work products in the SDLC, by giving examples using project and product types
2.2. compare the different levels of testing
2.3. compare four software test types using examples
2.4. identify and describe non-functional test types based on non-functional requirements
2.5. identify and describe test types based on the analysis of a software system's structure or architecture
2.6. compare maintenance testing to testing a new application with respect to test types, triggers for testing and amount of testing
2.7. describe the role of regression testing and impact analysis in maintenance

Content

- **Software Development Models (SDMs)**
- **Test Levels**
- **Test Types**
- **Maintenance Testing**

UNIT 3 – Static Techniques
(6 hours: *4hrs Lecture/Tutorial, 2Hrs Lab*)

Upon completion of Unit 3, the student should be able to:

3.1. describe the importance and value of considering static techniques for the assessment of software work products
3.2. explain the difference between static and dynamic techniques, considering objectives, types of defects to be identified, and the role of these techniques within the SDLC
3.3. explain the difference between different types of reviews: informal, technical, walkthrough and inspection
3.4. explain the factors for successful performance of reviews
3.5. describe, using examples, the typical benefits of static analysis

Content

- **Static Techniques and the Test Process**
- **Review Process**
- **Static Analysis by Tools**

UNIT 4 –Test Design Techniques
(12 hours: *8hrs Lecture/Tutorial, 4Hrs Lab*)

Upon completion of Unit 4, the student should be able to:

4.1. differentiate between a test design specification, test case specification and test procedure specification
4.2. compare the terms test condition, test case and test procedure
4.3. evaluate the quality of test cases in terms of clear traceability to the requirements and expected results
4.4. translate test cases into a well-structured test procedure specification at a level of detail relevant to the knowledge of the testers
4.5. explain the characteristics, commonalities, and differences between specification-based testing, structure-based testing and experience-based testing
4.6. write test cases from given software models using equivalence partitioning, boundary value analysis, decision tables and state transition diagrams/tables
4.7. describe the concept and value of code coverage
4.8. explain the concepts of statement and decision coverage
4.9. write test cases from given control flows using statement and decision test design techniques
4.10. assess statement and decision coverage for completeness with respect to defined exit criteria
4.11. compare experience-based techniques with specification-based techniques
4.12. classify test design techniques according to their fitness to a given context, for the test basis, respective models and software characteristics

Content

- **Test Development Process**
- **Categories of Test Design Techniques**
- **Specification-based/Black-box Techniques**
- **Structure-based/White-box Techniques**
- **Experience-based Techniques**
- **Technique Selection**

UNIT 5 – Test Management and Tool Support for Testing
(12 hours: *8hrs Lecture/Tutorial, 4Hrs Lab*)

Upon completion of Unit 5, the student should be able to:

5.1. explain the benefits and drawbacks of independent testing within an organization
5.2. differentiate between the subject of test planning for a system and scheduling test execution
5.3. write a test execution schedule for a given set of test cases, considering prioritization, and technical and logical dependencies
5.4. differentiate between two conceptually different estimation approaches: the metrics-based approached and expert-based approach
5.5. justify adequate entry and exit criteria for specific test levels and groups of test cases
5.6. explain and compare test metrics for test reporting and test control related to purpose and use
5.7. summarize how configuration management supports testing
5.8. distinguish between project and product risks
5.9. describe, using examples, how risk analysis and risk management may be used for test planning
5.10. write an incident report covering the observation of a failure during testing
5.11. classify different types of test tools according to their purpose and to the activities of the fundamental test process and the SDLC
5.12. explain the term test tool and the purpose of tool support for testing
5.13. summarize the potential benefits and risks of test automation and tool support for testing

Content

- **Test Organization**
- **Test Planning and Estimation**
- **Test Progress Monitoring and Control**
- **Configuration Management**
- **Risk and Testing**
- **Incident Management**
- **Types of Test Tools**
- **Effective Use of Tools**

4.0 LEARNING AND TEACHING APPROACHES

- Lectures provide coverage of software quality management topics to generate understanding
- Tutorials provide focused interactions for content for further understanding and application of knowledge
- Supervised practical in a lab environment to implement concepts, develop practical competence, problem solving, testing and debugging or real applications
- Unsupervised practical in a lab environment to facilitate independent discovery and self-directed learning
- Continuous weekly assessment to ensure program validity and accuracy during development
- Supplementary Lecture Notes

5.0 ASSESSMENT PROCEDURES

Coursework **60%**

Theory Test 1 *15%*
This test is designed to cover content in units 1 – 3

Theory Test 2 *15%*
This test is designed to cover content in units 4 – 5

Group Project *30%*
This assignment provides students the opportunity to demonstrate their knowledge and skills from all units to integrate quality management techniques in a software development project.

Project Continuous Assessment (10%)
Projects will be assessed on a weekly basis with respect to integration of quality management and team responsibilities.

Final Project Submission and Interview (20%)

Final Examination **40%**

6.0 BREAKDOWN OF HOURS

Lecture/Tutorial		26 (2 hours * 13)
Lab	– *supervised*	13 (1 hour * 13)
	– *unsupervised*	30 (2 hours * 15)
Assessment		6 (6 hours)
Total		***75 hours***

7.0 TEXTBOOKS AND REFERENCES

Required:

Software Testing and Quality Assurance Theory and Practice.
Kshirasagar Naik and Priyadarshi Tripathy. Wiley *(Latest Edition)*

Recommended:

Documentation for third party tools provided by the vendors of the tools

Platform Library Support and Documentation

8.0 NAME OF SYLLABUS

Tyrone A. Edwards (Writer)
Sean Thorpe (Evaluator)

9.0 DATE OF PRESENTATION OR REVISION

August 15, 2013

10.0 DATE OF ACCEPTANCE

_____
(Programme Director)

_____
(OCDE)