Subject Description Form

Subject Code	COMP 3222
Subject Title	Software Design Principles
Credit Value	3
Level	3
Pre-requisite/ Co-requisite/ Exclusion	Pre-requisites: COMP 2011, COMP 2021
Objectives	The objectives of this subject are to: 1. Educate students on the major principles and methods of software design based mainly on the object-oriented paradigm 2. Educate students on common methods, notations and tools for documenting software design
Intended Learning Outcomes	Upon completion of subject, students will be able to: *Professional/academic knowledge and skills* a. Analyze an information system problem using the object oriented paradigm b. Develop a solution for an information system problem using the object-oriented paradigm c. Document and present information system solutions using proper object-oriented notations and tools *Attributes for all-roundedness* d. solve complex problems in groups and develop group work.
Subject Synopsis/ Indicative Syllabus	1. *Principles of modularization:* modularization, cohesion, coupling, reuse, encapsulation, information hiding, abstract data types, objects. 2. *Principles of object-orientation*. Objects and classes, single and multiple inheritance, polymorphism, dynamic binding, abstract classes and interfaces. 3. *Requirements*: business models, use cases. 4. *Analysis*: boundary, control and entity classes; class modeling - class diagrams and responsibilities; dynamic modeling – state charts activity and interaction diagrams. 5. *Design*: Detailed design, attributes and methods, metrics. 6. *Implementation*: CASE tools, relevant UML diagrams, test cases. 7. *Design Patterns*: creational patterns, behavioral patterns, structural patterns, model-view-controller, frameworks supporting design patterns. 8. *Database issues*: mapping from OO design to relational databases. 9. *Advanced topics:* Distributed objects, pure object-oriented

systems such as Smalltalk, etc.

Teaching/Learning Methodology

Lectures, tutorials and laboratory

Core principles will be presented in lectures.

Class exercises and tutorials will be used to help students practice design principles taught in lectures.

Laboratories will be used to hands-on practice with tools and relevant programming exercises.

Sample designs and programs (both good ones and bad ones, and students' own work) will be used for analysis and practice.

Assignments and Projects

A term project will be used to practice the execution of an OO project. Students will be required to go through the complete requirement, design, implementation and testing software cycle.

Students will be required to submit the initial design, which will be marked and returned to the students as feedback, and to form the basis for revised design and subsequent implementation. Students will be required to produce a final report, and to present and demonstrate their project.

Students will be advised on important aspects of their term projects: team work, report writing, and presentation skills, at appropriate stages during the execution of the project.

The project will emphasize innovative user interaction rather than computational functionality.

Assessment Methods in Alignment with Intended Learning Outcomes

Specific assessment methods/ tasks	% weighting	Intended subject learning outcomes to be assessed			
		a	b	c	d
Class exercises, assignments, quizzes.	20%	✓	✓	✓	
Term project	35%	✓	✓	✓	✓
Examination	45%	✓	✓		
Total	100%				

Sample designs and programs (both good ones and bad ones, and students' own work) will be used for training and assessment on their skills in analyzing information system problems.

The students will be required to discuss the application of OO

	design principles in their term project, in their project reports – as a self- assessment of their skills in applying design principles. Students will be required to write a report and to verbally present their projects – as an assessment of their skills in reporting and presenting software design solutions. The project will be graded on functionality, application of design principles, effective presentation and proper reporting.	
Student study effort expected	Class Contact:	
	Lecture	39 hours
	Tutorial/Labs	0 hours
	Other student study effort:	
	Assignments, projects, self-study	66 hours
	Total student study effort	105 hours
Reading list and references	1. Stephen R. Schach. **Object-Oriented Software Engineering**, McGraw-Hill, International Edition, 2008. 2. Stephen R. Schach. **Object-oriented and Classical Software Engineering**, 7th Edition, McGraw-Hill, 2007. 3. Craig Larman. **Applying UML and Patterns**, Prentice-Hall, 2005. 4. Timothy C. Lethbridge & Robert Laganiere. **Object-Oriented Software Engineering – Practical software development using UML and Java**, McGraw-Hill, 2nd Edition, 2005. 5. Simon Bennet, Steve McRobb and Ray Farmer. **Object-oriented Systems Analysis and Design Using UML**, 4th Edition, McGraw-Hill, 2010.	