COURS OUTLINE

SOFTWARE ARCHITECTURE

Courses Title	**Software Architecture (SE 3244)**					
Instructor/	Engr. Mangong Clement Fosah					
contact N^{O}	+237 653 519 879 – mangong.clement@ictuniversity.org					
Duration of Course	Level	**Course Code**	**Credit value**	**Course type**	**Semester**	**Department**
	III	SE 3244			Fall	ICT
Course Description	The way that software components — subroutines, classes, functions, etc. — are arranged, and the interactions between them, is called architecture. In this course you will study the ways these architectures are represented, both in UML and other visual tools. This course provides the final elements necessary to the knowledge and practice of the profession. During the course you will explore the software engineering and application development practices that distinguish you as a software architect.					
Course Objective	Upon completing this course, the software engineering and application development architectural practices that set them apart as software architects will be explored by students.					
Expected Outcome	Student will be able to o Understand the importance of modelling and software design in the software development life cycle o Understand the big picture of UML; its history and components o Analyse software requirement and produce a several types of models (use case, static, dynamic interaction Models) o Produce a straightforward software design (a design from scratch) using the UML models o use of UML tools and on paper in designing a software model o Relate software code to its model and model back to the code o Apply design basic structural, creational and behavioural patterns to a software design o Apply design principles to object oriented software designs o					

WEEKLY	**TOPICS (Content)**

WORK LOAD	
Lesson 1	**Chapter One: Introduction to Software Architecture, Measuring Non-functional Requirements** • **Introduction to software Architecture** ✓ Overview of software architecture ✓ The main goals of software ✓ Designing a project to be successful ✓ The importance of software architecture ✓ Software Architecture vs Design ✓ Software connectors • **Measuring non-functional requirements** ✓ Overview of non-functional requirements and software quality attributes ✓ Classification of Non-functional requirements ✓ Representation of non-functional requirements ✓ Quality measures
Lesson 2	**Chapter Two: Software Architecture Documentation and Views** • **Short overview of architecture Documentation** ✓ Why document software architecture? ✓ Uses and audience for architecture ✓ Architecture and quality attributes ✓ Economics of architecture documentation ✓ The views and beyond "Method" ✓ Vues and beyond in an agile environment ✓ Architecture that changes faster than you can document them • **Seven rules for sound documentation**
Lesson 3	**Chapter Three: 4+1 View Model**

	• Architectural Model • View Model • Views, Models and Diagrams • The 4+1 view model ✓ Logical view, Process view, Development view, Physical view, Use case view ✓ Importance of 4+1 view model to a software architect • Why not called just 5 but 4+1 view • Relationship among the views • Decision made during modelling a system • Stakeholders' satisfaction with the 4+1 view model
Lesson 4	**Chapter Four: Software Architectural Styles and Patterns** • Over view of Software architecture styles and patterns • Origin of software architecture styles and patterns • Using software architecture patterns • Overusing architecture patterns • Understanding the difference between architecture styles and architecture patterns • Classification and application of software architectural styles and Patterns • Data - Cantered Software Architecture • Data Flow Software Architecture • Distributed Software Architecture • Interactive Software Architecture • Hierarchical Software Architecture

Lesson 5	**Chapter Six: Software Design Patterns** • Overview of software design patterns • Classification and some commonly used software design patterns • Creational design patterns • Structural design patterns • Behavioural design patterns
Lesson 6	**Chapter Eight: Designing Software Architectures** • Software architecture development processes • Designing software architecture
Lesson 7	**Chapter Ten: Evaluating Software Architectures** • Why evaluate an architecture? • When can an architecture be evaluated? • Who 's involved? • What results does an architecture evaluation produce? • For what qualities can we evaluate an architecture? • Why are quality attributes too vague for analysis? • What are the output of an architecture evaluation? • The architecture trade -off analysis method (ATAM) • The software architecture analysis method (SAAM) • Active Reviews of Intermediate Designs (ARID) • Attribute-Based Architectural Styles (ABAS) • Performance assessment of software architecture and software performance engineering (PE) • The cost benefit analysis method (CBAM) • Evaluating software architecture for stability and evolution
Lesson 8	**Chapter Eleven: Anti-patterns** • Overview of anti-pattern • Software development antipatterns • Software architecture antipatterns

Lesson 9	**Chapter Fourteen: Future challenges and emerging trends in software architecture discipline.**

Recommended Textbooks

- *Software Modelling & Design: UML, Use cases, Patterns & Software Architecture ; Hassan Gomaa*
- *Head First Design Patterns : A Brain-Friendly Guide; Eric Freeman & Elisabeth Robson with Kathy Sierra & Bert Bates*