

Addis Ababa Science and Technology University College of Electrical and Mechanical Engineering			
Course Code	SWEG3104		
Course Name	Software Requirements Engineering		
Coordinator	Department of Software Engineering		
Batch Info.	Semester II	A.Y: 2021/22	Year: III
Level	B.SC. In Software Engineering		
Location/s	Kilinto Akaki-Kaliti		
Instructor (s)	Behailu G. (Asst. Prof.)		
Credit Hour	3		
Prerequisites	SWEG3251		
Co-requisite (s)			
Synopsis	The purpose of requirements engineering is to develop a common understanding of the needs, priorities, and constraints relevant to a software system. Many software failures arise from an incomplete understanding of requirements for the software to be developed or inadequate management of those requirements. Specifications of requirements range in formality from completely informal (e.g., spoken) to rigorously mathematical (e.g., written in a formal specification language such as Z or first-order logic). In practice, successful software engineering efforts use requirements specifications to reduce ambiguity and improve the consistency and completeness of the development team's understanding of the vision of the intended software. Plan-driven approaches tend to produce formal documents with numbered requirements.		
Course Description	This course introduces students to the process of requirements engineering and helps them understand important issues in requirements engineering. It will also help them to learn and apply the RE concepts for elicitation, specification, modeling and analysis of software requirements. Important topics include Requirement engineering types, Requirements management and validation of requirements.		

Course objectives	Objectives of the course are to: • provide an overview of different Software Requirements types (functional vs non-functional (quality), constraints, business requirements, business rules, user and system requirements); • explain the Software Requirements Process (ISO/IEC/IEEE 29148 Standard for Requirements Engineering) also in agile contexts; • provide the instruments for the definition of user requirements; • describe different software requirements elicitation modalities; • provide approaches for requirements analysis and verification & validation; • provide approaches to manage requirements prioritization; • describe the software architecture and the relevance in the context of software requirements. • describe how to decompose system models: abstraction, & different system views for the definition of the software architecture from the requirements; • describe how to model Non-Functional Requirements (NFR);
Learning outcomes	At the end of the course students will: • have a clear understanding about processes, tools and techniques used in requirements engineering; • understand the concepts of software requirements elicitation, modeling, validation and verification; • be able to model software requirements rigorously according to the latest requirements engineering standards; • be able to conduct a prioritization process for software requirements according to different approaches; • be able to make a reasoned choice about the best approach for requirements modeling given the context of a project; • be able to properly manage requirements and their quality concerns; • understand the differences between different requirements modeling approaches (agile and non-agile); • be able to generate and maintain a software requirements specification document.
Syllabus	
Weeks Allotted	Chapters
1-2	1. Introduction to Requirements Engineering • Definitions, role in SDLC, layered model, types of requirements (Functional Requirements, Non-Functional

	Requirements), Modeling and RE, AS-IS vs TO-BE, RE process & activities, Requirements verification and validation.
3-4	2. Requirements Inception and Elicitation • Inception Requirements : ◦ Problem Analysis: ▪ Five Steps for Problem Analysis. ◦ Business Requirements: ▪ context diagram, ecosystem maps, events list, feature trees, the goal-design scale. • Elicitation Requirements: ◦ Goals, Risks, and Challenges, Sources of Requirements, Requirements Elicitation Tasks, Elicitation Problems. • Elicitation techniques: ◦ Analysis of Existing Systems, Interviews, Brainstorming, Prototyping, Use Cases (Stakeholders Analysis), Agile approaches with User Stories.
5-7	3. Developing Requirements Analysis • ISO/IEC/IEEE 29148 Standard for Requirements Analysis. • (C)lass (R)esponsability (C)ollaborators cards. Linking Requirements to UML Analysis Models. • Requirements Specification: ◦ Requirements document structure ◦ Requirements document standard and Guidelines. ◦ Examples: Case Studies
8-10	4. System Modeling Requirements • Goal modeling techniques (i*, KAOS etc.) • User Requirements: User Stories & Use cases modeling: ◦ Actor Role Models, Role Activity Diagrams (RADs) ▪ Case studies, examples and class activity ◦ Behavior-Driven Development (BDD): ▪ Feature-Driven Meta Modeling. ▪ Given-When-Then Approach • Decomposing system models: abstraction, & different system views. The Attribute-Driven Design (ADD) Method. • Features of Software/System Architecture Requirements
11-12	5. Requirements Verification & Validation • Model Checking: ◦ Consistency checks, CRUD checks, Acceptance Testing • Requirement Prioritization: ◦ Analytic Hierarchy (AHP) process, Software Quality Deployment Function (SQFD), the Agile Model Path Plans. • Quality Model Requirements: ◦ SQuaRE (Software product Quality Requirements and

	Evaluation) & ISO/IEC 25010.
13-14	6. Requirements Effort estimation & Project Velocity • Early models of effort estimation (LOCs based). ◦ Three modalities of estimation: COCOMO II, k-Nearest Neighbour, Planning Poker
Teaching Methods	• Lectures/Small Group Projects: ◦ Frontal lectures and a small interesting group project developed during lectures to see the practical application of theory and techniques seen during the course. • Frameworks and Tools: ◦ ReqIF Studio, Gherkin BDD, Cucumber BDD, Software Development IDE (E.g.: Eclipse IDE), Software Project Development Tools (E/g: PlantUML), Graph Modelling Framework (E.g: Graph Walker), Feature IDE (E.g: Feature Modeling Plugin, SPLOT, etc.).
Assessment Methods	• Attendance (5%), Tests (20%), Projects (25%), and Final Exam (50%). • Grade Policies: As per the University Legislation.

Text books:

- Pohl, Klaus. *Requirements engineering fundamentals: a study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant*. Rocky Nook, Inc., 2016.
- Sommerville, Ian. *Software Engineering GE*. Pearson Australia Pty Limited, 2016.

Recommended References:

- Young, Ralph Rowland. *The requirements engineering handbook*. Artech House, 2004.
- Winters, Titus, Tom Manshreck, and Hyrum Wright. *Software engineering at google: Lessons learned from programming over time*. O'Reilly Media, 2020.
- Macaulay, Linda A. *Requirements engineering*. Springer Science & Business Media, 2012.
- Elizabeth Hull, Ken Jackson and Jeremy Dick. "Requirements Engineering", 3rd Ed, Springer-Verlag London Limited, 2011.
- Bass, Len, Paul Clements, and Rick Kazman. *Software architecture in practice*. Addison-Wesley Professional, 2003.
- Axel Van Lamsweerde (2009) "Requirements Engineering: from system goals to UML Models to software specifications", Wiley, 2009.
- Cockburn, Alistair. "Writing effective use cases." *Addison-Wesley, Boston. Ferber D (2004) Synthetic biology. Microbes made to order. Science* 303 (2001): 158-161.
- Nesi, Hilary, and Sian Alsop. "Stories and scenarios: Lecturers' use of fantastic hypothetical events." *Journal of English for Academic Purposes* 53 (2021): 101022.
- Jeffries, Ron. "Extreme Programming and Agile Software Development Methodologies." (2003).