软件工程理论基础（英文）

课程名称/Course Title：软件工程理论基础（英文）/ Theoretical Foundations of Software Engineering

课程代码/Course Code：

任课教师/Instructor(s)：法赫德萨巴赫/ Fahad Sabah

开课学部（院）/Faculty：计算机学院

1．课程概要/Course Summary				
课程名称（中文） Course Title (Chinese)	软件工程理论基础（英文）			
课程名称（英文） Course Title (English)	Theoretical Foundations of Software Engineering			
适用层次(可多选) Applicable level(s)	☑学硕 (Academic master)	☐学博 (Academic Ph.D)	☑专硕 (Professional master)	☐专博 (Professional Ph.D)
授课语言 Teaching Language	英语 English		适用学科/专业学位类别(领域) Discipline	软件工程 Software Engineering
学分数 Course Credit(s)	48		开课学期 Semester	
总学时 Teaching Hours in Total	共 48 学时 48 teaching hours		实验/实践学时 Hours for Experiments /Practice	共 0 学时 0 teaching hours
预修课程要求 Pre-requisite Course(s)	面向对象程序设计 、离散数学 、软件工程基础 and 软件工程理论基础 Object Oriented Programming, Discrete Mathematics, Fundamentals of Software Engineering			
课程简介 Course Introduction （150-300 字）	本课程为期 12 周，每周 4 学时 ，聚焦软件工程的 形式化理论与核心原则 ，贯通抽象模型与真实系统设计的关联。通过形式化规约语言（Z 表示法、Alloy）、软件生命周期模型（瀑布模型、CSP 进程代数）和体系结构理论，学生将结合工具实践（SPIN、TLA+）与经典案例（如 Therac-25 医疗事故、Kubernetes 架构），探讨模块化、可扩展性与系统正确性的权衡。课程涵盖逻辑验证技术、质量度量指标（圈复杂度）及伦理框架，并深入 AI 代码生成、分布式系统等新兴领域。面向高年级本科生及研究生，课程强调以理论指导实践，培养批判性思维和伦理意识，为复杂软件工程的创新与责任奠定基础。 Spanning 12 weeks with 4 hours of engagement per week , this course delves into the theoretical foundations of software engineering , connecting			

	abstract principles to the design and verification of real-world systems. Students will explore formal methods (e.g., Z Notation, model checking), software lifecycle models (Agile, CSP), and architectural trade-offs through hands-on labs (SPIN, TLA+) and case studies like the Therac-25 failure and Kubernetes architecture. Emphasizing rigorous specification, ethical decision-making, and quality metrics (e.g., cyclomatic complexity), the curriculum also addresses emerging challenges such as AI-driven code generation and distributed systems. Designed for advanced undergraduates and graduate students, this course equips learners to critique methodologies, innovate theoretically grounded solutions, and navigate the societal impact of software technologies.

2. 授课团队/Teaching group

姓名 Name	法赫德萨巴赫 Fahad Sabah	职称 Title		工作证号 ID	
姓名 Name		职称 Title		工作证号 ID	
姓名 Name		职称 Title		工作证号 ID	

3. 教学目标/Course Objective （100-200 字）

本课程旨在帮助学生建立扎实的软件工程理论基础，使其能够运用形式化方法分析、设计和验证复杂软件系统。课程将深入探讨软件生命周期模型（如瀑布模型、敏捷开发、CSP 进程代数）、形式化规约语言（如 Z 表示法、Alloy）和软件体系结构描述方法（ADLs），使学生能够从理论层面理解软件工程的核心原则，并权衡抽象性、模块化和实际工程需求之间的关系。学生将学习使用逻辑验证工具（如 SPIN、TLA+）对系统行为进行建模和形式化验证，同时掌握软件质量保障技术，如变异测试和复杂度度量（圈复杂度、Halstead 度量）。此外，课程还将涵盖软件工程的经济与伦理维度，包括项目管理模型（COCOMO、EVM）以及人工智能代码生成等新兴技术的社会影响。通过结合经典案例（如 Therac-25 医疗事故、Kubernetes 架构分析），学生将培养批判性思维，能够应对分布式系统的挑战，倡导符合伦理的工程实践，并基于理论创新解决方案。

This course aims to equip students with a rigorous theoretical understanding of software engineering principles, enabling them to analyze, design, and verify complex software systems through formal methods. By exploring foundational theories such as software lifecycle models (e.g., Agile, CSP), formal specification languages (Z Notation, Alloy), and architectural description frameworks (ADLs) students will learn to critically evaluate trade-offs between abstraction, modularity, and practical implementation. They will apply logic-based tools (SPIN, TLA+) to model and verify system behavior, while mastering quality assurance techniques like mutation testing and complexity metrics (Cyclomatic, Halstead). Additionally, the course emphasizes ethical and economic dimensions of software engineering, including project management models (COCOMO, EVM) and the societal implications of emerging trends such as AI-driven code generation. By synthesizing theoretical frameworks with real-world case studies (e.g., Therac-25 failure, Kubernetes architecture), students will develop the ability to anticipate challenges in distributed systems, advocate for ethical practices, and innovate solutions grounded in formal rigor.

4. 教学内容及进度安排/Course Content & Schedule

课次 No.	教学周 Week	教学内容 Contents	作业/实验 Assignment
1	1	软件工程理论导论	Therac-25 事故分析

		• 软件工程的定义与范畴 • 理论视角与实践视角的对比 Introduction to Software Engineering Theory • Definition and scope of software engineering. • Theoretical vs. practical perspectives. Historical evolution	Analyzing the Therac-25 failure.
2	2	软件生命周期模型 • 传统模型（瀑布模型、V 模型） • 敏捷开发原则与理论权衡 • 形式化进程代数（如 CSP、Petri 网） Software Life Cycle Models • Traditional models (Waterfall, V-Model). • Agile principles and theoretical trade-offs. • Formal process algebras (e.g., CSP, Petri Nets).	敏捷方法与计划驱动方法对比 使用 Petri 网对简单流程建模 Agile vs. plan-driven approaches. Modeling a simple process using Petri Nets.
3	3	需求工程：传统方法 • 利益相关者分析、用例与场景 • 需求规约语言 Requirements Engineering: Traditional Methods • Stakeholder analysis, use cases, and scenarios. • Requirements specification languages	图书馆系统软件需求规格说明书（SRS）编写 Writing a software requirements specification (SRS) for a library system.
4	4	需求工程中的形式化方法 • 基于逻辑的规约（Z 表示法、Alloy 语言） • 模型检测基础 Formal Methods in Requirements • Logic-based specifications (Z Notation, Alloy Language). • Model-checking basics.	银行系统的 Z 规格说明编写 Writing Z schemas for a banking system.
5	5	软件设计原则 • 模块化、抽象与封装 • SOLID 原则与设计模式（如观察者模式、工厂模式） Software Design Principles • Modularity, abstraction, and encapsulation. • SOLID principles, design patterns (e.g., Observer, Factory).	基于耦合度与内聚度指标的软件设计质量评估 Evaluating design quality using coupling/cohesion metrics.
6	6	软件架构理论 • 架构风格（分层架构、MVC 架构、微服务架构） • 形式化架构描述语言（ADLs） Software Architecture Theory • Architectural styles (Layered, MVC, Microservices). • Formal architecture description languages (ADLs).	基于形式化架构描述语言的 Kubernetes 体系结构分析 Analyzing Kubernetes architecture using formal ADLs.
7	7	形式化验证与模型检测 • 时序逻辑（线性时序逻辑 LTL、计算树逻辑 CTL） • 工具：SPIN 模型检测器、TLA+形式化规约语言 Formal Verification & Model Checking	基于 SPIN 的交通灯系统验证 Verifying a traffic light system with SPIN.

		• Temporal logic (LTL, CTL). • Tools: SPIN, TLA+.	
8	8	软件测试理论 • 基于图的测试方法、变异测试 • 理论局限性（如停机问题） Testing Theory • Graph-based testing, mutation testing. • Theoretical limits (e.g., halting problem).	基于图覆盖的排序算法测试用例设计 Designing test cases for a sorting algorithm using graph coverage.
9	9	软件项目管理与经济学 • 构造性成本模型(COCOMO)、挣值管理(EVM) • 利益相关方谈判中的博弈论应用 Project Management & Economics • COCOMO model, Earned Value Management (EVM). • Game theory in stakeholder negotiation.	模拟实践：基于 COCOMO 模型的项目工期优化 Simulation: Optimizing project timelines with COCOMO.
10	10	软件度量与质量 • 复杂度度量（圈复杂度、Halstead 复杂度） • ISO/IEC 25010 质量模型 Software Metrics & Quality • Complexity metrics (Cyclomatic, Halstead). • ISO/IEC 25010 quality model.	开源项目度量指标计算 Calculating metrics for open-source projects
11	11	软件工程新兴趋势与伦理 • 人工智能在软件工程中的应用 • 伦理框架（如 ACM 伦理准则） Emerging Trends & Ethics • AI in software engineering. • Ethical frameworks (e.g., ACM Code of Ethics).	
12	12	课程复习与项目汇报 Review & Project Presentations	期末项目汇报 期末考试复习与答疑 Term project presentations. Final exam review and Q&A.

5. 课程考核及成绩评定/Course Assessment & Grading

考核指标* Assessment Criteria	权重 Percentage	评定标准 Assessment Standard
出勤 Attendance		
课堂表现 Participation	10	积极参与课堂讨论 Participation in group discussions
作业/实验 Assignment(s)	20	按时提交所有作业 Timely submission of all assignments 代码符合规范要求 Code meets specified standards 功能实现完整

		Complete implementation of required functions 报告内容清晰准确 Clear and accurate reports
课程论文 Course Paper		
课程考试 Course Exam(s).	30	期末考试（30%）：综合考核全部课程内容 Final (30%): Comprehensive assessment of all course content
项目 Project	40	项目完整性（16%）：实现所有要求功能 Completeness (16%): All required features implemented 代码质量（12%）：符合 PEP8 规范，结构清晰 Code quality (12%): PEP8 compliant, well-structured 文档（8%）：包含清晰的 README 和注释 Documentation (8%): Clear README and comments 创新性（4%）：体现创造性解决方案 Creativity (4%): Demonstrates innovative solutions
* 各项考核指标可自由设置，总权重为 100%。		

6. 教材/Textbook(s) （如使用自编讲义，请在“名称”列中备注说明）

序号 No.	名称 Title	作者 Author(s)	标准书号 ISBN	出版机构 Publisher	出版日期 Publication Date	是否必读 Mandatory or Elective
1	Software Engineering: A Practitioner’s Approach	Roger Pressman and Bruce Maxim	9781260548006	McGraw-Hill Education	2019	Mandatory
2	Engineering Software Products: An Introduction to Modern Software Engineering	Ian Sommerville	9781292376349	Pearson	2020	Elective

7.教学参考资料/Reading Materials and References

序号 No.	名称 Title	作者 Author(s)	标准号码 ISBN/DIO	出版机构 Publisher	出版日期 Publication Date	是否必读 Mandatory or Elective
1	Software engineering with uml	Bhuvan Unhelkar	9781351235181	Auerbach Publications	2017	Elective