Modul: Software Engineering

Dekan/in:	Prof. Dr. Oliver Koch	
Modulleitung:	Prof. Dr. Oliver Linssen	
Praesenzstd:	52,0	UE
Eigenstudium:	66,00	ZStd
Student Consulting / Praxistransfer:	45,00	ZStd
Workload:	150,00	ZStd
ECTS:	6	

Modulziel

Die Studierenden können nach erfolgreichem Abschluss des Moduls:

- Unterschiedliche Lösungsmöglichkeiten für typische Probleme der Softwareentwicklung auswählen und anwenden,
- Methoden des Software Engineerings problem- und praxisgerecht auswählen und diese in verschiedenen Situationen anwenden,
- die Methodik des Software Engineerings erklären und die unterschiedlichen Vorgehensweisen vergleichen,
- Anforderungen an Softwaresysteme ermitteln und dokumentieren,
- verschiedene Modelle der UML im Entwicklungsprozess einer Software auswählen und anwenden,
- unterschiedliche Aspekte des Entwurfs von Software (z. B. Modularisierung, Information Hiding, hohe Bindung (Kohäsion), geringe Kopplung) erklären und anwenden,
- Methoden zur Sicherstellung der Qualität erklären, vergleichen und anwenden,
- Methoden zur Aufwandsschätzung im Software Engineering gegenüberstellen.

Arbeitsmarktrelevanz

Die Arbeitsmarktrelevanz von Software Engineering und Projektmanagement ist bei immer komplexer werdenden IT-Systemen und der immer weiter fortschreitenden Digitalisierung aller betrieblichen und privaten Lebensbereiche sehr hoch und wird in der Zukunft weiter zunehmen. Softwaresysteme können hier als die Motoren des 21. Jahrhunderts betrachtet werden. Dabei ist das in der Praxis vorhandene Wissen über die Methodik der Softwareentwicklung häufig stark verbesserungswürdig. Das Modul versetzt die Studierenden in die Lage, sich in Richtung einer/eines methodisch breit qualifizierten Softwareentwicklerin/Softwareentwicklers zu entwickeln. Dabei werden insbesondere Schnittstellenkompetenzen zwischen Informatik und Betriebswirtschaftslehre zu nutzen sein. Das erworbene Wissen kann zum erfolgreichen Umsetzen komplexer IT-Projekte entscheidend beitragen. Die hier vermittelten Kenntnisse und Fähigkeiten sind auch außerhalb der Softwareentwicklung in einer Vielzahl unterschiedlicher Kontexte (Qualitätsmanagement, Beschaffung, Produktentwicklung, Projektmanagement, Planung

etc.) von hohem Wert.

Lehrmethodik

- Vorlesungen
- Themenbezogene Diskussionen
- Übungen und Fallstudien
- Online-Campus

Curriculum

Einführung

- Was Software Engineering ist und warum Software Engineering notwendig ist
- Umfang und Themen der Disziplin Software Engineering

Vorgehensmodelle

- Prinzipielle Vorgehensmodelle
- V-Modell XT und Rational Unified Process
- Agiler Ansatz, Agiles Manifest, Agile Prinzipien
- Scrum
- Software-Kanban und Extreme Programming

Requirements Engineering

- Problemstellung und Aufgaben, funktionale und nicht-funktionale Anforderungen
- Anforderungsermittlung
- Use-Case-Spezifikationen, User Storys
- Anforderungsschablonen und Gliederung von Anforderungsdokumenten
- Prüfen von Anforderungen, Aufgaben im Anforderungsmanagement

Modellierung von Softwaresystemen

- UML: Geschichte, Aufbau, Verwendung der Diagramme, typische Abfolge von Diagrammen in der Entwicklung
- Use-Case-Diagramm
- Aktivitätsdiagramm
- Klassendiagramm
- Sequenzdiagramm
- Zustandsdiagramm
- Übrige Diagramme der UML
- Modellgetriebene Entwicklung (CASE, MDA), andere Ansätze zur Modellierung

Software-Entwurf

- Problemstellung, Ziele im Entwurf, Entwurf und Software-Architektur
- Prinzipien des Architekturentwurfs: Modularisierung, Information Hiding, Kohäsion und Kopplung, hierarchische Gliederung, Trennung von Zuständigkeiten

- Objektorientierter Entwurf, SOLID-Kriterien
- Entwurfs-Pattern: Idee, Dokumentation, Systematik der Gang-of-Four-Pattern
- Architektur-Pattern: Schichten, Pipe-and-Filter, MVC, SOA, Microservice
- Frameworks, Referenzarchitekturen, Interaction Design

Implementierung und Integration
- Eigenschaften professioneller Programmierung, Clean-Code-Ansatz
- Integration: Problemstellung, Integrationsstrategien (Big Bang Integration, Top-Down Integration, Bottom-Up Integration, inkrementelle Integration, fortlaufende Integration)

Software-Qualität und Software-Test
- Qualitätsmodell für Software, analytische und konstruktive Maßnahmen, statische und dynamische Verfahren der Qualitätssicherung
- Manuelle Prüfverfahren: Reviews, Inspektion, Walkthrough
- Systemtisches Testen, Testprozess, Testarten und Teststufen
- Auswahl von Testfällen: Aquivalenzklassen, Überdeckungsmaße

Software-Konfigurationsmanagement
- Begriffe, Aufgaben des Konfigurationsmanagements, Versionskontrolle
- Software-Metriken: McCabe, Halstead, Chidamber &Kemerer

Wartung und Wiederverwendung
- Software-Wartung
- DevOps
- Re-Engineering, Reverse-Engineering, Refactoring
- Software-Wiederverwendung

Aufwandsschätzung
- Problemstellung, Bottom-Up-Verfahren, Top-Down-Verfahren
- Analogieverfahren, Schätzung auf Grundlage der Aufwandsverteilung auf Phasen
- COCOMO, Function-Point-Verfahren
- Planning Poker

Prüfung und Benotung
Klausur 90 Minuten (100% der Modulnote)
Transferaufgabe in der Klausur (etwa 10% des Klausurumfangs):
In der Veranstaltung werden Arbeits- und/oder Rechercheaufgaben mit explizitem Transferbezug zum praktischen Umfeld der Studierenden gestellt. Im Rahmen der Klausur wird dann durch angelehnte (nicht zwingend identische) Fragestellungen die Übertragung wissenschaftlicher Inhalte und Methoden auf konkrete betriebliche oder gesellschaftliche Probleme reflektiert.

Teilnahmevoraussetzungen und Vorkenntnisse

BWI

Die Teilnahme an folgenden Modulen wird empfohlen:

- Wirtschaftsinformatik Basics
- Konzepte d. prozeduralen Programmierens
- Konzepte d. objektorientierten Programmierens
- Algorithmen &Datenstrukturen
- Datenbankmanagement

Dieses Modul bereitet vor auf

BWI

- Keine

Literatur

Pflichtliteratur:

- Ludewig, J., &Lichter, H. (2013). Software Engineering: Grundlagen, Menschen, Prozesse, Techniken (3. Aufl.). Heidelberg: dpunkt Verlag.
- Kleuker, S. (2018). Grundkurs Software-Engineering mit UML: Der pragmatische Weg zu erfolgreichen Softwareprojekten (4. Aufl.). Wiesbaden: Springer Vieweg.

Ergänzende Literatur:

- Anderson, D. J., &Reinertsen, D. G. (2010). Kanban: Successful Evolutionary Change for Your Technology Business. Sequim, Washington: Blue Hole Press.
- Bourque, P., &Fairley, R. E. (Hrsg.). (2014). Guide to the Software Engineering Body of Knowledge, Version 3.0. Washington DC: IEEE Computer Society. Abgerufen 2. Januar 2018, von https://www.computer.org/web/swebok/v3
- Essigkrug, A., &Mey, T. (2007). Rational Unified Process kompakt (2. Aufl.). Heidelberg: Spektrum Akademischer Verlag.
- Grechenig, T., Bernhart, M., Breiteneder, R., &Kappel, K. (2010). Softwaretechnik: Mit Fallbeispielen aus realen Entwicklungsprojekten. München: Pearson Studium.
- IREB International Requirements Engineering Board e.V. (2017). IREB Certified Professional for Requirements Engineering. Foundation Level. Lehrplan Version 2.2.1 v. 24.7.2017. Abgerufen 2. Januar 2018, von https://www.ireb.org/en/downloads/#syllabus-foundation-level
- Object Management Group (2003). MDA Guide Version 1.0.1. OMG Document omg/2003-06-01. Abgerufen 2. Januar 2018, von http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf
- Object Management Group (OMG). (o. J.). UML 2.5. Abgerufen 2. Januar 2018, von http://www.omg.org/spec/UML/2.5/
- Pohl, K., &Rupp, C. (2015). Basiswissen Requirements Engineering: Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level (4., überarb. Aufl.). Heidelberg: dpunkt Verlag.

- Rubin, K. S. (2014). Essential Scrum: Umfassendes Scrum-Wissen aus der Praxis. Heidelberg u.a.: mitp.
- Rupp, C., Queins, S., &Zengler, B. (2007). UML 2 glasklar: Praxiswissen für die UML-Modellierung (3., aktual. Aufl.). München: Hanser Verlag.
- Rupp, C., &SOPHISTen, die. (2014). Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil (6., aktual. u. erw. Aufl.). München: Hanser Verlag.
- Schwaber, K., Sutherland, J. (2017). Der Scrum Guide. Der gültige Leitfaden für Scrum: Die Spielregeln. Abgerufen 2. Januar 2018, von https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-German.pdf
- Sommerville, I. (2015). Software Engineering, Global Edition (10. Aufl.). Harlow, Essex: Pearson Education.
- Sommerville, I. (2012). Software Engineering (9., aktual. Aufl.). Hallbergmoos: Pearson Studium.
- Staud, J. L. (2009). Unternehmensmodellierung: Objektorientierte Theorie und Praxis mit UML 2.0. Berlin: Springer.
- Verein zur Weiterentwicklung des V-Modell XT e.V. (o.J.). V-Modell XT. Abgerufen 2. Januar 2018, von http://www.cio.bund.de/Web/DE/Architekturen-und-Standards/V-Modell-XT/vmodell_xt_node.html

Weitere Literatur wird durch den am Standort zuständigen Dozenten bekannt gegeben.