Modulbeschreibung

1	**1.1 Modulbezeichnung (dt. / engl.)** **Fortgeschrittenes Software Engineering**	**1.2 Kurzbezeichnung (optional)**	**1.3 Modul-Code** (aus HIS-POS)
2	**2.1 Modulturnus:** **Angebot in ☒ jedem SoSe, ☐ jedem WiSe,** **anderer Turnus, nämlich:**	**2.2 Moduldauer:** **☒ 1 Semester ☐ 2 Semester**	
3	**3.1 Angebot für folgenden Studiengang/folgende Studiengänge**	**3.2 Pflicht, Wahlpflicht, Wahl**	**3.3 Empfohlenes Fachsemester**
	Master Wirtschaftsinformatik	Pf	2

4 Workload

		SWS je Lehrform	Std. pro Semester je Lehrform/ angegebener Form 1 SWS darf als 15 Zeitstunde angesetzt werden, d. h. 1 SWS = 1 UStd. x 15 Semesterwochen	Workload insgesamt: Arbeitsaufwand in Std. (Workload) Summe Kontaktzeit + Summe Selbststudium in Std.	Workload insgesamt: Leistungspunkte (Credits) i. d. R. 30 Std. = 1 LP; nur ganze Zahlen zulässig!
Kontaktzeit (z. B. Vorlesung, Übung, Praktikum, seminaristischer Unterricht, Projekt-/ Gruppenarbeit, Fallstudie, Planspiel, kreditiertes Tutorium) (weitere Zeilen möglich)	Seminaristischer Unterricht	2	30	**150**	**5**
	Übung	1	15		
	Summen	Summe Kontaktzeit in SWS	Summe Kontaktzeit in Std.		
Selbststudium (z. B. Tutorium, Vor-/ Nachbereitung, Prüfungsvorbereitung, Ausarbeitung von Hausarbeiten, Recherche)	Entwicklung eines Prototyps und/oder Ausarbeitung eines Konzeptes		65		
	Erstellung einer Präsentation		20		
	Selbststudium		20		
	Summen		Summe Selbststudium in Std.		

5 5.1 Lernziele (Was sollen Studierende nach Abschluss des Moduls können? Bietet das Modul neben fachlichen Lernzielen Gelegenheiten, außerfachliche Kompetenzen zu entwickeln? Wofür sind die beschriebenen Ziele relevant (z. B. Voraussetzung für weitere Studienelemente oder für bestimmte berufliche Tätigkeiten)?)

Die Absolventen können fortgeschrittene Aufgabenstellungen des Software Engineerings bearbeiten, die aus dem Einsatz bereits bestehender Software oder neuartiger Ansätze resultieren. Die Absolventen besitzen Kenntnisse, wie sich die Implementierung, das Design bzw. die Anforderungen bestehender Software mit Hilfe von Werkzeugen analysieren lässt. Sie können den Aufbau, die Anwendung und den Nutzen neuerer Technologien (Container, Blockchain etc.) darlegen und anwenden sowie für verschiedene weitere spezielle Programmieransätze (aspekt-orientiert, funktional, polyglott etc.) abwägen, ob deren Einsatz förderlich oder kontraproduktiv ist. Sie können die wesentlichen Anforderungen an sog. kritische Software-Systeme wiedergeben und mögliche materielle und immaterielle Auswirkungen von Software-Fehlern benennen. Die Absolventen können eine gegebene Aufgabenstellung bzgl. ihrer Kritikalität einordnen und entsprechende qualitätssteigernde Maßnahmen zu deren Umsetzung vorschlagen bzw. bewerten und die Resultate verifizieren.

5.2 Lerninhalte

- Methoden und Werkzeugunterstützung in Software-Projekten
- Ansätze der Virtualisierung
- Reverse Engineering
- Aspektorientierte Software-Entwicklung
- Polyglotte Programmierung
- Funktionale Programmierung
- Blockchain-Technologien
- Entwicklung kritischer Systeme (Grundlagen, Spezifikation, Umsetzung, Validierung)
- Bewertung von Zuverlässigkeit

→ zu den Details: siehe Vorlesungsverzeichnis, Lehrveranstaltungsplan etc.

5 **5.3 Modulkurzinformation** (Dieser Absatz [max. 250 Zeichen] wird auf der FH-Webseite veröffentlicht, um Studieninteressierte bei der Wahl ihres Studiengangs zu unterstützen. Fokussieren Sie sich auf wesentliche Inhalte und Ziele, gern verbunden mit Aussagen zur Bedeutung des Moduls für das weitere Studium oder berufliche Tätigkeiten. Bitte formulieren Sie ganze Sätze, sprechen Sie die Adressaten direkt an und vermeiden Sie Fachtermini.)

Sie lernen weitere Ansätze und Techniken der Programmierung und des Reverse Engineerings kennen. Darüber hinaus verstehen Sie die wesentlichen Anforderungen an sog. kritische Software-Systeme sowie Ansätze zur Bewertung von Zuverlässigkeit.

6 **6.1 Teilnahmevoraussetzungen** (*Formal:* Prüfung in Modul XY muss bestanden sein o. ä.; *Inhaltlich:* Modul XY sollte absolviert sein, folgende Kenntnisse sollten vorhanden sein, …)

6.2 Voraussetzungen für die Vergabe von Leistungspunkten (z. B. Bestehen der Prüfung, erfolgreicher Abschluss einer Studienleistung, regelmäßige und aktive Teilnahme)

Bestehen der Prüfung

6.3 Prüfungsformen und -umfang (z. B. Klausur, mündliche Prüfung, Hausarbeit, Präsentation, Portfolio, Dauer der Prüfung in Min.)

Seminararbeit
Präsentation

6.4 Voraussetzungen für die Zulassung zur Prüfung

6.5 Gewichtung der Note bei Ermittlung der Endnote

s. Prüfungsordnung/ -en für oben (Zeile 3) genannte Studiengänge*

*Die Prüfungsordnungen der Studiengänge finden Sie in den Amtlichen Bekanntmachungen der FH Münster unter dem folgenden Link https://www.fh-muenster.de/hochschule/aktuelles/amtliche_bekanntmachungen/index.php?p=2,7.

7 **7.1 Veranstaltungssprache/n**

☒**Deutsch** ☐ **Englisch** ☐ **Weitere, nämlich:**

7.2 Modulverantwortliche/r

Prof. Dr. Claus Grewe

7.3 Hauptamtlich Lehrende (optional)

Prof. Dr. Claus Grewe und optional wissenschaftliche Mitarbeiter

7.4 Maximale Teilnehmerzahl (optional)

7.5 Ergänzende Informationen (optional) (z. B. Literaturempfehlungen, weitere beteiligte Personen etc.)

Sommerville, I.: Software Engineering. Pearson Studium. 2018
Fowler, M.; et al.: Refactoring: Improving the Design of Existing Code. Addison-Wesley, 1999
Subramanian, V.: Functional Programming in Java. Harnessing the Power of Java 8 Lambda Expressions. Dallas, Texas: The Pragmatic Programmers, 2014