

# Modul

## SE - Software Engineering

Bachelor Elektrotechnik 2020

---

Version: 1 | Letzte Änderung: 27.10.2019 21:01 | Entwurf: 0 | Status: vom Modulverantwortlichen freigegeben | Verantwortlich: Kreiser

### ^ Allgemeine Informationen

<b>Anerkannte Lehrveranstaltungen</b>	<a href="#">SEA Kreiser</a>
<b>Fachsemester</b>	4
<b>Modul ist Bestandteil der Studienschwerpunkte</b>	<a href="#">SE - Smart Energy</a> , <a href="#">AU - Automatisierungstechnik</a>
<b>Dauer</b>	1 Semester
<b>ECTS</b>	5
<b>Zeugnistext (de)</b>	Software Engineering Automatisierungstechnik
<b>Zeugnistext (en)</b>	Software Engineering Automatisierungstechnik
<b>Unterrichtssprache</b>	deutsch und englisch
<b>abschließende Modulprüfung</b>	Ja

### Modulprüfung

<b>Benotet</b>	Ja
<b>Frequenz</b>	Jedes Semester

### Prüfungskonzept

Mündliche Prüfung nach schriftlicher Vorbereitung.

Anhand einer natürlichsprachlichen Beschreibung eines realitätsnahen Automatisierungssystems angemessener Komplexität modellieren die Studierenden das Systemmodell eines zur Lösung der Automatisierungsaufgabe geeigneten Softwaresystems und begründen und bewerten die

wesentlichen Eigenschaften ihres Entwurfs. Zur Begründung und Bewertung nehmen die Studierenden Bezug auf die spezifischen Anforderungen an das Automatisierungssystem sowie auf grundlegende Qualitätskriterien für automatisierungstechnische Softwaresysteme (System-, Entwicklungs-, Betriebs-, Service- und Wartungsanforderungen) und zeigen dabei an ausgewählten Modellartefakten insbesondere, dass sich und wie sich das Systemmodell in ein Softwaremodell und anschließend in ein Implementierungsmodell transformieren lässt und welche Konsequenzen ihr Entwurf für die Modelle der nachfolgenden Entwurfsphasen hat.

## ^ Allgemeine Informationen

### Inhaltliche Voraussetzungen

<b>IP - Informatik Projekt</b>	Erfassen einer in natürlicher Sprache gegebenen Softwarespezifikation Programmieren in einer prozeduralen Programmiersprache
<b>PI2 - Praktische Informatik 2</b>	Erfassen einer in natürlicher Sprache gegebenen Softwarespezifikation Programmieren in einer objektorientierten Programmiersprache Klassen und Objekte
<b>EPR - Erstsemesterprojekt</b>	zielgerichtetes Arbeiten im Team

### Kompetenzen

Kompetenz	Ausprägung
Erkennen, Verstehen und analysieren technischer Zusammenhänge	diese Kompetenz wird vermittelt
Technische Zusammenhänge darstellen und erläutern	diese Kompetenz wird vermittelt
Finden sinnvoller Systemgrenzen	diese Kompetenz wird vermittelt
Abstrahieren	diese Kompetenz wird vermittelt
Technische Systeme analysieren	diese Kompetenz wird vermittelt
Technische Systeme entwerfen	diese Kompetenz wird vermittelt
Technische Systeme realisieren	diese Kompetenz wird vermittelt
Technische Systeme prüfen	diese Kompetenz wird vermittelt
Informationen beschaffen und auswerten	diese Kompetenz wird vermittelt
Arbeitsergebnisse bewerten	diese Kompetenz wird vermittelt
Komplexe technische Aufgaben im Team bearbeiten	diese Kompetenz wird vermittelt
Lernkompetenz demonstrieren	diese Kompetenz wird vermittelt

Sprachliche und interkulturelle Fähigkeiten anwenden	diese Kompetenz wird vermittelt
Gesellschaftliche und ethische Grundwerte anwenden	diese Kompetenz wird vermittelt
In unsicheren Situationen entscheiden	diese Kompetenz wird vermittelt
Sich selbst organisieren und reflektieren	diese Kompetenz wird vermittelt
Betriebswirtschaftliches und rechtliches Grundwissen benennen, erklären und anwenden	Voraussetzungen für diese Kompetenz (Wissen,...) werden vermittelt

## ^ Vorlesung / Übungen

### Exemplarische inhaltliche Operationalisierung

System- und Softwaremodellierung anhand ausgewählter Struktur- und Verhaltensnotationen der Unified Modeling Language (UML2) und/oder weiterer/anderer in der industriellen Praxis gebräuchlicher Notationen. Der Fokus der Betrachtungen liegt auf den frühen Projektphasen der Softwareentwicklung bis zur Konzeptphase, da hier der größte Teil der Lebenszykluskosten des Softwareprodukts verursacht wird.

### Separate Prüfung

keine

## ^ Praktikum

### Exemplarische inhaltliche Operationalisierung

Einsatz eines professionellen UML2-Modellierungswerkzeugs, das Round-Trip-Engineering unterstützt, z.B. Visual Paradigm. Einsatz einer professionellen Softwareentwicklungsumgebung für C++, z.B. Microsoft Visual Studio oder Eclipse-basierte Umgebungen. Als Laufzeitsystem kann z.B. ein PC mit einer Geräte- oder Anlagenemulation oder ein reales technisches Zielsystem (Gerät, Anlage) mit eingebetteter Steuerung zum Einsatz kommen.

### Separate Prüfung

<b>Benötet</b>	Nein
<b>Frequenz</b>	Einmal im Jahr
<b>Voraussetzung für Teilnahme an Modulprüfung</b>	Ja

## Prüfungskonzept

Round-Trip-Engineering: nachvollziehbare, auf Basis der Anforderungen und Rahmenbedingungen begründete Transformationen zw. Systemmodell, Softwaremodell, Implementierungsmodell und Quellcode durchführen.

## ^ Projekt

### Exemplarische inhaltliche Operationalisierung

Entwicklung des Systemmodells und ggfs. wesentlicher Artefakte des Softwaremodells / der Softwarearchitektur eines zur Lösung einer realitätsnahen Automatisierungsaufgabe geeigneten Softwaresystems. Die Komplexität und der erwartete Arbeitsumfang zur Lösung der Aufgabenstellung richten sich nach dem verfügbaren Zeitkontingent des Projektteams (abh. von der Teamgröße).

### Separate Prüfung

<b>Benotet</b>	Nein
<b>Frequenz</b>	Einmal im Jahr
<b>Voraussetzung für Teilnahme an Modulprüfung</b>	Ja

## Prüfungskonzept

Anhand einer natürlichsprachlichen Beschreibung (Englisch) eines realitätsnahen Automatisierungssystems angemessener Komplexität modellieren die Studierenden das Systemmodell eines zur Lösung der Automatisierungsaufgabe geeigneten Softwaresystems und begründen und bewerten die wesentlichen Eigenschaften ihres Entwurfs. Zur Begründung und Bewertung nehmen die Studierenden Bezug auf die spezifischen Anforderungen an das Automatisierungssystem sowie auf grundlegende Qualitätskriterien für automatisierungstechnische Softwaresysteme (System-, Entwicklungs-, Betriebs-, Service- und Wartungsanforderungen) und zeigen dabei an ausgewählten Modellartefakten insbesondere, dass sich und wie sich das Systemmodell in ein Softwaremodell und anschließend in ein Implementierungsmodell transformieren lässt und welche Konsequenzen ihr Entwurf für die Modelle der nachfolgenden Entwurfsphasen hat.