



Angebote an Themen für Studienarbeiten, Stand: September 2011

Vorgehen

Bei Interesse an einem oder mehreren der Themen ist das Verfahren das Folgende:

- Die Ansprechpartner des Lehrstuhls stehen für Rückfragen zu den Themen zur Verfügung.
- **Bei Interesse an einem oder mehreren Themen muss jeweils ein schriftliches Exposé am Lehrstuhl abgegeben werden.**
- Aus den abgegebenen Exposés werden die besten für jedes Thema ausgewählt.

Das Exposé für das gewünschte Thema sollte beinhalten:

- Angaben zur Person und zu Vorkenntnissen
- Motivation und Problemstellung des gewünschten Themas
- Theoretischer Hintergrund und erster Literaturüberblick
- Geplante Vorgehensweise, Verfahren und Werkzeuge
- Zeitplan und Organisation der Arbeit

Ansprechpartner:

	E-Mail	Raum	Telefon
Marco Müller	marco.mueller@s3.uni-due.de	SGW 102	(0201) 183-4682
Michael Striewe	michael.striewe@s3.uni-due.de	SGW 102	(0201) 183-4682

Themenbereich „Softwarearchitekturen“

Architekturadaption auf Basis von dynamischen Quality-of-Service-Annotationen

- Ansprechpartner: Marco Müller
- Art der Arbeit:
 - Projektseminar und Bachelor-Abschlussarbeit
- Beschreibung: Bei der Verwendung von externen Services in Service-orientierten Architekturen kann die aktuelle Qualität des Services eine wichtige Rolle für das Gesamtsystem spielen. In einem Projektseminar oder Studienprojekt soll eine Technik entwickelt und implementiert werden um die Qualität eines Services zur Laufzeit zu messen und die Ergebnisse den Servicenutzern zur Verfügung zu stellen. In einer darauf aufbauenden Abschlussarbeit sollen diese Informationen dazu genutzt werden, um die Qualität von Services zur Laufzeit zu vergleichen und ggf. die Systemarchitektur mit Rücksicht auf die Qualitätsanforderungen anzupassen.
- Erforderliche Vorkenntnisse: Java, ggf. Serviceorientierung, ggf. komponentenbasierte Entwicklung

Auffinden von Services durch Schnittstellenvergleiche

- Ansprechpartner: Marco Müller
- Art der Arbeit: Projektseminar
- Beschreibung: In komponentenbasierten Systemen werden Komponenten miteinander verknüpft um gemeinsam ein Gesamtsystem zu bilden. Die einzelnen Module kommunizieren dabei über Schnittstellen, indem Sie angeben, welche Schnittstellen sie fordern und welche Sie anbieten. Es existiert ein prototypischer Ansatz zum Vergleich von geforderten und angebotenen Schnittstellen zum Auffinden von geeigneten Komponenten zur Entwicklungszeit. Der Prototyp soll so angepasst werden, dass der Vergleich und die Bindung der Komponenten zur Laufzeit geschieht.
- Erforderliche Vorkenntnisse: Java, ggf. komponentenbasierte Entwicklung

Intermodell-Verifikation von Sequential Contracts

- Ansprechpartner: Marco Müller
- Art der Arbeit: Master-Abschlussarbeit
- Beschreibung: In serviceorientierten und komponentenbasierten Systemen werden funktionale Einheiten miteinander verknüpft um gemeinsam ein Gesamtsystem zu bilden. Die einzelnen Module kommunizieren dabei über Schnittstellen. Required Interfaces beschreiben benötigtes Verhalten, während Provided Interfaces angebotenes Verhalten beschreiben. Es gibt verschiedene Level von Schnittstellen, von denen typischerweise nur der Unterste erreicht wird. Zu diesen Leveln gehört die formale Angabe von Informationen über Aufrufsequenzen und Parallelität. In dieser Arbeit soll zunächst untersucht werden, welche Techniken bekannt sind, um diese Synchronization Level Interfaces zu definieren. In einem weiteren Schritt soll untersucht werden, wie diese Verträge von Required und Provided Interfaces verglichen werden können, auch über Modelltypgrenzen hinweg.
- Erforderliche Vorkenntnisse: Java, komponentenorientierte Softwareentwicklung, ggf. servicebasierte Softwareentwicklung, Modellierung

Entwicklung eines Editors zum Bearbeiten von eingebetteten erweiterten Zustandsautomaten

- Ansprechpartner: Marco Müller
- Art der Arbeit: Projektseminar
- Beschreibung: In serviceorientierten und komponentenbasierten Systemen werden funktionale Einheiten miteinander verknüpft um gemeinsam ein Gesamtsystem zu bilden. Die einzelnen Module kommunizieren dabei über Schnittstellen. Required Interfaces beschreiben benötigtes Verhalten, während Provided Interfaces angebotenes Verhalten beschreiben. Es gibt verschiedene Level von Schnittstellen, von denen typischerweise nur der Unterste erreicht wird. Zu diesen Leveln gehört die formale Angabe von Informationen über Aufrufsequenzen. Eine Möglichkeit der Beschreibung von sog. Sequential Contracts sind Schnittstellen-Automaten. Im Rahmen dieses Projektseminars soll ein grafischer Editor entwickelt werden, mit dem Schnittstellen-Automaten erstellt und bearbeitet werden können, und es ermöglicht, die Interaktion von Schnittstellen-Automaten zu verifizieren.
- Erforderliche Vorkenntnisse: Java, Modellierung, grafische Oberflächen in Java

Themenbereich „E-Learning und E-Assessment“

Methoden zur Erkennung von Code-Plagiaten

- Ansprechpartner: Michael Striewe
- Art der Arbeit: Hauptseminar
- Beschreibung: Es existieren verschiedene Ansätze, um Programmcode auf Gleichheit zu prüfen. Im Rahmen von automatischen Prüfungssystemen kommen diese zum Einsatz, um Code-Plagiate zu entdecken und die Studierenden so zu ermuntern, Aufgaben selber zu lösen, anstatt Lösungen abzuschreiben. In diesem Hauptseminar sollen existierende Ansätze aus der Fachliteratur recherchiert und mit einander verglichen werden. Dazu sollen aus dem Einsatzzweck relevante Vergleichskriterien abgeleitet werden.
- Erforderliche Vorkenntnisse: Java

Statistische Methoden zur Analyse von Lösungen für Programmieraufgaben

- Ansprechpartner: Michael Striewe
- Art der Arbeit: Bachelor-Abschlussarbeit
- Beschreibung: In der Lehrveranstaltung "Programmierung" werden von den Studierenden zu manchen Übungsaufgaben mehr als 1.000 Lösungen eingereicht. Diese lassen sich einzeln automatisiert bewerten und durch Softwareproduktmetriken mit Kennzahlen versehen. Ziel der Arbeit soll es sein, statistische Verfahren zu recherchieren und für den konkreten Anwendungsfall anzupassen, mit denen eine fundierte Analyse der Lösungsmenge möglich ist. Auf diese Weise sollen beispielsweise besonders häufige, aber falsche Lösungsansätze identifiziert werden.
- Erforderliche Vorkenntnisse: Java

Metriken für UML-Diagramme

- Ansprechpartner: Michael Striewe
- Art der Arbeit: Bachelor-Abschlussarbeit
- Beschreibung: In der Softwaretechnik wurden zahlreiche Metriken entwickelt, um die Komplexität und Größe eines Programms durch Kennzahlen zum Programmcode zu messen. Ähnliche Techniken können jedoch auch auf Modelle angewandt werden, sind aber weniger verbreitet. Bei der automatischen Bewertung von Modellierungsaufgaben könnten solche Metriken mit zwei Zielsetzungen eingesetzt werden: Zum einen, um Lösungen danach zu beurteilen, ob sie unnötig groß, unnötig komplex oder vielleicht auch kompakter als eine Musterlösung sind. Zum anderen, um verschiedene Aufgabenvarianten durch Vergleich ihrer Musterlösungen auf Gleichwertigkeit zu überprüfen. In dieser Bachelorarbeit sollen daher verschiedene Metriken aus der Literatur recherchiert und auf reale Lösungen von Modellierungsaufgaben angewandt werden. Wo keine passenden Metriken in der Literatur gefunden werden, sollen neue geeignete Metriken entworfen werden. Als Endergebnis sollen geeignete Metriken beschrieben und als lauffähige Komponente für ein Prüfungssystem bereitgestellt werden.
- Erforderliche Vorkenntnisse: Java, UML

Automatische Prüfung von Aufgaben zum Programmverständnis

- Ansprechpartner: Michael Striewe
- Art der Arbeit: Projektseminar
- Beschreibung: Zu den grundlegenden Fähigkeiten eines Programmierers gehört nicht nur das Schreiben von Programmcode, sondern auch das Lesen und Verstehen von existierendem Programmcode. Diese Fähigkeit kann beispielsweise dadurch trainiert werden, dass zu kurzen Programmstücken ein vollständiger Trace manuell erzeugt werden soll. In diesem Projektseminar soll daher für ein bestehendes Prüfungssystem eine geeignete Erweiterung der GUI und der Prüffunktionen erstellt werden, mit der solche Aufgaben angezeigt, bearbeitet und geprüft werden können. Insbesondere soll die Prüffunktion in der Lage sein, hilfreiche Hinweise zu geben, wenn die Aufgabe nicht korrekt gelöst wurde.
- Erforderliche Vorkenntnisse: Java

Themenbereich „Analyse von Programmcode“

Design Recovery für Modell-Spezifikationen in Bytecode

- Ansprechpartner: Michael Striewe
- Art der Arbeit:
 - Studienprojekt
- Wenn Programmcode kompiliert wird, geht die Semantik der Programmiersprache zu einem bedeutenden Teil verloren. Jedoch sind zahlreiche Informationen vorhanden, die Rückschlüsse auf das ursprüngliche Programm zulassen. Das Forschungsgebiet der „Design Recovery“ beschäftigt sich mit der Nutzung solcher Informationen, um aus kompiliertem Code das ursprüngliche Programm und die dahinterliegende Spezifikation wiederherzustellen. Das ist relevant für den Fall, dass Softwaresysteme lange Zeit im Einsatz sind und Dokumentation und Quellcode nicht mehr vorhanden bzw. aktuell sind. In dieser Arbeit soll für Design Recovery mit Java Bytecode gearbeitet werden, der aus Programmen entstanden ist, die nach bestimmten Pattern basierend auf Modellen entwickelt wurden (z.B. Zustandsautomaten, Prozessmodelle). Es soll versucht werden, in einem ersten Schritt bekannte Patterns vollständig zu rekonstruieren; die Vorgehensweise hierfür ist bereits größtenteils spezifiziert und soll prototypisch implementiert werden. Im zweiten Schritt soll aufbauend hierauf ein Design-Recovery-Werkzeug entwickelt werden, das auch potentiell unbekanntem Code untersucht und möglicherweise vorhandene Modelle soweit wie möglich rekonstruiert.
- Erforderliche Vorkenntnisse: Java