Formale und semiformale Methoden für eingebettete Software

Inhalt

Der Begriff “formale Methoden” fasst eine Vielzahl von **Techniken zur mathematischen Modellierung und Verifikation** von Computersystemen zusammen: formale Spezifikationssprachen (z.B. ‘Z’), Automaten, Temporallogik, Prozesskalküle, Model-Checking usw. Sie werden in der Softwaretechnik und in der industriellen Praxis in sicherheitskritischen Bereichen angewendet um Fehlerfreiheit zu beweisen oder plausibel zu machen, um Betriebssicherheit zu gewährleisten und um technische Systeme auf die Erfüllung von Anforderungen zu überprüfen.

Sowohl formale als auch semiformale Methoden haben eine strikt definierte **Syntax und Semantik** für Sprachelemente. Semiformale Modelle eignen sich typischerweise zur unzweideutigen Kommunikation von Systemanforderungen und -eigenschaften, darüber hinaus auch zu einfachen Modellanalysen, z.B. einer Duplikatserkennung.

Formalen Methoden liegt zusätzlich ein **Kalkül** zugrunde, darin unterscheiden sie sich von semiformalen Ansätzen und gängigen Programmiersprachen. Man kann mit formalen Modellen also “rechnen”: Man kann sie ineinander und in kanonische Darstellungen transformieren, sie so vergleichen und weitergehende Eigenschaften, auch Systemeigenschaften, analysieren. Die zugrunde liegende Annahme ist: besitzt das Modell eine Eigenschaft, dann besitzt das modellierte System sie auch.

Ist die Erstellung eines korrekten formalen Modells weniger aufwendig als die fehlerfreie Beschreibung des gewünschten Verhaltens in einer Programmiersprache, so ermöglicht die formale Spezifikation u.U. nicht nur eine **frühzeitige Analyse** des geplanten Systems, sondern ist auch kosteneffizient. Der Einsatz formaler Methoden in der industriellen Software-Entwicklung steht jedoch noch am Anfang.

Voraussetzungen

- In diesem Seminar sind Bachelor- und Masterstudierende zugelassen. Für Teilnehmer des Bachelor-Studiengangs ist das Proseminar Voraussetzung.
- Ggf. ist Vorwissen für die Bearbeitung einzelner Themen von Vorteil.
- **Bitte geben Sie relevantes Vorwissen bei Ihrer Anmeldung mit an, um Ihre Chance auf Zuteilung zu erhöhen.**

Themen

- Pushing to the Top
- Reachability Modulo Theories
- Self-Adaptive Software Needs Quantitative Verification at Runtime
- Regression Verification: Proving the Equivalence of Similar Programs
- Contracts-refinement proof system for component-based embedded systems

Die o.g. Aufsätze sind Beispiele aus vorangegangenen Semestern, um einen Eindruck von den Themen zu geben, die üblicherweise in diesem Seminar-Track behandelt werden.

From:
https://embedded.rwth-aachen.de/ - **Informatik 11 - Embedded Software**

Permanent link:
https://embedded.rwth-aachen.de/doku.php?id=lehre:sose18:formal

Last update: **2018/07/15 21:02**