

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.

Themen

Die hier genannten Themen sind Beispiele und zeigen die Richtung der verfügbaren Themen. An der Auswahl der Papiere kann sich bis zur Einführungsveranstaltung noch ändern. Insgesamt werden 10 Themen und damit 10 Plätze zur Verfügung stehen.

- [Synthesis of Loop-free Programs](#)
- [Synthesizing Safe Bit-Precise Invariants](#)
- [Software Model Checking via IC3](#)
- Static Analysis Tools as Early Indicators of Pre-Release Defect Density

From:

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

Permanent link:

<https://embedded.rwth-aachen.de/doku.php?id=lehre:wise1617:formal>

Last update: **2016/06/22 11:16**

