Summer School Marktoberdorf (1970-2010)
Software and Systems Safety: Specification and Verification

Muhammad Taimoor Khan

Doktoratskolleg Computational Mathematics
Johannes Kepler University
Linz, Austria

October 20, 2010

Outline

- Introduction
- Organization
- Lectures
- Tutorials
- Model-Driven Development of Reliable Services

Introduction

- History
  - NATO Software Engineering Conference in Germany (1968)
  - Tony Hoare and E.W. Dijkistra
- Introduction
  - For two weeks (August 3-15, 2010)
  - Academic Activities
  - Entertainment

Model-Driven Development of Reliable Services by *Manfred Broy*

- detail on coming slides.

Unifying Models of Data Flow by *Tony Hoare*

▶

Model Checking by *Doron Pelad*

- Modeling of software and hardware systems
- Software specification using temporal logic and Buchi Automata
- Translation between logic and automata
- Model Checking Algorithms
- How to make it work in practice: abstraction/reduction/BDDs

Issues of Adaptable Software for Open-World Requirements by *Carlo Ghezzi*

- Specifications and service level agreements among different stakeholders and subsystems
- Functional and non-functional qualities
- Architecture: how do the requirements for dynamic adaptation aspect software composition
- Language support to dynamic adaptation
- Modelling and analysis: development time requirements vs runtime requirements

Requirements Models for System Safety and Security by *Connie Heitmeyer*

- Modeling and formal specification of requirements
- Consistency and completeness checking of requirements
- Simulation of requirements to check their validity
- Generating invariants from requirements specifications
- Formal verification of requirements
- Testing and automatic code generation based on an operational requirements model
- Modeling and analyzing systems for critical properties (e.g. security and fault-tolerance)

Formal Methods and Argument-based Safety Cases by *John Rushby*

-

Abstraction for System Verification by *Susanne Graf*

- Appropriate abstraction is the key for successful verification of programs/systems
- General verification is of high complexity task (state explosion)
- General framework for abstraction
- Using abstractions to (meaningfully) reason about large composed systems
- General contract framework to prove stronger properties
- Proving properties with top-down design constraints and bottom-up abstractions

Model-based Testing by *Ed Brinksma*

- Model-based testing (terminology and concepts)
- Derivation of functional tests from models in the form of input/output transition systems
- Theory and tools can be extended to deal with real-time behaviour in specifications, implementations and tests
- Test selection and coverage

From Concurrency Models to Numbers: Performance, Dependability, Energy by *Holger Hermanns*

-

Formal Verification by *John Harrison*

▶

Model-based Verification and Analysis for Real-Time Systems by *Kim Larsen*

-