SEMINAR »VERIFICATION AND TESTING OF COMPLEX SYSTEMS«

Dr. Günter Kniesel – Uni Bonn

Dr. Michael Gerz – Fraunhofer FKIE

Kickoff Meeting

October 17th, 2016

Günter Kniesel

Institut für Informatik III

Römerstraße 164

53117 Bonn

0228-73-4511

gk@cs.uni-bonn.de

- Software Analysis and Transformation
- Software Design and Architecture
- Model-Driven Software Development
- Domain Specific Langauges
- Agile Software Engineering Methodologies

Michael Gerz

Fraunhofer FKIE

Fraunhoferstraße 20

53343 Wachtberg (south of Bonn)

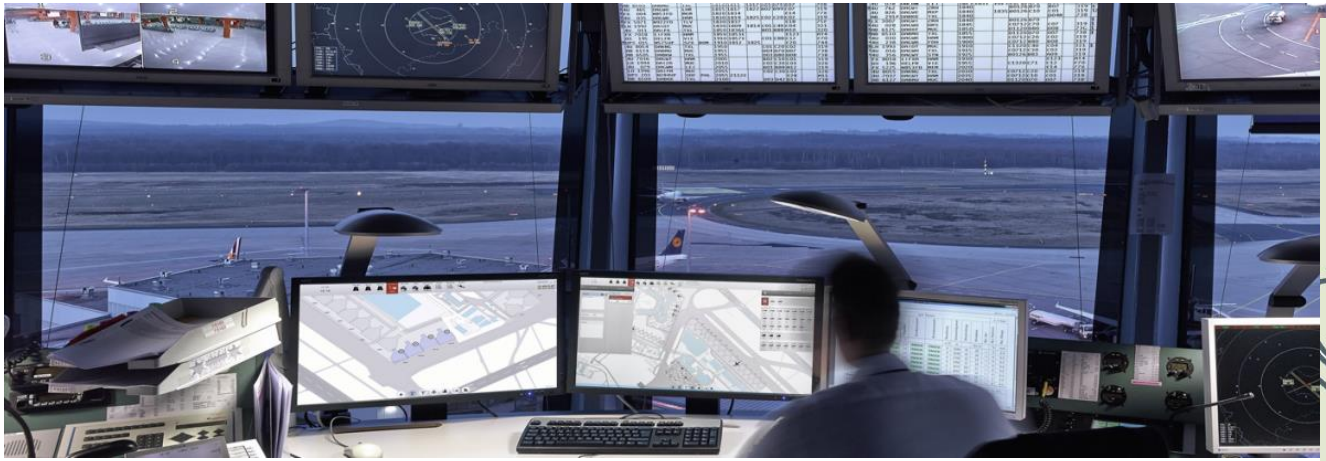
0228-9435-414

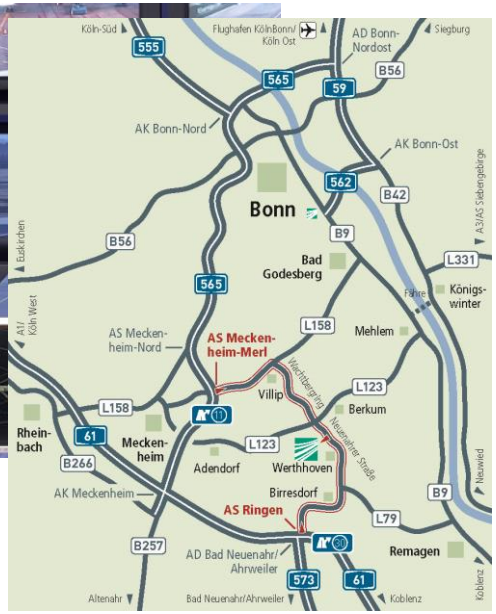
michael.gerz@fkie.fraunhofer.de

- 1990 - 1996: Study of Computer Science, University of Koblenz-Landau, Enhanced courses in computational linguistics
- 10/1996 - 09/2001: Research Assistant at the Institute for Telematics, Medical University of Lübeck
- 10/2001 - 01/2003: Research Assistant at the Institute for Telematics e.V., Trier
- 01/2003 - 10/2003: Research Assistant at the Chair of Prof. Meinel, University of Trier
- 04/2003: Dissertation at the University of Göttingen
- since 01/2004: Head of Group „Interoperability & Testing“ Department on Information Technology for Command and Control, Fraunhofer FKIE

Fraunhofer FKIE

Fraunhofer Institute for Communication, Information Processing and Ergonomics FKIE





- Applied research on all aspects of Defense and Security
  - Reconnaissance, communication networks, command & control, cyber defense
- »We are working on making our world safer. Our goal is to detect, minimize, and manage life-threatening risks.«
- Visit http://www.fkie.fraunhofer.de

Fraunhofer FKIE

You (?)

- Students of Master Programme in Computer Science
- … with (some) focus on Inform. & Communication Management track
- Max. 8 participants

Background knowledge:

- Software construction, programming
- Logic, mathematical formalisms

Aim:

- Seminar with grade in fulfillment of module requirements

Scope of the seminar

- Techniques for analyzing the correctness of complex (software) systems
- Theoretical foundations for such techniques
- Application of practical tools

Aims of Seminar, Capabilities

- Study scientific literature
  - Information gathering, search and comprehension
  - Retrieve further literature for additional/background information
  - ➡ Stay in touch with supervisor
- Write a scientific article (in English)
  - Summarize the content of the provided papers
  - Provide own explanations and examples
  - ➡ Write seminar paper (12–15 pages)
- Give a scientific talk (in English)
  - Present findings on the assigned topic, based on paper
  - ➡ Give seminar talk (about 40 minutes), answer questions
- Discuss scientific subjects
  - ➡ *Active* participation in moderated group discussions

Seminar Organization

- Assignment of topics: this week
- Self-study of literature: as soon as possible (next two to three weeks)
- Talks to be held new year (probably two per meeting)
- Preparation phase (dates to be adjusted for holidays):
  - 6 – 8 weeks before talk: First meeting with supervisor (might be at FKIE)
  - at least 3 weeks b.t.: Submission of draft paper
  - at least 10 days b.t.: Distribution of final paper to all participants
  - at least 1 week b.t.: Discussion of presentation with one of us
  - At any time: Discussions on demand

A Few (Not Completely Random) Remarks

- Agree with supervisor on what should go into essay and talk
- Don’t be shy asking questions (during preparation & discussions)
- Take notes during preparation meetings
- Discuss the structure of paper and presentation in advance
- Use your own words, examples, explanations
- Provide a motivation, don’t just array fact after fact.
- Use notation and terminology consistently (in particular when using different sources)
- Use a spell-checker (please!)
- Adhere to deadlines! When submitting a final version, make sure it is really final
- The main addressees of your presentations are your fellow students.

Introduction To
Basic Concepts

Definitions

- Verification
  - "The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase." [IEEE Standard 610]
  - "Are we building the product right?"
- Validation
  - "The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements." [IEEE Standard 610]
  - "Are we building the right product?"
- Testing
  - "The process of operating a system or component under specified conditions, observing or recording the results and making an evaluation of some aspects of the system or component." [IEEE Standard 610]
  - Test can unveil the existence of errors but cannot prove their absence!

Verification – Non-Formal Approach

- Detection of specification violations by means of testing
  - Also known as *manual* verification
  - Most common in practice
- Pros
  - Suitable for complex / huge systems
  - No specific training / education in formal methods required
  - Less expensive than formal approach (time & costs)
- Cons
  - Absence of bugs (“correctness”) cannot be proven
  - Less amenable to automation

Verification – Formal Approach

- Methods
  - Model Checking
  - Theorem Proving
- Pros
  - Supports correctness proofs (i.e., that certain properties DO hold)
  - Method(s) of choice for highly reliable systems (Ariane 5 crash)
  - High degree of automation
  - Shows limits of automation (→ scientific insight)
- Cons
  - Requires specific training / education
  - Tricky when applied to complex / huge systems
  - In practice (much) more expensive than testing

Model Checking

- Given
  1. Finite state machine M (→ abstract model of the system under test)
  2. Temporal formula F (→ specification of a desired property)
- *Model Checking Verification Problem*: Show that M semantically entails F:

  $M \models F$ ("M is a model of F")
- Algorithmic approach: Explore the set of reachable states of M to ensure that F holds
- Termination requires that the set of reachable states is finite

Theorem Proving (1)

- Expressions of the form

$$\{F_{pre}\}\ P\ \{F_{post}\} \qquad \text{(Hoare calculus)}$$

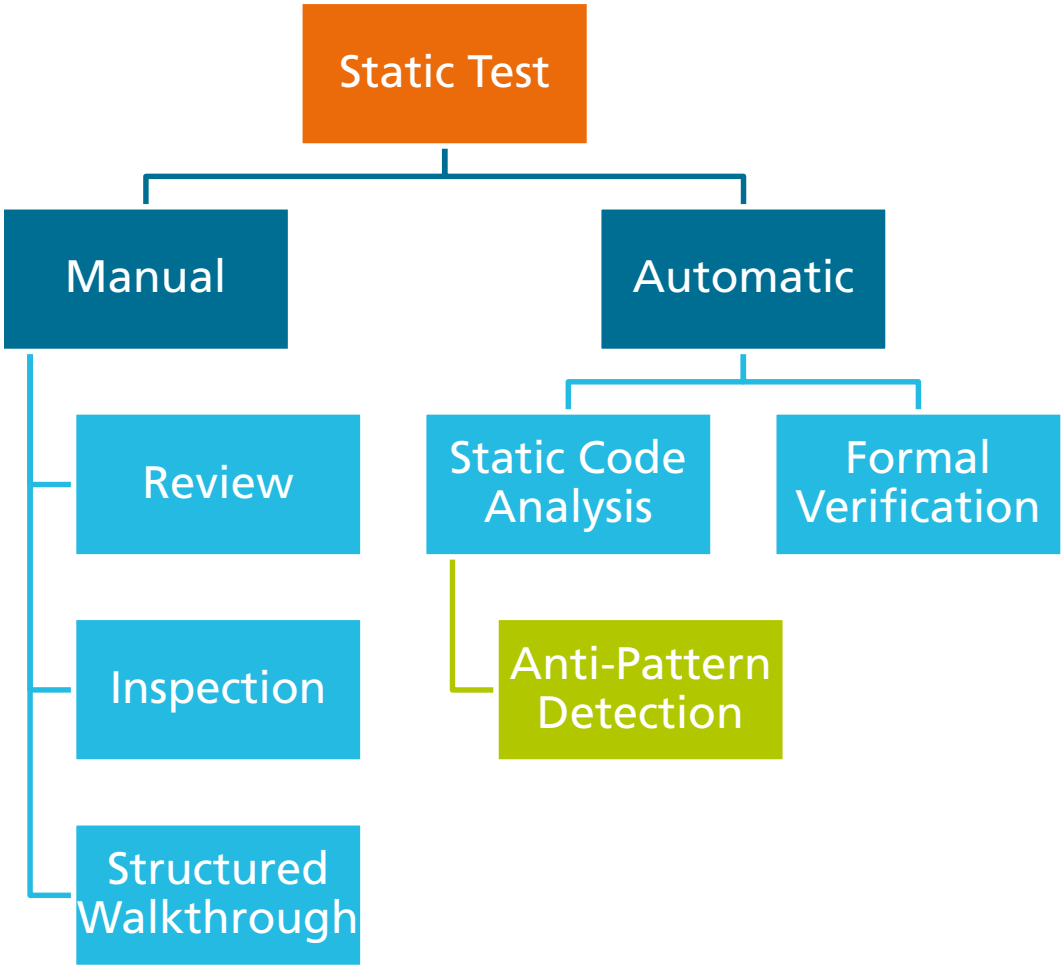
  - meaning "if property F_{pre} holds before program P starts, F_{post} holds after the execution of P"
  - P can refer to an entire program or to an atomic action, depending on the unit to be verified
- Calculus consists of *axioms* and *inference rules* which are used to derive F_{post} based on F_{pre} and P
- Example inference rule:

$$\frac{\{F_{pre}\}\ P\ \{F_{inter}\},\quad \{F_{inter}\}\ P'\ \{F_{post}\}}{\{F_{pre}\}\ P;\ P'\ \{F_{post}\}}$$

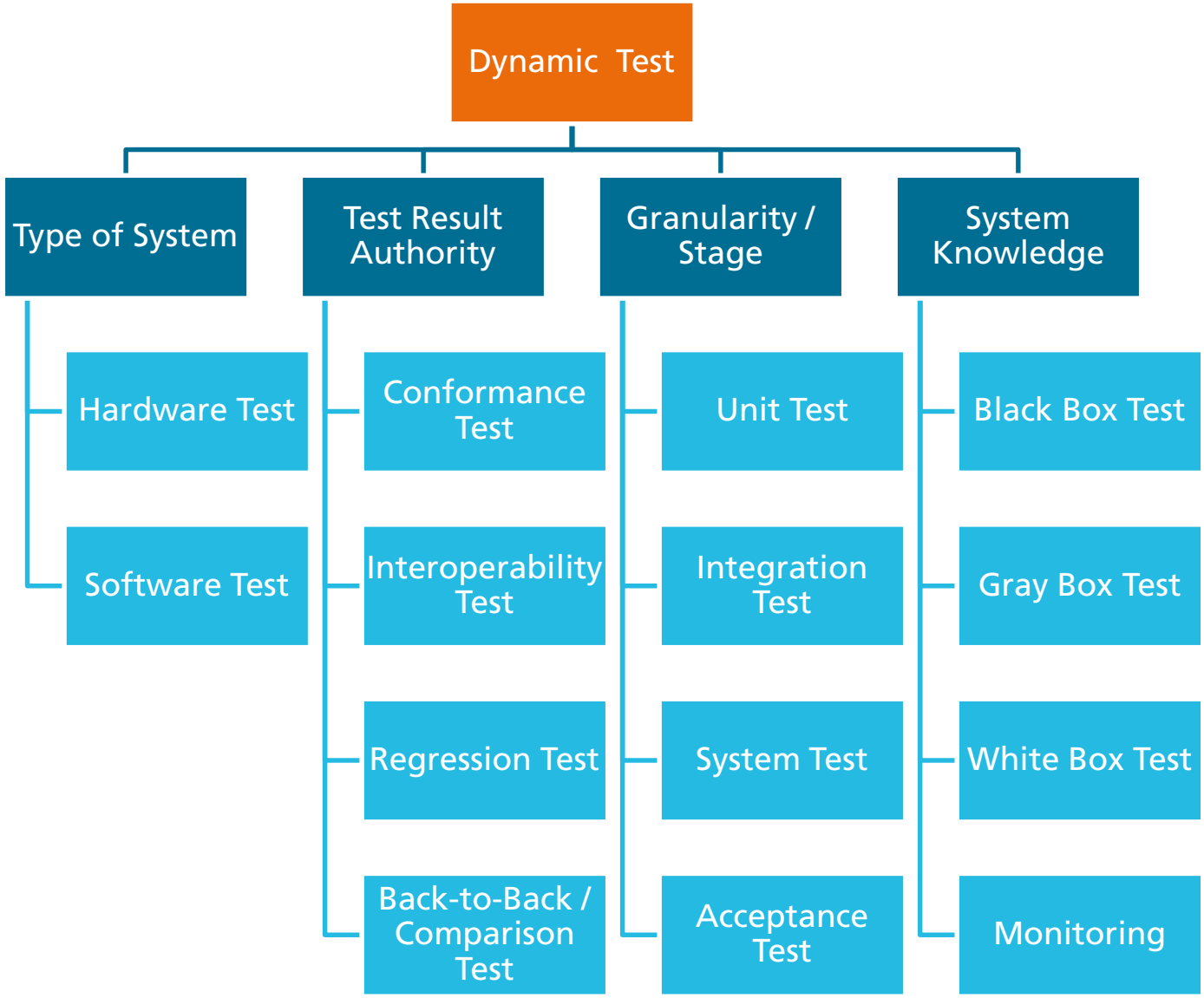
Theorem Proving (2)

- Most theorem provers used to prove program properties are based on variations of Hoare calculus
- Key difference between theorem proving approach and model checking approach (to software verification):
  - Theorem provers do not need to exhaustively visit the program's state space to verify properties
  - … can reason about infinite state spaces (e.g., state spaces involving complex datatypes and recursion)
- Drawbacks:
  - Proof system for a system of practical size can be extremely large
  - Generated proofs can be large and difficult to understand
  - Require a great deal of user expertise and effort

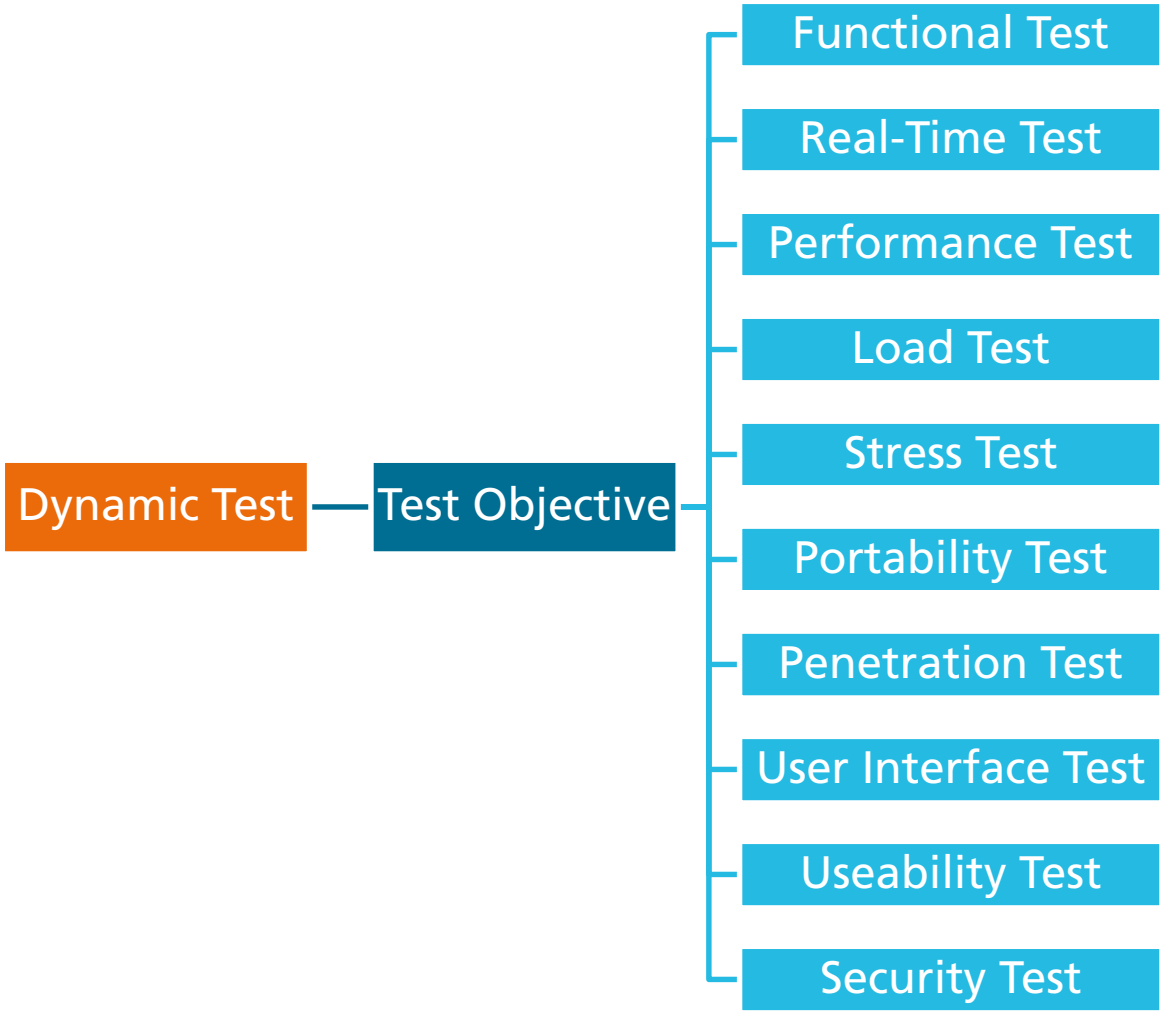
Classification of Test Methods (1)



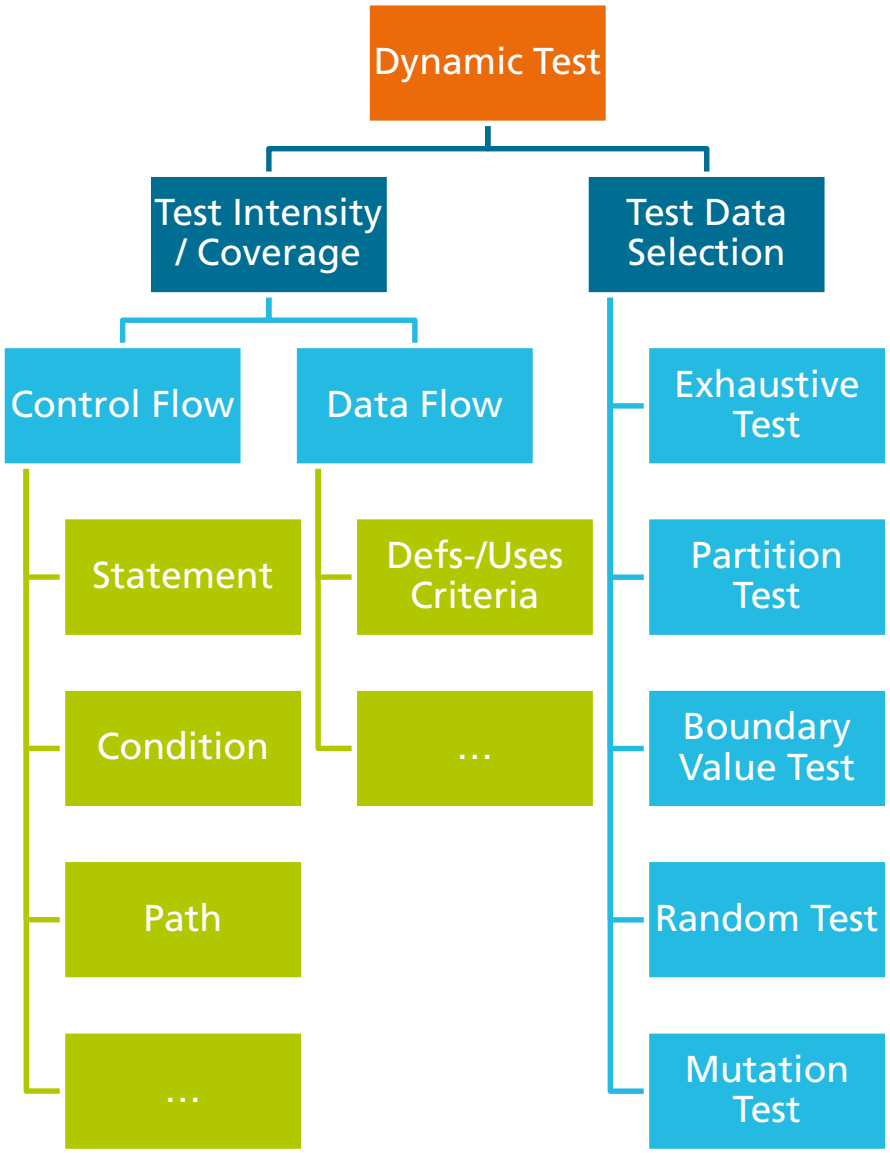
Classification of Test Methods (2)



Classification of Test Methods (3)



Classification of Test Methods (4)



Selected Topics

Topic 1: Testing and Test Control Notation 3 (TTCN-3)

- Subject
  - Standardized testing language
  - European Telecommunication Standards Institute (ETSI)
  - Supports automated and distributed testing
- Objectives
  - Present the key concepts of the test language
  - Develop sample test suite in TTCN-3
    - Use case: Bank withdrawal, account balance request
  - Analyze tool support (Eclipse Titan)
- References
  - http://www.ttcn-3.org
  - https://projects.eclipse.org/projects/tools.titan

Topic 2: Jnario – Executable Specifications for Java

- Subject
  - Testing, specification, and documentation framework
  - Executable unit, integration, and acceptance specifications
  - Orchestration of specifications
  - Developed at BMW Car IT
- Objectives
  - Present the key concepts of Jnario
  - Explain how Domain Specific Languages (DSLs) can be defined with Xbase/Xtext
  - Live demonstration based on self-defined use case
- References
  - http://jnario.org/
  - https://github.com/sebastianbenz/Jnario/issues/168
  - https://github.com/borisbrodski/Jnario/tree/no_xtend_xtext2.9

Topic 3: SAT-Based Formal Verification

- Subject
  - SAT-based formal verification (propositional logic)
  - How to use SAT solvers for model checking
  - Formulating a verification problem as a SAT problem
- Prerequisite:
  - Knowledge of math. logic & complexity theory
- Objectives
  - Understand, present & explain new results
- References
  - M. Prasad, A. Biere, and A. Gupta. A survey of recent advances in SAT-based formal verification. Int J Softw Tools Technol Transfer (2005) 7: 156-173.

Topic 4: Continuous Integration & Automated Testing

- Subject
  - Continuous integration - integrate code early and often into code mainline
  - Automated regression testing
- Objectives
  - Explain how testing can be coupled with continuous integration
- References
  - M. Goedicke, T. Menzies, S. Motoshi. Communicating continuous integration servers for increasing effectiveness of automated testing. In: Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering (ASE 2012), ACM New York, USA, 2012, pages 374-377.
  - S. Stolberg. Enabling Agile Testing through Continuous Integration. In: Proceedings of Agile Conference, 2009 (AGILE '09). Pages 369-374.
  - R. A. Razak, F. R. Fahrurazi: Agile testing with Selenium. In: Proceedings of the 2011 Malaysian Conference in Software Engineering, IEEE, 2011, pages 217-219.

Topic 5: Testing Grid and Cloud Infrastructures

- Subject
  - Testing of grid and cloud infrastructures
- Objectives
  - Present the specific requirements for testing cloud/grid computing environments
  - Present work on standardized test framework
- References
  - T. Rings, J. Grabowski, S. Schulz. On the Standardization of a Testing Framework for Application Deployment on Grid and Cloud Infrastructures. Proceedings of the 2nd International Conference on Advances in System Testing and Validation Lifecycle (VALID), 2010.
  - K. Inçki, I. Ari, H. Sözer: A Survey of Software Testing in the Cloud. IEEE Sixth International Conference on Software Security and Reliability Companion (SERE-C), 2012.
  - St. Herbold, A. De Francesco, J. Grabowski, et al.: The MIDAS Cloud Platform for Testing SOA Applications. IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), 2015.

Topic 6: Test Generation Based on Finite State Machines

- Subject
  - Finite State Machines to model specifications
  - Various methods to derive test cases
    - addressing fault models
    - minimizing the size of test cases
- Objectives
  - Describe the underlying assumptions and constraints
  - Present different approaches to generate tests based on FSMs
  - Provide examples
- References
  - H. Ural: Formal methods for test sequence generation. Computer Communications. Volume 15, Issue 5, June 1992, Pages 311 – 325, http://people.cs.aau.dk/~kgl/TOV04/ural.pdf
  - M. C. Yalcin, H. Yenigun: Using Distinguishing and UIO Sequences Together in a Checking Sequence. Proceedings of Testing of Communicating Systems. Lecture Notes in Computer Science Volume 3964, 2006, pp 259-273

Topic 7: Model-Based Testing With Spec Explorer

- Subject
  - Testing environment developed by Microsoft Research
  - Reactive Systems = Non-deterministic
- Objectives
  - Understand & present key ideas
- References
  - M. Veanes, C. Campbell, W. Grieskamp, W. Schulte, N. Tillmann, and L. Nachmanson. Model-Based Testing of Object-Oriented Reactive Systems with Spec Explorer. Lecture Notes in Computer Science, 2008, volume 4949, 39-76.

Topic 8: Lazy Systematic Unit Testing

- Subject
  - Semi-automatic unit test generation
  - “Tests for full conformance to a lazy specification, which is inferred on-the-fly from the code, by static and dynamic analysis, and from hints supplied by the programmer“
- Objectives
  - Understand & present approach
  - Showcase »JWalk«
- References
  - Journal and conference publications at http://staffwww.dcs.shef.ac.uk/people/A.Simons/jwalk/
- Note:
  - Request academic license from author in time!

Topic 9: Mobile Testing as a Service (MTaaS)

- Subject
  - Validation of mobile Apps and SaaS applications on mobile web
  - High complexity due to diversity of mobile devices and computational resources
- Objectives
  - Provide a survey of the challenges, technologies, approaches, and infrastructures
- References
  - A. Malini; N. Venkatesh; K. Sundarakantham; S. Mercyshalinie: Mobile application testing on smart devices using MTAAS framework in cloud. International Conference on Computer and Communications Technologies (ICCCT), 2014
  - J. Gao, W.-T. Tsai, R. Paul, X. Bai, T. Uehara: Mobile Testing-as-a-Service (MTaaS) -- Infrastructures, Issues, Solutions and Needs. IEEE 15th International Symposium on High-Assurance Systems Engineering, 2014.
  - I. K. Villanes, E. A. Bezerra Costa, A. C. Dias-Neto: Automated Mobile Testing as a Service (AM-TaaS). IEEE World Congress on Services. 2015

Topic 10: Regression Test Selection

- Subject
  - Regression testing as an expensive maintenance activity
  - Test selection based on control flow graphs for a (modified) program
- Objectives
  - Present the concepts and provide examples
- References:
  - G. Rothermel, M. J. Harrold: A Safe, Efficient Regression Test Selection Technique, ACM Transactions on Software Engineering and Methodology, Vol. 6, No. 2, 1997.

Topic 11: Practical Application of Model Checking

- Subject
  - Use model checking to find serious errors in file systems
  - Find corner-case errors by exploring the system's state space
  - File systems as a use case:
    - Errors are most serious
    - Hard to test whether a systems recovers after any crash
- Objectives
  - Introduce basic concepts of model checking
  - Describe the techniques and tools applied
  - Summarize the findings of the use case
- References
  - J. Yang, P. Twohey and D. Engler, M. Musuvathi: Using Model Checking to Find Serious File System Errors. ACM Transactions on Computer Systems, November 2006, vol. 24, no. 4. (https://web.stanford.edu/~engler/osdi04-fisc.pdf)

Way ahead

Reminder: What we expect from you

- Information gathering
  - Summarize the content of the provided papers
  - Retrieve further literature for additional/background information
- A 12–15 pages summary of the topic (in English)
- A 40 minutes presentation of the findings (in English)
  - Followed by a moderated group discussion
- *Active* participation in group discussions (and *regular* attendance)

Topic Assignment

Topic #	Student
1 – Testing and Test Control Notation 3 (TTCN-3)	
2 – Jnario – Executable Specifications for Java	
3 – SAT-Based Formal Verification	
4 – Continuous Integration & Automated Testing	
5 – Testing Grid and Cloud Infrastructures	
6 – Test Generation Based on Finite State Machines	
7 – Model-Based Testing With Spec Explorer	
8 – Lazy Systematic Unit Testing	
9 – Mobile Testing as a Service (MTaaS)	
10 – Regression Test Selection	
11 – Practical Application of Model Checking	

Schedule

- Meetings take place
  - on Monday, 12.15 – 13.45,
  - Römerstraße 164, room A121
- Kickoff Meeting (today)
  - Assignment of topics
- Preparation phase
  - 24-Oct, 31-Oct, 07-Nov, 14-Nov, 21-Nov, 28-Nov, 05-Dec, 12-Dec, 19-Dec
  - Meetings after consultation
- Presentation phase
  - 09-Jan, 16-Jan, 23-Jan, 30-Jan, 06-Feb, 13-Feb
  - Possibly two presentations per session

Red = M. Gerz not available