Software testing tools: from real-world software projects to educational contents

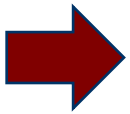
Klaus Bothe

Institute of Informatics, Humboldt University – Berlin, Germany,
bothe@informatik.hu-berlin.de

Workshop

Cooperation at Academic Informatics Education across Balkan Countries and Beyond
***Jelsa, Croatia, 2nd – 6th* September 2019**

Contents



- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

Our problem

Generally accepted:

- Software testing is a key discipline in software engineering
- Sources for educational contents:
  textbooks, recommendations from authorities (e.g. IEEE)
- Software testing without tool support not successful in practice
- Thus, students should become familiar with some of them
  – but which ones?

Which kind of testing tools should
be used in educational environments
based on main issues of software testing?

Our approach

Own experiences should be included.

- Evaluate (own) typical real-world software projects
- Main issues of our real-world software projects determine selection of testing tools

Our solution

There is no unique (best) software testing tool:
it depends on the activity to be supported

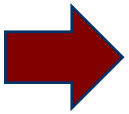
Main idea: testing tools for different needs:

-> Regression testing

-> Selection of test cases

-> Check the completeness of test cases

Contents



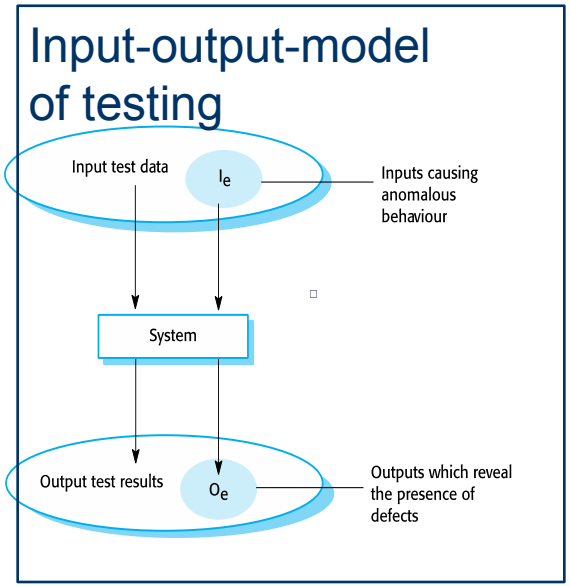
- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

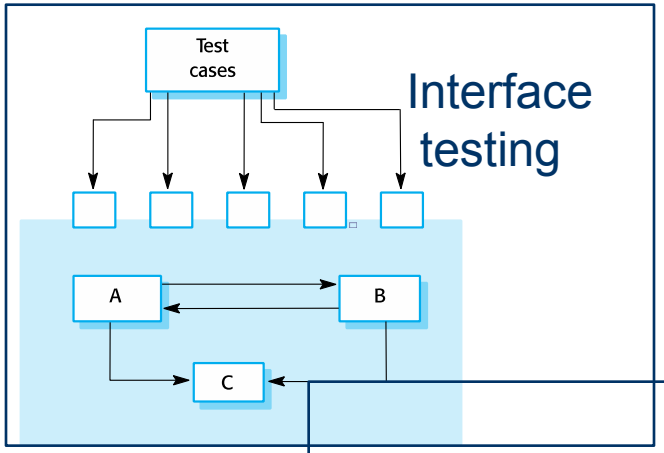
Textbooks

on general Software Engineering

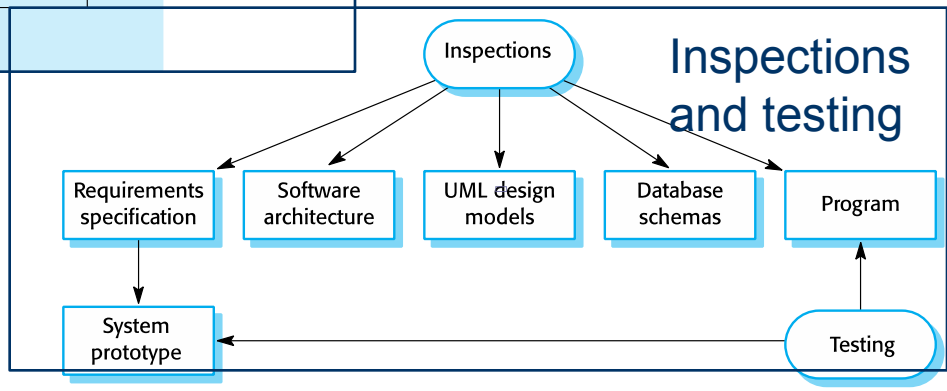
Example: Sommerville, Software Engineering, 10th Edition, 2016

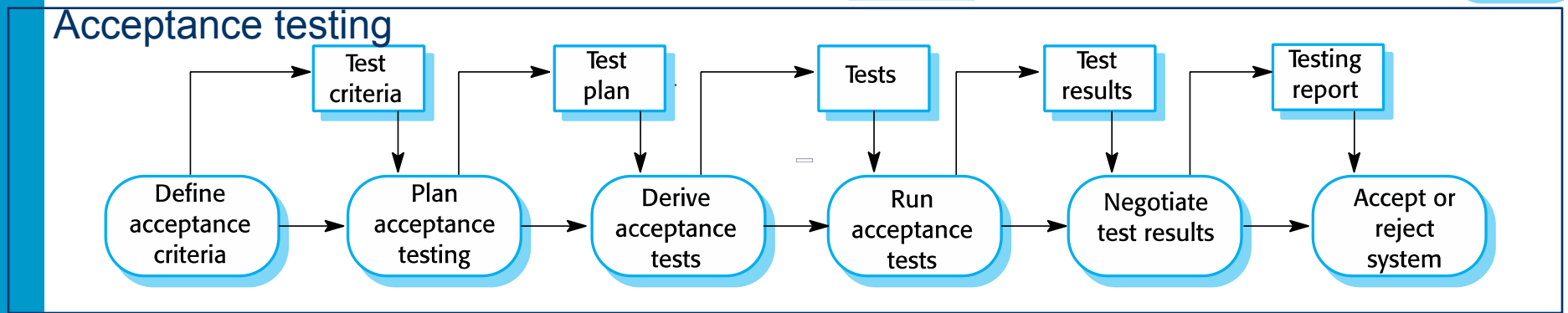
Educational contents in software testing: the case of Sommerville





Good overview of all kinds of testing problems (knowledge on general abstract level)





Textbooks

on Software Testing

Google

introduction to software testing software testing techniques

Anmelden

Alle | Bilder | Videos | News | Shopping | Mehr | Einstellungen | Tools

Introduction to Software Testing / Wird auch oft gesucht

Introduction to Software Testing / Wird auch oft gesucht

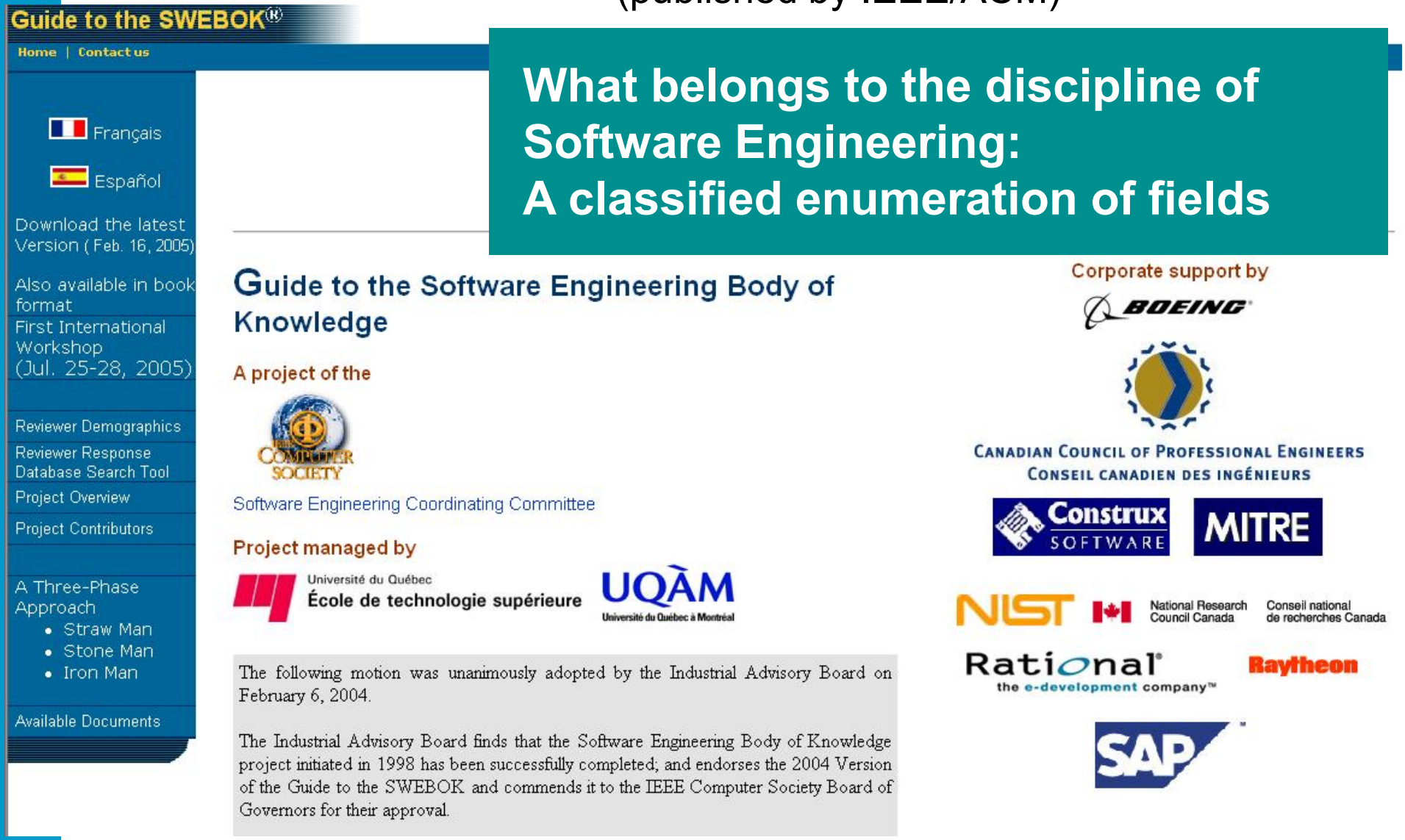
Introduction to Software Testing / Wird auch oft gesucht

Contents

- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

SWEBOK:
Software Engineering Body of Knowledge

(published by IEEE/ACM)



What belongs to the discipline of Software Engineering:
A classified enumeration of fields

http://www.swebok.org/

SWEBOK 2014: Testing knowledge areas

Chapter 4: Software Testing

1. Software Testing Fundamentals

1.1. Testing-Related Terminology
1.2. Key Issues
1.3. Relationship of Testing to Other Activities

2. Test Levels

2.1. The Target of the Test
2.2. Objectives of Testing

3. Test Techniques

3.1. Based on the Software Engineer's Intuition and Experience
3.2. Input Domain-Based Techniques
3.3. Code-Based Techniques
3.4. Fault-Based Techniques
3.5. Usage-Based Techniques
3.6. Model-Based Testing Techniques
3.7. Techniques Based on the Nature of the Application
3.8. Selecting and Combining Techniques

4. Test-Related Measures

4.1. Evaluation of the Program Under Test
4.2. Evaluation of the Tests Performed

5. Test Process

5.1. Practical Considerations
5.2. Test Activities

6. Software Testing Tools

6.1. Testing Tool Support
6.2. Categories of Tools

http://www.swebok.org/

ISTQB®: Certified tester

- → http://www.istqb.org/
- → ISTQB®:
  International Software Testing Qualifications Board
- → Founded in 2002
- → Registered in Belgium
- → Until June 2017:
  740.000 exams with 535.000 certifications in more than 120 countries

A lot of material and recommendations: What to select?

Contents

- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

(Own) Real-world software projects

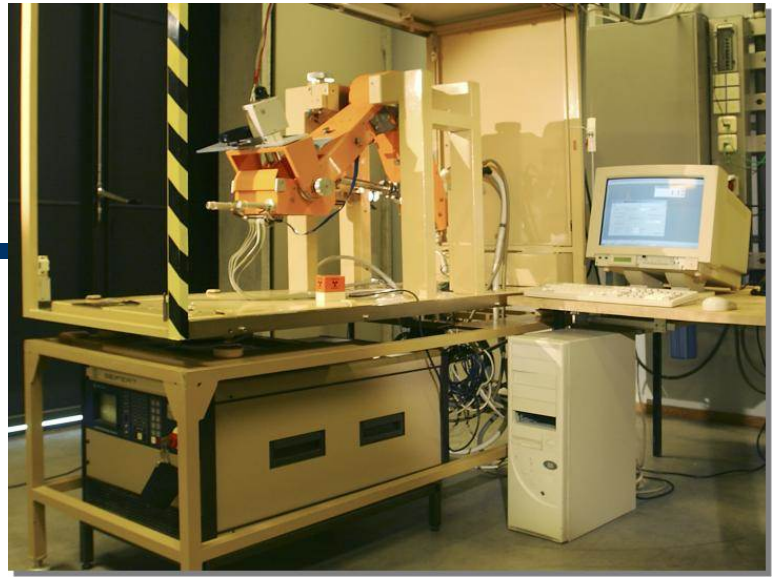
Practical software projects (which are safety-critical)

-> Bachelor and Master thesis with industry
(e.g. with car industry, suppliers to car industry)

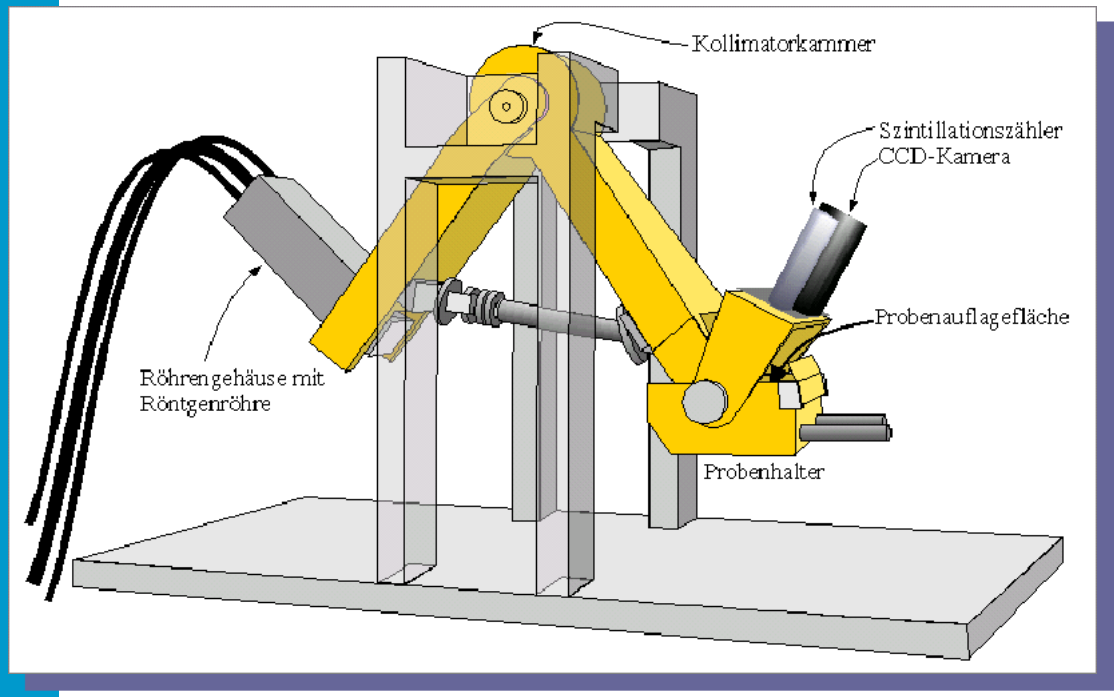
-> ATEO: experiments in psychology

-> XCTL: control a device for experiments in physics
(safety-critical, long-term practical project, 2001-2014,
real customer: Institute of Physics, HU)

Real-life project XCTL (control a device for experiments in physics)

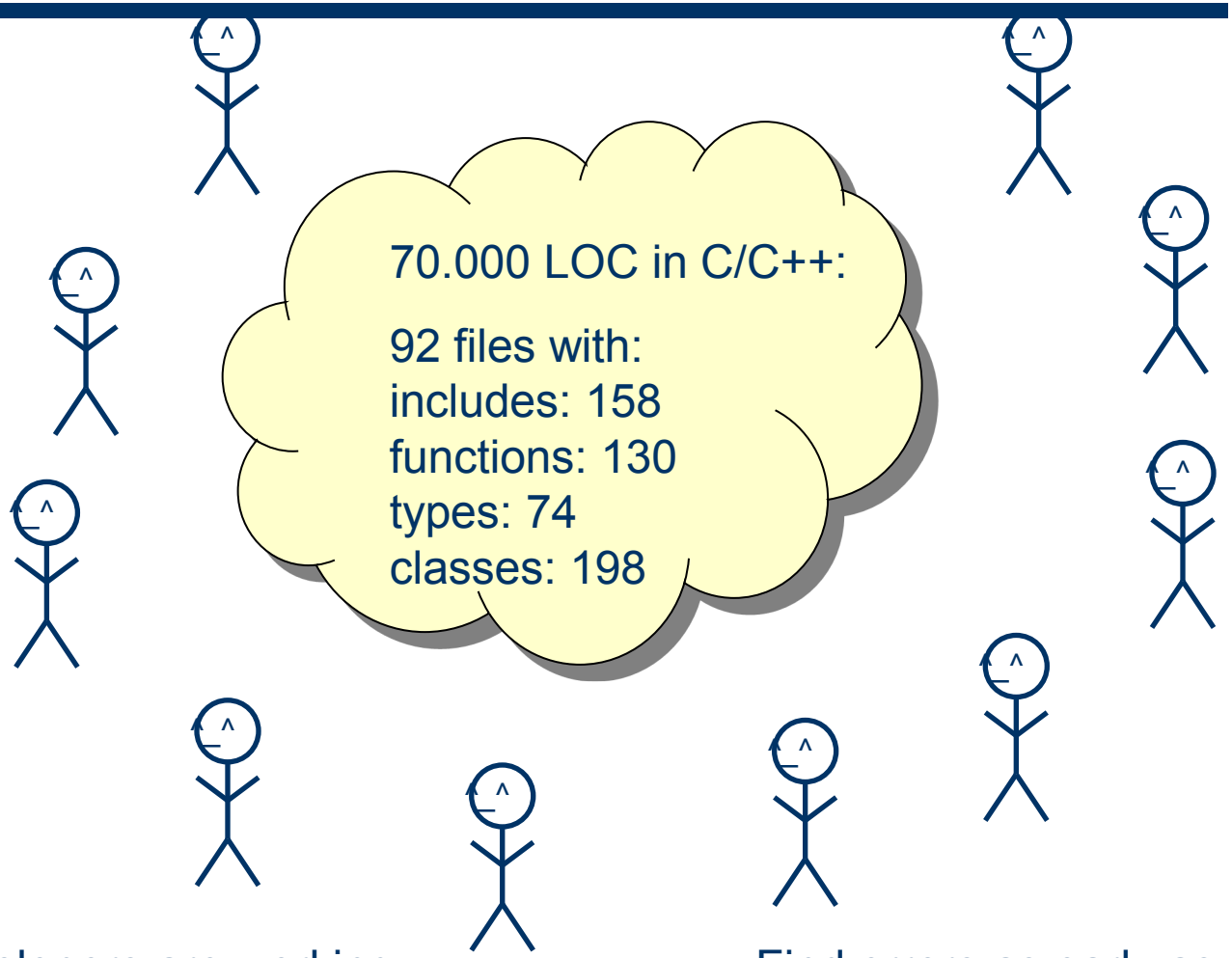


working place



X-ray topography camera

Real-life project XCTL (control a device for experiments in physics)



Many developers are working at different parts of the system

Find errors as early as possible: test very often (but: test takes 2 days)

Main issues of software testing in our projects (esp. XCTL)

- How to automate regression testing,
  i.e. to prove that there is no regression after modifications of software?
- How to support the development test cases?
- How to check the completeness of test cases?

Contents

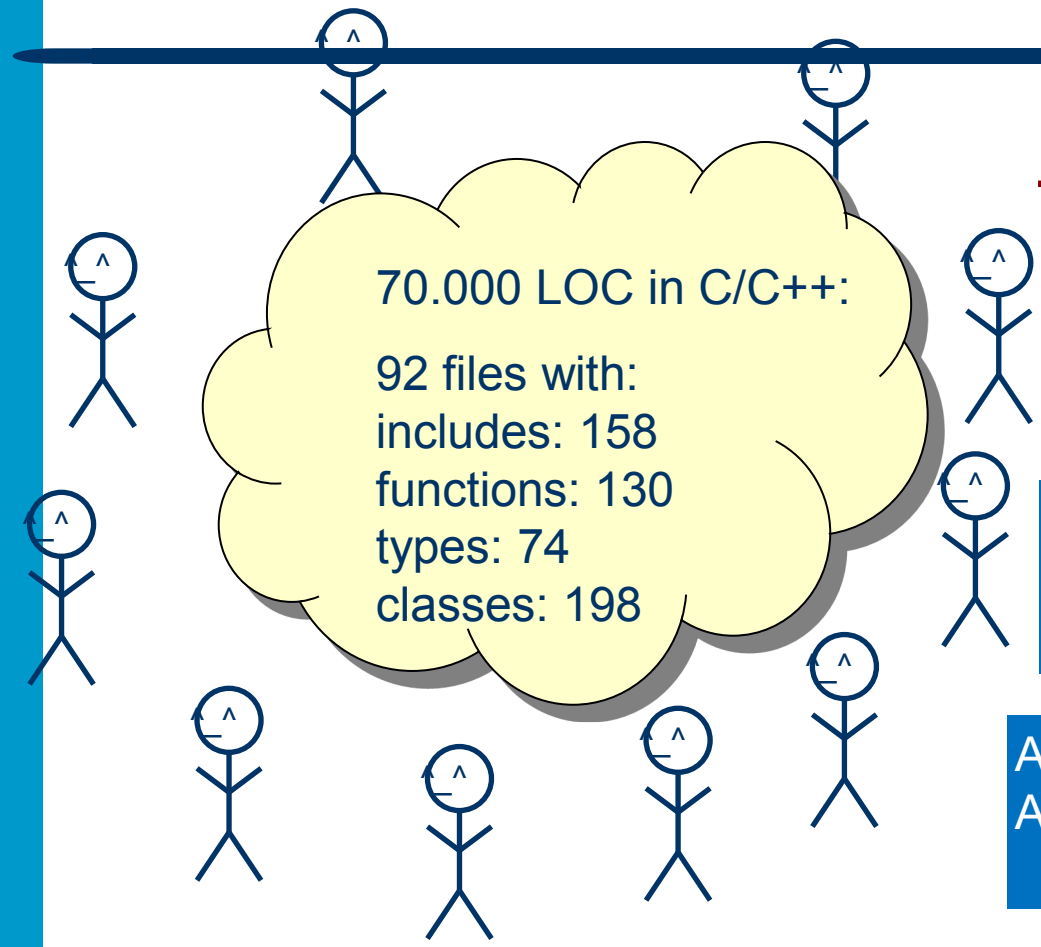
- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

Main issues of software testing in our projects

- How to automate regression testing,
  i.e. to prove that there is no regression after modifications of software?
- How to support the development test cases?
- How to check the completeness of test cases?

→ **Each issue should be supported by a testing tool**

Real-life project XCTL: three testing tools



Testing tools for different needs:

TESTONA:
Find all necessary test cases

SOTA:
Check the quality of test cases:
measure coverage degrees

ATOS:
Automatic regression testing
(repeat the test after modification)

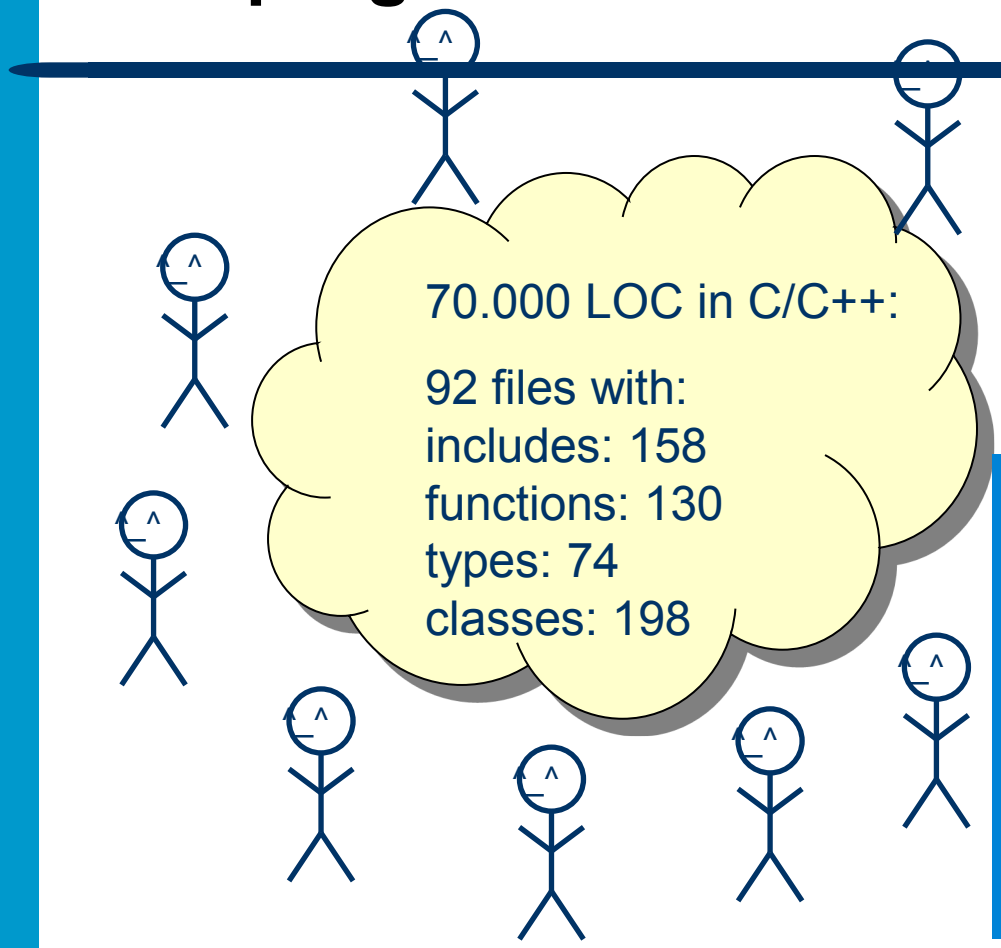
Many developers are working
at different parts of the system

Find errors as early as possible:
test very often (but: test takes 2 days)

Main issues of software testing in our projects

- How to automate regression testing,
  i.e. to prove that there is no regression after modifications of software?
- How to support the development test cases?
- How to check the completeness of test cases?

Problem: systematic testing after each program modification



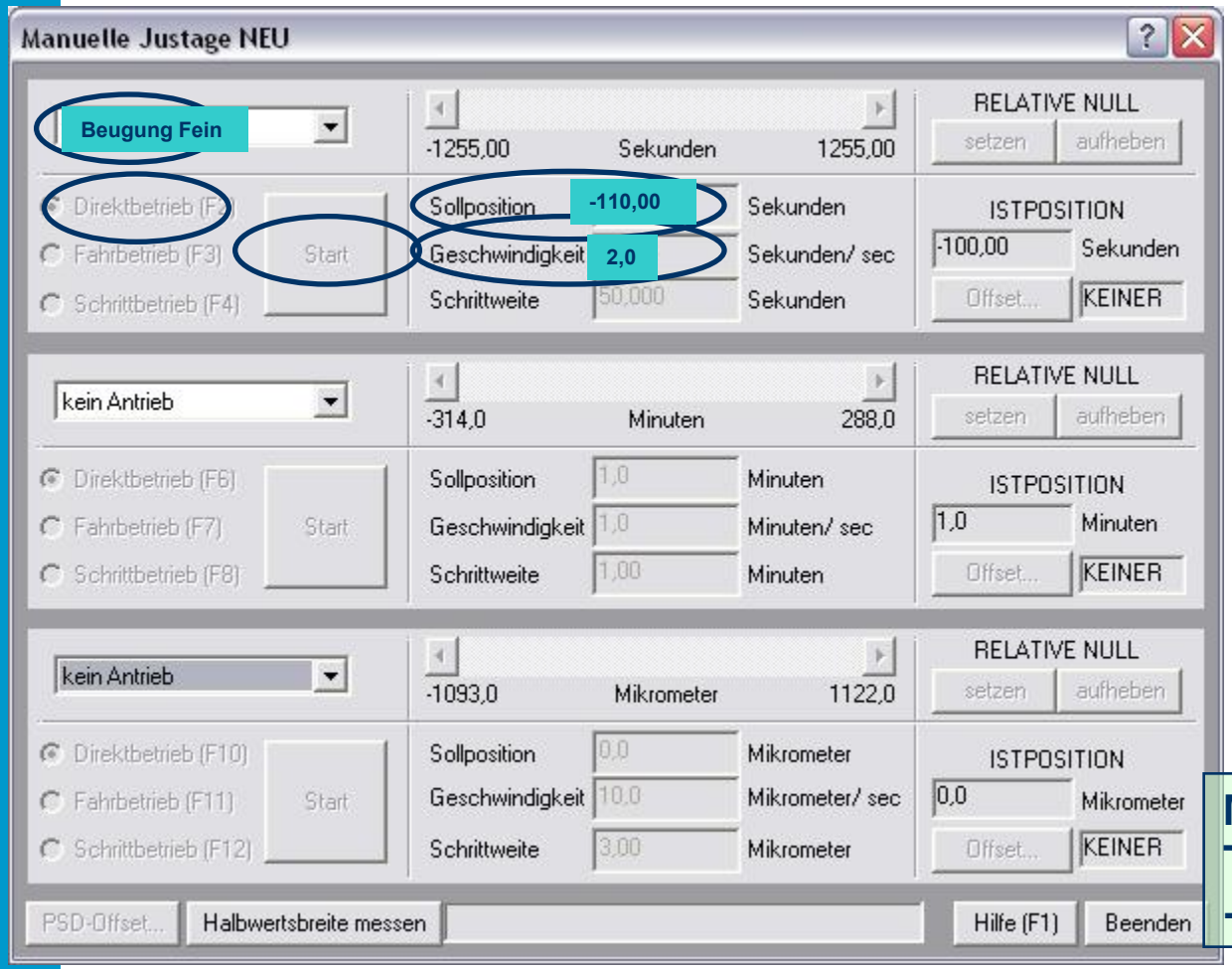
Regression testing

Regression testing:
- Testing that after modifications there are no new errors included (no step backward = no regression)
- The same test cases before and after modifications
- Tool support: Automatic run and documentation

Many developers are working at different parts of the system

Instead of manual testing 2 days: now only 1 hour

GUI-oriented automatic regression testing: sequence of test activities

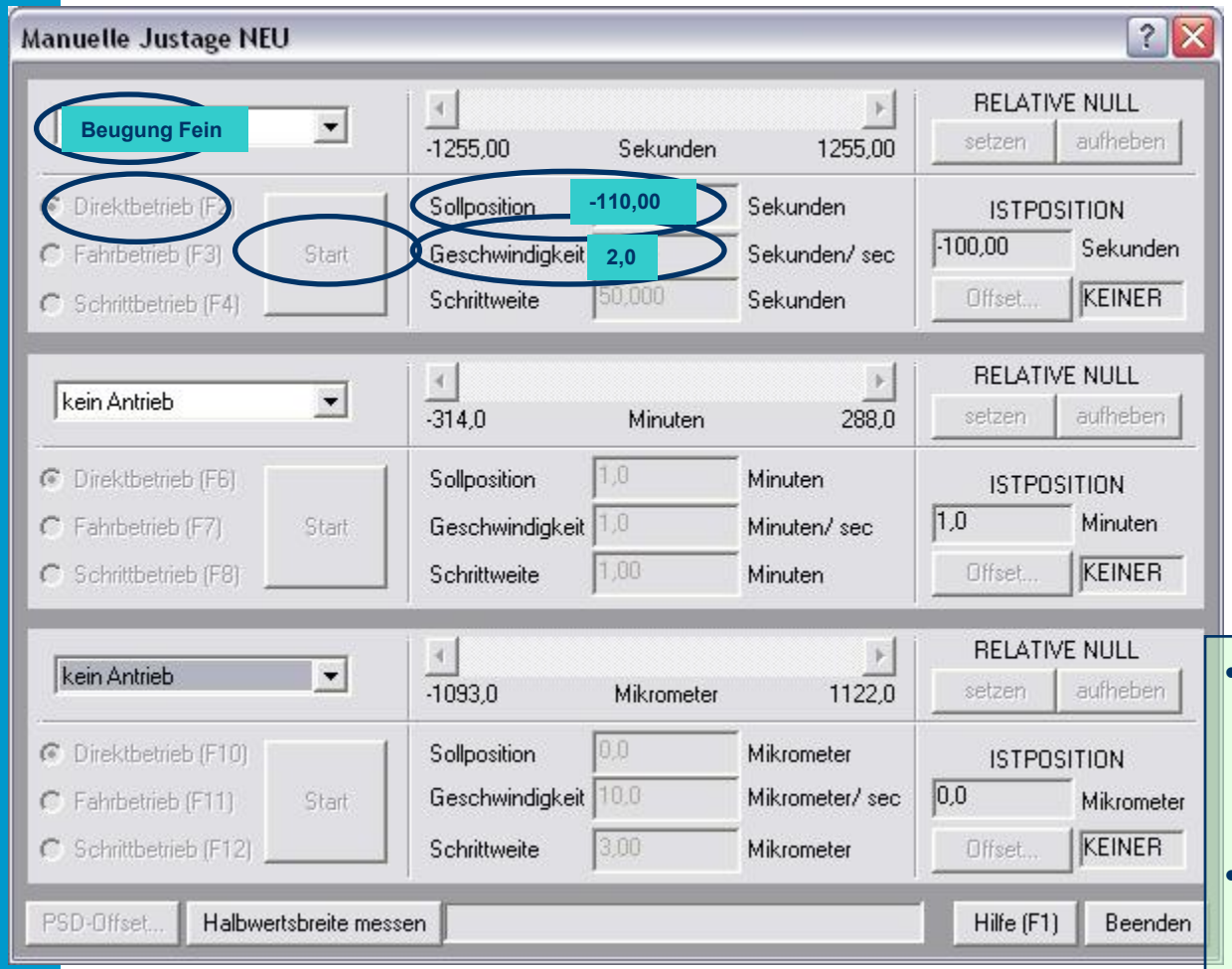


Test sequence:

- Start the system XCTL
- Open window ‘Manuelle Justage‘
- Initial state: of dialogue box; source: .ini-File
- Activities on the dialogue box
  - Combo box: select motor (Beugung Fein)
  - Radio button: Direktbetrieb
  - Edit box: Sollposition: -110,00
  - Edit box: Geschwindigkeit: 2,0
  - Button: Start
- Close dialogue box ‘Manuelle Justage‘
- Stop the system XCTL

Manual input into the GUI: Time-consuming, error-prone - If you repeat it again and again

GUI-oriented automatic regression testing: Capture and replay tool ATOS for GUI systems



ATOS WinRunner

Test sequence:

- Start the system XCTL
- Open window ‘Manuelle Justage‘
- Initial state: of dialogue box; source: .ini-File
- Activities on the dialogue box
  - Combo box: select motor (Beugung Fein)
  - Radio button: Direktbetrieb
  - Edit box: Sollposition: -110,00
  - Edit box: Geschwindigkeit: 2,0
  - Button: Start
- Close dialogue box ‘Manuelle Justage‘
- Stop the system XCTL

- **Capture:**
  Manual input into the GUI: The test tool ATOS stores the input in a test script file.
- **Replay:**
  Later on, under the control of ATOS, this script file directs the run of the system (regression testing)

Main issues of software testing in our projects

- How to automate regression testing,
  i.e. to prove that there is no regression after modifications of software?
- How to support the development of test cases?
- How to check the completeness of test cases?

TESTONA:
Classification Tree Method

Developed by Daimler-Chrysler:

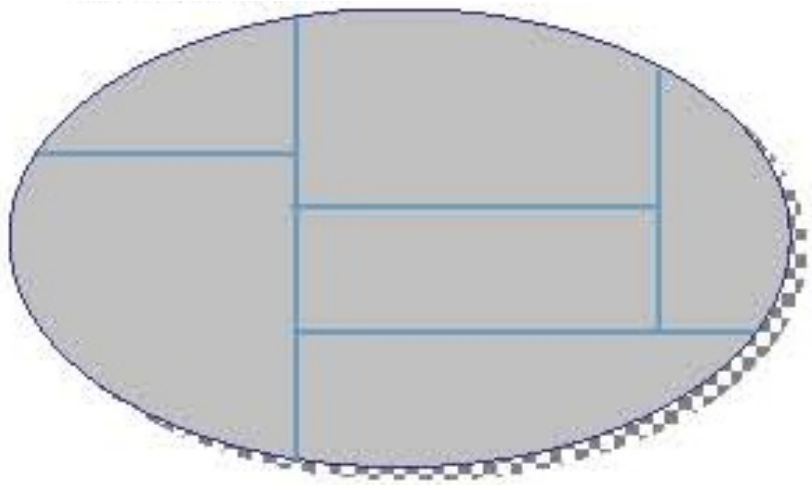
- **Originally: Embedded into a test system for car industry**

  http://www.testona.net/en/index.html

- **Widely propagated usage:**
  Airbus, Mercedes, BMW, Audi etc.

- **Free academic version**

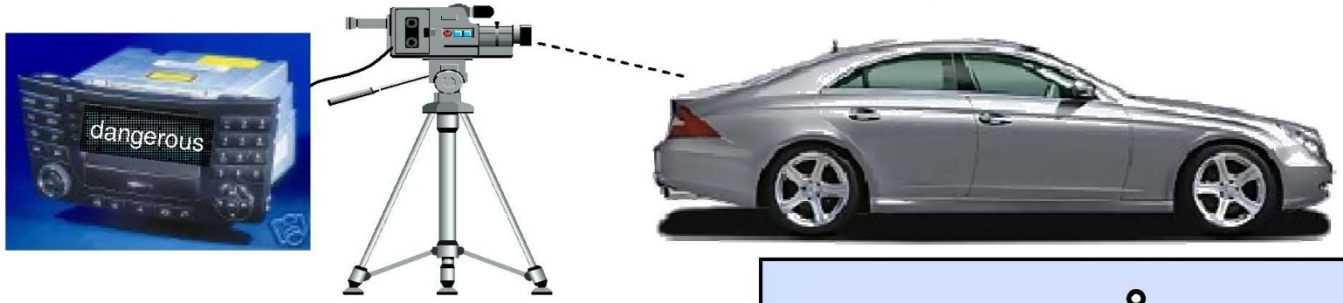
Classification Tree Method



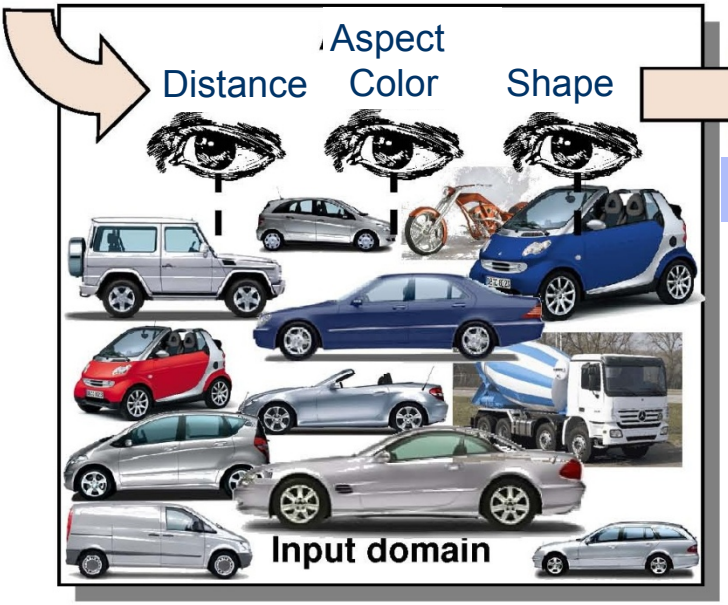
Basic principle:
Classification of input data space

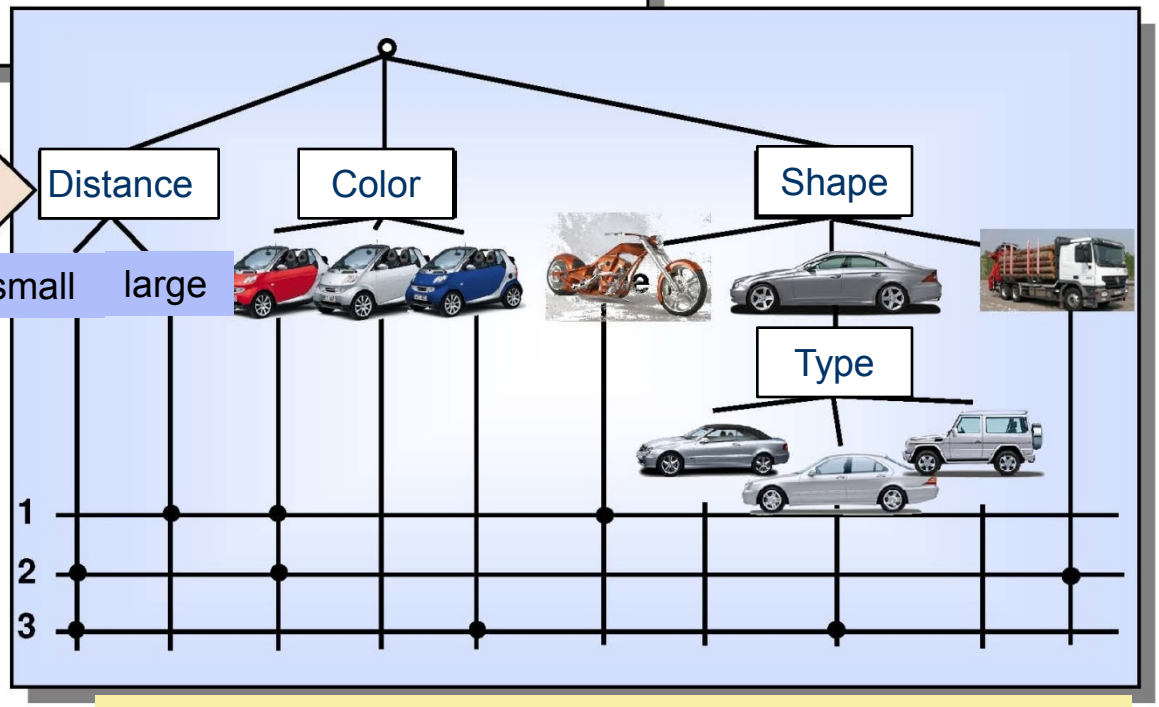
Driving assistant: Determine the distance between objects

Exemplary test object: computer-vision system, distance to the preceding vehicle determined



Autonomous car driving





Source of case-study: Wegener, DaimlerChrysler

TESTONA

Editions

TESTONA Light	TESTONA Express	TESTONA Professional	TESTONA Enterprise
Free Edition with reduced functionality	Edition for the cost-effective entry into professional software testing	Professional Edition conceived for sophisticated software testing	Enterprise-Edition with full scope of performance and all features tailored to individual needs
AddOns not available	AddOns not available	Alle AddOns available	All AddOns included (e.g. AUTOSAR, Matlab)
Support & Maintenance not available	Support & Maintenance available	Support & Maintenance package available (incl. project support and more)	Incl. 2-days On-site Training
Only Node Locked Licence	Only Node Locked Licence	Node Locked, Hard Locked, Floating-Licence	Floating-Licence incl. Borrow-Function
Details about Features Features	Details about Features Features	Details about Features Features	Details about Features Features
		Additional features included such as Requirements tracing, colouring	
		Numerous Import/ Export-connections (e.g. DOORS, QualityCenter)	
Free Download	Buy TESTONA Express	Buy TESTONA Professional	Buy TESTONA Enterprise

Main issues of software testing in our projects

- How to automate regression testing,
  i.e. to prove that there is no regression after modifications of software?
- How to support the development test cases?
- How to check the completeness of test cases?

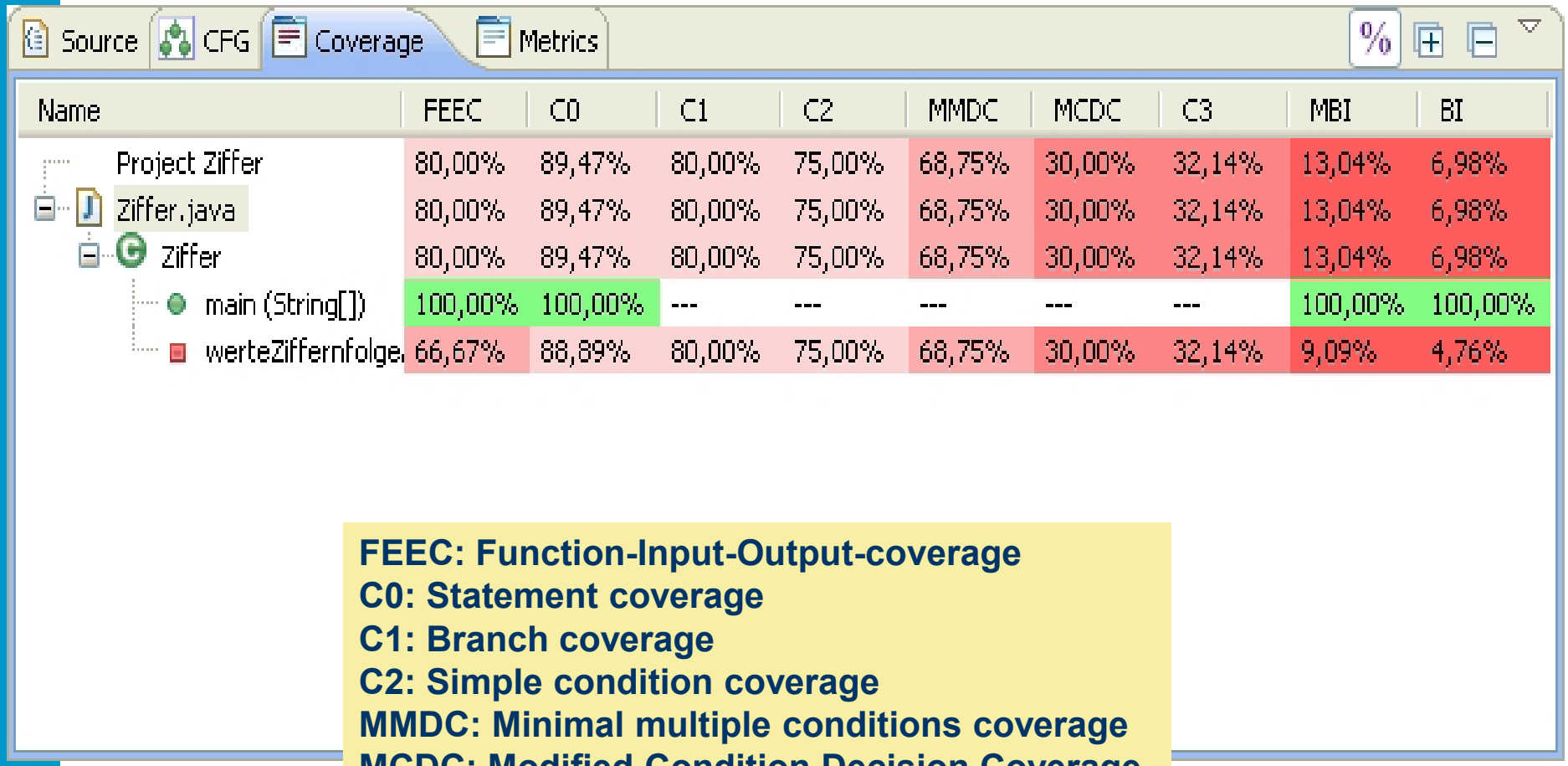
SOTA …

- **Delivers coverage degrees**
  for a given test data set and a program
  (based on *structure-oriented testing / white box testing)*
- **Visualization of coverage**

→Quality of test data assessed

Author: Ronny Treyße, Diploma thesis, HU

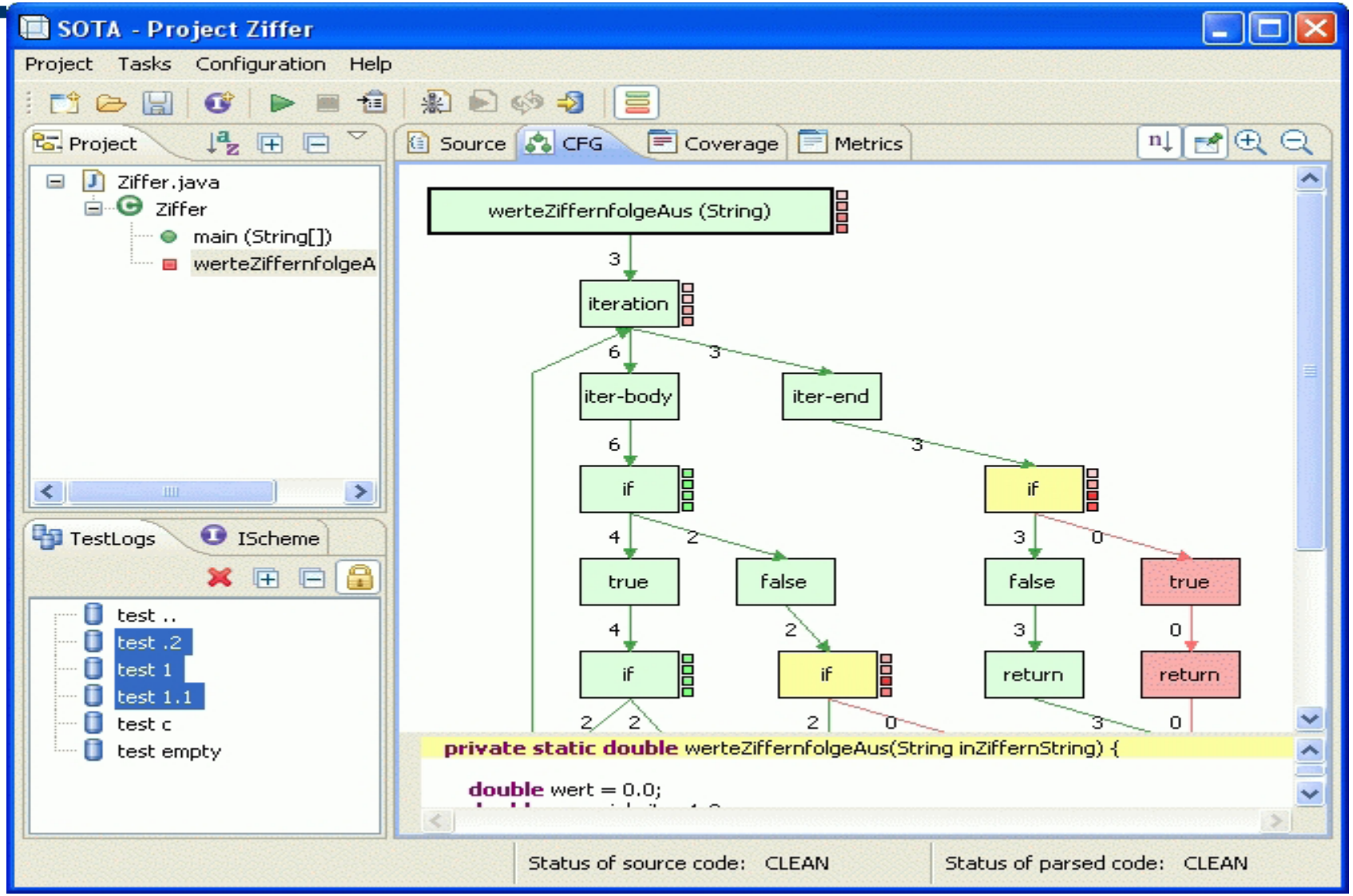
SOTA: coverage degrees for nine criteria



Name	FEEC	C0	C1	C2	MMDC	MCDC	C3	MBI	BI
Project Ziffer	80,00%	89,47%	80,00%	75,00%	68,75%	30,00%	32,14%	13,04%	6,98%
Ziffer.java	80,00%	89,47%	80,00%	75,00%	68,75%	30,00%	32,14%	13,04%	6,98%
Ziffer	80,00%	89,47%	80,00%	75,00%	68,75%	30,00%	32,14%	13,04%	6,98%
main (String[])	100,00%	100,00%	---	---	---	---	---	100,00%	100,00%
werteZiffernfolge	66,67%	88,89%	80,00%	75,00%	68,75%	30,00%	32,14%	9,09%	4,76%

FEEC: Function-Input-Output-coverage
C0: Statement coverage
C1: Branch coverage
C2: Simple condition coverage
MMDC: Minimal multiple conditions coverage
MCDC: Modified Condition Decision Coverage
C3: Multiple conditions coverage
MBI: Modfied Boundary-Interior path coverage
BI: Boundary-Interior-Path Test

Control-flow graph and coverage visualization



Contents

- Introduction
- Textbooks: What are the main issues of software testing?
- Other sources for testing issues
- Main issues and problems for testing in (our) software projects
- Which tools: TESTONA, ATOS, SOTA
- Summary

Conclusion

There is no unique (best) software testing tool:
it depends on the activity to be supported

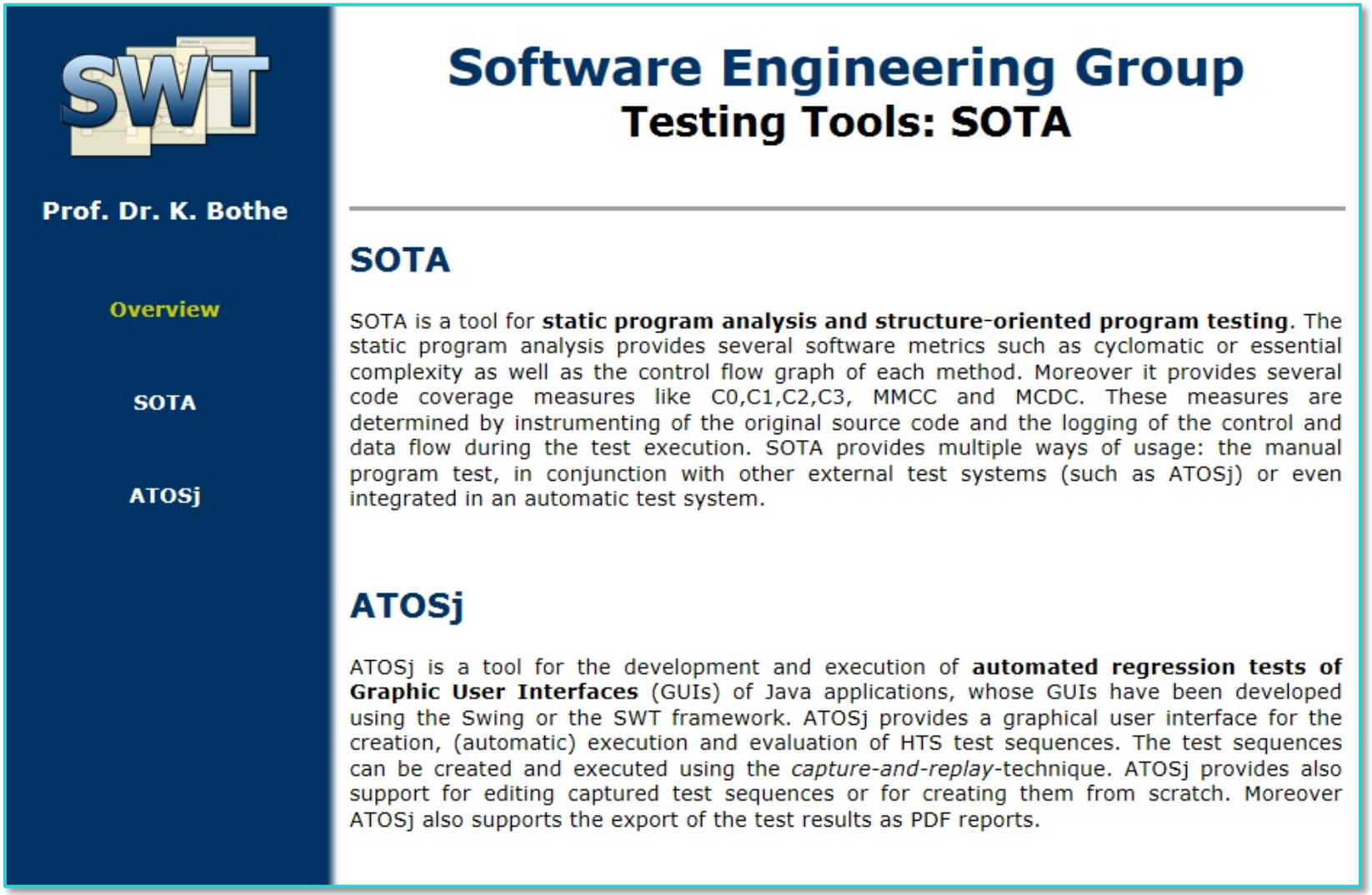
Selection of three different kinds of testing tools
for different purposes:

- ATOS/ATOSj *) : Regression testing for GUI software
- TESTONA +): Systematic selection of test cases
- SOTA *): Check the completeness of test cases
  with respect to different criteria of structure-oriented testing
  (white box testing, e.g. branch coverage)

*) own development
+) industry product (Daimler, free academic version)

➔Students worked with three different tools in practical assignments

Own free software tools



https://www2.informatik.hu-berlin.de/~wwwcompsoft/lehre/TestTools/

Thank you