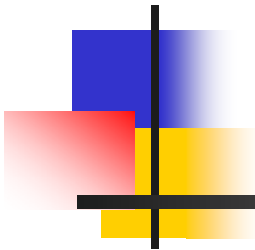


Do students like tool usage in a Software Engineering course?

A decorative graphic on the left side of the slide, consisting of overlapping colored squares (blue, red, yellow) and a black crosshair.

Michael Ritzschke, Olga Schiemangk, Klaus Bothe

9th Workshop "Software Engineering Education and Reverse Engineering"
Neum, Bosnia and Herzegovina, 31 August – 5 September 2009

Contents

- Tools in SE course at HU (Overview)
- In detail: assignment – tool support
- Students feedback
- Summary

Which tools did we use in 2009?

Assignments

1. Review requirements specification "SemOrg"
2. Function points
3. Develop an OOA model **objectiF**
4. Formal specifications **Z/EVES**
5. Metrics **CCCC**
6. Select test cases functionally by the **CTE**
7. Select regression test cases by **ATOS**
8. Test coverage with **SOTA**

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Motivation: Why using tools?

- Students learn to work with different tools, like in practical software development
- Software companies even expect that students are familiar with tools
- Tools: implement theoretical ideas of software engineering
- Solving assignments: we hope tools will introduce fun and liveliness

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Motivation: Why using tools? (cont.)

- Students get an impression about benefits and shortcomings of tools, e.g.
 - Bugs in tools
 - Stability
 - Online help
 - Usability
 - Platform dependencies

Tools and developers?

■ Commercial Software

- **objectiF**: Tool for Model-Driven Development with UML in Java, .NET and C++ (Company: microTOOL)
- **CTE XL: Classification Tree Editor eXtended Logics** (DaimlerChrysler); supports functional testing

■ Open-source Software

- **CCCC**: Tool for metrics related to source code (by Tim Littlefair)
- **Z/EVES**: Supports the Z notation; originally available from ORA Canada (by Mark Saaltink), its status is unknown now.

■ Developed at HU

- **ATOS**: GUI-oriented regression testing, capture-and-replay technique
- **SOTA**: Test coverage tool

This Is objectiF®



objectiF

objectiF Eclipse Edition

objectiF Visual Studio
.NET Edition

objectiF Enterprise
Edition

Download

Tutorials

Films

Feature Lists

objectiF in Detail

The Right objectiF
Variant for You

Web Development

SOA Development

From Use Case
to Code

Model-Driven
Development

Software Technology
Services

Success Stories

Prices

Buy

Brochure in PDF format

objectiF – The Tool for Model-Driven Development with UML in Java, .NET and C++

In objectiF you will find everything you need for efficient development. This includes development of enterprise, SOA and web applications as well as client-server applications and embedded software.

- General support of UML
From Use Case to Code ▶
- Automatic transformation of domain-oriented into technical models with source code following the concepts of MDA
Model-Driven Development with objectiF ▶
- Business Process Modeling with BPMN, graphical orchestration of the process with Web services and code generation of WS-BPEL – all combined with MDD
objectiF as Software Factory for SOA ▶
- Technical automation of MDD for highly efficient Web development
objectiF as Software Factory for Web applications ▶

Choose between the following objectiF editions:

Take a look at the development of web applications with objectiF...



From use case to domain-oriented model

- ▶ Watch film (German)
- ▶ Download film (German)
- ▶ Read article



From a domain-oriented model to a technical model and to code

- ▶ Watch film (German)
- ▶ Download film (German)
- ▶ Read article



[Customer-Login](#)

[Contact us](#)

[Site map](#)

[Imprint](#)

[Privacy Policy](#)

Company

Automotive

MESSINA
modularHiL
MODENA
▶ CTE
Engineering
Consulting
References/Partners

Transportation

Aeronautics

Space & Defence

Events

Press

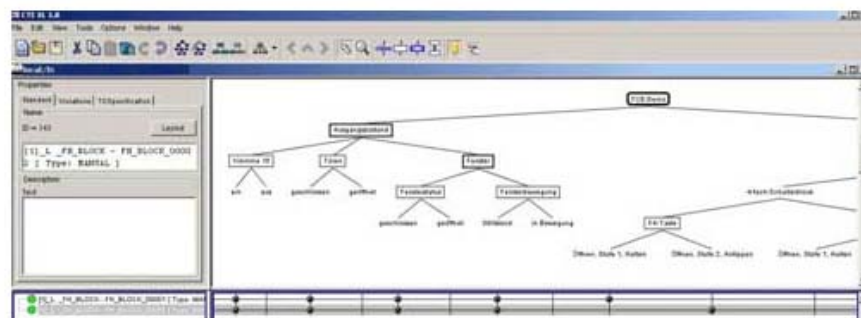
Information Material

Careers

Contact us

Home

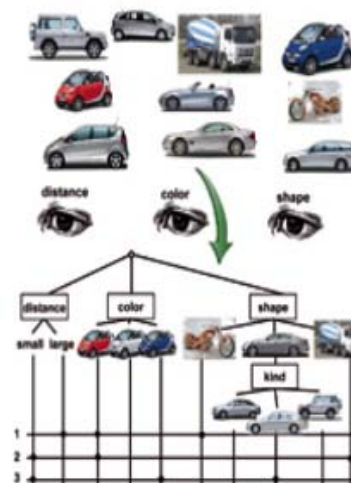
CTE XL Efficiently generating Systematic Test Specifications



The classification-tree editor CTE XL is an effective tool for applying the classification-tree method in systems and software development. An easily operated, context-sensitive graphical editor reliably guides the user through the entire process of generating the classification-tree. By means of combination rules the tester is able to define both the test coverage and the test focuses. On the basis of these combination rules the tester automatically generates the test cases at the push of a button.

Fields of Application

- Requirements and test engineering
- Test management (resource optimized testing)
- Automated generation of formal and semiformal test specifications
- All vehicle domains (reactive and continuous functions)



[CTE XL FAQs](#)

[Download](#)

systematic-testing.com

www.systematic-testing.com

[Home](#) [Functional Testing](#) [Evolutionary Testing](#) [Static Analysis](#) [Further Resources](#) [Contact](#) [TAV](#) [Impressum](#)

News

18.10.2007: CTE/XL
V1.8 released
[download here](#)

systematic-testing.com

This site offers access to selected papers on software testing. The main focus lies on papers related to functional and evolutionary testing as well as their automation. Most of the papers are available in pdf-format or are linked to the original source.

Furthermore, this site offers a free download of the **Classification-Tree Editor CTE/XL**.

CCCC: cccc.sourceforge.net



CCCC - C and C++ Code Counter

A free software tool for measurement of source code related metrics by Tim Littlefair

The CCCC tool was developed as a testing ground for a number of ideas related to software metrics in an MSc project. The research project is described at <http://www.chs.ecu.edu.au/~tlittlef>

My research project is now (hopefully) coming to an end. The descriptive page will remain on the net, and some material (e.g. the finished thesis) may be added to it, but when my registration as a student expires, soon afterward I would expect to lose the ability to change material on this site, hence the effort to get this one up and running as a forum to cover the onward development of the CCCC tool (for as long as there is any). Many thanks to SourceForge for providing this forum to me at no cost for this purpose. Check out <http://sourceforge.net> if you are interested in their policy of providing free web hosting for open source projects.

In addition to hosting the page you are reading at present, SourceForge support a range of services for their projects. <http://sourceforge.net/projects/cccc> is their standard summary page related to the CCCC project, which provides access to all of these services. In the future I hope to use SourceForge to host anonymous CVS access. For the present, I have set up mailing lists and bug tracking and http download access to the most recent beta release cccc-3pre48.tar.gz.

I have set up a number of mailing lists related to CCCC on the SourceForge site: there is [one for announcements](#), [one for discussions about the use of present versions of CCCC](#), and [another for discussions relating to the future development process](#) (including proposals for new features or changes to old ones).

The interface to search or add to the bug tracking database for CCCC is available via https://sourceforge.net/bugs/?group_id=7763.

The last version of CCCC which I released was 2.1.4, which I put out some time in September 1997. Between then and December 1998, I tried to concentrate on writing up the project thesis. Since December 1998 I have been working on a new release of CCCC, which I plan to designate 3.0. The new version fixes a number of bugs reported in 2.1.4, but also has a lot of new code, and will presumably introduce many new bugs.



ORA Canada

ORA Canada

[Home](#)

[Contact us](#)

[What's new?](#)

Products and services

[Z/EVES](#)

[Documentation](#)

[Screen shots](#)

[EVES](#)

[Ada'95](#)

Reports and Collections

[ORA Canada](#)

[Bibliography](#)

[Automated Deduction](#)

[Bibliography](#)

Z/EVES

As of June 2005, ORA Canada can no longer distribute Z/EVES.

Z/EVES 2.4.1 was the last version released. This version includes a graphical user interface that allows Z specifications to be entered, edited, and analysed in their typeset form; supports the incremental analysis of specifications; and manages the synchronization of the analysis with modifications to the specification. Some [screen shots](#) are available.

Z/EVES uses state-of-the-art formal methods techniques from Europe and North America, integrating a leading specification notation with a leading automated deduction capability. The resulting system supports the analysis of Z specifications in several ways:

- syntax and type checking,
- schema expansion,
- precondition calculation,
- domain checking, and
- general theorem proving.

What's New?

ORA Canada has been inactive since 2005.

Unfortunately, we are unable to distribute EVES or Z/EVES any longer, because we do not own the intellectual property in EVES and no longer have rights to it. However, Mark Saaltink has been working on connecting the Z part of Z/EVES to a new prover. This should be released very soon (in Spring 2009). While the new system is not as powerful, it is a start, and the eventual hope is to make it all open source, so that interested users can improve it.

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Contents

- Tools in SE course at HU (Overview)
- In detail: assignment – tool support
- Students feedback
- Summary

Assignments overview (website)

Software Engineering

Prof. K. Bothe

Summer semester 2009

Assignments

Precondition for admission to examination: 75% of reachable points.

Points: you can get maximum 10 points for each assignment.

Mode of delivery: printed on paper.

The annotated solutions to the assignments are distributed for discussion during the class and they are collected after the class as a basis for examination.

Teamwork: Assignment tasks are normally solved in groups of three people. Please talk about deviations to that rule with Dr. Ritzschke before.

Assignments overview

.	Theme	Beginning	Delivery	Evaluation	Tool	Demo. in Lec.
Assignment 1	Review Requirements specifications	21.04.09	11.05.09	20.05.09	-	-
Assignment 2	Function point method	04.05.09	18.05.09	27.05.09	-	-
Assignment 3	OOA model	11.05.09	03.06.09	10.06.09	objectF	11.05.09
Assignment 4	Formal software specification	18.05.09	10.06.09	17.06.09	Z/EVES	-
Assignment 5	Classification tree method	25.05.09	15.06.09	01.07.09	CTE	25.05.09
Assignment 6	Test coverage	03.06.09	22.06.09	01.07.09	SOTA	03.06.09
Assignment 7	GUI oriented regression test	10.06.09	29.06.09	15.07.09	ATOSj	10.06.09
Assignment 8	Metrics	15.06.09	06.07.09	15.07.09	cccc	-

Tools (website)

Software Engineering

Prof. K. Bothe

Summer semester 2009

Tools

In the course of the lecture you will work with some software engineering tools. Several of them are demonstrated during the lectures.

OO CASE TOOL: [objectiF](#)

TEST TOOLS:

- **CTE XL** - Classification Tree Editor
 - [Users guide](#)
- **SOTA**
 - [Installation hints](#)
 - [User manual](#) (pdf, 1,9MB)
 - [Setup of test environment for SOTA assignment](#) (HUSemOrg)
 - [Setup of test environment for sample program1](#)
 - [Setup of test environment for sample program2](#)
- **ATOSj**
 - [Installation hints](#)
 - User manual (pdf: [one slide per page](#); [two slides per page](#))
 - HTS language specification([pdf-file](#))
 - [Project setup for HUSemOrg](#)

Information about:

- User manuals
- Installation guides
- Download information

Z-SPECIFICATION: [Z/EVES](#)

General activities of students to use tools

A decorative graphic consisting of overlapping colored squares (yellow, red, blue) and a black crosshair.

- First contact: Tool demonstration in the lectures
 - examples, hints, remarks, overview of documents
- Preparation: Download, installation, test with sample application
- Become familiar with tool: Get practical know-how by using the tool, work with tutorials and applications, use the help-instructions and comments to find answers to questions
- Solving the assignments: hopefully with more liveliness and fun ...

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

objectiF

- Students get the description of a

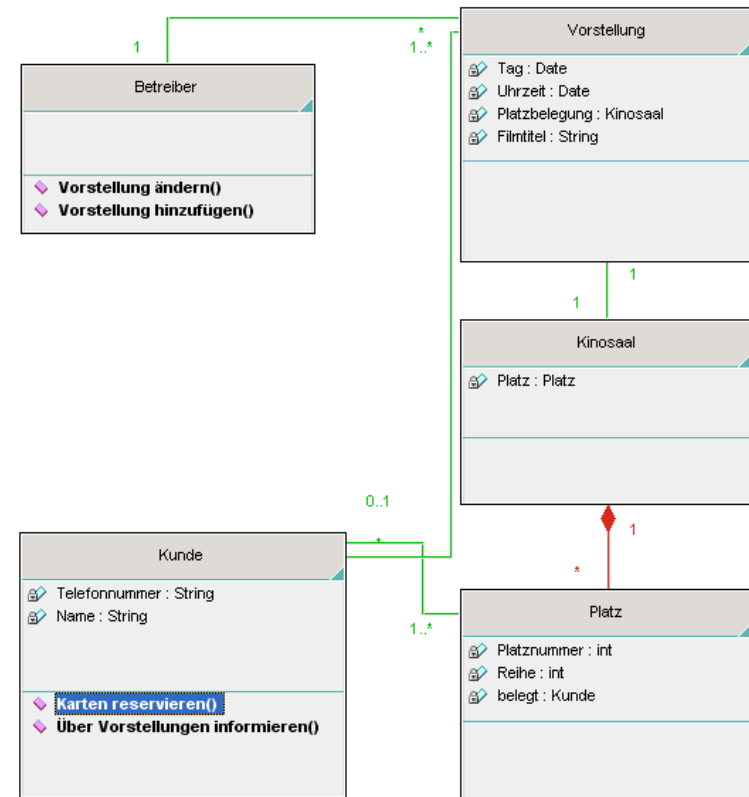
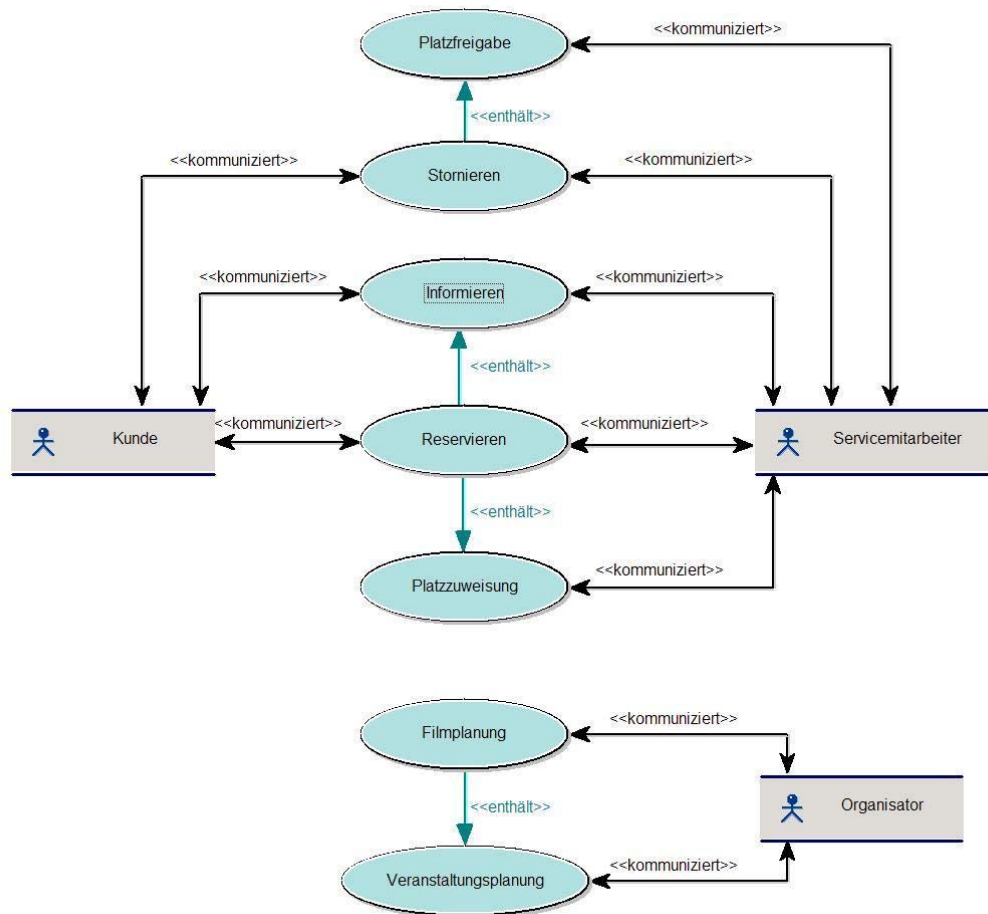
Cinema booking system

- seat reservation, some cinema halls
- booking: row and place number, possibility to ask for neighbouring places
- ...

- Tasks: Use the tool objectiF to develop a
 - a) use case diagram
 - b) class diagramm
-

Tool support: draw graphical elements (use cases, classes)

objectiF: sample solution



A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Z/EVES

- Students get the description of a queue

create	generates an empty queue
extend	appands a new element at the end
get_first	affords the first element
delete	cuts the first element
not_empty	tests, if the queue is not empty (empty queue afords false)

- Tasks: Describe the queue
 - a) use the Z-notation
 - b) algebraic specification
-

Tool support: Z Mini Editor, Syntax check



Z/EVES

Z/EVES - Z:/software_engineering/z_eves/data/ZQueue

File Edit Command Window Abort Eager Lazy

Syntax Proof Specification

? ?

Queue

queue_elements : seq \mathbb{Z}

Z/EVES Mini Editor

File Edit Enabled Disabled

- Clear
- Schema Box
- Axiom Box
- Generic Box
- Copy
- Cut
- Paste

Δ

$\langle \rangle$

Logical Symbols $\neg \wedge \vee \Rightarrow \Leftarrow \nabla \exists \cdot$

Basic Symbols $\mathbb{P} \times \in \equiv \langle \rangle ; > \dagger \langle \rangle \lambda \mu \theta \Delta \exists$

Box Drawing $\ulcorner _ _ _ _ _ _ _ _ _ _$

Subscripts 0 1 2 3 4 5 6 7 8 9

Toolkit Symbols $\neq \subseteq \emptyset \subset \supset \cup \cap \cup \cap \rightarrow \leftrightarrow \circ \triangleleft \triangleright \triangleleft \triangleright$

$() \oplus \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \sim + *$

$\mathbb{N} \mathbb{Z} < > \leq \geq F \rightarrow \rightarrow \langle \rangle \sim 1 \ll \ll$

Special Words **let if then else local global dom ran id div mod**

seq iseq prefix suffix in disjoint partition bag inbag pre

Queue

queue_elements : seq \mathbb{Z}

new

Δ *Queue*

queue_elements' = $\langle \rangle$

is_empty

\exists *Queue*

msg! : {true, false}

msg! = true \Leftrightarrow *queue_elements* = $\langle \rangle$

enqueue

Δ *Queue*

inp? : \mathbb{Z}

queue_elements' = *queue_elements* \sim \langle *inp?* \rangle

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

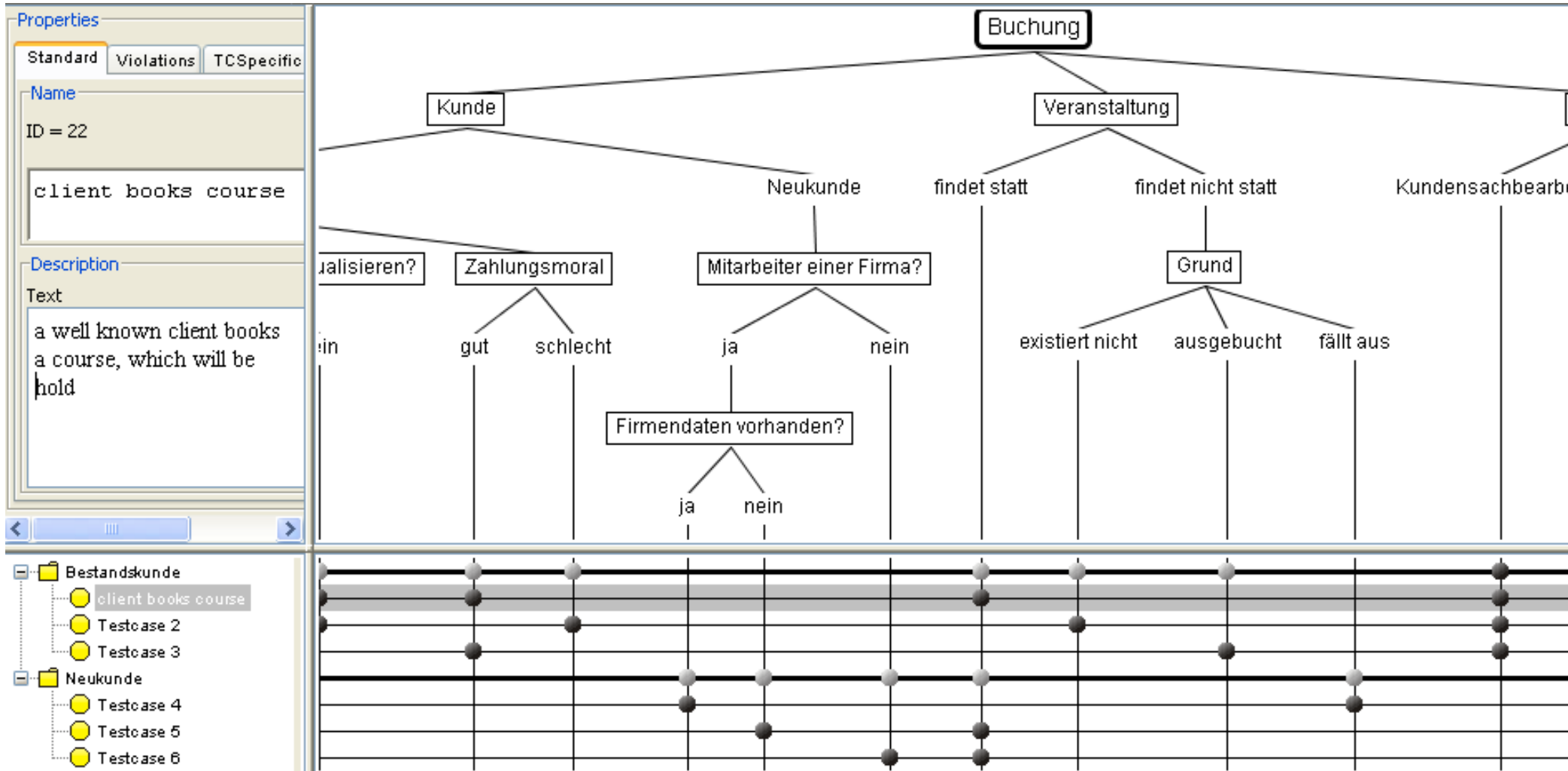
CTE XL

- Students know the functional specification document "Seminar organization" and the description of the use case /F20/ *From the registration of a client to the reservation of a course*
- Tasks
 - a) use the classification tree method to prepare a functional test
 - b) find out which persons/objects/situations are relevant for the booking process and build classes
 - c) generate test cases (combine classes)

Tool support: Drawing classification tree, generating and marking test cases, creation of documents ...



CTE XL



A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

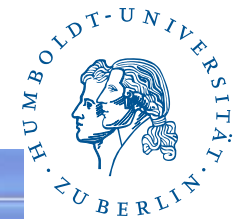
SOTA

 (Detail Information by M.Hildebrand)

- Realize a structure-oriented test for the test object *ClientWindow.java* from the semorg.gui-package. *ClientWindow* implements the Window "client".
- Tasks
 - a) use SOTA, Eclipse, HUSemorg and MySQL to realize different test runs for HUSemOrg (use Buttons, text inputs, arrays ...)
 - b) use SOTA to check the result (e.g. Instruction coverage) and start further test runs till the coverage reaches a given border (e.g. 90%)

Tool support: Instrumentation of java-files

SOTA



The screenshot displays the Eclipse IDE environment. On the left, the Package Explorer shows a project structure with a package named 'semorg'. The main editor window shows the source code for 'ClientWindow.java'. The code includes several 'import' statements for classes in the 'semorg.sql' package, followed by a class declaration and a method definition. A GUI window titled 'Seminarorganisation' is overlaid on the code, showing a form for entering customer data. The form has fields for 'Nummer', 'Name' (with sub-fields for 'Anrede' and 'Titel'), 'Vorname', 'Nachname', 'Adresse' (with sub-fields for 'Strasse', 'PLZ/Ort', and 'Land'), and 'Zusatz'. The 'Anrede' field is set to 'Herr' and 'Titel' is set to 'Prof'. The 'Nachname' field contains 'Bothe'. The 'Strasse' field is empty. The 'PLZ/Ort' field is split into two sub-fields, both empty. The 'Land' field is empty. The 'Zusatz' field is empty. The GUI window also has a 'Kontakt' section on the right side.

```
43 import semorg.sql.table
44 import semorg.sql.table
45 import semorg.sql.table
46 import semorg.sql.util
47
48 /** The window for edit
49 public class ClientWind
50
51     /**
52     * Inner class whic
53     * the lecturer wi
54     */
55     class InputChange
56         SWTCalendarLis
57
58     /**
59     * Appends to the
60     * changed. Furthe
61     * {@link semorg.g
62     */
63     private void input
64     /* ASC */ int asc_pe
65
66     /* ASC */ ASCLogger
67
68     /* ASC */ ASCLogger
69
70     /* ASC BEGIN DECLARAT
71     /* ASC */ boolean as
72     /* ASC */ boolean as
```

Tool support: Evaluation the test runs (different metrics, graphical view of instruction coverage, ...)



SOTA

The screenshot displays the SOTA - Project husemorg interface. The main window is divided into several panes:

- Project Explorer:** Lists Java files such as ClientBookingTableProvider.java, ClientBookingWindow.java, ClientListWindow.java, ClientTableProvider.java, ClientWindow.java, Company.java, CompanyBooking.java, CompanyBookingListWindow.java, CompanyBookingTableProvider.java, and CompanyBookingWindow.java.
- Source View:** Shows the source code for ClientWindow.java. The code is color-coded by coverage: green for 100% coverage and red for 0% coverage. The code includes a constructor and a listener method.
- Coverage View:** A table showing coverage metrics for various classes and methods.
- TestLogs:** Lists test runs for husemorg clientwindow 1 through 5.

Name	FECC	C0	C1	C2	MMDC	MCDC	C3	MBI
ClientBookingWindow.java	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
ClientListWindow.java	20,00%	16,05%	11,11%	8,33%	6,25%	0,00%	8,33%	13,89%
ClientTableProvider.java	87,50%	91,94%	91,18%	80,00%	80,00%	60,00%	80,00%	88,24%
ClientWindow.java	37,66%	78,08%	36,89%	38,79%	41,91%	10,34%	37,50%	0,00%
ClientWindow	62,96%	84,50%	51,25%	55,88%	57,69%	17,65%	55,22%	0,00%
ClientWindow (Shell)	100,00%	100,00%	---	---	---	---	---	100,00%
ClientWindow (Shell, Clie	100,00%	100,00%	---	---	---	---	---	100,00%

```
Shell instance used the as parent shell of the window.  
*/  
private ClientWindow(Shell parent) {  
  
    shell = new Shell(parent, SWT.CLOSE);  
  
    shell.addListener(SWT.Close, new Listener() {  
        public void handleEvent(Event event) {  
            if (inputChanged) {  
                if (confirmClose()) {  
                    if (inputEditingEnabled && input != null)  
                        Client.lockedIds.remove(new Integer(input.getId()));  
                } else  
                    event.doit = false;  
            } else if (input != null && inputEditingEnabled)  
                Client.lockedIds.remove(new Integer(input.getId()));  
        }  
    });  
}
```

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

ATOSj

- Realize a part of regression test for the use case */F20/ From the registration of a client to the reservation of a course* of the program HUSemOrg.
 - Tasks
 - a) use the capture function from ATOSj to register your inputs in different windows (e.g. seminary typ, docent, client, public seminar ...)
 - b) start the test sequences (replay) and prepare them manually so, that they runs without any mistakes
-

Tool support: Capture the inputs



ATOSj

The screenshot displays the ATOSj - HUSemOrg application window. The main area shows a test sequence editor with the following table:

Nummer	Kommando
1	START
2	ACTION, MAIN, MENU, "MainMenu", SELECT, SUBITEM, "&Ersterfassung", "SeminarTyp"
3	ACTION, "SeminarTypeWindow", EDITBOX, "ShortTitleText", EDIT, "typ123"
4	ACTION, "SeminarTypeWindow", EDITBOX, "TitleText", EDIT, "TESTEN MIT ATOSj"
5	ACTION, "SeminarTypeWindow", EDITBOX, "TopicText", EDIT, "Testen mit ATOSj"
6	ACTION, "SeminarTypeWindow", EDITBOX, "DurationText", EDIT, "6"
7	ACTION, "SeminarTypeWindow", EDITBOX, "AudienceText", EDIT, "Informatikstudenten"

Below the test sequence editor, there are two overlapping windows:

- Seminarorganisation**: A window with a menu bar (Anwendung, Stammdatenlisten, Ersterfassung, Geschäftsprozesse, Administration, Hilfe) and a blue background.
- Neu - SeminarTyp***: A form window for creating a new seminar type. It contains the following fields:
 - Nummer:
 - Kurztitel:
 - Titel:
 - Zielsetzung:
 - Tagesablauf:
 - Dauer:
 - Unterlagen:

Tool support: Replay the test sequence and check, if the functionality in a later release has changed



ATOSj

The screenshot displays the ATOSj software interface with several windows open. The main window shows a project tree on the left and a test execution summary table on the right. A dialog box titled 'Testpaket "Testpaket1" ausführen' is open, showing test settings and a detailed execution log. Another dialog box titled 'Neu - Dozent*' is open, showing a form for entering contact information.

Testpaket "Testpaket1" ausführen

Testeinstellungen

- Testsequenz bei einem Fehler beenden
- Testpaket bei einer fehlerhaften Testsequenz beenden
- Protokolldatei anlegen
- Interaktive Testsequenzen ignorieren
- 100 Verzögerung in ms
- Testsequenzen ohne Cleanup ignorieren

Nummer	Testsequenz	Interaktiv	Cleanup	Status	Dauer
1	Vorbereitung			Fehler	16
2	Programmstart			Erfolg	3281
3	Seminartyp			Erfolg	3359
4	Dozent			Erfolg	5297
5	ÖffentlicheVeranstaltung			Läuft	0

Nummer	Kommando	Status	Durchläufe	Dauer
1	ACTION, MAIN, MENU, "MainMenu", SELECT, SUB...	Fehler	1	516
2	ACTION, "PublicPresentationWindow", CHECKBOX...	Fehler	1	10016
3	ACTION, "PublicPresentationWindow", BUTTON, "	Fehler	1	10016

Nummer	Kommando	Status	Durchläufe	Dauer
19	ACTION, "LecturerWindow", TABFOLDER, "Tabs", ...	Erfolg	1	
20	ACTION, "LecturerWindow", TABFOLDER, "Tabs", ...	Erfolg	1	
21	ACTION, "LecturerWindow", TABFOLDER, "Tabs", ...	Erfolg	1	
22	ACTION, "LecturerWindow", TABFOLDER, "Tabs", ...	Erfolg	1	
23	ACTION, "LecturerWindow", EDITBOX, "DailyFeeT...	Erfolg	1	
24	TEST, "LecturerWindow", TABFOLDER, "Tabs", IT...	Erfolg	1	
25	TEST, "LecturerWindow", BUTTON, "&OK", TEXT, "...	Erfolg	1	16
26	TEST, "LecturerWindow", BUTTON, "&Neu", TEXT, ...	Erfolg	1	15
27	TEST, "LecturerWindow", BUTTON, "&Übernehme...	Erfolg	1	16
28	TEST, "LecturerWindow", BUTTON, "&Abbrechen,..."	Erfolg	1	16
29	TEST, "LecturerWindow", BUTTON, "Adressaufkle...	Erfolg	1	0
30	ACTION, "LecturerWindow", BUTTON, "&OK", SEL...	Erfolg	1	515

Neu - Dozent*

Form fields: Nummer, Name, Anrede (Herr), Titel, Vorname, Nachname (Testdozent), U, Kontakt (Telefon, Handy, Fax, E-Mail: test@dozent.de).

A decorative graphic consisting of overlapping colored squares (yellow, red, blue) and a black crosshair.

CCCC-Tool

- CCCC.exe calculates different software metrics (for Java and C)
 - Tasks
 - a) use the tool to investigate the MVG-metric (McCabe's Cyclomatic Complexity) for two given java-Files
 - b) for high MVG-values: look to the program-structure and discuss. Have the programs also a high complexity? Is the maintenance difficult?
-

Tool support: Investigates different metrics for whole modules and all functions



CCCC-Tool

Detailed report on module Error

Metric	Tag	Overall	Per Function
Lines of Code	LOC	43	*****
McCabe's Cyclomatic Number	MVG	57	*****
Lines of Comment	COM	6	*****
LOC/COM	L_C	7.167	
MVG/COM	M_C	9.500	

Functions

Function prototype	LOC	MVG	COM	L_C	M_C
error(int, int, int) definition Error.java:8	34	57	3	11.333	19.000

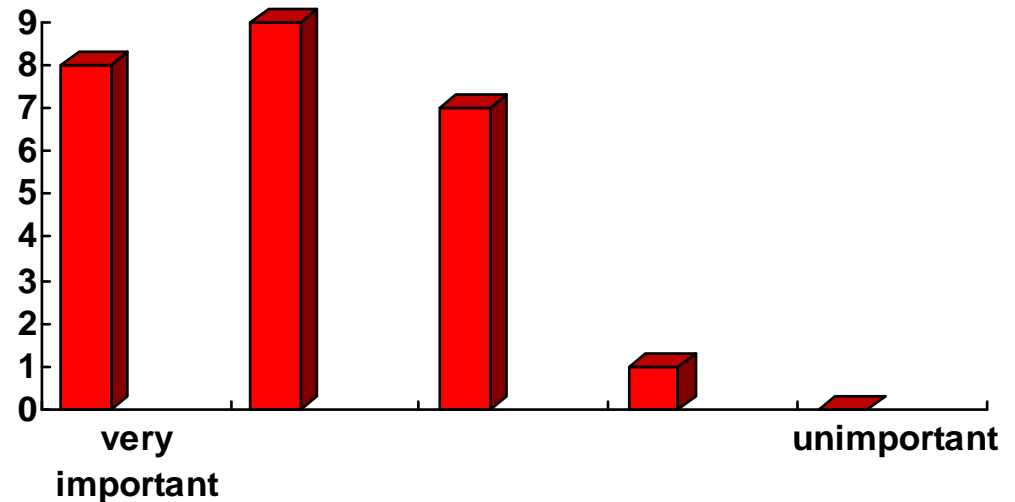
A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Contents

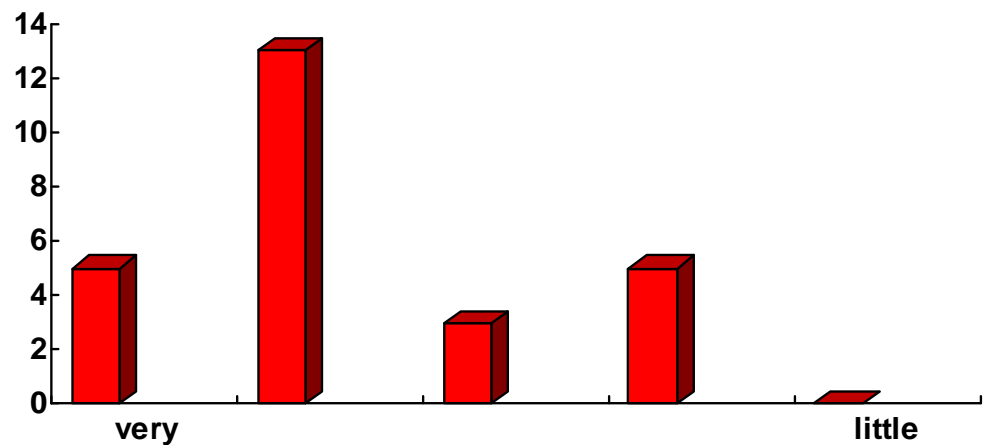
- Tools in SE course at HU (Overview)
- In detail: assignment – tool support
- Students feedback
- Summary

Generally questions

1. Relevance:
Is the use of software tools in the course important for your professional career?

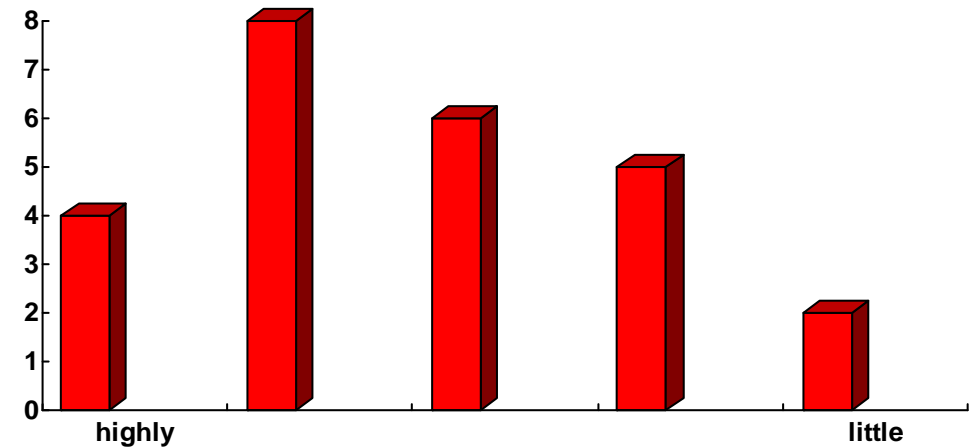


2. Competence: Did you get more professional competence about software tools by our assignments?

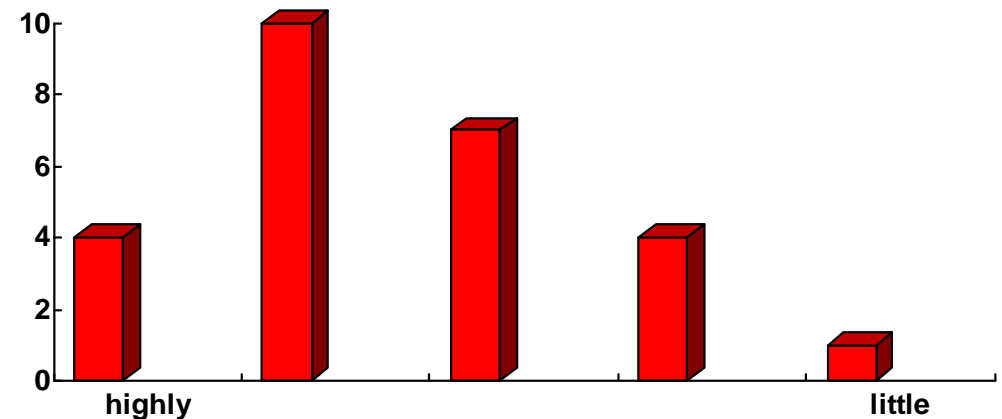


Generally questions

**3. Technical profit:
Did the use of the tools
have relevance for
solving the assignments
(effect on efficiency)?**

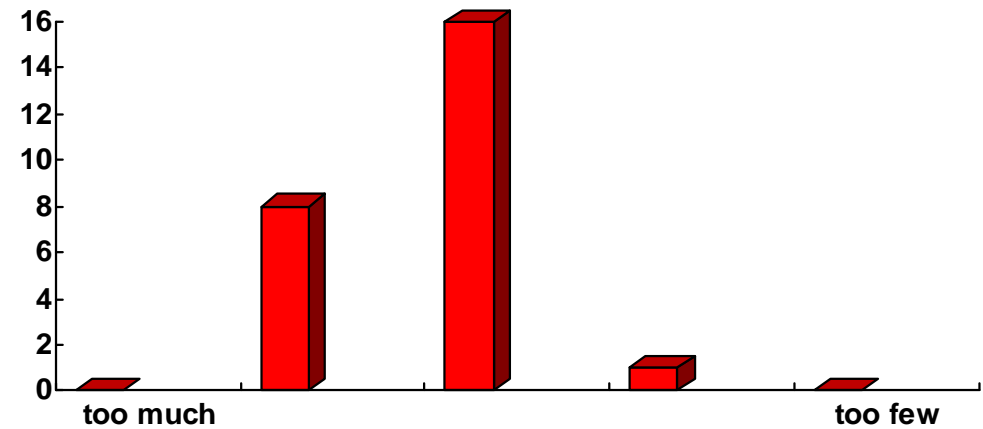


**4. Functional profit:
Had the use of tools
relevance for better
understanding the
theoretical concepts
from the lecture?**

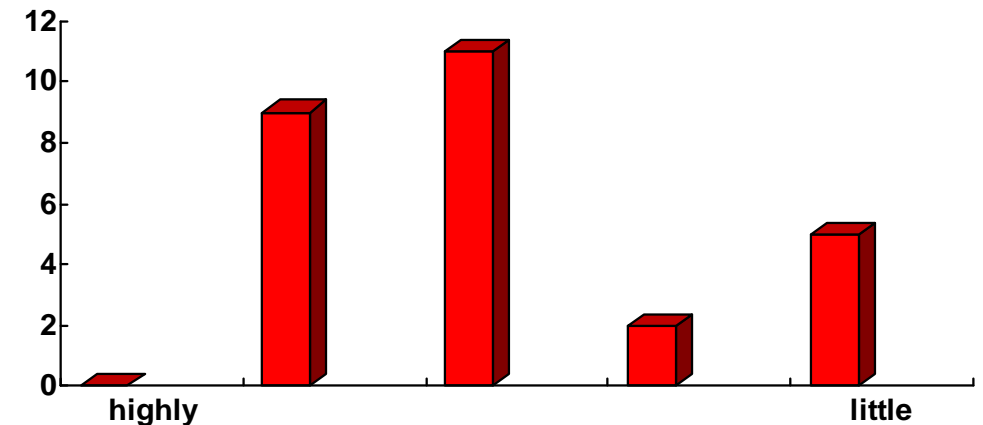


Generally questions

5. Was the use of software tools time-consuming?

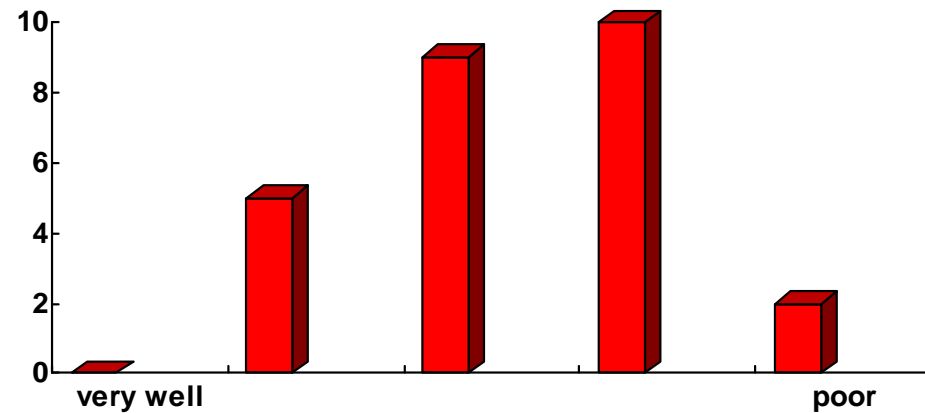


6. Fun: Did you use the tools with fun?



Generally questions

7. Operability: How did you experience the tool operability in general?



A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

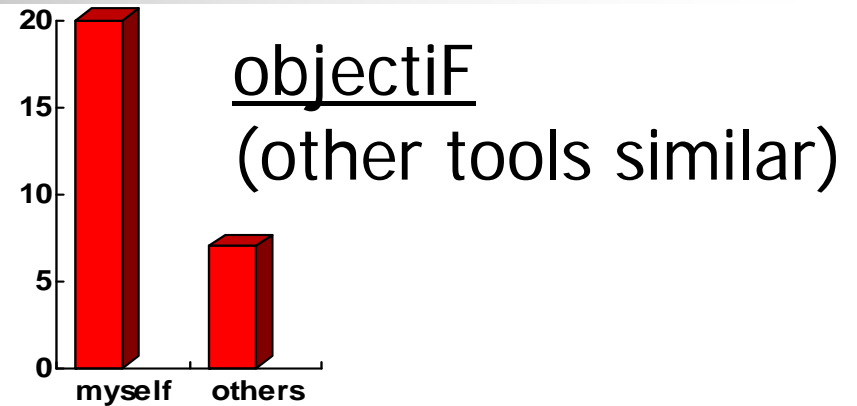
Individual questions

Similar questions for each individual tool:

- Technical profit: working with CCCC, CTE, SOTA, ATOS, Z/EVIS, objectiF
- Fun: working with CCCC, CTE, SOTA, ATOS, Z/EVIS, objectiF
-

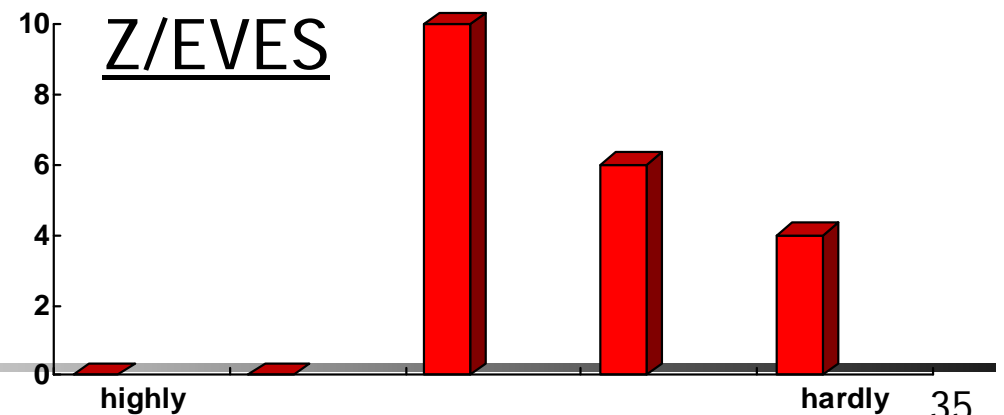
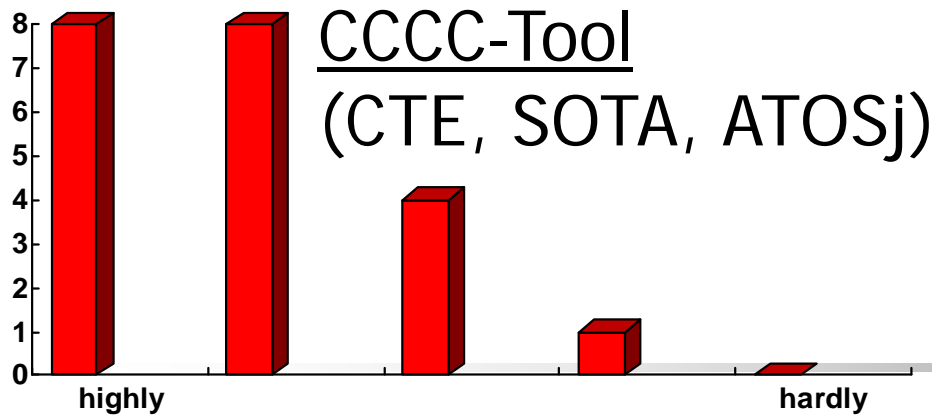
Individual Questions

0. Who did work with the tool?



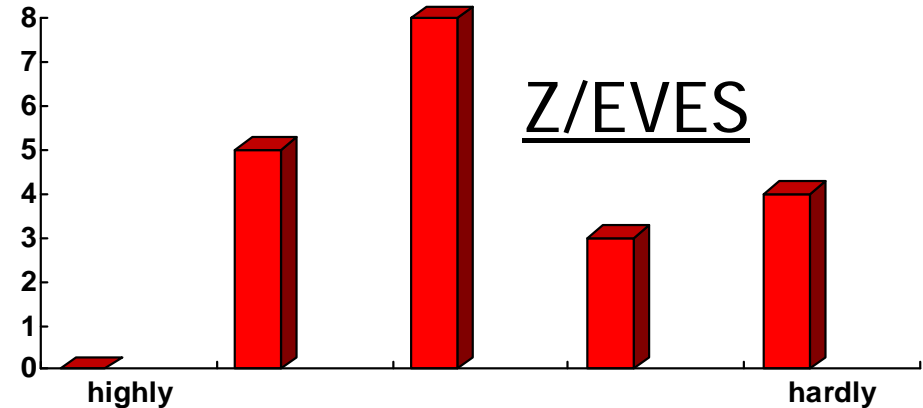
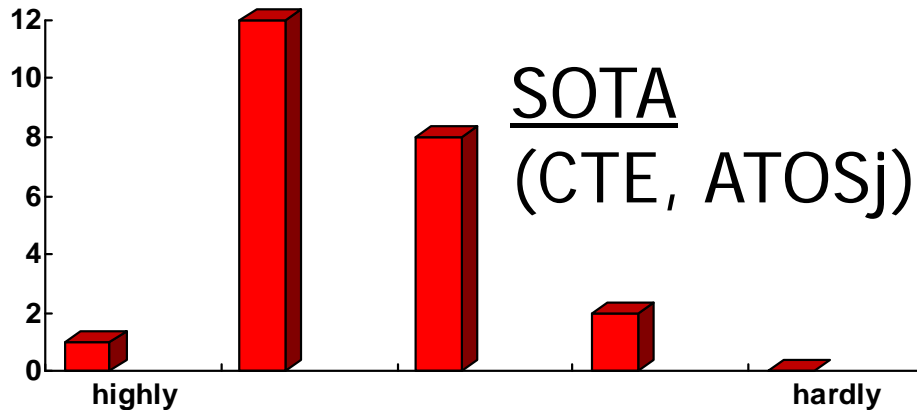
1. Technical profit:

Did the use of the tool have relevance for solving the assignments (effect on efficiency)?

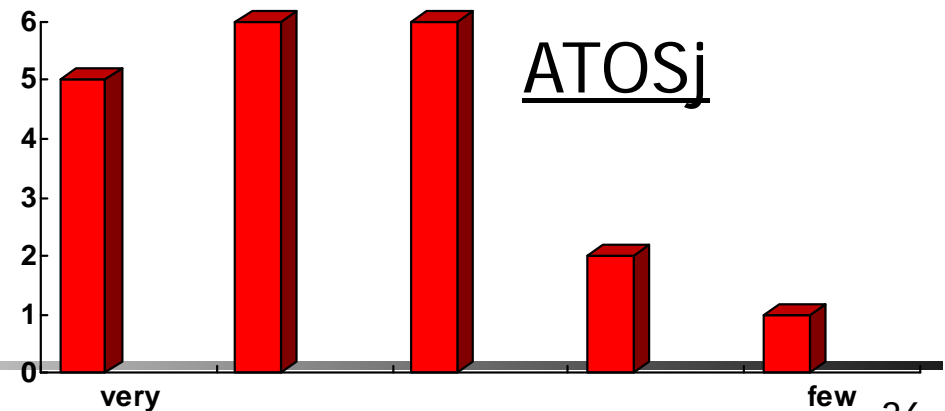
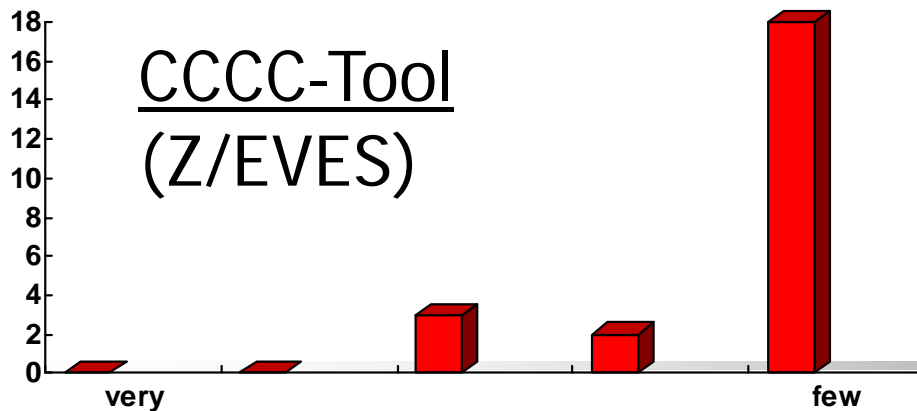


Individual Questions

2. Relevance: Is the using that tool important for your professional career?

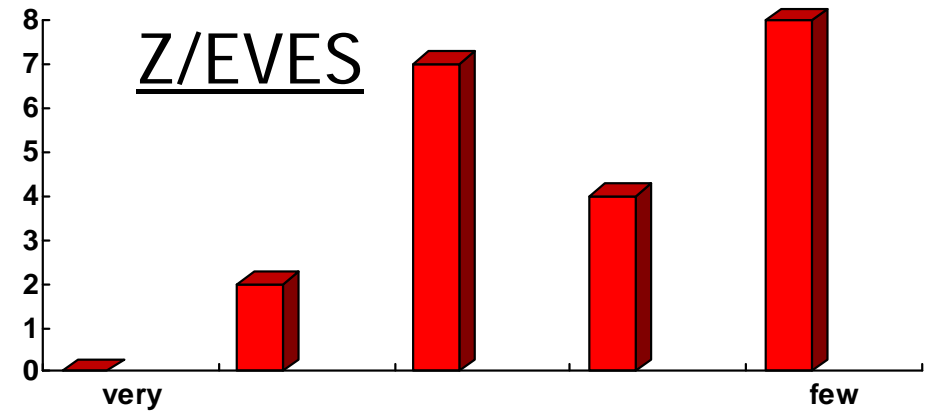
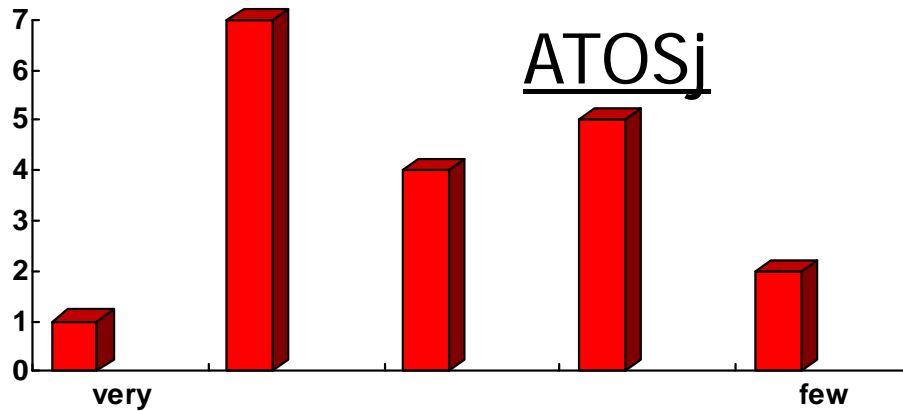


3. Was the using of software tools time-consuming?

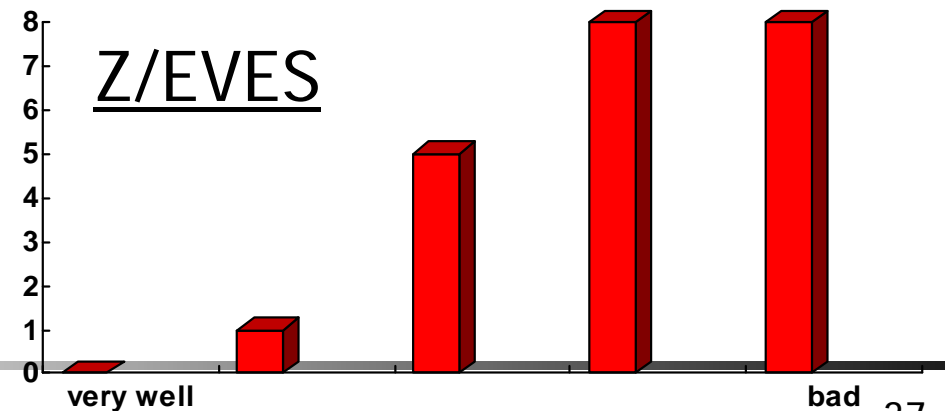
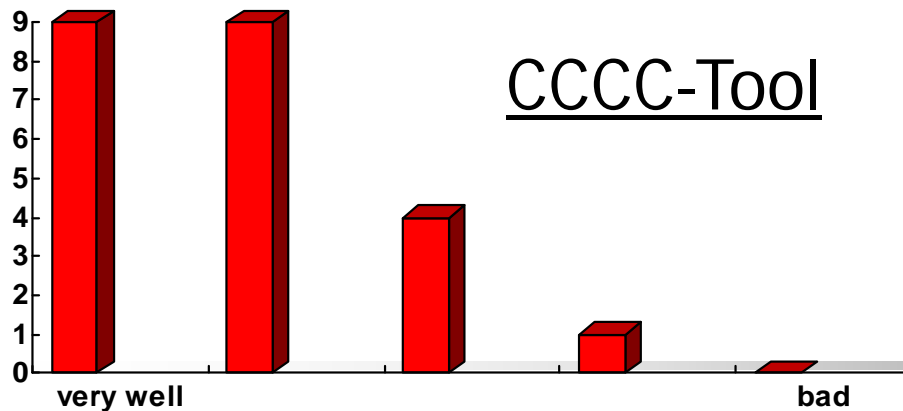


Individual Questions

4. Fun: Did you use the tool with fun?

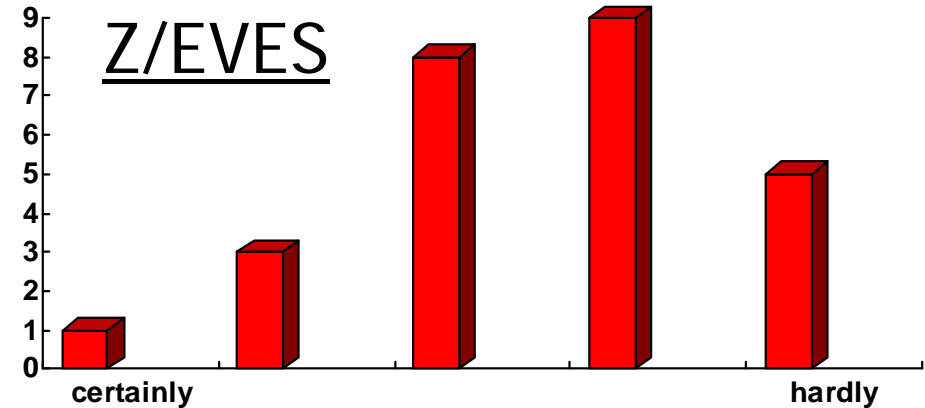
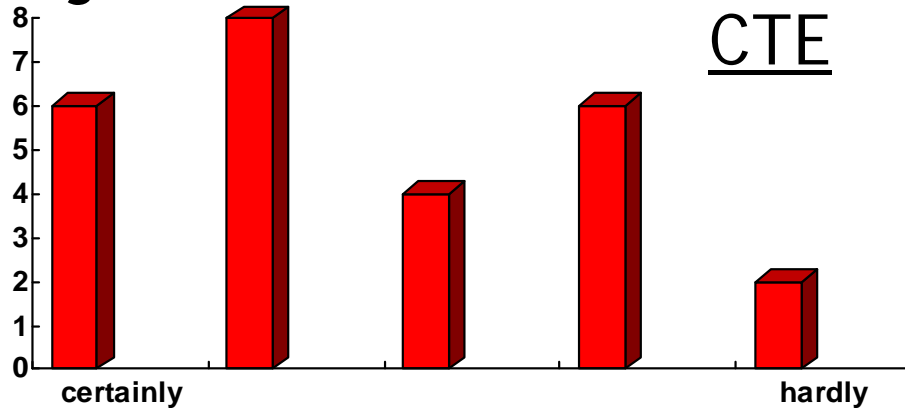


5. How did you experience the tool operability?

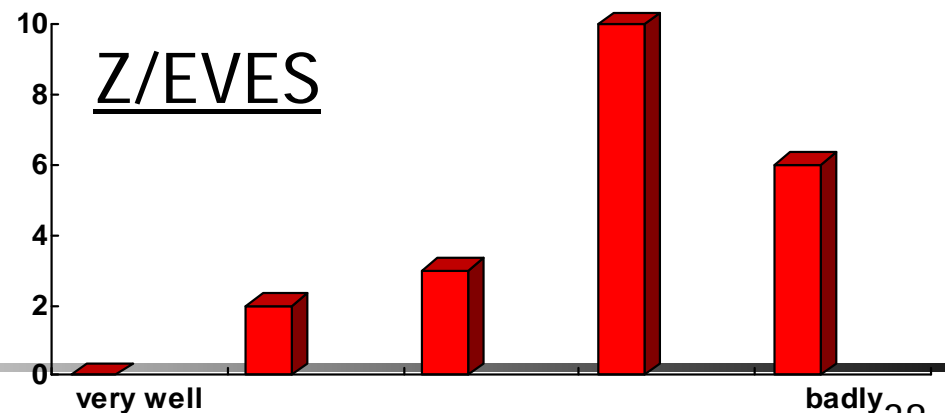
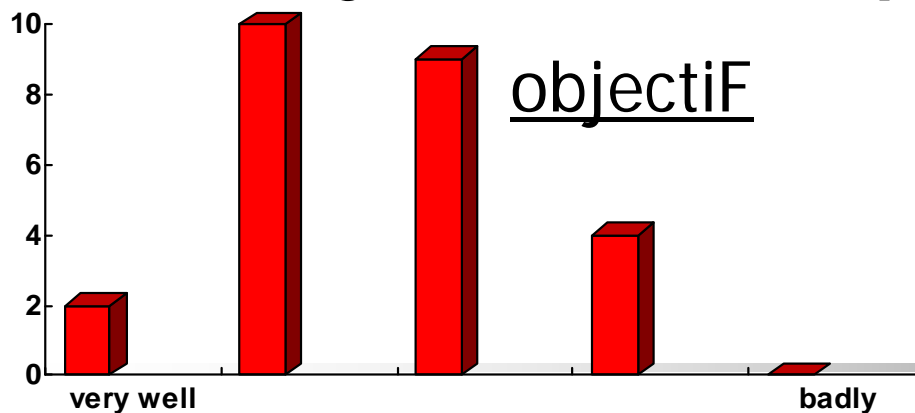


Individual Questions

6. Sustainability: Would you use that tool to realize projects in the future?



7. How do you assess the performance of that tool?



A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Students comments

- Other tools also useful: Junit, SVN (subversion)
- Tools mentioned in job offers should be selected
- Widely used tools should be taken instead of in-house products (HU: ATOS, SOTA)
- Some tools were applicable only under Windows (students partly works with Linux)
- More time necessary for the installation than for use in the assignment
- Assignments with tool usage were well-prepared

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Contents

- Tools in SE course at HU (Overview)
- In detail: assignment – tool support
- Students feedback
- Summary

Summary

- Tools are necessary for lively lessons in SE
- Working in teams is valuable for preparation of professional work
- Software engineering requires the using of models and tools
- Efforts for the staff:
 - installation of tools in computer lab
 - information on tools at website
 - about tool installation, usage and downloads
 - assignments: solved by staff before

A decorative graphic consisting of overlapping yellow, red, and blue squares with a black crosshair.

Summary

Thank you for your attention!