Do students like tool usage in a Software Engineering course?

Michael Ritzschke, Olga Schiemangk, Klaus Bothe
Contents

- Tools in SE course at HU (Overview)
- In detail: assignment – tool support
- Students feedback
- Summary
Which tools did we use in 2009?

<table>
<thead>
<tr>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Review requirements specification “SemOrg”</td>
</tr>
<tr>
<td>2. Function points</td>
</tr>
<tr>
<td>3. Develop an OOA model</td>
</tr>
<tr>
<td>4. Formal specifications</td>
</tr>
<tr>
<td>5. Metrics</td>
</tr>
<tr>
<td>6. Select test cases functionally by the CTE</td>
</tr>
<tr>
<td>7. Select regression test cases by ATOS</td>
</tr>
<tr>
<td>8. Test coverage with</td>
</tr>
</tbody>
</table>
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.
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 – 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:
CTE XL
Efficiently generating Systematic Test Specifications

The classification-free editor CTE XL is an effective tool for applying the classification-free 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-free. 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)

systematic-testing.com
www.systematic-testing.com

News
18.10.2007: CTE/XL V1.8 released
download here

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 - 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/~tittlef](http://www.chs.ecu.edu.au/~tittlef).

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](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/ccc](http://sourceforge.net/projects/ccc) 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 [ccc-3pre48.tar.gz](http://sourceforge.net/projects/ccc/ccc-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](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.
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.
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

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Theme</th>
<th>Beginning</th>
<th>Delivery</th>
<th>Evaluation</th>
<th>Tool</th>
<th>Demo. in Lec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment 1</td>
<td>Review Requirements specifications</td>
<td>21.04.09</td>
<td>11.05.09</td>
<td>20.05.09</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Assignment 2</td>
<td>Function point method</td>
<td>04.05.09</td>
<td>18.05.09</td>
<td>27.05.09</td>
<td>objectIF</td>
<td>-</td>
</tr>
<tr>
<td>Assignment 3</td>
<td>OOA model</td>
<td>11.05.09</td>
<td>03.06.09</td>
<td>10.06.09</td>
<td>Z/EVES</td>
<td>11.05.09</td>
</tr>
<tr>
<td>Assignment 4</td>
<td>Formal software specification</td>
<td>18.05.09</td>
<td>10.06.09</td>
<td>17.06.09</td>
<td>CTE</td>
<td>25.05.09</td>
</tr>
<tr>
<td>Assignment 5</td>
<td>Classification tree method</td>
<td>25.05.09</td>
<td>15.06.09</td>
<td>01.07.09</td>
<td>SOTA</td>
<td>03.06.09</td>
</tr>
<tr>
<td>Assignment 6</td>
<td>Test coverage</td>
<td>03.06.09</td>
<td>22.06.09</td>
<td>01.07.09</td>
<td>ATOSj</td>
<td>10.06.09</td>
</tr>
<tr>
<td>Assignment 7</td>
<td>GUI oriented regression test</td>
<td>10.06.09</td>
<td>29.06.09</td>
<td>15.07.09</td>
<td>ccccc</td>
<td>-</td>
</tr>
<tr>
<td>Assignment 8</td>
<td>Metrics</td>
<td>15.06.09</td>
<td>06.07.09</td>
<td>15.07.09</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
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
  - User's guide
- SOTA
  - Installation hints
  - User manual (pdf, 1.9MB)
  - Setup of test environment for SCTA assignment (HUSEmOrg)
  - Setup of test environment for sample program
  - Setup of test environment for sample program
- ATOS
  - Installation hints
  - User manual (pdf: one slide per page; two slides per page)
  - HTS language specification (pdf-file)
  - Project setup for HUSEmOrg

Z-SPECIFICATION: Z/EVES
General activities of students to use tools

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

objectiF: sample solution
Students get the description of a queue

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>create</td>
<td>generates an empty queue</td>
</tr>
<tr>
<td>extend</td>
<td>appends a new element at the end</td>
</tr>
<tr>
<td>get_first</td>
<td>affords the first element</td>
</tr>
<tr>
<td>delete</td>
<td>cuts the first element</td>
</tr>
<tr>
<td>not_empty</td>
<td>tests, if the queue is not empty (empty queue affords false)</td>
</tr>
</tbody>
</table>

Tasks: Describe the queue
a) use the Z-notation
b) algebraic specification
Tool support: Z Mini Editor, Syntax check

```
Queue

queue_elements : seq Z

new

\Delta Queue

queue_elements' = \langle \rangle

is_empty

exists Queue

msg! : \{true, false\}

msg! = true \iff queue_elements = \langle \rangle

enqueue

\Delta Queue

inp? : Z

queue_elements' = queue_elements \cup \langle inp? \rangle
```
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

Properties
- Standard
- Violations
- TCSpecific

Name
ID = 22
client books course

Description
Text:
a well known client books a course, which will be held

Buchung

Kunde
Neukunde
Veranstaltung
findet statt
findet nicht statt
Kundensachbearbeiter

Zahlungsmoral
Mitarbeiter einer Firma?
ja
nein

Firmendaten vorhanden?
ja
nein

Grund
existiert nicht
ausgebucht
fällt aus

Bestandskunde
client books course
Testcase 2
Testcase 3

Neukunde
Testcase 4
Testcase 5
Testcase 6
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
Tool support: Evaluation the test runs (different metrics, graphical view of instruction coverage, ...)

SOTA

```java
private ClientWindow(Shell parent) {
    // Shell instance used as parent shell of the window.
    // ...
    Shell shell = new Shell(parent, SWT.CLOSE);
    // ...
    shell.add Listener(SWT.Close, new Listener() {
        public void handle Event(Event event) {
            if (input Changed) {
                if (confirm Close) {
                    if (input Editing Enabled && input != null)
                        Client.lockedIds.remove(new Integer(input.getId()));
                    } else
                        event.doit = false;
                } else if (input != null && input Editing Enabled)
                    Client.lockedIds.remove(new Integer(input.getId()));
            } else
                // ...
        }
    });
}
```
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. seminar 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
Tool support: Replay the test sequence and check, if the functionality in a later release has changed.
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

<table>
<thead>
<tr>
<th>Metric</th>
<th>Tag</th>
<th>Overall</th>
<th>Per Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lines of Code</td>
<td>LOC</td>
<td>43</td>
<td>****</td>
</tr>
<tr>
<td>McCabe’s Cyclomatic Number</td>
<td>MVG</td>
<td>57</td>
<td>****</td>
</tr>
<tr>
<td>Lines of Comment</td>
<td>COM</td>
<td>6</td>
<td>****</td>
</tr>
<tr>
<td>LOC/COM</td>
<td>L_C</td>
<td>7.167</td>
<td></td>
</tr>
<tr>
<td>MVG/COM</td>
<td>M_C</td>
<td>9.500</td>
<td></td>
</tr>
</tbody>
</table>

Functions

<table>
<thead>
<tr>
<th>Function prototype</th>
<th>LOC</th>
<th>MVG</th>
<th>COM</th>
<th>L_C</th>
<th>M_C</th>
</tr>
</thead>
<tbody>
<tr>
<td>error( int, int, int )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>definition Error.java:8</td>
<td>34</td>
<td>57</td>
<td>3</td>
<td>11.333</td>
<td>19.000</td>
</tr>
</tbody>
</table>
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?
7. Operability: How did you experience the tool operability in general?
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)?

- **objectiF** (other tools similar)
- **CCCC-Tool** (CTE, SOTA, ATOSj)
- **Z/EVES**
Individual Questions

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

![Graph showing relevance for SOTA (CTE, ATOSj)]

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

![Graph showing time-consuming for CCCC-Tool (Z/EVES)]
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?
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
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
Thank you for your attention!