**Introduction:**

- Motivation
- Testing
- the Plan
- Tools
- what we did
- What could we not do?
- Reasons
- Theme versus Actual Semester Goal
- Conclusion1
- Timeline
- Conclusion2

**Motivation**

Why is testing important?

- Stable Product
- Tracking effects of code changes
- Independant quality ensurance

Methods

- Repeatable Tests
- One time Tests
- Automatic tests

**Testing:**

What is Testing?

The two main aims to be achieved are

- to check whether final product satisfies the specification,
- to help the programmers in that way.

It was already clear to all of us from the start that aiding the development is as essential as quality assurance.

**The Plan:**

The testers thought to get real detailed specification. This was should have been a general specification about the whole project and even more specified ones for the single projects PHP1/2 & Java.

The plan was to test different stages of the groups&cute; implementations e.g. the db-access, the data handling, the communication between components and on top the practical tests of the product when the development is close to the end.

Unfortunately we did not get detailed specification.

After spotting the web we discovered some interesting tools, ...

**Tools:**

..., that seemed to be potentially useful in praxis, but possibly will not help in the actually algorithmic not very demanding CoMa. In order to learn something we wanted to test these tools anyway for demonstration an educational purposes. After the project we got confirmed that the tools did not help.

**Tools in use**

Web Application Tests

- Puretest [Thiago]
- TestMaker [Olle]

Unittests

- PHPUnit [Oliver]
- jUnit [Olle]

Test (-Group-) Coordination

- bugzilla

**Testmaker:**

provides

- web application tests (utilized but buggy respectively not well developed)
- web service tests based on SOAP standard
- template-creator for jUnit testcases (no big help)

(demonstration)

**Puretest:**

Similar tool + supports phpUnit tests.

- Both tools are platform independent. (In fact even practically!)

**jUnit:**

provides

"test framework"

- (some) GUIs
- techniques to organize and manage a whole lot of tests

**Bugzilla:**

Although hated in the beginning - it is (can be) a great tool/help if it is well configured. But a not so bad configured bugzilla is already useful. We don't know an alternative solution being capable of providing such bug management.

**What we did:**

We all did a lot of function tests on the (nearly) ready CoMa. Although not aided by web application tests the functionality of what works could be tested for proper operation.
Bugs are mainly detected in this way, and that shows, how easy in fact this task (CoMa) was.
What we did together in our testgroup: When the "real work" began there was not much to do or plan within the scope of the test group.

**What we could not do:**

Due to lack of precise specification we actually did not take noteworthily part in the *active* implementation of CoMa. We were only tester and no helper, in contrast to our goal.

**Reasons for Problems**

Specification

- was changed,
- was not global
- was not authoritative
- was not detailed enough.

Foresight

- awareness of problems and necessities
- general idea: 'Testers start delayed'

**Specification**

| Programming in the Many | Earning a certificate |
|---|---|
| • Interaction of modules<br>• Definition of classes<br>• Definition of functions<br>• Definition of interfaces | • working product |

**Conclusion**

Specification

- Binding
- Detailed
- Global

Advice

- Dependencies
- Scheduling

**Timeline**

- Specification -> First half of December (No Testing)
- First code -> Begin 2005 (Some Testing)
- Basic functionality -> End of january (Heavy Testing)
- Beta Version -> 8. February (Testing not ended)

**Conclusion**

- Specification >> Deadline 1.12.2004
- Beta version >> Deadline 1.1.2005
- Stable final version >> Deadline 8.2.2005
- Adjust certificate conditions