Using Unit Testing to Drive Effective Programming

Martin Lippert
Software Engineering Group
Computer Science Department
University of Hamburg &
APCON Workplace Solutions GmbH
http://www.jwam.de/
lippert@acm.org

Frank Meyer
UBS AG
www.ubs.com
frank.meyer@ubs.com

Keywords:
Unit Testing, Frameworks, eXtreme Programming

Abstract:
Framework development is a challenging task and a powerful approach to the development of enterprise critical and highly innovative software. Frameworks facilitate the work of application developers by encapsulating complexity. Therefore the quality of frameworks is essential. But how can we reach this extremely high quality?
We use several techniques to improve the quality, some of them are well known by literature and approved by many developers (like Design by Contract), other are relative new. One of the more recent ones is eXtreme Programming, from which we use a selection of techniques. We use aggressive refactoring and short development cycles as well as continuous integrations more than once a day. We have gained a great flexibility using these techniques. But there is also the danger of doing something wrong. You might change one small part of the system which affects another part but you are not aware of this dependency. So you could change some part of the system without knowing whether the system is still working after your change or not.
Hence we need the possibility to check the complete system, our framework - after every change. It makes no difference whether the change is small or big, affects only one class or a complete package. And the system check must be fast, extremely fast, because we have to do it many times a week, twice or more a day or sometimes after one hour of development. If we could check the complete system after every little change we can be sure that we have a running system every time. The problem is, that the framework is of considerable size. It consists of more than 850 classes and we are not able to test the complete system by manual tests.
Aggressive Unit Testing provides the best matching techniques for these challenges. We can use unit testing to assure the correctness of the complete framework automatically and quickly. In our context automatically means, that the tests run automatically and the developer has nothing to do but to start a tool to test the complete framework. But the test cases are not automatically generated from the source code. The developer has the responsibility to write the test cases. Ideally he first writes the test class, then the class itself.

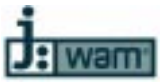
In our talk we describe our experiences with unit testing and what kind of benefits we have gained from using this technique. We discuss the pros and cons of using unit testing for the framework development for large scale software development.

Background:

Martin Lippert works as a framework architect in the software engineering group at the University of Hamburg which developed the JWAM framework, a Java framework for large scale interactive software. In parallel he works as consultant and software architect at APCON Workplace Solutions GmbH. He his part of the development team building the professionalized version of the framework. They use the XP techniques since the beginning of 1999 and have observed great improvements in quality (understandability, simplicity, correctness, robustness etc.) since then. They call the usage of XP techniques for framework development and application „eXtreme Frameworking“ (XF).
Frank Meyer works at UBS AG in a team, which develops a smalltalk framework for client applications in the banking business. The framework emerges in close collaboration with the application developers, that means every failure in the framework interferes with their work, so testing is essential. When he joined the team, he implemented the SUnit testing framework and integrated it with the existing testing tool "Test Mentor". He has established a test culture by providing an easy to use and supportive environment for testing

Literature

- *The JWAM-framework*: http://www.jwam.de (in English)
- Kent Beck: *Extreme Programming Explained - Embrace Change*. Addison-Wesley Pub Co; ISBN: 0201616416
- Kent Beck, Erich Gamma: *Testing Framework*. http://www.armaties.com/testfram.htm
- Martin Fowler: Refactoring: *Improving the Design of Existing Code*. Addison-Wesley Pub Co; ISBN: 0201485672
- Robert Binder: *Testing Object-Oriented Systems: Models, Patterns, and Tools*. Addison-Wesley Pub Co; ISBN: 0201809389
- Bertrand Meyer: *Object-Oriented Software Construction*. Second Edition. Prentice Hall. New Jersey. 1997.
- WiKi-Web: http://c2.com/cgi/wiki.
- XP-Newslist: http://www.egroups.com/group/xp-forum
- SilverMark‘s Test Mentor: http://www.silvermark.com/



Using Unit Testing to Drive Effective Programming

ICSTEST 2000
International Conference on Software Testing

Martin Lippert

Software Engineering Group
Computer Science Department
University of Hamburg
&
APCON Workplace Solutions GmbH

lippert@jwam.de

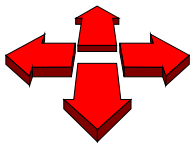
Frank Meyer

UBS AG

frank.meyer@ubs.com

UBS

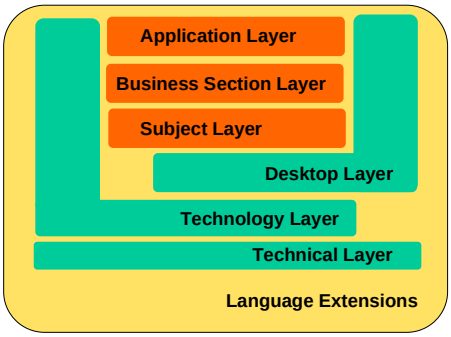
Overview



- **Context**
- **Characterictics of frameworks and components**
- **Design techniques to gain high quality of code**
- **Parts of eXtreme programming**
- **Test Infected:**
  - Refactoring
  - How to find test cases?
  - Continuous integrations
  - Collective Code Ownership and Pair Programming
- **How to establish automated class tests?**

Context I: Development of the JWAM framework

- **What is the JWAM framework**?
  - Application-oriented framework for the *Tools-&-Materials*-approach.
  - Flexible architecture and technology for changing environments.
  - Provides a component infrastructure for scalability: small framework kernel and powerful components extending the framework.
  - Developed by a research group at the University.
  - Further professional development provided by APCON Workplace Solutions.
  - Used within several professional projects
  - Version 1.4
    kernel: 107 classes
    components: 757 classes

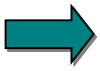


Context II: Development at UBS

- **Who is UBS**?
- **Applications for credit centers**
- **Architecture and IT Environment**
  - Service architecture with CORBA (ISI)
  - C++ Server (integrates services and legacy systems)
  - Smalltalk Client (fat)
- **Client Framework**
  - white box framework
  - main parts: base classes for models and views, interface to the server
  - evolved from a WAM framework
  - used by one main and some smaller addional applications
  - developed in parallel with the applications

Characterictics of frameworks and components

- **The quality** of a framework/component is essential.
- Frameworks and components are **extremely innovative** (no old and good known technology).
- They are part of **mission critical** applications.
- They **facilitate the work of application developers** by encapsulating complexity.



- **eXtreme Programming** is suitable for this kind of projects!
- **Unit Testing** is one of the most important techniques we use to assure high quality!

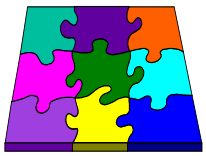
Design techniques to gain high quality of code

We use **several design techniques** to assure a high quality of the code:

- Design by Contract.
- Disciplined exception handling.
- Application oriented usage of inheritance.
- Parts of eXtreme Programming.
- Design Patterns
- Periodical reviews of the framework kernel.
- Reviews of the JWAM-components before they are admitted to the framework package.

- **AND**: a simple but perfected **testing environment !!!**

Parts of eXtreme programming



- **Lifecycle**: Listening, Testing, Coding, Design
- Continuous **Refactoring**
- **Pair-Programming** for the JWAM kernel and all important components
- short **integration** cycles
- Keep it simple!
- You aren‘t gonna need it!
- Planning Game

- and **Testing**

Test Infected

- Write a test class for every class inside the framework.
- Ideally, first write the test class, than the class itself.
- Use inheritance for testing:
  - Tests written for a superclass will also be executed for all subclasses.
  - Subclasses have to define tests for new behavior and different usages
  - To test abstract framework classes, you need concrete implementations: implement simple subclasses only for testing.
- Provide testdriver and testdata through the test framework for the application developers.
- The test should be automated.
- Short testing cycles. The minimum is, that at the end of the day all test cases of the framework run without problems.

➔ **We have a test class for every class inside the JWAM framework.**

➔ **Tests for the framework can be reused for the applications.**

➔ **They can be processed automatically by a small test tool.**

Test Infected: How to find test cases?

- First write the test class, than the class itself: This paradigm leads to specify the expected behavior of the class without analysing the class itself.
- Invoke every operation of the class at least once.
- Look at state diagrams: at least one test for every state transition
- Boundary value analysis
- Coverage analysis
- Every time you find a bug in the class: first extend the test class to find the bug, than correct the class itself.

Test Infected: Refactoring

- It is not a shame to write bad code.
- But it is a shame not to change it.
- The old guideline: „Never change a running system“ leads to huge problems and should be given up.
- To restructure a piece of software keeping the old behavior is called: „Refactoring“.
- You can control whether the new behavior is the same as the new one with regression tests.
- The scope of the software part to be restructured should be chosen as small as possible to enable the restructure process to be happened within a few hours.

What we do:

➔ Continuous refactoring of the JWAM framework.

➔ **This is possible only because of the unit tests!!!** They ensure that the refactoring did not influenced other parts of the framework and did not introduce new bugs into the framework.

Test Infected: Continuous integrations

- Changes to the framework are integrated many times a day.
- At the end of every integration cycle the framework version of the integration machine is running. (at least at the end of the day.) Why?
  - **Every member of the team profit from the changes very fast.**
  - New code can be tried out and examined by the other members right after the integration.
  - The risk of double development gets minimized.

What we do:

➔ Most of the coding tasks are small enough to finish them within less than one day.

➔ Every coding task is integrated into the framework version immediately after the development.

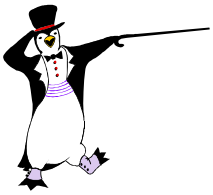
➔ **This is possible only because of the unit tests!!!** They ensure that the new code pieces did not influenced other parts of the framework and did not introduce new bugs into the framework.

Test Infected: Collective Code Ownership and Pair Programming

- The code (and all other development documents) belongs to the team and not to a single person.
- Every developer is able and is allowed to change everything every time.

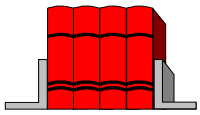
- Make use of the Pair Programming idea.
- Along with that we have a continuous review at coding time.
- It leads to a significant higher quality:
  - reduced number of bugs
  - better class interfaces
  - less redundancies

How to establish automated class tests?

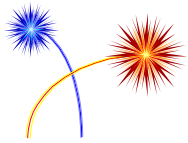


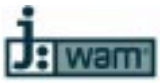
- Provide a test framework (for example JUnit).
- Support the automatic testing with tools.
- Integrate the tests into the development environment.
- Define development guidelines and check the adherence.
- Test Reviews.

Literature and more ...



- *The JWAM-framework*: http://www.jwam.de (in English)
- Kent **Beck**: *Extreme Programming Explained - Embrace Change*. Addison-Wesley Pub Co; ISBN: 0201616416
- Kent **Beck**, Erich **Gamma**: *Testing Framework*. http://www.armaties.com/testfram.htm
- Martin **Fowler**: *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Pub Co; ISBN: 0201485672
- Robert **Binder**: *Testing Object-Oriented Systems: Models, Patterns, and Tools*. Addison-Wesley Pub Co; ISBN: 0201809389
- Bertrand **Meyer**: Object-Oriented Software Construction. Second Edition. Prentice Hall. New Jersey. 1997.
- WiKi-Web: http://c2.com/cgi/wiki.
- *XP-Newslist*: http://www.egroups.com/group/xp-forum
- *SilverMark's Test Mentor*: http://www.silvermark.com/





Thank you for your attention!

If you have any comments or questions or if you would like to know more about the JWAM framework, please refer to:

Martin Lippert

Software Engineering Group
Computer Science Department
University of Hamburg
&
APCON Workplace Solutions GmbH

lippert@jwam.de

Frank Meyer

UBS AG

frank.meyer@ubs.com