

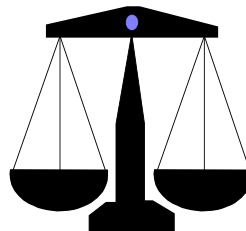
# Test-driven Development aus dem Open Source Regal

Christian Dedek (dedek@oio.de)  
Sabine Winkler (winkler@oio.de)

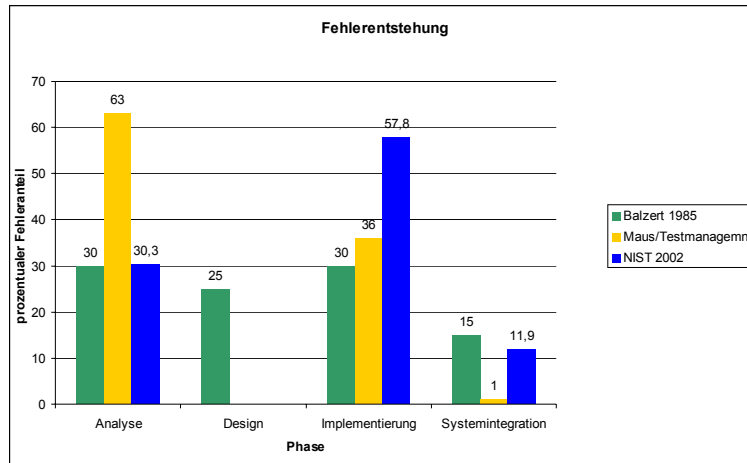
Orientation in Objects GmbH  
Weinheimer Str. 68  
68309 Mannheim  
www.oio.de

## Qualitätskosten ein Minimierungsproblem

- Softwarekosten = Herstellungskosten + Qualitätskosten
- Qualitätskosten = Qualitätssicherungskosten + Fehlerkosten



## Fehlerentstehung



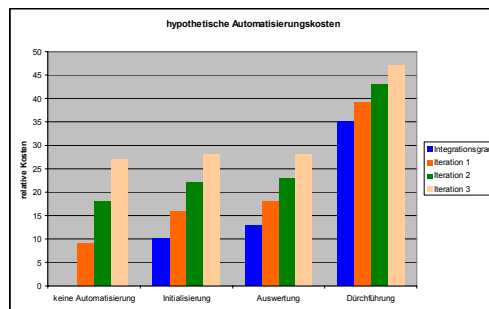
## Der Hund und Testen ?

- Der Hund kackt im Jahr etwa das 3,5-fache seines eigenen Körpergewichts.
- Die Aussage ist frei erfunden.
- Möglicherweise liegt sie gar nicht so schlecht.
- Einen Nachweis habe ich nicht gegeben.
- Die meisten Bücher über das Testen sind voll von Äußerungen solcher Qualität.



## Ökonomische Gründe zum Werkzeugeinsatz I

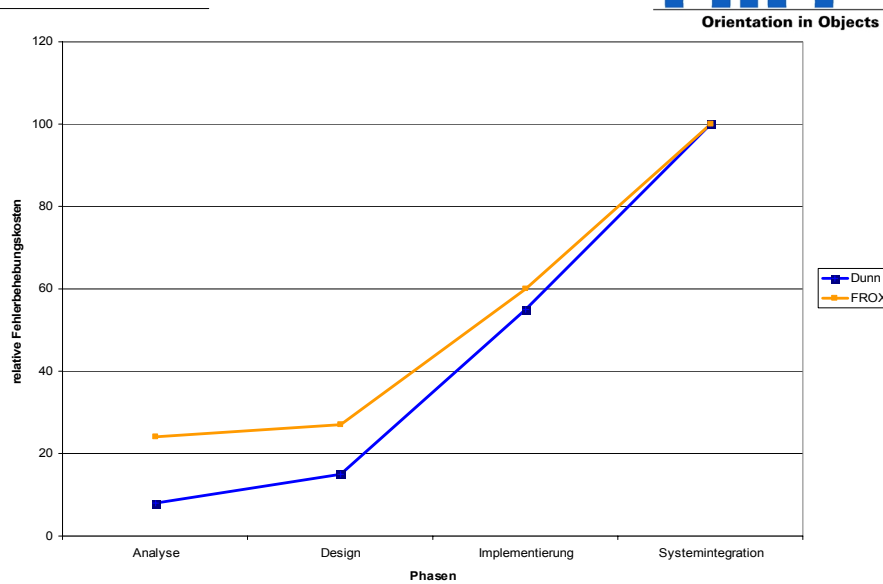
- „Bereits im Anwendungsentwicklungsprojekt lassen sich durch TPA bis zu 10 - 15% der Gesamttestkosten einsparen“ [SQS]
- „Bei Vorgehensmodellen mit laufend wiederkehrenden Testausführungen amortisiert .. nach 4 bis 12 Retests“ [SQS]
- Kostenreduktion durch Engineering - „Testware“
  - Wiederverwendung
  - Kapselung
  - Modularisierung
  - Integration



Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## Ökonomische Gründe zum Werkzeugeinsatz II



Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## Die Vorgeschichte eines OS-Werkzeugs



- Idee
  - „*Every good work of software starts by scratching a developer's personal itch.*“ - Eric Raymond
- 1. Version
  - „*It's fairly clear that one cannot code from the ground up in bazaar style. One can test, debug and improve in bazaar style, but it would be very hard to originate a project in bazaar mode.*“ - Eric Raymond

## OIO Tooling philosophy



1. Define clearly what is a tools job.
2. There is always a tool doing the job right. A project team has to find out, how to use it right.
3. If no tool doing the job right could be found by the team, the extension of a tool found during the evaluation is the solution.
4. Integration of tools doing their jobs right is the key to lasting success.

## XP - the central principles



- Fast and detailed feedback
  - test driven design with acceptance tests
  - Customer On-site
  - pair programming
- Common understanding
  - planning game
  - simple design
  - System metaphor
  - collective code ownership
  - coding conventions
- Continuous process:
  - continuous integration
  - refactoring
  - frequent & small releases
- Developer Welfare
  - a week has 40 hours

Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## Strategien im Qualitätsmanagement



- Bottom-up
  - einfach in die Entwicklung zu integrieren(test first ?)
  - Unterstützung durch Unit-Frameworks
  - Anlehnung an andere Ingenieurwissenschaften



- Top-down Ansätze
  - in Form von C/R-Ansätzen
  - ineffiziente Bsp. bekannt
  - in anderen Ingenieurwissenschaften teilweise sehr spektakulär und teuer



Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## „Regal“-Kategorien



- Funktionaler Test
  - Komponententest
  - integrierter Test
  - Systemtest
- Nicht-funktionaler Test

Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## Lebenszyklus eines Unit Tests



- Erzeugung eines oder mehrerer Objekte
- Objekte in definierten Anfangszustand bringen
- Ausführung einer Folge von Methoden, deren Wirkung getestet werden soll
- Endzustände, bzw. Wirkung auf Umwelt wird überprüft



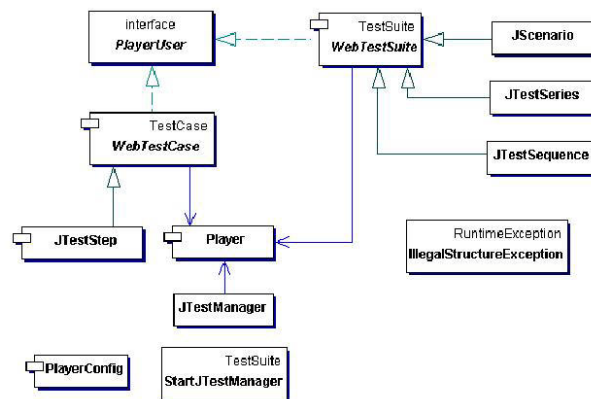
*keep the bar green to keep the code clean...*

Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## Testdriver

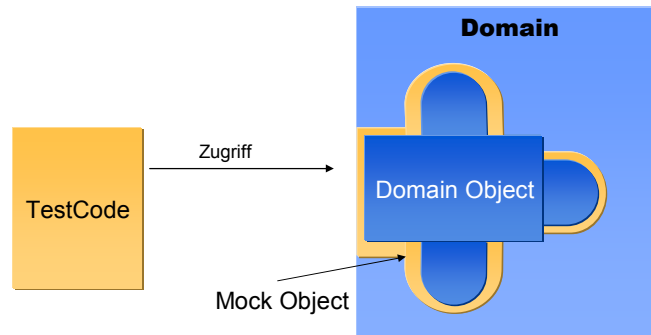
- JUnit Framework (HttpUnit - Schritt I)



## JUnit Decorator

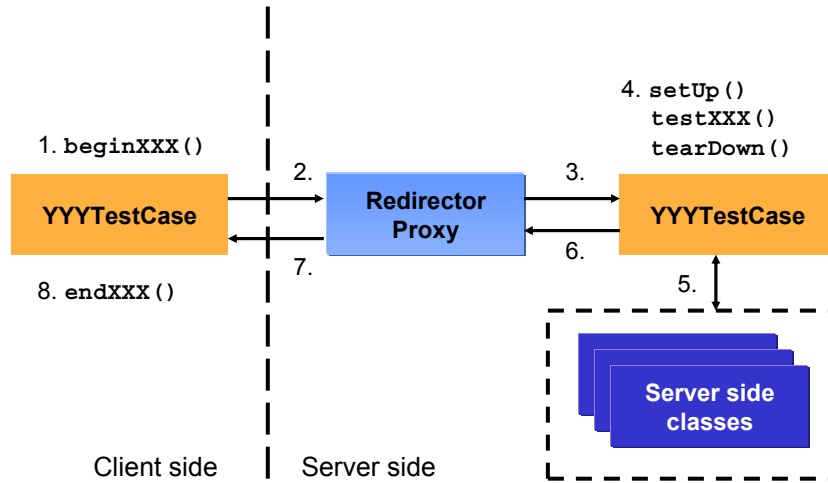
- JUnitPerf
  - TimedTest Sicherstellung des Maximalen Zeit des Zeitverbrauchs eines Testfalls
  - LoadTest Simulation mit Threads

## Mockobjects



## Mock Objects

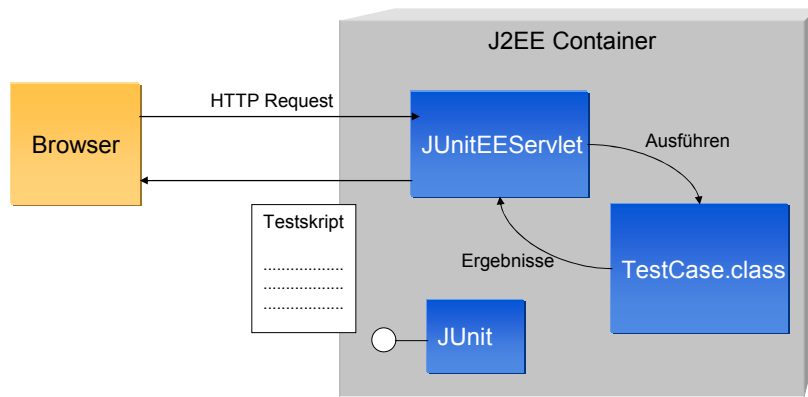
- früheres Testen von integrierten Objekten
- Abtrennung Komponente Container in Testumgebung möglich
- Anwendungsdesign verändert sich (BIT vs. lose Kopplung)
- Einsparungen durch Codegeneration von Mock-Klassen aus schnittstellen - Funktionalität der Mock-klassen sind Testkosten
- Kopplung zwischen Mocks vermeiden, funktionsreiche Mocks umstritten
- Toolkits: MockCreator, Mockmaker, MockObjects



- Test von Java Komponenten in der VM des Web Containers
  - z.B. Servlets, JSPs, Filter, EJBs (über local interfaces), Tags
- Integrations- und Systemtests
- Spezieller Ant Support



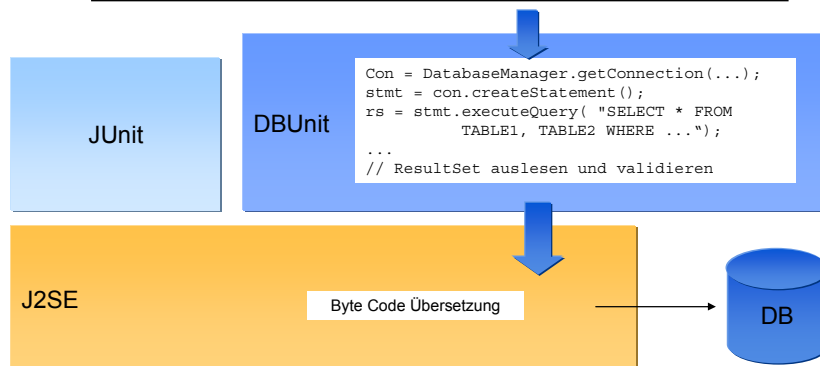
# JUnitEE



# DBUnit

## DBUnit Testskript:

```
ITable actual = getConnection().createQueryTable("RESULT_NAME",  
"SELECT * FROM TABLE1, TABLE2 WHERE ...");  
Assertion.assertEquals(expected, new  
CompositeTable(expected.getTableMetaData(), actual));
```



## DBUnit



- Ablage von Testdaten in XML
  - Ausgangsdaten des Systems
  - Erwartungswerte des Systems
- einfache Abfragen und Updates der relationalen Datenbestände
- Antintegration
- Problem:
  - Handling von Nullwerten
  - Portabilität der XMI-Datenbestände
  - Speicher und Zeitverbrauch bei großen Testdatenbeständen

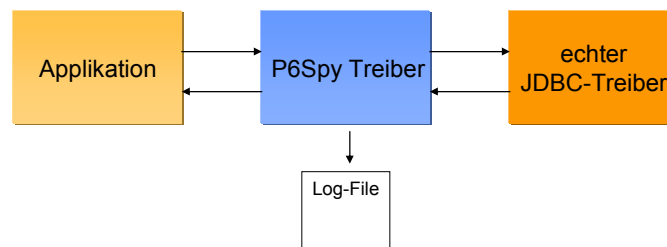
Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## P6Spy



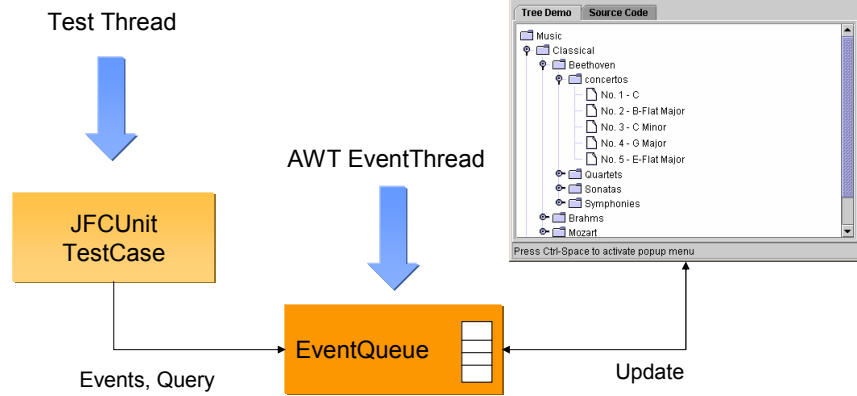
- Betrachtung der Performance-kritischen Bestandteile
  - Logging sämtlicher JDBC Operationen mit P6Log Module
  - Zeitabhängiges Filtern von JDBC Statements mit P6Outage Module



Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## JFCUnit



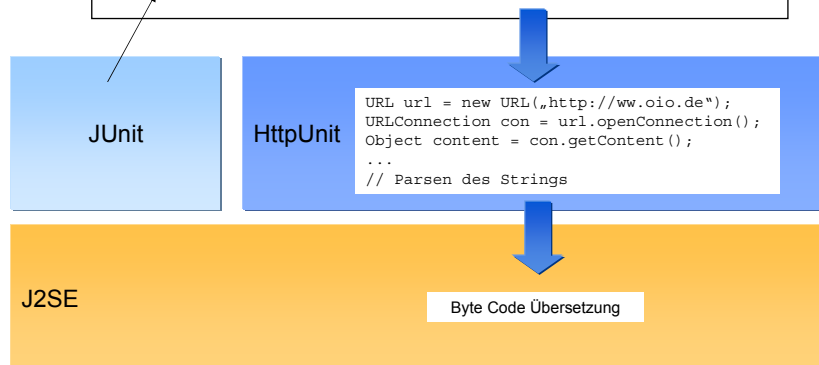
## JFCUnit

- API zum Suchen von JFC-Komponenten
- API zum Auslösen von Events
- besonderer Support für Single-Thread Architektur
  - asynchrone Eventbearbeitung nicht unterstützt
- Problem:
  - komplexe GUI-Komponenten(JTree,JTable)
  - modale Dialoge

## HttpUnit

### HttpUnit Testskript:

```
WebConversation wc = new WebConversation();  
WebResponse wr =  
wc.getResponse( "http://www.oio.de" );  
wr.getLinkWith(„Testen“).click();  
assertEquals( resp.getTitle(), "OIO" );
```



## HttpUnit

- JUnit Aufsatz zur Erstellung von Testskripten die
  - Steuerung des HTTP Protokolls
  - Behandlung von Sessions, Cookies, Frames
  - Bereitstellung von Textinhalten, XML DOM Dokumenten zur Bearbeitung via JUnit
- Systemtest von Webanwendung

JWebUnit Testskript:

```
getTestContext().setBaseUrl("http://www.oio.de");  
beginAt("/");  
clickLinkWithText("HttpUnit");  
assertTitleEquals("HttpUnit");
```

**JWebUnit**

```
WebConversation wc = new WebConversation();  
WebResponse wr =  
wc.getResponse("http://www.oio.de");  
wr.getLinkWith("Testen").click();  
assertEquals(resp.getTitle(), "OIO");
```

**JUnit** **HttpUnit**

```
URL url = new URL("http://ww.oio.de");  
URLConnection con = url.openConnection();  
Object content = con.getContent();  
...  
// Parsen des Strings
```

**J2SE**

Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

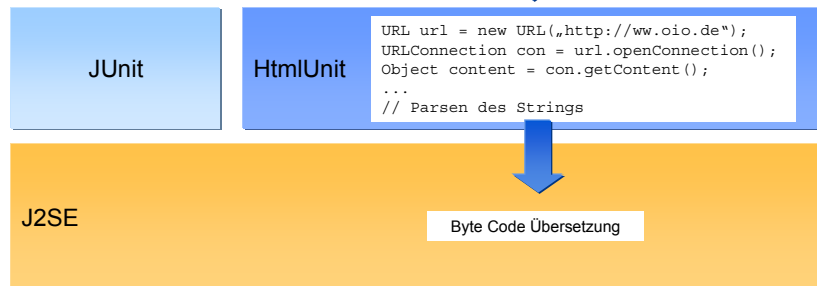
- Aufsatz auf HttpUnit, Rhino, Nekohtml
- Navigations API
  - Unterstützung für Formulare, Links, Frames und PopUp inklusive Navigation innerhalb dieser
- Assertions API
  - Validierung von Texten, Textbestandteile des HTML Dokuments, Zugriff auf Tabelleninhalte und HTML Elementen
  - Einbindung von Erwartungswerten über Properties ermöglicht
- Systemtest von Webanwendung

Test-driven Development aus dem Open Source Regal

© 2004 Orientation in Objects GmbH

## HtmlUnit Testskript:

```
WebClient webClient = new WebClient();
URL url = new URL("http://www.oio.de");
HtmlPage pageOIO = (HtmlPage)webClient.getPage(url);
HtmlForm form = page1.getFormByName("search");
HtmlSubmitInput button =
    (HtmlSubmitInput)form.getInputByName("submitbutton");
HtmlTextInput textField =
    (HtmlTextInput)form.getInputByName("searchString");
textField.setValueAttribute("xxxxx");
HtmlPage oioSearchPage = (HtmlPage)button.click();
```



- Parallele Entwicklung zu HttpUnit
- Fokus: HTML Dokumente
  - Java Wrapper für eine Vielzahl von HTML Elementen
    - z.B. `HtmlBoby`, `HtmlButton`, `HtmlHeader1`, ...
- Testausführung über JUnit
- Systemtest von Webanwendung

## TagUnit



- „Taglibs“ als Komponenten testbar
  - Überprüfung zwischen Deklaration (TLD) und Umsetzung (Taglib-Klassen)
  - Testumgebung zur Ausführung im Container - TagUnit Tags
  - Beispiel:

```
<tagunit:assertEquals name="Simple test of generated content">  
  <tagunit:expectedResult>...</tagunit:expectedResult>  
  <tagunit:actualResult>...</tagunit:actualResult>  
</tagunit:assertEquals>
```

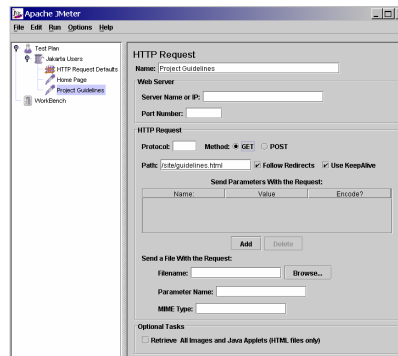
## JMeter



- Support für Proxy, Authorization, Website Spider
- Testplanung Remote und zeitgesteuert durchzuführen
- Ant Treiber Support
- Verteilung der Testprozesse über RMI
- Auswertung: Antwortzeit und Durchsatz

## JMeter

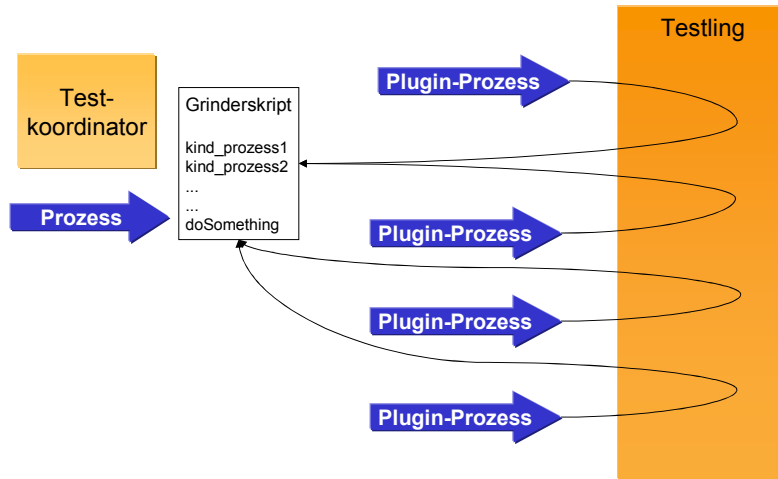
- Testpläne basiert auf eigener Sprache
- Erstellung wird grafische unterstützt
- Capture & Replay für HTTP
- Swing und JDBC Tests
- Threadgroup
- Controller
  - Logic Controller
  - Sampler



## The Grinder

- Grinder Testskripte steuern Aktivitäten in verschiedenen Prozessen über mehrere Rechner verteilt gezielt auf die Anwendung
  - Nicht-funktionale Systemtests
- *Grinder 2*: Plugin Mechanismus
  - Plugins für HTTP(S), RMI, Plain Java
- *Grinder 3*: basierend auf Jython
  - Ausführung über Java
- Problem: Auswertungsunterstützung
  - nur Transaktionszahl bereitgestellt

## The Grinder



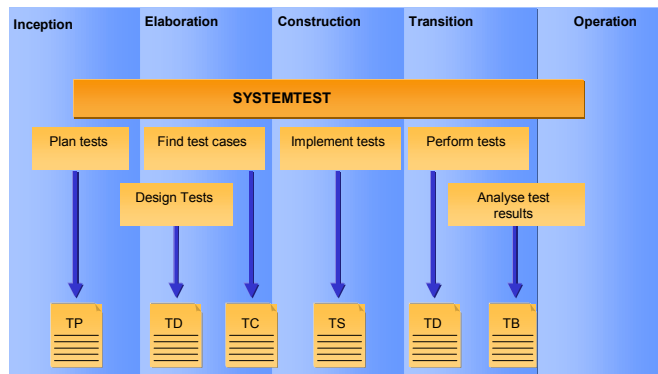
## TestMaker

- Grinder3 ähnliches Konzept
- C&R Funktionalität z.B. für Http
- kein multi-client support
- Monitoring nur auf Response ausgerichtet
- Problem: Auswertung
- usw. Loadsim auf Muffin Proxy

## Weitere Kategorien

- Integration
  - Ant, JUnit, Eclipse, ...
- Aktivität des Projekts
  - Mailing list, CVS, ...
  - Dokumentation
- Relevanz
  - Wiederverwendung in anderen Produkten ?
- Lizenz
- ...

## Werkzeugunterstützung ?



TP - Test plan  
TD - Test design specification  
TC - Test case specification  
TS - Test scripts  
TE - Test execution protocol  
TR - Test report

Vielen Dank für Ihre Aufmerksamkeit !

**Orientation in Objects GmbH**  
Weinheimer Str. 68  
68309 Mannheim  
<http://www.oio.de>  
[info@oio.de](mailto:info@oio.de)