

# Testing

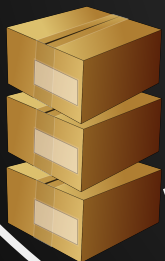
Tests und Testpyramide, Testframeworks

# Produktentwicklungszyklus

Anforderungen

verstehen  
bearbeiten  
zerlegen

Features / Storys



Entwicklungsiteration (Sprint / Projekt)

programmieren  
erstellen  
verifizieren



releasen (veröffentlichen)  
deploy (installieren)

# Übersicht

professionelle Software-  
Entwicklung

CI

CD

Prozesse

Methoden

VCS

Scrum

XP

Build Tools

Kanban

Code Reviews

Package Layout

Specification by  
Example

TDD

Tests

# Allgemeine Definition Test

“Ein Test ist ein Versuch, mit dem größere Sicherheit darüber gewonnen werden soll, ob ein technischer Apparat oder ein Vorgang innerhalb der geplanten Rahmenbedingungen funktioniert beziehungsweise ob bestimmte Eigenschaften vorliegen oder nicht.”

# Tests in der Softwareentwicklung

“Ein Softwaretest prüft und bewertet Software auf Erfüllung der für ihren Einsatz definierten Anforderungen und misst ihre Qualität. Die gewonnenen Erkenntnisse werden zur Erkennung und Behebung von Softwarefehlern genutzt. Tests während der Softwareentwicklung dienen dazu, die Software möglichst fehlerfrei in Betrieb zu nehmen.”

Quelle: <http://de.wikipedia.org/wiki/Softwaretest>

# Grenzen von Tests

„Program testing can be used to show the presence of bugs, but never show their absence!“ (Das Testen von Programmen kann die Existenz von Fehlern zeigen, aber niemals deren Nichtvorhandensein)

Edsger W. Dijkstra

# Wozu werden Tests geschrieben:

“Globales Ziel des Softwaretestens ist das Messen der Qualität des Softwaresystems.”

(Quelle: <http://de.wikipedia.org/wiki/Softwaretest#Ziele>)

- fixieren das Verhalten des Codes
- Fehler beheben (über Nachweis des Vorhandenseins von Fehler)
- erleichtern bzw. ermöglichen überhaupt erst Umarbeiten der Quelltextes (Refactoring)

# Bugs




9/9

0800 Antan started  
 1000 " stopped - antan ✓ { 1.2700 9.037 847 025  
 1300 (032) MP-MC 2.130476415 9.037 846 995 connect  
 (033) PRO 2 2.130476415 4.615925059(-2)  
 connect 2.130676415

Relays 6-2 in 033 failed special speed test  
 in relay 10.000 test.

Relays changed

1100 Started Cosine Tape (Sine check)  
 1525 Started Multi Adder Test.

1545  Relay #70 Panel F  
 (moth) in relay.

First actual case of bug being found.

1630 Antan started.  
 1700 closed down.

Relay 3375

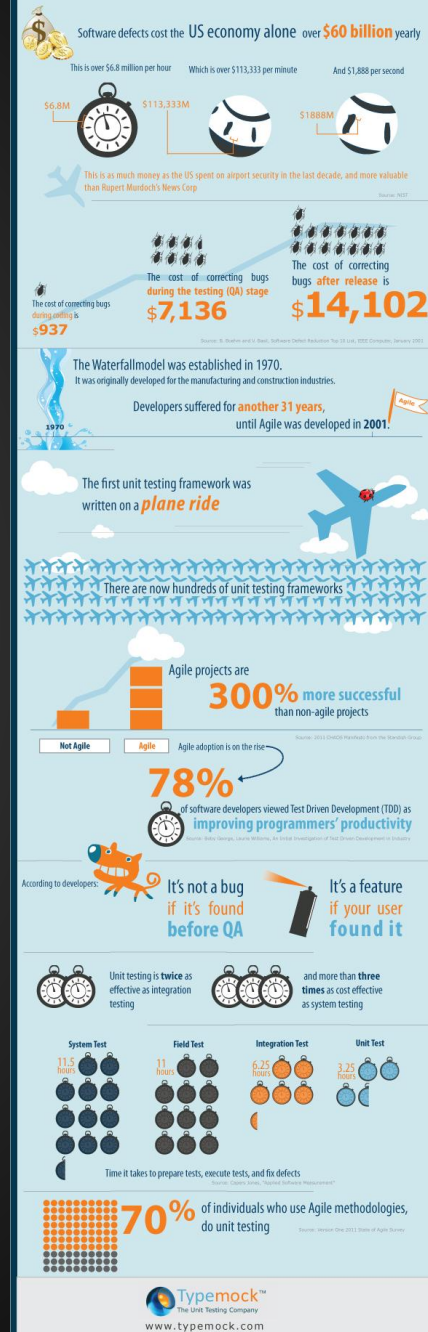
"In 1946, when (Grace) Hopper was released from active duty, she joined the Harvard Faculty at the Computation Laboratory where she continued her work on the Mark II and Mark III. Operators traced an error in the Mark II to a moth trapped in a relay, coining the term bug. This bug was carefully removed and taped to the log book. Stemming from the first bug, today we call errors or glitch's [sic] in a program a bug."

Quelle: [http://en.wikipedia.org/wiki/Software\\_bug](http://en.wikipedia.org/wiki/Software_bug)

# Kosten von Fehlern

Softwarefehler sind teuer! Die Fehlerbehebung kostet umso mehr je später der Fehler im Entwicklungs- und Lieferprozess gefunden wird!

## The Severity of Bugs: Are We Doomed?





Software defects cost the US economy alone over **\$60 billion** yearly

This is over \$6.8 million per hour

Which is over \$113,333 per minute

And \$1,888 per second



This is as much money as the US spent on airport security in the last decade, and more valuable than Rupert Murdoch's News Corp

Source: NIST



The cost of correcting bugs during coding is **\$937**



The cost of correcting bugs during the testing (QA) stage

**\$7,136**



The cost of correcting bugs after release is

**\$14,102**

Source: B. Boehm and W. Basili, Software Defect Reduction Top 10 List, IEEE Computer, January 2001

The Cost of Bugs by Bradley Jones - Published 11/11/2012

<http://www.codeguru.com/blog/category/programming/the-cost-of-bugs.html>

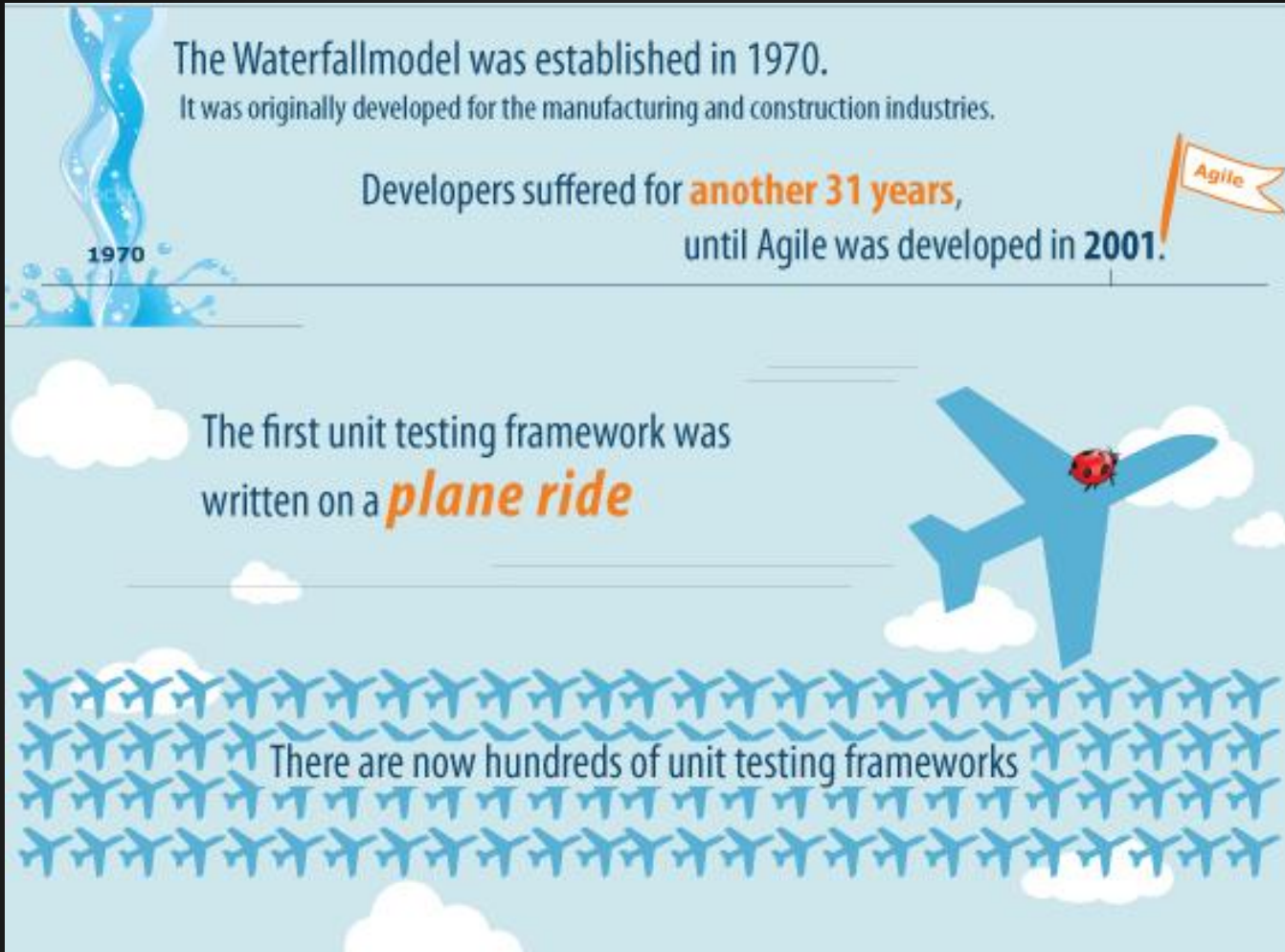
# Kosten von Fehlern aka Bugs

Cambridge University Study States Software Bugs Cost Economy \$312 Billion Per Year

Software development's taboo subject annually costs more than double the total Eurozone Bailout, Released by Undo Software

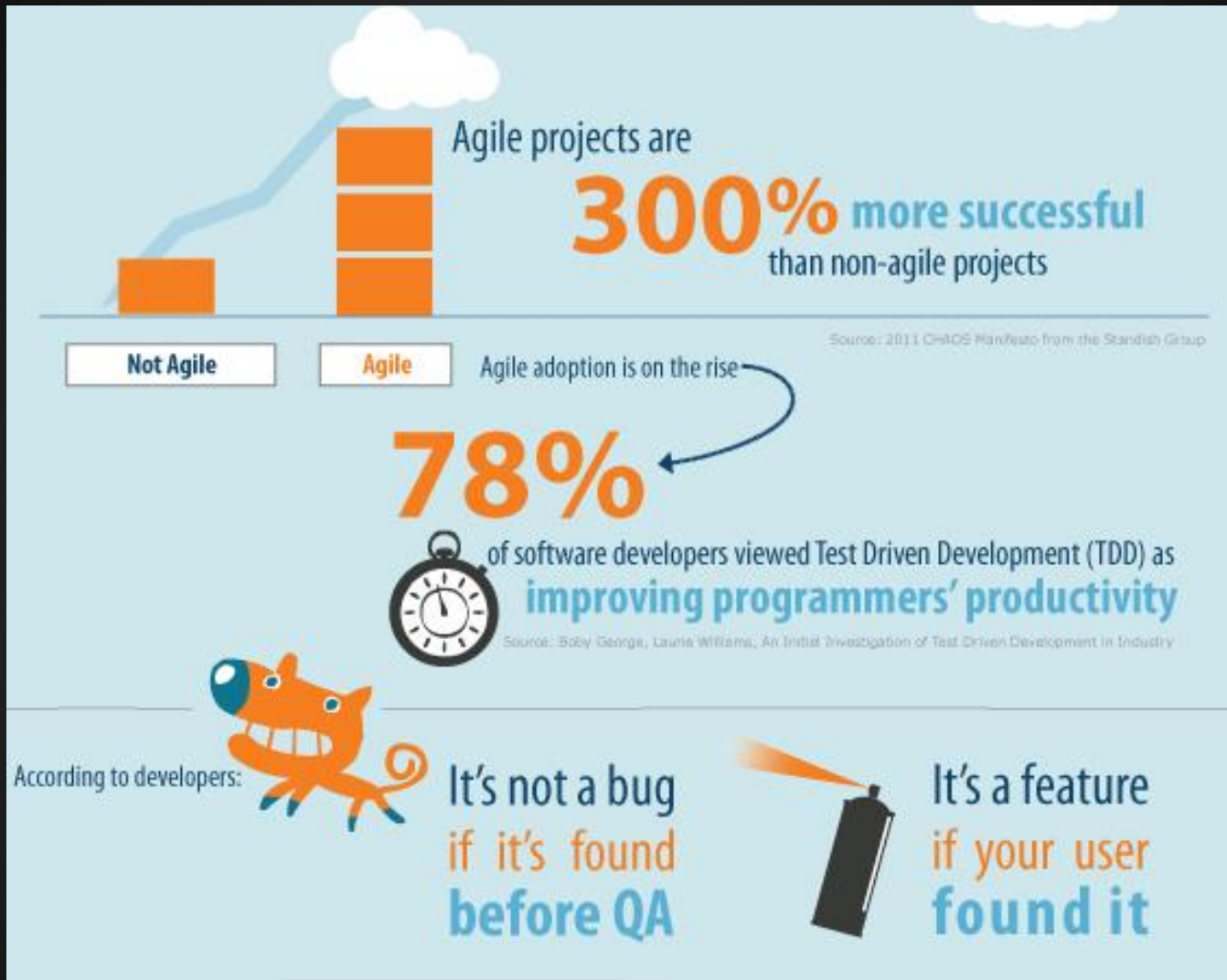
Quelle: <http://www.prweb.com/releases/2013/1/prweb10298185.htm>





The Cost of Bugs by Bradley Jones - Published 11/11/2012

<http://www.codeguru.com/blog/category/programming/the-cost-of-bugs.html>



The Cost of Bugs by Bradley Jones - Published 11/11/2012  
<http://www.codeguru.com/blog/category/programming/the-cost-of-bugs.html>



Unit testing is **twice** as effective as integration testing



and more than **three times** as cost effective as system testing

#### System Test



#### Field Test



#### Integration Test

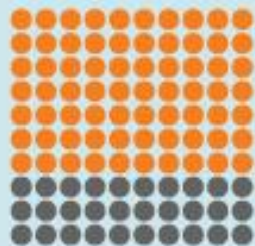


#### Unit Test



Time it takes to prepare tests, execute tests, and fix defects

Source: Capers Jones, "Applied Software Measurement"



**70%**

of individuals who use Agile methodologies, do unit testing

Source: Version One 2011 State of Agile Survey

The Cost of Bugs by Bradley Jones - Published 11/11/2012

<http://www.codeguru.com/blog/category/programming/the-cost-of-bugs.html>

# Entwicklung moderner Test-Methoden

- 1997 - JUnit -Framework
- Martin Fowler - Selftesting Code (PDF 1998)
  - <http://martinfowler.com/distributedComputing/duckling.pdf>
- CppUnit for C++ durch Michael Feathers (erste Portierung von JUnit in eine andere Sprache)
- Kent Beck - Artikel (Test in SmallTalk):
  - <http://www.xprogramming.com/testfram.htm>



# Testen von funktionalen und nicht-funktionalen Anforderungen

## funktionale Anforderungen

- Fachlichkeit
- z. B. Button auf Webseite

## nicht-funktionale Anforderungen

- Rahmenbedingungen (Sicherheit, Gebrauchstauglichkeit, Zuverlässigkeit)
- z. B. XSS / Ausfallsicherheit / etc.

# Manuell vs. Automation

## Manuell

- Exploratives Testen
- Session-based Testing
  - Session-based testing is a software test method that aims to combine accountability and exploratory testing to provide rapid defect discovery, creative on-the-fly test design, management control and metrics reporting. The method can also be used in conjunction with scenario testing. Session-based testing was developed in 2000 by Jonathan and James Bach. (Quelle: [http://en.wikipedia.org/wiki/Session-based\\_testing](http://en.wikipedia.org/wiki/Session-based_testing))

## Automatisch

- Tests nach Testpyramide (Unit / Component / System-Integration, etc.)

**Automatisch ausführbare Tests sind  
die Voraussetzung für CI !**



# Literatur / Links

- <http://watirmelon.com/2012/01/31/introducing-the-software-testing-ice-cream-cone/>
- <http://www.martinfowler.com/bliki/SelfTestingCode.html>
- <http://www.martinfowler.com/bliki/TestDrivenDevelopment.html>
- [http://www.jamesshore.com/Agile-Book/test\\_driven\\_development.html](http://www.jamesshore.com/Agile-Book/test_driven_development.html)
- <http://www.jamesshore.com/Blog/Lets-Play> (TDD)

