Let's make this test suite run faster!

Sponsors Platiniums 2010

OXiane

valtech

NOVEDIA VISION

zenika ARCHITECTURE INFORMATIQUE

OBJET DIRECT

FastConnect

IBM ibm.com/developerWorks

Xebia

Sfeir Groupe

Who am I?

David Gageot

CTO algodeal.com
The Crowd Sourced
Quant Hedge Fund

@dgageot
javabien.net

Let's make this test suite run faster!

Why?

http://farm1.static.flickr.com/158/349497988_fb751a5e3a_z.jpg

You test early and often

Don't you?

http://farm1.static.flickr.com/97/246816211_573c2901e1_z.jpg?zz=1

Continuous integration, continuous testing...

...even continuous deployment

JUnit Max

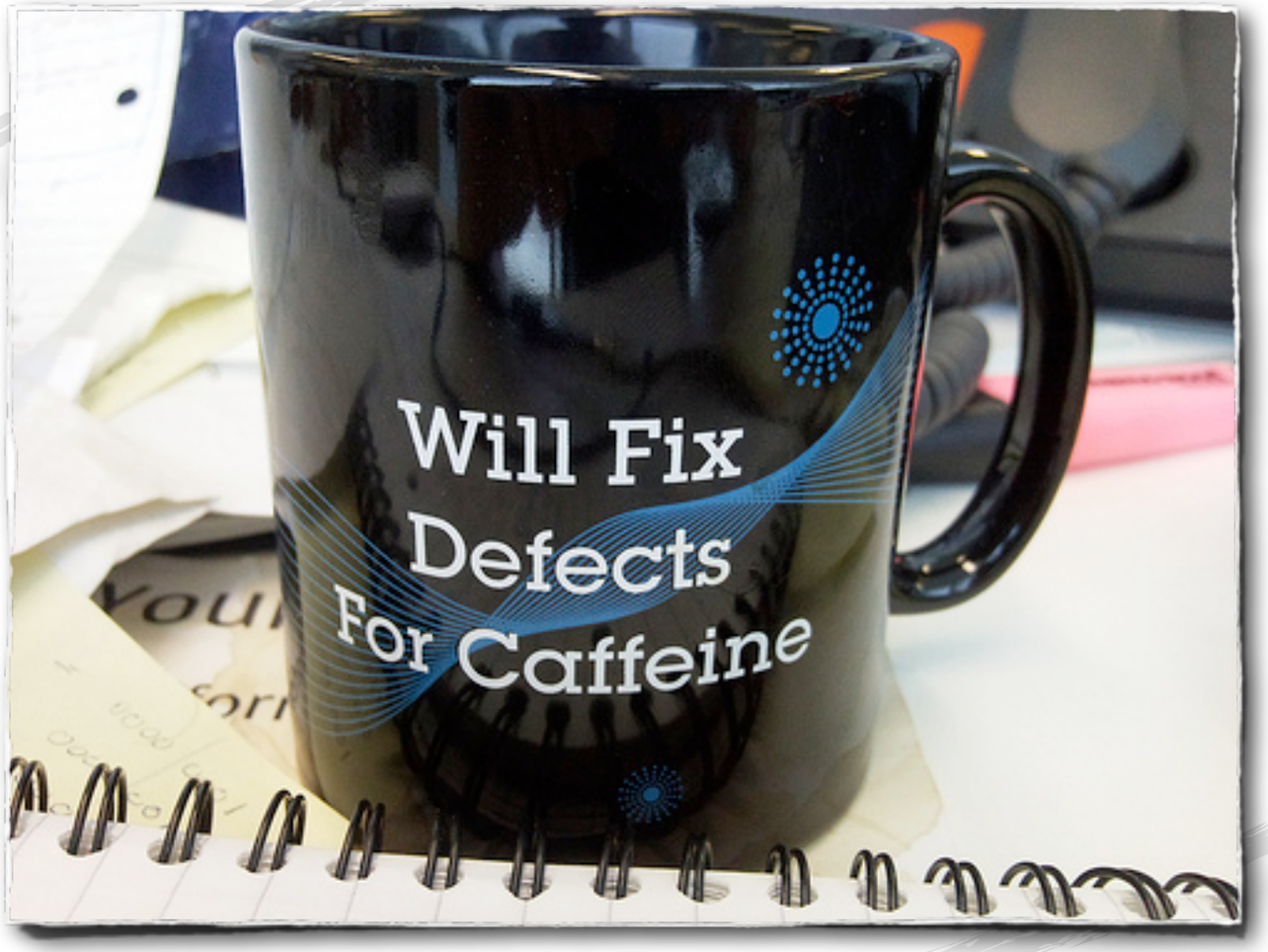
infinitest

cruisecontrol

http://vator.tv/images/attachments/150609195630ctd.gif

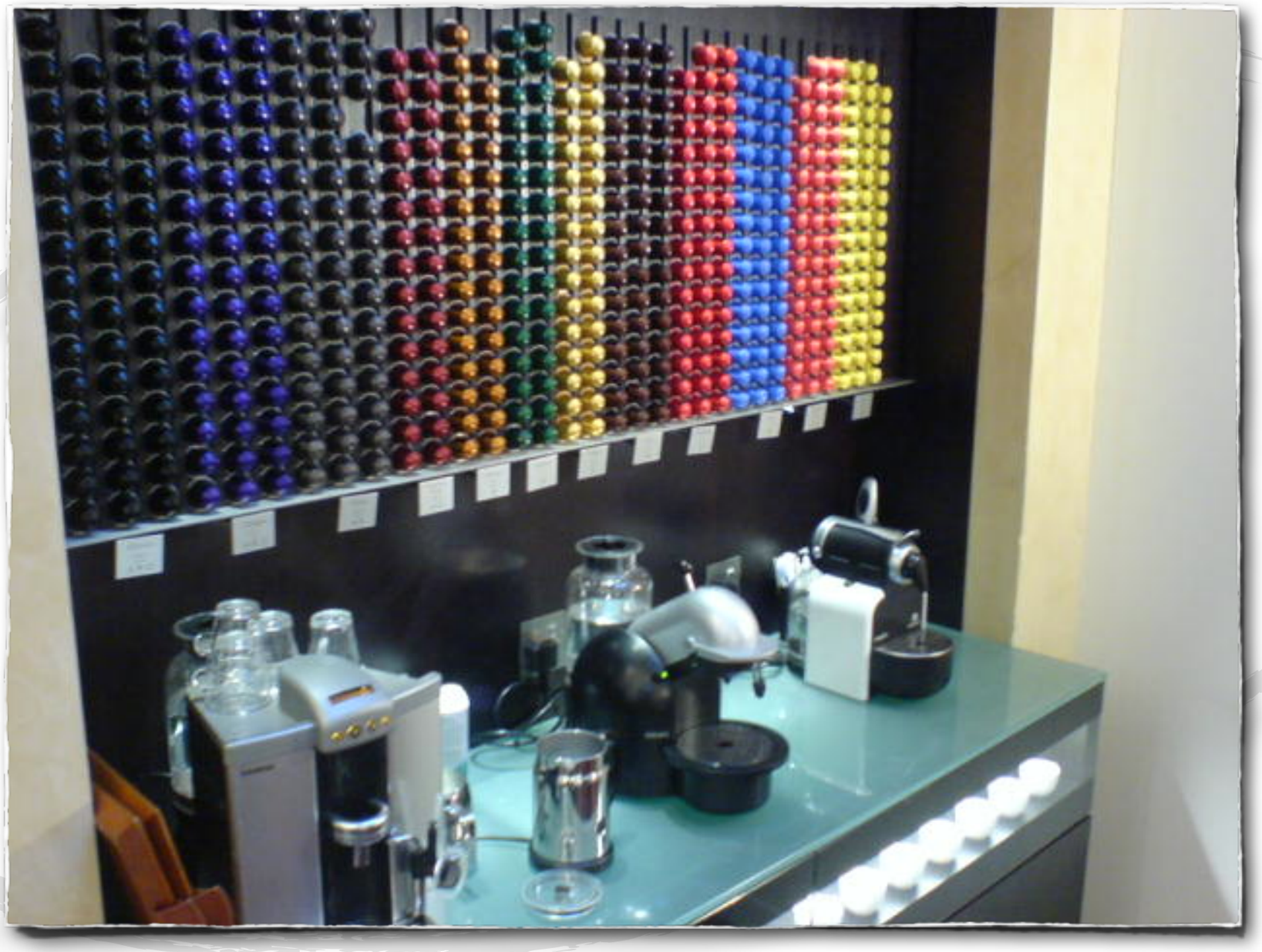
Testing often

Often means waiting a lot



http://farm3.static.flickr.com/2505/3772118924_11fe195ef1.jpg

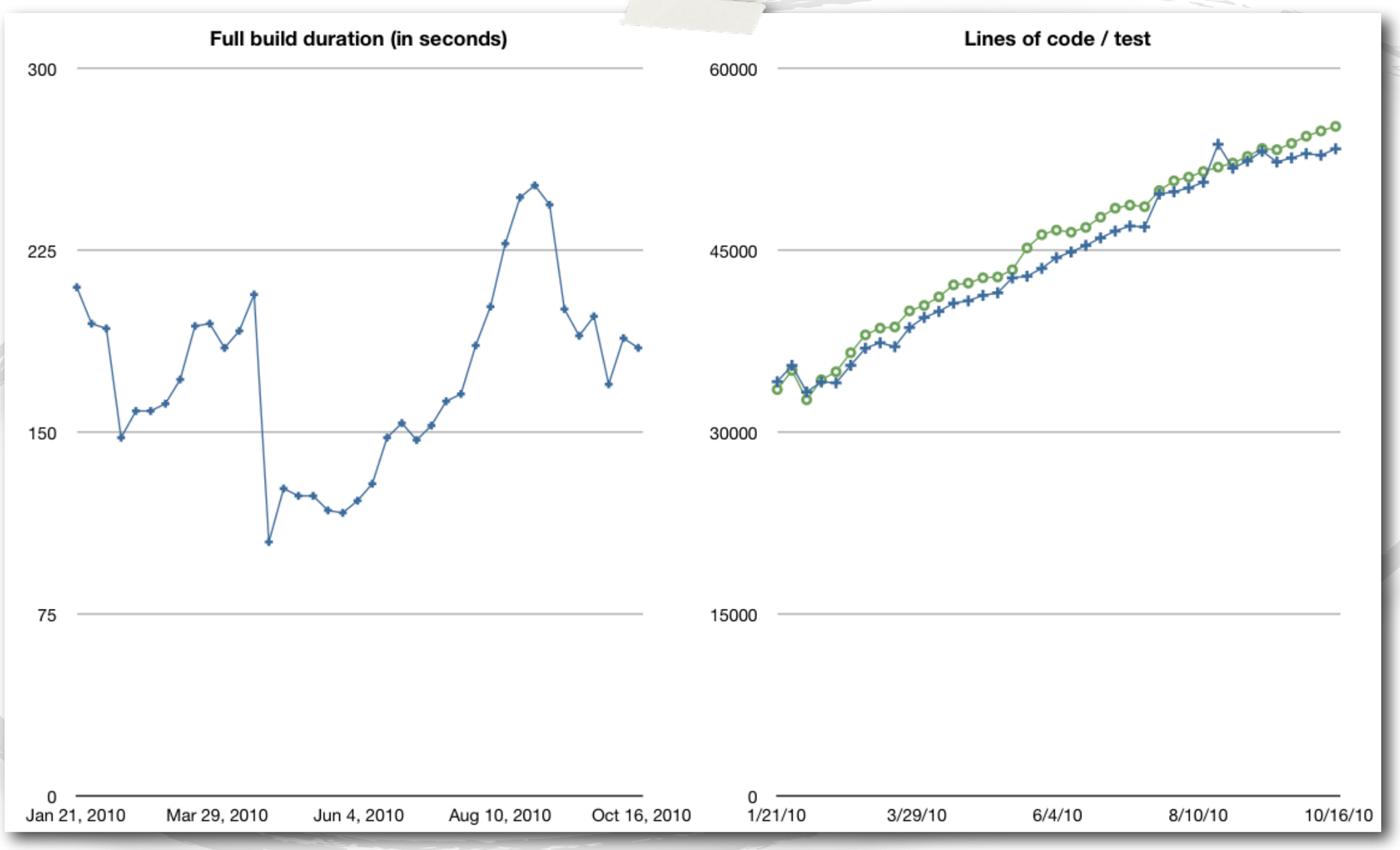
Your coding room doesn't have to look like this



http://upload.wikimedia.org/wikipedia/commons/3/3f/Nespresso_boutique.JPG

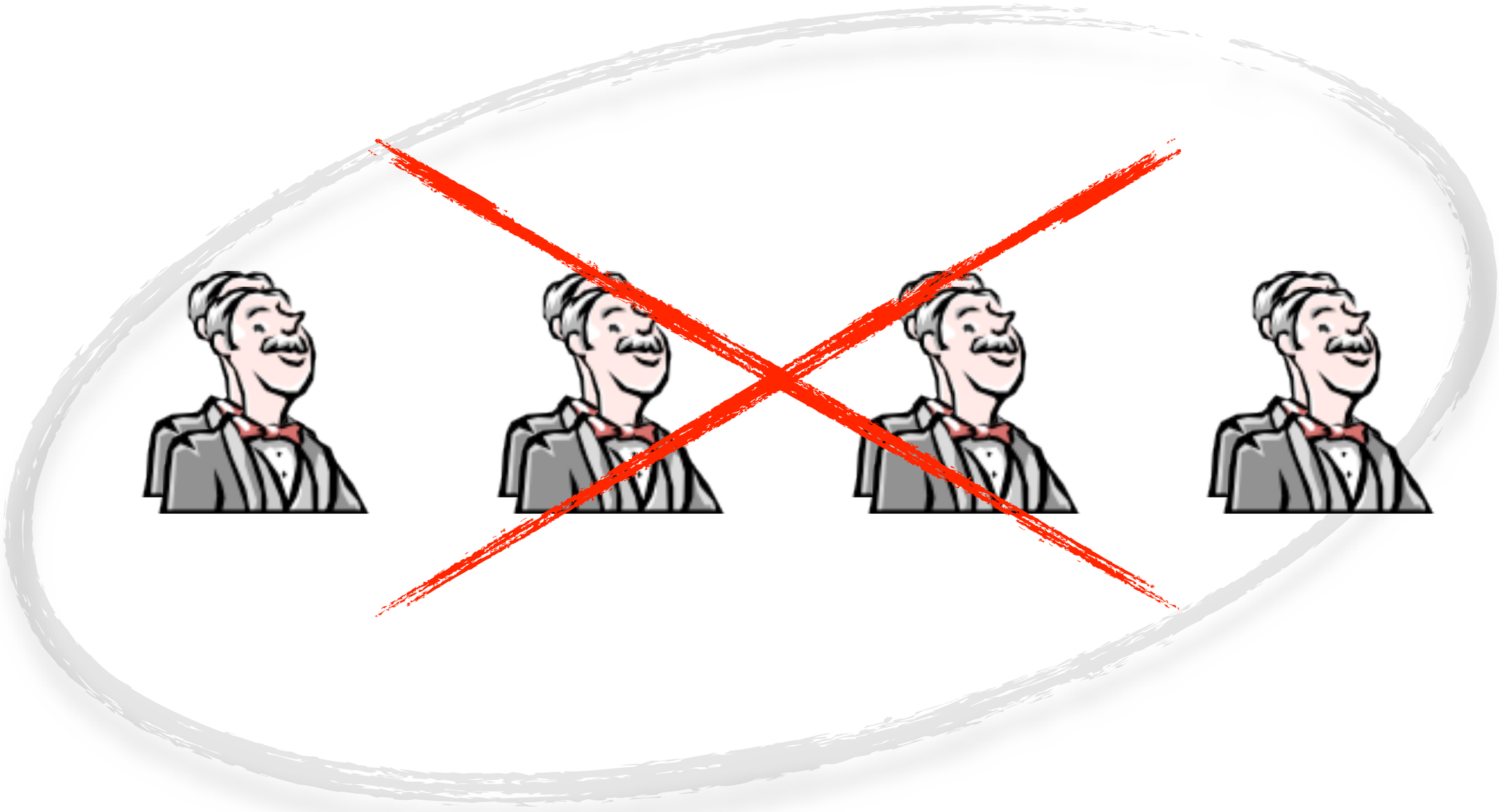
Tests can be fast

Even if lines of code keep growing



Easy!

Let's distribute across multiple hudson servers



It doesn't have to be complicated

I'll share a few simple tricks

It can make your product better

Simple to test, means simple to write, deploy and USE

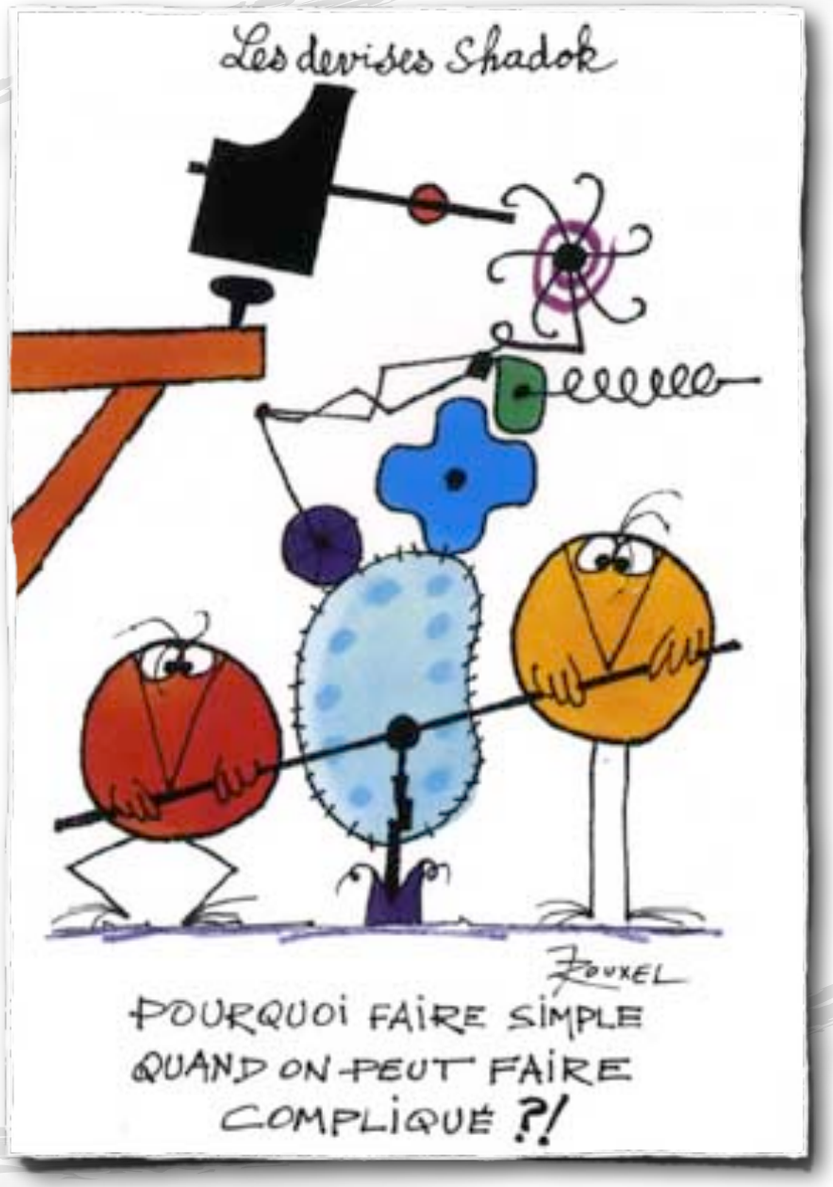
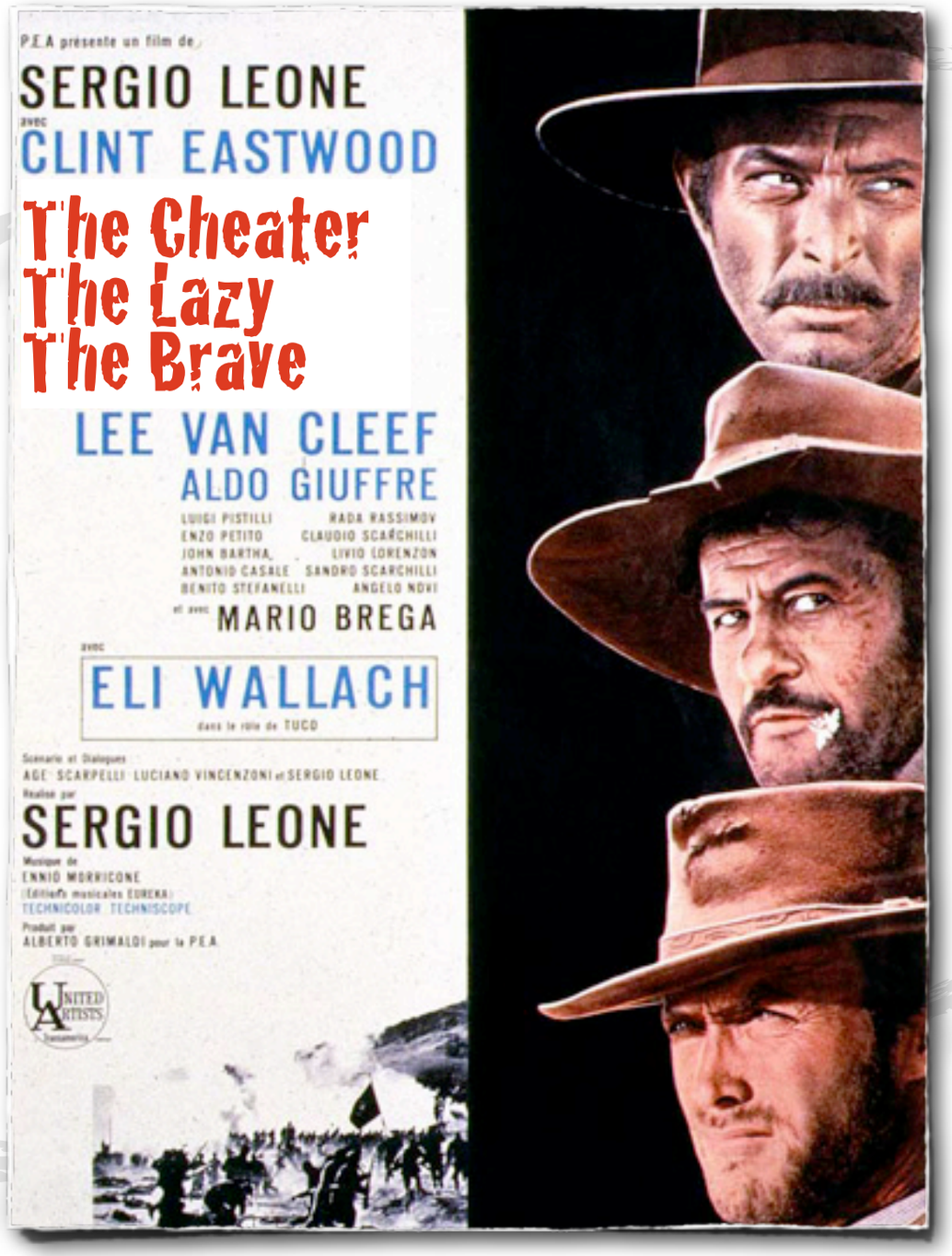
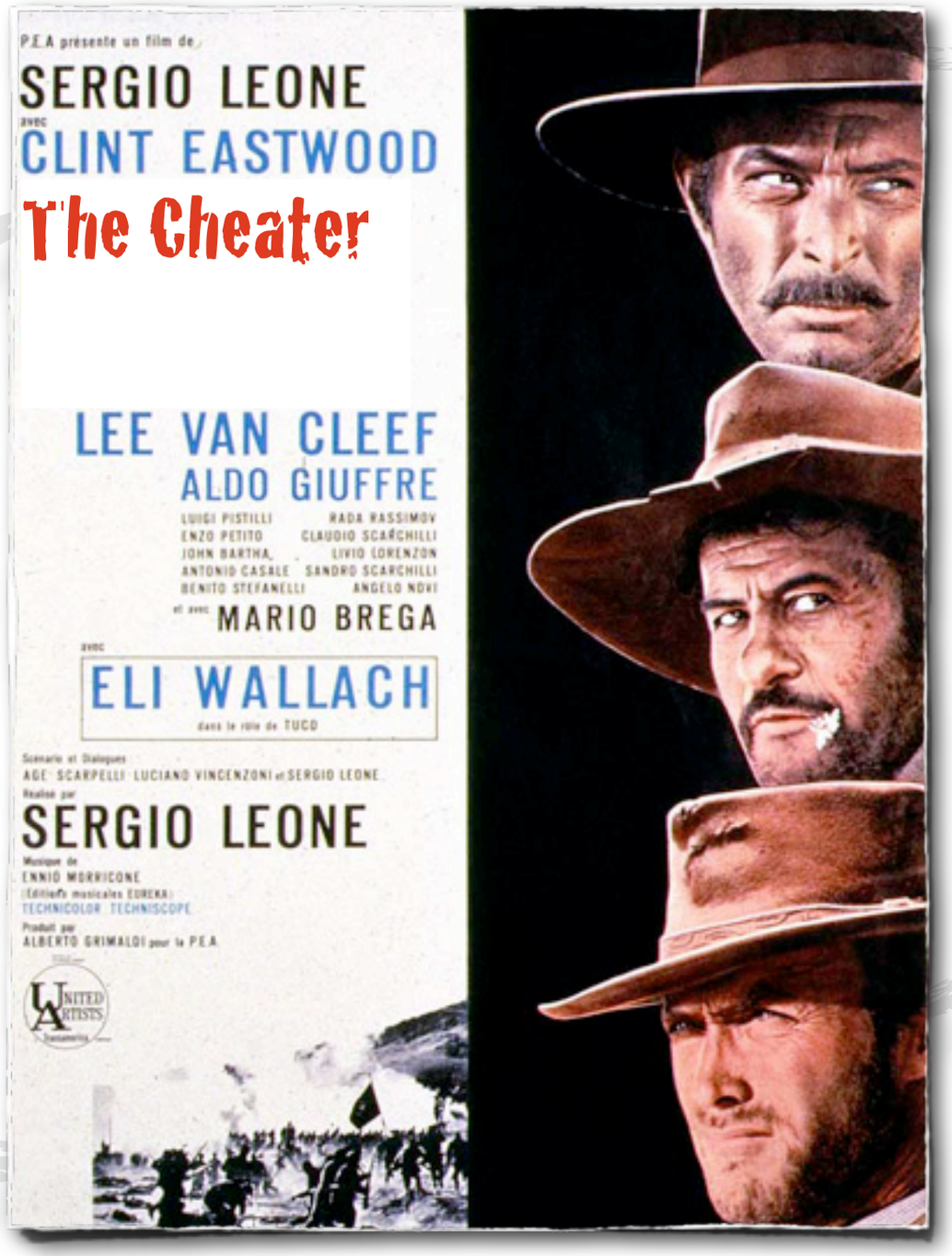


Photo: http://www.flickr.com/photos/zoutedrop/2317065892/

How can we accelerate the tests?



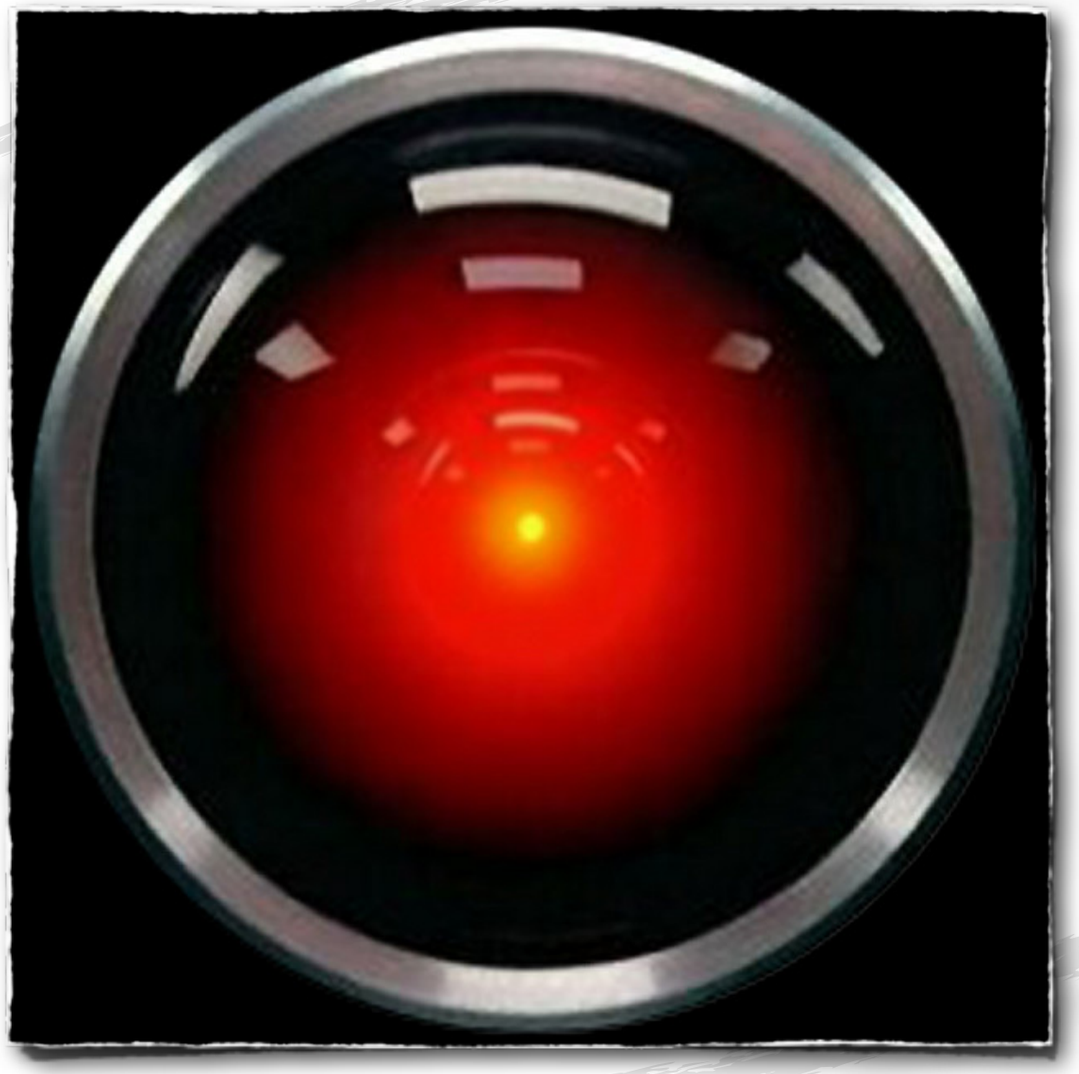
http://all-the-movies.cowblog.fr/images/arton1122.jpg



http://all-the-movies.cowblog.fr/images/arton1122.jpg

Buy a faster machine

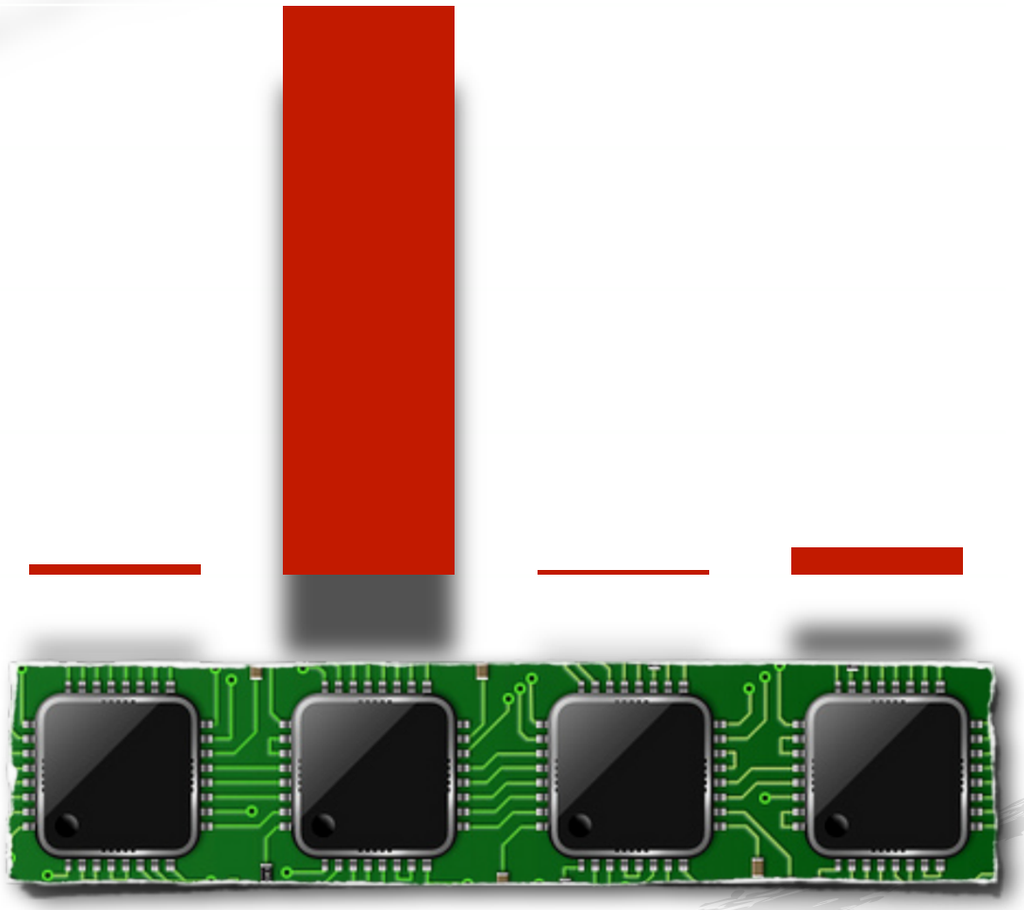
Tests are cpu/memory bound



http://farm1.static.flickr.com/28/93569705_1c562b413a_z.jpg?zz=1

Be warned

Single threaded tests get slower over time



http://farm5.static.flickr.com/4096/4819945812_735744e0fc_z.jpg

Use all the cores

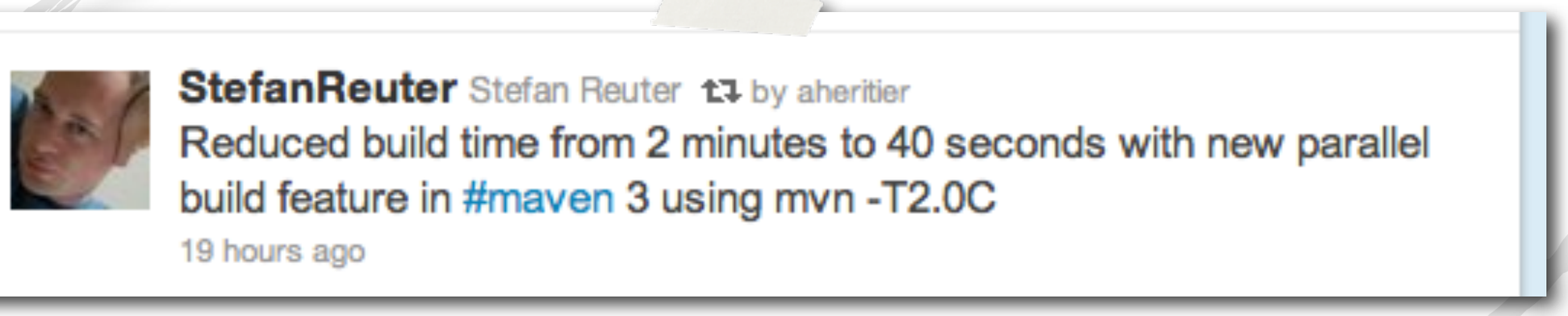
parallel build with maven3

mvn -T1 clean install : 5:05s

mvn -T4 clean install : 3:10s

Use all the cores

parallel build with maven3

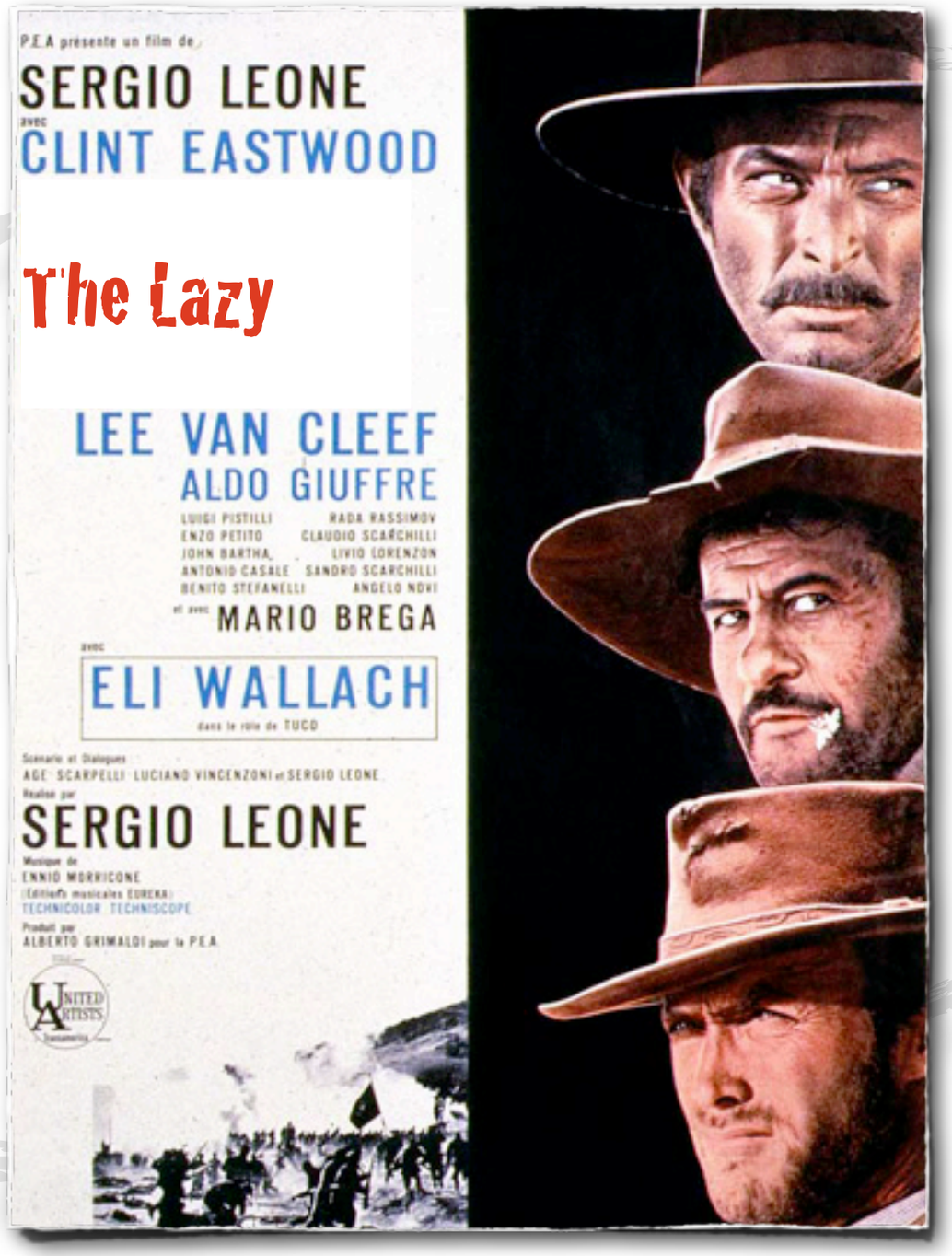


Use all the cores

For JUnit/TestNG tests

```
<dependency>
    <groupId>org.jdogma.junit</groupId>
    <artifactId>configurable-parallel-computer</artifactId>
    <version>1.5</version>
    <scope>test</scope>
</dependency>
```

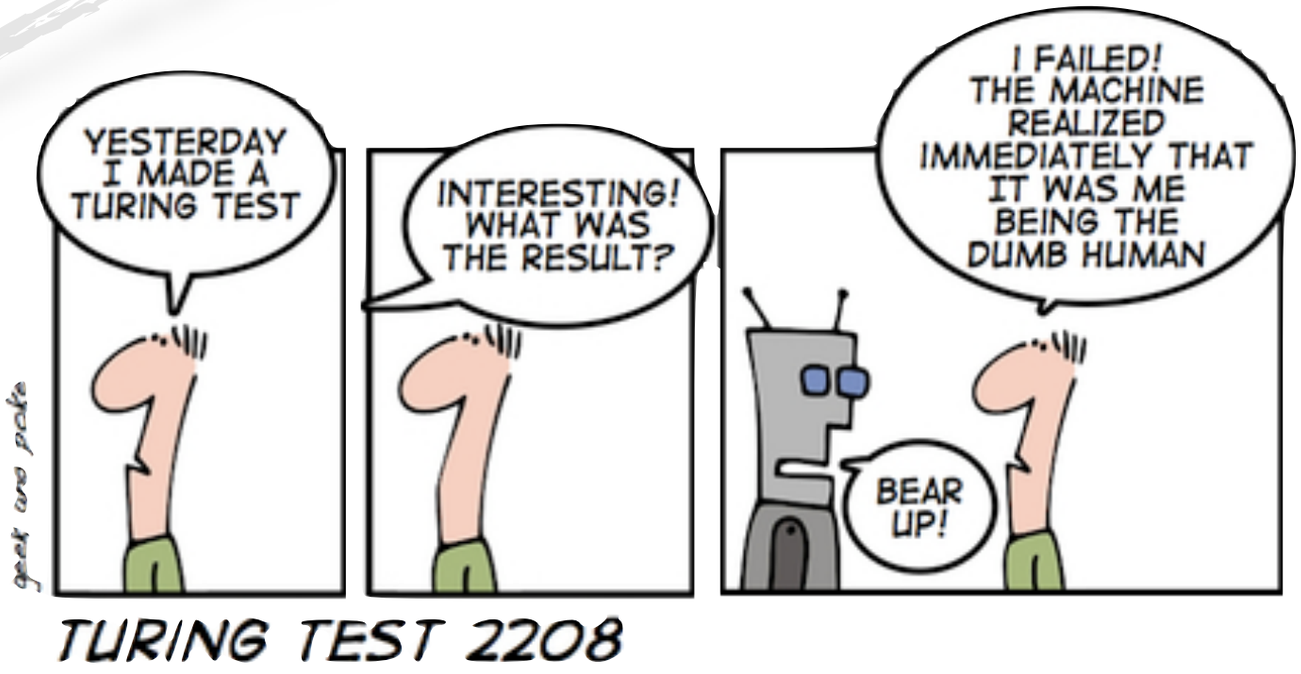
```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-surefire-plugin</artifactId>
    <configuration>
        ....
        <parallel>classes</parallel>
        <threadCount>2</threadCount>
    </configuration>
</plugin>
```



http://all-the-movies.cowblog.fr/images/arton1122.jpg

Delete redundant tests

It's so simple, we don't do it...



http://geekandpoke.typepad.com/geekandpoke/images/2008/04/24/turingtest.jpg

Even better, delete dead code

To delete yet other useless tests

http://farm1.static.flickr.com/58/191393602_3d9b643ab0_z.jpg

Work in a sandbox

The network is too slow

In-memory database: H2

```
<bean id="dataSource" class="org.springframework.jdbc.datasource.SingleConnectionDataSource">
    <constructor-arg value="org.h2.Driver" />
    <constructor-arg value="jdbc:h2:mem:test" />
    <constructor-arg value="sa" />
    <constructor-arg value="" />
    <constructor-arg value="true" />
    <property name="autoCommit" value="false" />
</bean>
```

Behaves more like MySql than Hsqldb

It's Not only SQL

If going with NoSQL, take a server that can run in-process

eg. Voldemort

In-memory SMTP Server

Ethereal

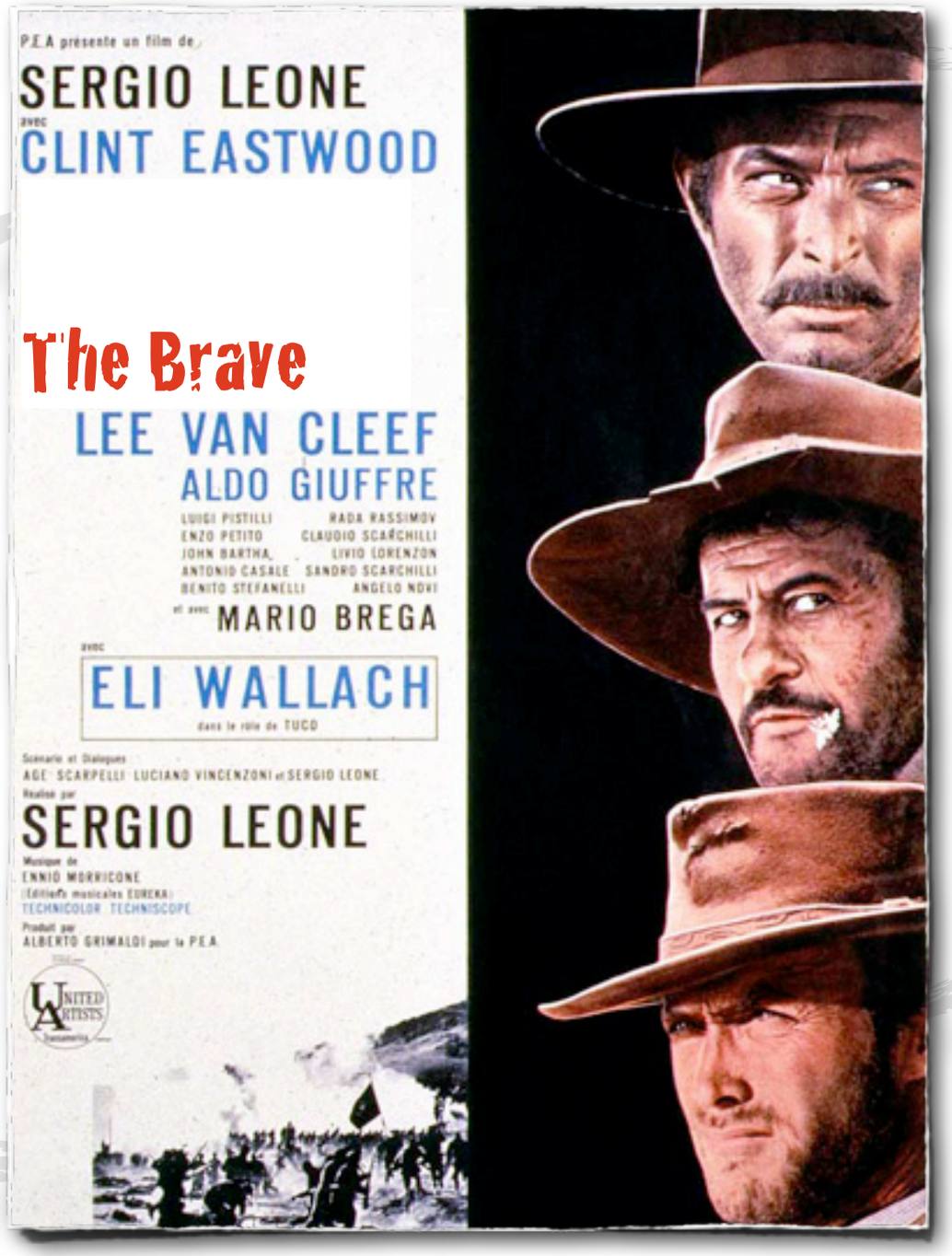
```
public class MailQueueTestFunctional extenas AbstractTestFunctional {
    private static Wiser wiser;
    private MailQueue mailQueue;

    @BeforeClass
    public static void startSmtpServer() {
        wiser = new Wiser(2525);
        wiser.start();
    }

    @Test
    public void canSendEmail() {
        List<WiserMessage> messages = wiser.getMessages();

        WiserMessage email = emails.get(0);
        assertThat(email.getEnvelopeSender()).isEqualTo("no-reply@algodeal.com");
    }

    @AfterClass
    public static void stopSmtpServer() {
        wiser.stop();
    }
}
```

Everything local and in-memory
Files

Apache VFS (Virtual File System)

Spring Resource...

As a bonus, tests will run smoother



http://all-the-movies.cowblog.fr/images/arton1122.jpg

Don’t test business rules in integration tests

Unit tests is often a better place

http://upload.wikimedia.org/wikipedia/commons/b/bd/Contortionist_Ravi_standing.jpg

Don't test business rules in integration tests

Unit tests is often a better place

```
@Test
public void displayDotsNumberWhenThereAreLotsOfData() {
    mockSolutionDaoToReturnNSolutions(51);
    beginAtAsQuant("/users/backtestHistory.html");

    assertTextPresent("page 1 2 3 4 5 ... 11 next");
    assertTextNotPresent("newest");

    clickLinkWithExactText("next");
    assertTextPresent("page prev 1 2 3 4 5 ... 11 next");
    assertTextNotPresent("newest");

    clickLinkWithExactText("next");
    assertTextPresent("page prev 1 2 3 4 5 ... 11 next");
    assertTextNotPresent("newest");

    clickLinkWithExactText("next");
    assertTextPresent("page prev 1 2 3 4 5 6 ... 11 next");

    clickLinkWithExactText("prev");
    assertTextPresent("page prev 1 2 3 4 5 ... 11 next");
    assertTextNotPresent("newest");

    clickLinkWithExactText("prev");
    assertTextPresent("page prev 1 2 3 4 5 ... 11 next");
    assertTextNotPresent("newest");

    clickLinkWithExactText("11");
    assertTextPresent("page prev 1 ... 7 8 9 10 11");
    assertTextNotPresent("next");
    assertTextNotPresent("oldest");

    clickLinkWithExactText("prev");
    assertTextPresent("page prev 1 ... 7 8 9 10 11 next");
    assertTextNotPresent("oldest");

    clickLinkWithExactText("prev");
    assertTextPresent("page prev 1 ... 7 8 9 10 11 next");
    assertTextNotPresent("oldest");
```

Functional test
(on a web page)
10s

```
@Test
public void canPaginate11Pages() {
    assertThat(pages(11, 11)).isEqualTo("prev 1 ... 7 8 9 10 11");
    assertThat(pages(10, 11)).isEqualTo("prev 1 ... 7 8 9 10 11 next");
    assertThat(pages(9, 11)).isEqualTo("prev 1 ... 7 8 9 10 11 next");
    assertThat(pages(8, 11)).isEqualTo("prev 1 ... 6 7 8 9 10 11 next");
    assertThat(pages(7, 11)).isEqualTo("prev 1 ... 5 6 7 8 9 ... 11 next");
    assertThat(pages(6, 11)).isEqualTo("prev 1 ... 4 5 6 7 8 ... 11 next");
    assertThat(pages(5, 11)).isEqualTo("prev 1 ... 3 4 5 6 7 ... 11 next");
    assertThat(pages(4, 11)).isEqualTo("prev 1 2 3 4 5 6 ... 11 next");
    assertThat(pages(3, 11)).isEqualTo("prev 1 2 3 4 5 ... 11 next");
    assertThat(pages(2, 11)).isEqualTo("prev 1 2 3 4 5 ... 11 next");
    assertThat(pages(1, 11)).isEqualTo("1 2 3 4 5 ... 11 next");
}
```

Unit test
0.01s

Take the longer integration test

Break it in one faster integration test
and a lot of small unit tests

```
 Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.015 sec
Running com.algodeal.util.collections.CountTest
Tests run: 10, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.008 sec
Running com.algodeal.util.numbers.NumberUtilsTest
n: 13, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.02 sec
com.algodeal.util.collections.EnumUtilsTest
n: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.01 sec
com.algodeal.util.misc.ThreadedTest
n: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.019 sec
com.algodeal.util.functions.MoreStringUtilsTest
n: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.004 sec
com.algodeal.util.misc.ChronoTest
n: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.038 sec
com.algodeal.util.io.ResourceUtilsTest
n: 9, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.013 sec
com.algodeal.util.misc.SequencerTest
n: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.003 sec
com.algodeal.util.predicates.StringPredicatesTest
n: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.005 sec
com.algodeal.util.io.UncategorizedIOExceptionTest
n: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.008 sec
com.algodeal.util.primitives.ToDoubleTest
n: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.002 sec
com.algodeal.util.collections.CachedIterableTest
n: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.017 sec
com.algodeal.util.io.FilenameFiltersTest
n: 7, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.035 sec
com.algodeal.util.suppliers.AbstractRefreshSupplierTest
n: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.02 sec
com.algodeal.util.functions.MoreFunctionsTest
n: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.007 sec
com.algodeal.util.proxy.CacheProxyTest
n: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.048 sec
Running com.algodeal.util.misc.RegexUtilsTest
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.005 sec
Running com.algodeal.util.io.LocalFileAccessorImplTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.017 sec
Running com.algodeal.util.csv.CsvFileTemplateTest
Tests run: 7, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.012 sec
Running com.algodeal.util.io.FileTemplateTest
Tests run: 31, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.208 sec
```


http://farm2.static.flickr.com/1168/3166709586_96f9a3fd95.jpg

Or mock the slowest layers

eg. functional test with Spring and Mockito

```
public class MockitoSpyPostProcessor implements BeanPostProcessor {
    private final Set<String> namesOfBeansToSpy;

    public MockitoSpyPostProcessor(Set<String> namesOfBeansToSpy) {
        this.namesOfBeansToSpy = namesOfBeansToSpy;
    }

    @Override
    public Object postProcessBeforeInitialization(Object bean, String beanName) {
        return bean;
    }

    @Override
    public Object postProcessAfterInitialization(Object bean, String beanName) {
        if (namesOfBeansToSpy.contains(beanName)) {
            return Mockito.spyBean(bean);
        }

        return bean;
    }
}
```

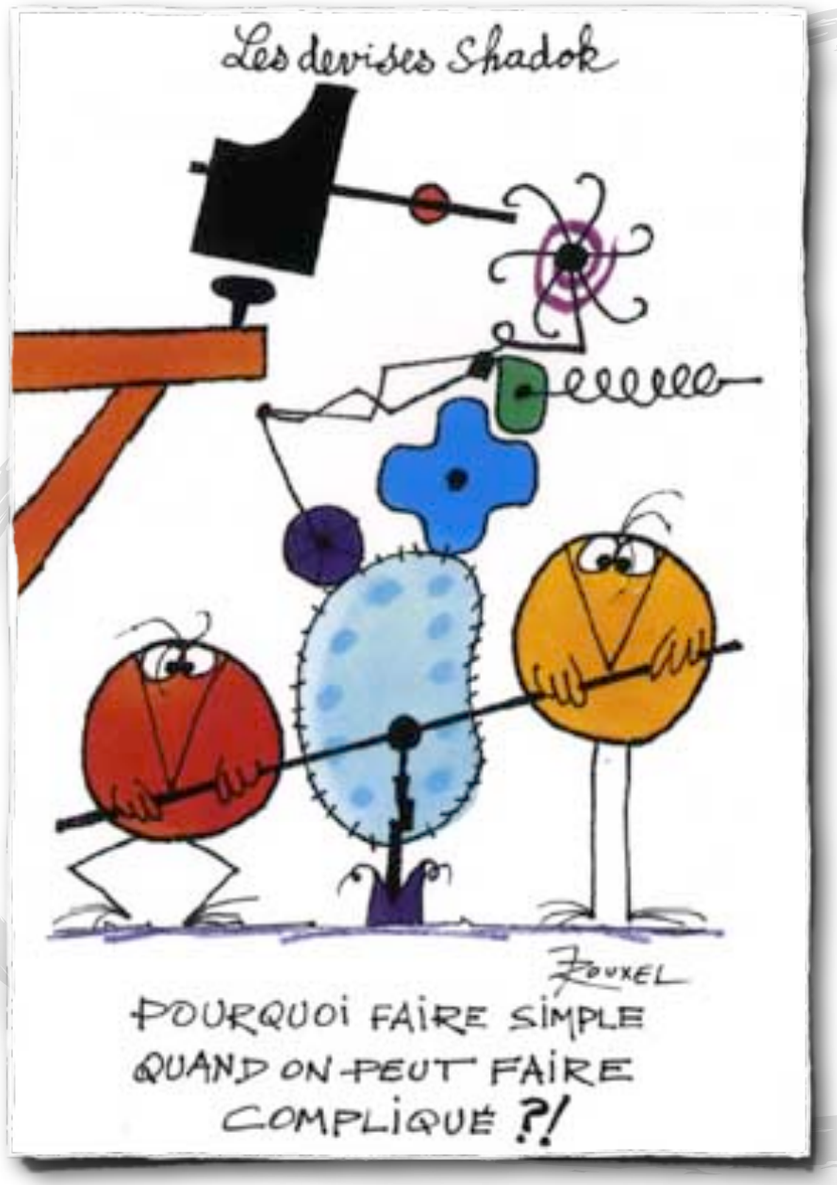
Or mock the slowest layers

eg. functional test with Spring and Mockito


```
@Test
public void userLoggedInWithAdminPasswordCanDownloadTheOrders() {
    doReturn(zip("<Orders>")).when(getStore()).retrieveOrders(QUANT, SOLUTION_ID);

    logIn(QUANT, ADMIN_PASSWORD, "/users/backtests.html");
    clickLink("Download Orders");

    assertThat(getResponse()).isEqualTo("<Orders>");
}
```

Don't test through the browser

Selenium is often overkill



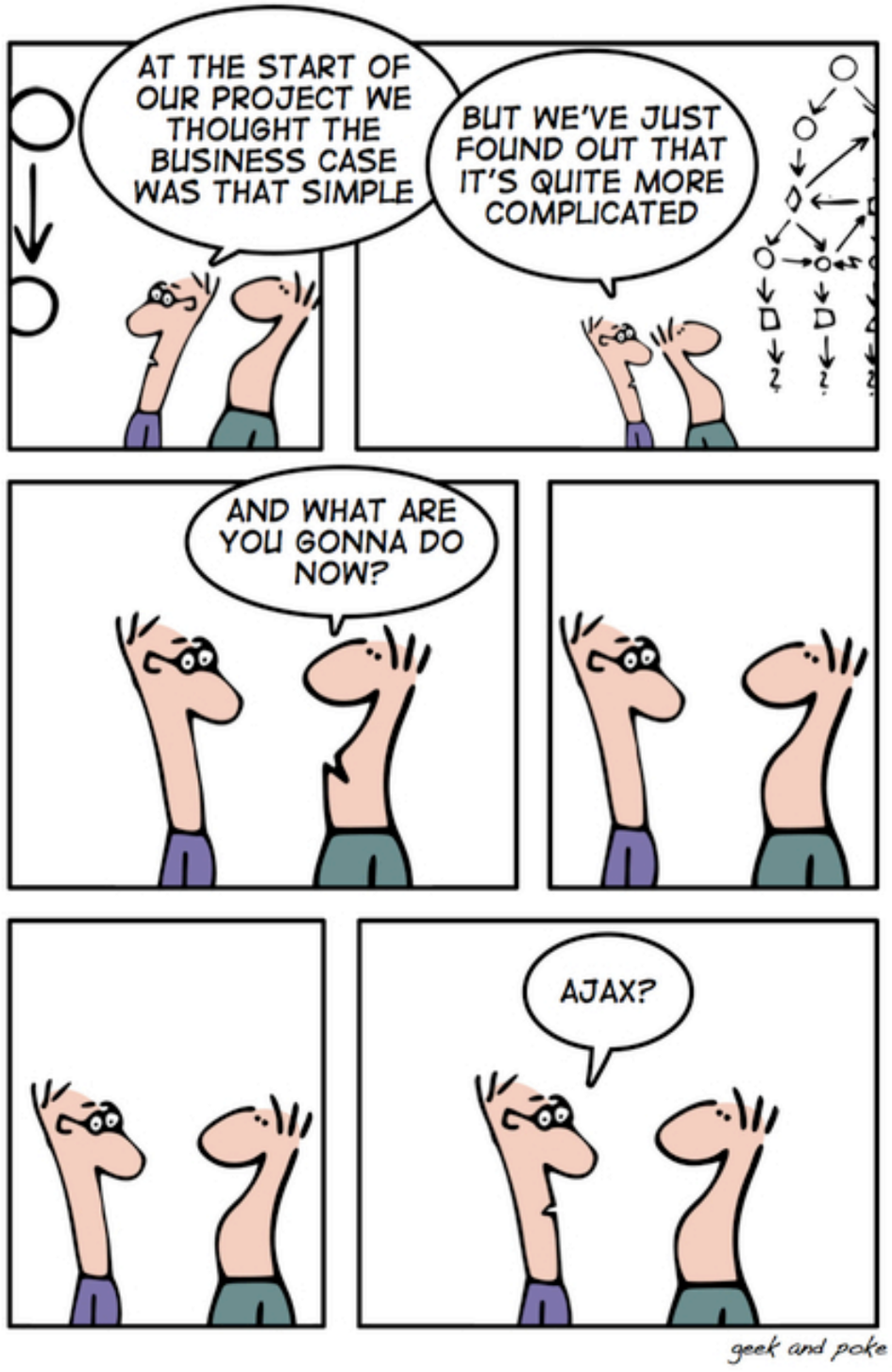
«But my application is complex!»

«My users want complex features,
My users want Ajax»

«I need to test browser compatibility!»

Photo: http://www.flickr.com/photos/zoutedrop/2317065892/

Really ?



http://i.zdnet.com/blogs/complexity-and-false-hope.jpg

Complexity has a cost

That you pay each time a test runs

http://thehongkongfixereng.files.wordpress.com/2008/07/couteau-suisse.jpg

Test through the browser the strict minimal

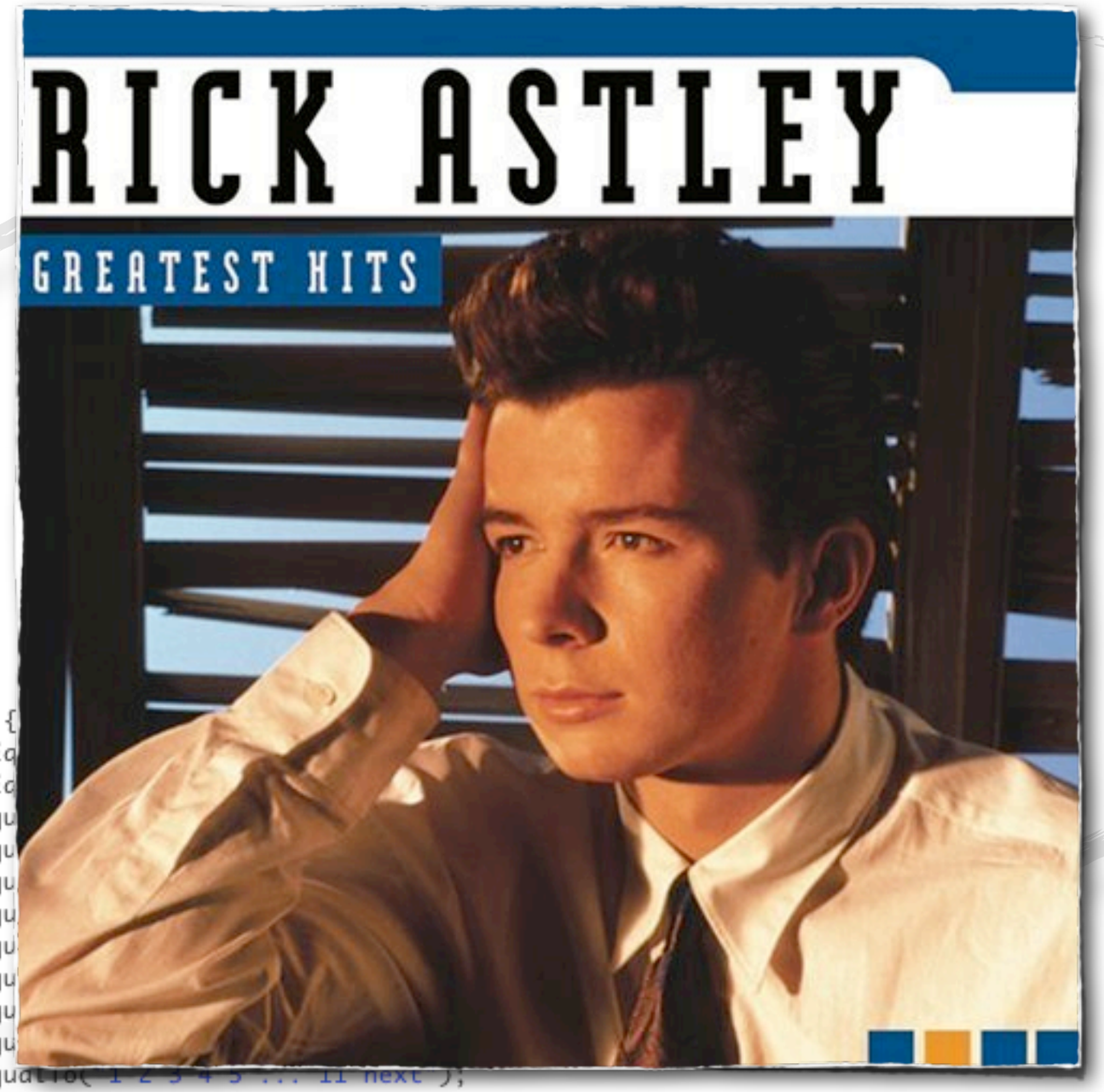
Use javascript unit tests for the rest

```
<html>
 <head>
  <title>Test Page for multiplyAndAddFive(value1, valu
  <script language="javascript" src="jsUnitCore.js"></
  <script language="javascript" src="myJsScripts.js"><
 </head>
 <body>
  <script language="javascript">
    function testWithValidArgs() {
        assertEquals("2 times 3 plus 5 is 11", 11, mul
        assertEquals("Should work with negative number
    }
    function testWithInvalidArgs() {
        assertNull("A null argument should result in n
        assertNull("A string argument should result in
    }
    function testStrictReturnType() {
        assertNotEquals("Should return a number, not a
    }
    function testWithUndefinedValue() {
        assertNull("An undefined argument should result in null", multiplyAndAddFive(2, JSUNIT_UNDEFINED_VALUE));
    }
  </script>
 </body>
</html>
```

http://farm2.static.flickr.com/1168/3166709586_96f9a3fd95.jpg

I tend to be old school

And write server-side code most of the time




```
@Test
public void canPaginate11Pages() {
    assertThat(pages(11, 11)).isEq
    assertThat(pages(10, 11)).isEq
    assertThat(pages(9, 11)).isEqu
    assertThat(pages(8, 11)).isEqu
    assertThat(pages(7, 11)).isEqu
    assertThat(pages(6, 11)).isEqu
    assertThat(pages(5, 11)).isEqu
    assertThat(pages(4, 11)).isEqu
    assertThat(pages(3, 11)).isEqu
    assertThat(pages(2, 11)).isEqu
    assertThat(pages(1, 11)).isEqual
}
```


http://www.celebridiot.com/wp-content/uploads/2009/06/rick_astley_death_hoax.jpg

One more thing...

Simplify and optimize your code

Tests will run faster

Thank you

Q/A

Sponsors Platiniums 2010

OXiane

valtech

NOVEDIA VISION

zenika ARCHITECTURE INFORMATIQUE

OBJET DIRECT

FastConnect

IBM ibm.com/developerWorks

Xebia

Sfeir Groupe