Cutting the right corners

Balancing effort and payoff for GUI test automation

Alexandra Schladebeck, BREDEX GmbH
Once upon a time
  • Fixed price

But then one day…
  • Time and materials

Happily ever after?
  • Attempted solutions and their success
Introductions – BREDEX GmbH

- Software development and consulting
- Focus on quality
  Test tools: GUIdancer and Jubula
- Eclipse Strategic Member
Introductions – Context for Talk

- Customer project which began in 2006
  "Administration and control of measuring equipment"

- Original pilot project for GUI test automation

- Long-running project
  Progress and quality monitored throughout:
  lifecycle – different phases
  changing resources
  changing teams
This talk is about being pragmatic!

In an ideal situation, we have

- a continuous focus on testing
  And a continuous tester presence

But some testing is better than

- no testing at all
  Aim for better, deal with less

And so our story begins…
Once upon a time...

GUI test automation is economically viable and worthwhile.

Extremely high-quality release
But then ...

Move to time & materials

Fewer testing resources – money and people

Tests left to stagnate

More manual testing necessary

Quality suffers

Manual test effort increases
We need a hero!

- **The quest…**
  To get the most information about quality from automated tests as possible

- **The challenges**
  Limited budget and time
  Backlog of tests
  Complex program
Quality information

![Graph showing quality over time](image)

- Quality information
- Handover
- Time
- r = release
- QA = Quality Assurance
- D = development
- automated test
- non automated tests
- Eclipse Testing Day 2012

10.09.2012
Hero candidate number one

- The guy from support
  + people working in support make great testers
  - alongside other duties
  - too many other things to do
  - priority for tests too low
  - practically no introduction
  - no plan made to follow

- Progress slow, practically no benefits
Hero candidate number two

- **Student in Practical Semester (StiP)**
  - full week (40 h) for 1 – 2 months, 3x a year
  - no costs for team
  - may not have testing / area experience

  + preparation
  + 3 day tool training incl. conventions, best practices
  + introduction to software under test by the team
  + systematic test plan created and followed

- had to write new tests (old tests not run now for > 2 years)
StiP does the trick?

- **Progress after first StiP (1 month)**
  - 2000 test steps
  - Test runs for 70 minutes

- **Documentation for following StiP**
  - Current state
  - Use Cases
  - Software-specific conventions
Waiting for next StiP
StiP number two

- **Preparation**
  - Training
  - Introduction to software and existing tests
  - Documentation from previous StiP
  - Discussion of test plan

- **Progress**
  - Addition of new tests, expansion of existing tests: 8000 steps
  - Test runs for 3 hours
Progress

Test Execution History

- StiP 1
- guy from support
- maintenance period between StiPs
- StiP 2
- maintenance period after StiP 2

no tests running

Executed
- FULLTEST

Expected
- FULLTEST
### Progress

![Properties Window]

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Branch Coverage</strong></td>
<td></td>
</tr>
<tr>
<td>Branch Coverage ratio</td>
<td>45.27%</td>
</tr>
<tr>
<td>Covered branches count</td>
<td>8612</td>
</tr>
<tr>
<td>Total branch count</td>
<td>19022</td>
</tr>
<tr>
<td><strong>Class Coverage</strong></td>
<td></td>
</tr>
<tr>
<td>Class Coverage Ratio</td>
<td>78.28%</td>
</tr>
<tr>
<td>Covered Classes Count</td>
<td>1532</td>
</tr>
<tr>
<td>Total Classes Count</td>
<td>1957</td>
</tr>
<tr>
<td><strong>Instruction Coverage</strong></td>
<td></td>
</tr>
<tr>
<td>Covered Instruction Count</td>
<td>142421</td>
</tr>
<tr>
<td>Instruction Coverage Rate</td>
<td>61.68%</td>
</tr>
<tr>
<td>Total Instruction Count</td>
<td>230900</td>
</tr>
<tr>
<td><strong>Method Coverage</strong></td>
<td></td>
</tr>
<tr>
<td>Covered Methods Count</td>
<td>8310</td>
</tr>
<tr>
<td>Method Coverage Ratio</td>
<td>67.57%</td>
</tr>
<tr>
<td>Total Methods Count</td>
<td>12299</td>
</tr>
</tbody>
</table>
Another example
Counter example...

Test Execution History

- Unknown quality
- We haven't known about these test steps for a week
- Don't know if previous errors got fixed
- Looks like they didn't

#Steps

5690

0
The moral of the story

**Nice-to-Haves**
- Full time tester
- Qualified tester
- Constant test growth

**Required**
- Good testers (…)
- Preparation
  - Training in tool
  - Good test design
  - Intro to software
  - Intro to tests
- Closeness to dev team
- Maintenance during “quiet” phases
- Continuous Integration (…)
What is a good tester?

- Qualifications
  - Exams
  - Courses

- Experience
  - User experience
  - Other applications

- Character
  - Curious
  - Courageous
  - Intelligent
  - Motivated and responsible
  - Inventive
Who makes a good tester?

User support

Documentation

Customer liaison

Students*

Trainers
The continuous integration hurdle
The continuous integration hurdle

- **CI must be in place**
  - Quick feedback from new tests
  - Monitoring of quality during maintenance phase
  - → Regression tests must run daily

- **Test environment must be stable**
  - Dedicated machines
  - Known status

- **Test results require reaction!**
  - Daily analysis
  - Fix bugs / alter tests
Strategies for cutting the right corners

1. Set up a continuous build and test process
2. Have “bursts” of new test additions
   Choose your tester carefully
   Provide sufficient familiarization
3. React to regression test results in quiet phases
And they all lived happily ever after