Case study AllianzNet

Automated versus manual testing
Pleased to meet you

• Hans van Loenhoud
  – secretary TestNet
  – managing partner INQA
    • Quality management
    • Test management
    • Service management
    • Project management

• Guido van Leeuwen
  – consultant INQA
    • Test support / test management AllianzNet project
Contents

• The context: Allianz and AllianzNet
• Testing AllianzNet: the strategy
• Automated testing: what did we do?
• And here are the results …
• Next time, we will …
Allianz

- A company of the Allianz Group, one of the largest integrated financial service providers in the world, with over 60 million clients in over 70 countries

- Allianz Schadeverzekering NV
  - in the Dutch general insurance market, one of the largest players, in particular in the fire and motor classes
  - distribution exclusively through the intermediary channel (insurance agents)
AllianzNet – the past

agent

back office

front office

customer
AllianzNet – the future
AllianzNet – the application

• Internet application, developed on COTS Java based web engine
  – replaces current mainframe based front office system
• Automated input forms for 6 insurance types
  – home, car, motor, caravan, boat, legal assistance
• Online ...
  – completion, verification and validation of data
  – calculation of premiums
  – acceptance of insurance
• Work flow between agents and Allianz
AllianzNet – the project

- First phase 1½ years (proposals & new orders)
  - >25 man years spent on IT work
    - of which >25% on testing
  - separate development teams
    - vendor
      - enhancements to web engine
    - Allianz
      - infrastructure
      - customisation of web engine
      - interfaces between web based front office and mainframe based back office system
  - step-wise development
    - multiple releases, builds and patches
  - November 7, 2005: live!
Testing strategy - risk

- Extremely high risk project
  - new business process
  - new technology
  - new methodology
  - new organisation (both business and IT)
  - high complexity
  - high volumes
  - high potential damage (e.g. acceptance of bad risks)
  - high external visibility
Testing strategy - master test plan

- **Unit testing**
  - developer teams
- **System testing**
  - dedicated test team
- **Integration testing**
  - dedicated test team
- **Acceptance testing**
  - dedicated business team (UAT)
  - exploitation & maintenance team (PAT)
  - outsourced (performance, security)
Testing strategy - test automation

• High complexity (30 years of insurance knowledge) ⇒
  – current front office system contains “the truth”
  – automated output prediction from front office system
    • validations
    • premiums

• High volumes (~ 1000 concurrent agents) ⇒
  – automated performance tests

• Multiple releases ⇒
  – automated regression tests

• Existing tool: Winrunner
Let’s test!
What did we do …

• Automated
  – premium calculation
  – performance
  – regression testing

• Manual
  – premium calculation
  – workflow
  – field validations
  – verification rules
  – acceptance rules
  – external connections
What did we do - time line

Test project

- Preparation
  - Sept 2004

- System test
  - Dec 2004
  - April 2005

- Integration test
  - Oct 2005
  - Sept 2005

- Acceptance test
  - Today

- Live!
  - Today

- Testing strategy
- Building scripts

- Premium testing
  - June 2005
  - Today

- Test conditions
- Test cases
The figures

• Conditions, cases and findings

<table>
<thead>
<tr>
<th></th>
<th>Conditions</th>
<th>Cases</th>
<th>Findings</th>
<th>Effort (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated</td>
<td>140</td>
<td>450</td>
<td>90</td>
<td>1350</td>
</tr>
<tr>
<td>Manual</td>
<td>100</td>
<td>1300</td>
<td>810</td>
<td>1250</td>
</tr>
</tbody>
</table>

• Effort on automated testing
  – 40% during system testing
  – 20% during integration testing
  – 5% during acceptance testing
The findings

- Total amount of findings 900
  - of which found by automated testing 90
Evaluating the testing strategy - test automation

- Automated premium calculation tests
  → Succeeded, at a cost:
  manual testing: 3,5 hours per finding
  automated testing: 15,0 hours per finding

- Automated performance tests
  → Succeeded, could not have done without
    (reusing existing scripts)

- Automated regression tests
  → Did not utilize, except for premium testing
Lessons learned

- (To) Early start of building scripts
  - time lost in rebuilding scripts

- Quality of test scripts determines value of automated testing
  - need for structured design of test scripts

- Be selective
  - consider benefits versus costs
Recommendations

- Choice for automated testing should depend on
  - reusability of test scripts
  - stability of application
  - need for flexibility

- When choosing for automated testing, make sure
  - there is a clear and smart scope for automated testing
  - there is a structured design approach for developing standardised test scripts
  - enough expertise on automated testing is available
  - maintenance of test scripts is embedded