Test Confessions

What Eclipsers Think & Do About Testing

Arie van Deursen
Michaela Greiler
Margaret-Anne Storey

Presentation for the symposium to celebrate 25 years of West Consulting
The TU Delft
Software Engineering Research Group

Education
• Programming, software engineering
• MSc, BSc projects

Research
• Software architecture
• Software testing
• Repository mining
• Collaboration
• Services
• Model-driven engineering
• End-user programming
The primary goal of Ada-Nederland is to stimulate the use of Ada, to act as a bridge between companies that use or want to use Ada, and to be the contact point for international Ada organizations like Ada-Europe. Ada-Nederland strives at a balanced membership of representatives of industry, government and academia. Ada-Nederland is chaired by Rob Westermann (West Consulting). Members of the board are Pieter Verduin (DEC), Jan van Katwijk (TU Delft), Dick Fikkert (TEL-TNO), and At Hijwegen (Data Sciences). Ada-Nederland currently has over 100 individual members. Anyone who is interested in active participation is invited to join.
Crawljax: Automated Testing of Ajax Applications
(Testing) Plug-in Architectures

Create series of tailored products by combining, configuring, & extending plug-ins

[Images of logos for Eclipse, OSGi Alliance, Firefox, and WordPress]
List of Eclipse-based software

The Eclipse platform can be extended by adding different plug-ins, for example:

- Adobe Flex Builder, Adobe IDE based on Eclipse for building Flex applications for the Flash Platform
- AnyLogic, a simulation modeling tool developed by XJ Technologies.
- Aptana, Web IDE based on Eclipse (commercial and community version)
- Avaya Dialog Designer, a commercial IDE to build scripts for voice self-service applications.
- Bioclipse, a visual platform for chemo- and bioinformatics.
- BIRT Project, open source software project that provides reporting and business intelligence capabilities for rich client and web applications.
- Bonita Open Solution relies on Eclipse for the modeling of processes, implementing a BPMN and a Web form editors.
- Borland JBuilder 2007, based on Eclipse
- ColdFusion, a plug-in for Eclipse that provides an Integrated Development Environment and editor for the ColdFusion Programming Language.
- Eclipse Rich Text Processing or RText editor
- Compuware OptimalJ, a model-driven development environment for Java
- EasyEclipse, bundled distributions of the Eclipse IDE
- g-Eclipse, an integrated workbench framework to access the power of existing Grid infrastructures
- Insomniac, an Eclipse-based, cross-platform, peer-to-peer instant messaging tool
- Kollabor, a collaborative development framework for Eclipse
- Eclipse integration for multiple functions
- Eclipse plug-in for deploying cloud applications
- eXtreme, a framework for design, development, analysis, testing, profiling and deployment of Java-based applications
- MOFclipse, a business governance solution for creating, modifying, governing, finding and reusing any type of model content
- Orion, a business process modeling tool to test Web, Java, .NET, Siebel, SAP and Oracle applications
- Compatible with Eclipse, it supports design with UML.
- Quantrum, a business performance management tool that supports the presentation and cause of system performance bottlenecks.
- Eclipse and the delivery platform
- Reliable Software development and delivery platform
- Eclipse plug-in to deploy cloud applications
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process modeling
- Eclipse plug-in for business performance management
- Eclipse plug-in for business governance
- Eclipse plug-in for business process model
Underneath: OSGi

- Routers, Modems, Gateways, Control Panels, Phones, Cars, Trains, Trucks, Healthcare devices...
Eclipse @ West: Overture

VDM Projects

Switch perspective

VDM Model files

VDM Editors

Outline view

Problem view
One Product = Many Plug-ins
Set of Plug-ins = Many Products

What are the Test Implications?

How should we test plug-in architectures?
What do Eclipsers Think about Testing?
Research Questions

1. What testing practices are prevalent in the Eclipse community?

2. Does the plug-in nature of Eclipse have an impact on software testing?

3. Why are certain practices adopted, and why are others not adopted?

4. Are there additional compensation strategies used to support testing of plug-ins?
## 25 Interviews

<table>
<thead>
<tr>
<th>Domain</th>
<th>Project and/or Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDEs, Eclipse Distribution</td>
<td>Yoxos, EclipseSource</td>
</tr>
<tr>
<td>SOA</td>
<td>Mangrove, SOA, Inria</td>
</tr>
<tr>
<td>GUI Testing Tool</td>
<td>GUIDancer, Bredex</td>
</tr>
<tr>
<td>Version Control Systems</td>
<td>Mercurial, InlandSoftware</td>
</tr>
<tr>
<td>Modeling</td>
<td>xtext, Itemis</td>
</tr>
<tr>
<td>Persistence layer</td>
<td>IMP, University of Amsterdam</td>
</tr>
<tr>
<td>Domain Specific Language</td>
<td>CDO</td>
</tr>
<tr>
<td>BPM Solutions</td>
<td>Spoofax, TU Delft</td>
</tr>
<tr>
<td>GUI Testing Tool</td>
<td>GMF, BonitaSoft</td>
</tr>
<tr>
<td>Coverage Analysis</td>
<td>Q7, Xored</td>
</tr>
<tr>
<td>Modeling</td>
<td>EclEmma</td>
</tr>
<tr>
<td>BPM Solutions</td>
<td>EMF, Itemis</td>
</tr>
<tr>
<td>Scientific data acquisition</td>
<td>RCP product, AndrenaObjects</td>
</tr>
<tr>
<td>Runtime platform</td>
<td>OpenGDA, Kichacoders</td>
</tr>
<tr>
<td>Task Management system</td>
<td>RAP, EclipseSource</td>
</tr>
<tr>
<td>Embedded Software</td>
<td>Mylyn, Tasktop</td>
</tr>
<tr>
<td>RCP product</td>
<td>MicroDoc</td>
</tr>
<tr>
<td></td>
<td>EclipseSource</td>
</tr>
</tbody>
</table>
Grounded Theory

• Systematic procedure to discover theory from (qualitative) data

Grounded

• Theoretical sensitivity
• Open coding
• Theoretical coding
• Theoretical sampling
• Constant comparative method
• Selective coding
• Memoing


Unit testing is popular

“Unit testing is where you find the most bugs”

“Ultimately, unit test are our best friends”

“At least 70% of our test effort is spent on unit testing.”
Other forms of testing are less popular

“We think that with a high test coverage through unit tests, integration tests are not necessary.”

“The Capture and Replay nature of QF-tests was too rigid when the system was evolving.”

“We haven’t been 100% satisfied with capture-replay: too much is captured.”
First Findings

• Common practice to have no separate test teams

• Eclipsers are proud of their unit testing

• Eclipsers tend to dislike system, integration, UI, and acceptance testing
  – Substantially less automation
Cross plug-in testing is optional

“We do bug-driven cross plug-in testing”

“We have no automated tests for cross plug-in testing, but we do manual testing.”
Version testing is minimal

“A lot of people put version ranges in their bundle dependencies, and they say they can run with 3.3 up to version 4.0 of the platform.”

“But I’m willing to bet that 99% of the people do not test that their stuff works.”
Findings 2: Plug-ins

• Testing deferred to `application engineering’
  – No special effort during `product line engineering’

• Integration testing on demand:
  – Bug occurring in the field

• No test effort aimed at integration faults per se
  – Versions, configurations, interactions, ...
“It’s complicated to integrate Junit with the build. Another framework? I didn’t want to take the trouble”

“And you never know, once you write a good test, then it will become obsolete with the next version of Eclipse”

“Especially for plug-ins, we would need some best practices.”
Findings 3: Barriers

- Responsibility for integration unclear
- Requirements for composite unclear
- Lack of ownership
- Insufficient plug-in knowledge
- Set-up of test infrastructure too complicated
- Test execution too long
- Poor testability of the platform
Community Testing (I)

Testing is done by the user community. [...] We have more than 10,000 installations per month. If there should be a bug it gets reported immediately.”

“The community helps to test the system for different operating systems, and versions. They are very active with that.”
Community Testing (II)

“I would say the majority of the bug reports come from the community. [...] We have accepted more than 800 patches.”

“We make all infrastructure available, [...] so that somebody who writes a patch has the opportunity to run the same tests [...]”
“Downstream Testing

“We’re a framework. If the user downloads a new version and lets his application run with it, then this is already like a test.”

“They have extensive unit tests, and so I am quite sure that when I break something, somebody downstream very rapidly notices and reports the problem.”
Findings 4: “Compensation Strategies”

- Community plays key role in finding and reporting issues.
- Downstream testing (manual and automatic) provides additional tests of upstream framework.
- Open test infrastructure facilitates patching.
Summary: Findings

1. (Automated) unit testing is widely adopted; Integration, system, UI and acceptance testing are much less automated.

2. The plug-in nature has little direct impact on test practices.

3. Barriers to adopt techniques include unclear ownership, accountability, and test effort & execution time.

4. Limited integration testing is compensated by community.
Scope

• Beyond the participants:
  – Challenged results in survey among 150 Eclipsers

• Beyond Eclipse:
  – Open source, developer centric, plug-in architecture, services, ...

• Beyond the people:
  – Repository mining, code analysis
Implications

1. Community tolerance for failures determines (integration) test effort

2. Need to strengthen community

3. Need to strengthen plug-in architecture with “self testing” capabilities

4. Test innovations must address adoption barriers
The next 25 years...

Who is the integrator?

“To engineer is human”

How can we improve ourselves?
Further Reading

Michaela Greiler, Arie van Deursen & Margaret-Anne Storey.


Technical report TUD-SERG-2011-010, Delft University of Technology

www.se.ewi.tudelft.nl