How to Profit from Static Analysis

Elena Laskavaia, Research In Motion, Eclipse Con, March 2012
How to Profit?

- Use Static Analysis Tools
- Sell Static Analysis Tools
  - not discussing today 😊
Where is the Profit?

![Bar chart showing the comparison of development cost, profit, and extra cost between Before, Ideal, Real, Real II, and Real III.]
What is Static Analysis?

Analysis of source code without running it

- Bug Detection
- Security
- Code Style
- Compliance
- Reverse Engineering
- Visualization
- Metrics
- X-Ref
- Refactoring
- Unused Code
- Code Clones
- Code Cleanup
## Cost of fixing one bug

<table>
<thead>
<tr>
<th>Stage</th>
<th>Bug Life</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>As you type</td>
<td>1 s</td>
<td>1 cent</td>
</tr>
<tr>
<td>Developer build</td>
<td>10 sec</td>
<td>10 cent</td>
</tr>
<tr>
<td>Developer testing</td>
<td>10 min</td>
<td>$3</td>
</tr>
<tr>
<td>SCM check in</td>
<td>4 h</td>
<td>$10</td>
</tr>
<tr>
<td>Integration build</td>
<td>1 d</td>
<td>$40</td>
</tr>
<tr>
<td>Integration testing</td>
<td>10 days</td>
<td>$200</td>
</tr>
<tr>
<td>In the field</td>
<td>from 30 days</td>
<td>$1000+</td>
</tr>
<tr>
<td>In outer space</td>
<td>3 years</td>
<td>$100 million*</td>
</tr>
</tbody>
</table>

![Graph showing the cost of fixing bugs at different stages](image)
Gain

- Finding of bugs for “Free”
- Detection of security vulnerabilities
- Enforcement of code style
- Assistance on software certification and audit
- Help with testing and regression testing
- Automation of code review
- Generation of software metrics trends
Cost

- Cost of Tools
- Installation and Setup
- Users Training
- Build Integration and Maintenance
- Tuning and Customization
- Problems Triage
Thinking to buy a tool?

- Use maximum of your IDE and compiler
- If you need external tools try to pick ONE
- Evaluate tool before buying
- Pay for integration
- Tune it
- Customize it
Good tool should

- Integrate into existing software lifecycle!
- Be customizable
- Hide false positives without code modification
- Have ability to report new errors only
- Use adaptive error reporting
- Auto-correct errors
- Explain itself well
Why so many False Positives

- Incorrect integration
- Failure to tune: noise
- Obscure code
- Bug in tool
Integration

- Developers Workstation
  - Integrated as you type
  - In local build
  - As unit tests
  - As extra “tool” to run on code
Integration cont’d

- **Code Review**
  - Bot code reviewer (gives −1 or +1 on review 😊)

- **Commit Hook**
  - Veto on code submission

- **Integration Build**
  - Send auto-alerts on new bugs
  - Provide trends on sw quality, maintainability, size

- **Run on Demand**
  - Audit – FDA, DO-178B, MISRA
  - Find bugs for external triage
Myths

- All bugs have to be fixed
- Free tool means no cost
- The more tools we buy, the more bugs they find, the better it is for us
- If we pay for the tools they will find all our bugs
- Tool just works out of the box
- Developers love to use a new tool
How to make it successful

- Zero Tolerance
- Warnings are Evil
- Progressive Exposure
- Stop the Bleeding
- Pick ONE tool for bug detection
- Smart code style
- Don’t argue, fix the code!
Ways to make it non-profitable

- Buy 5 tools and add another 5 free ones
- Add all errors directly into bug tracking database
- Report code style errors instead of auto-correct
- Enforce obsolete and irrelevant code standards
- Enable errors that you don’t care about
- Enable errors which have limited applicability
# How to pick problem types

<table>
<thead>
<tr>
<th>Type</th>
<th>How it affects the code</th>
<th>F.P. Rate</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming Errors</td>
<td>Cause Program Misbehaviour</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Code Style</td>
<td>Fail Certification</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Security Violations</td>
<td>Cause security exploits</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Unused Code</td>
<td>Bigger Code Footprint</td>
<td>Low</td>
<td>Med</td>
</tr>
<tr>
<td>Visibility Reduction, Code Clean-up</td>
<td>Poor Maintainability</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Performance</td>
<td>Code Executes Slower</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>
Eclipse Integrations
Turn Warnings into Errors in JDT
Enable Format on Save

- **Save Actions**
  - Enable project specific settings
  - Perform the selected actions on save
    - Format source code:
      - Format all lines
      - Format edited lines
    - Configure the formatter settings on the *Formatter* page.
  - Organize imports
    - Configure the organize imports settings on the *Organize Imports* page.
  - Additional actions:
    - Add final modifier to private fields
    - Add missing '@Override' annotations
    - Add missing '@Override' annotations to implementations of interface methods
    - Add missing '@Deprecated' annotations
    - Remove unnecessary casts

- [OK] [Apply] [Restore Defaults] [Cancel]
Code Analysis in CDT

- Finds errors as you type
- Provides quick fixes
Klocwork C/C++
public FileScopeComposite(Composite parent, final IProblem problem, IResource resource) {
    super(parent, SWT.NONE);
    if (problem == null)
        throw new NullPointerException();
    this.setLayout(new GridLayout(2, false));
}
Find Bugs Java

```java
this.ast = ast;

public IASTTranslationUnit getAST() throws OperationCanceledException, CoreException {
    return getAST(tu);
}

public IASTTranslationUnit getAST(ITranslationUnit tu)
    throws OperationCanceledException, CoreException {
    if (!this.tu.equals(tu)) {
        throw new IllegalArgumentException();
    }
    if (ast == null) {
        getIndexed();
        ast = tu.getAST(index, PARSE_MODE);
    }
    return ast;
```
UCDetector (Unused Code)

```java
public static void log(String message) {
    getDefault().getLog().log(new Status(IStatus.ERROR, PLUGIN_ID, message));
}

/**
 * @param e
 * @return
 */
public static IStatus getStatus(Throwable e) {
    return new Status(IStatus.ERROR, PLUGIN_ID, e.getLocalizedMessage(), e);
}
```

Problems

<table>
<thead>
<tr>
<th>Description</th>
<th>Path</th>
<th>Location</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCDetector Marker - References (1 item)</td>
<td>/org.eclipse.cdt...</td>
<td>line 75</td>
<td>UCDetector ...</td>
</tr>
<tr>
<td>Method &quot;CodanCheckersActivator.log(String)&quot; has 0 references</td>
<td>/org.eclipse.cdt...</td>
<td>line 75</td>
<td>UCDetector ...</td>
</tr>
<tr>
<td>UCDetector Marker - Visibility: Use private (3 items)</td>
<td>/org.eclipse.cdt...</td>
<td>line 23</td>
<td>UCDetector ...</td>
</tr>
<tr>
<td>Change visibility of Constant &quot;CodanCheckersActivator.PLUGIN_ID&quot; to private</td>
<td>/org.eclipse.cdt...</td>
<td>line 23</td>
<td>UCDetector ...</td>
</tr>
<tr>
<td>Change visibility of Method &quot;CodanCheckersActivator.getDefault()&quot; to private</td>
<td>/org.eclipse.cdt...</td>
<td>line 64</td>
<td>UCDetector ...</td>
</tr>
<tr>
<td>Change visibility of Method &quot;CodanCheckersActivator.getStatus(Throwable)&quot; to private</td>
<td>/org.eclipse.cdt...</td>
<td>line 83</td>
<td>UCDetector ...</td>
</tr>
</tbody>
</table>
Summary

- Maximize gain
  - Figure out what you want
  - Find the right tool(s)
- Minimize cost
  - Integrate
  - Tune
  - Enforce
  - Educate
Thank You!

Elena Laskavaia, elaskavaia@qnx.com