ABL source code analysis

Code quality management with SonarQube
Code quality
Audience survey
Why?

- Because you have to maintain your code
  - Low quality code is harder to maintain and to improve

- And you want to maximize profit!

- Unless you’re working by the hour and your customers are happy with bad code, it will cost money on the long term
Technical debt

Term coined by Ward Cunningham in 1992

Metaphor around the time it takes to work with low quality code:

- If technical debt is high, you’ll pay high interest (in terms of time spent) each time you work on the project
- Pay down the principal by refactoring code
- Increase the debt if you add quick and dirty code
SonarQube

- Code quality management platform
- Open source
- Extensible
- Over 20 languages (from COBOL to Javascript)
## SonarQube

### Apache Commons Collections

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>221</td>
</tr>
<tr>
<td>Security</td>
<td>1</td>
</tr>
<tr>
<td>Maintainability</td>
<td>1,301</td>
</tr>
<tr>
<td>Coverage</td>
<td>82.9%</td>
</tr>
</tbody>
</table>

### Reliability
- **Bugs**: 0
- **New Bugs**: 0
- **Reliability Rating**: A
- **Remediation Effort**: 0d

### Security
- **Vulnerabilities**: 1
- **New Vulnerabilities**: 0
- **Security Rating**: A

### Maintainability
- **Code Smells**: 307
- **New Code Smells**: 0
- **Maintainability Rating**: A
- **Technical Debt**: 22d
- **Added Technical Debt**: 0.2%
- **Technical Debt Ratio**: 1.7%
- **Technical Debt Rate on New Code**: 42.3%
- **Effort to Reach Maintainability Rating A**: 0d

### Coverage
- **Line Coverage**: 100.0%
- **Unit Tests**: 16,066
- **Uncovered Lines on New Code**: 0
- **Uncovered Conditions on New Code**: 0
- **Uncovered Conditions on Old Code**: 0
- **Uncovered Loops on New Code**: 0
- **Uncovered Loops on Old Code**: 0
- **Uncovered Methods on New Code**: 1
- **Uncovered Methods on Old Code**: 11,011
OpenEdge plugin

- Open source plugin written by Riverside Software
- Common metrics (LOC, comments, complexity, …)
- Code duplication
- Code coverage
- XREF and code parsing
- Extensible
OpenEdge plugin - Requirements

▪ Automated build process
  ▪ PCT is strongly recommended

▪ Analysis triggered by Java
  ▪ Continuous integration server is strongly recommended
Analyzing legacy projects

- Technical Debt: 16kd Debt, 239k Issues
- Duplications: 24.2% Duplications, 12k Duplicated Blocks
- Structure: OpenEdge 100.0%, 672k Lines of Code
OpenEdge plugin - Integration
Lexer and parsers

- KEYWORD : 'define' | 'variable' | 'property' | 'as' | 'character' | 'integer'
- WS : ( ' ' | '\t' | '\n' )* -> HIDDEN
- ID : [a-zA-Z]*
- EOS : '.'

define variable foo as integer.
define property bar as character.
Lexer and parsers

- defVar: KW_DEFINE KW_VARIABLE ID KW_AS DATATYPE KW_NO_UNDO? (KW_LABEL QUOTED_STRING)?
- defProp: KW_DEFINE (KW_PUBLIC | KW_PROTECTED | KW_PRIVATE)? (KW_STATIC | KW_ABSTRACT)? KW_OVERRIDE? KW_PROPERTY ID KW_AS (DATATYPE | CLASSNAME)
Lexer and parsers
Beta period

- Full support until end of June 2016
- Rules available for free
- Looking for feedback in terms of metrics, rules and bugs

- Try it!
  - https://github.com/Riverside-Software/sonar-openedge
Under the hood

- SonarQube is (almost) entirely written in Java
- ABL Parser based on Proparse
- Extensible rules engine (Java only)
- Much faster than Prolint
- Other parsers based on ANTLR4
Questions ?
Reference?

▪ Sonar Source: http://www.sonarsource.com

▪ Riverside Software: http://riverside-software.fr
▪ Contact: contact@riverside-software.fr