



Thomas Zierer

Combining SonarQube and PL/I V5 for Continuous Code Quality Inspection

October 12, 2017, GSE EM Working Group Meeting, Nürnberg

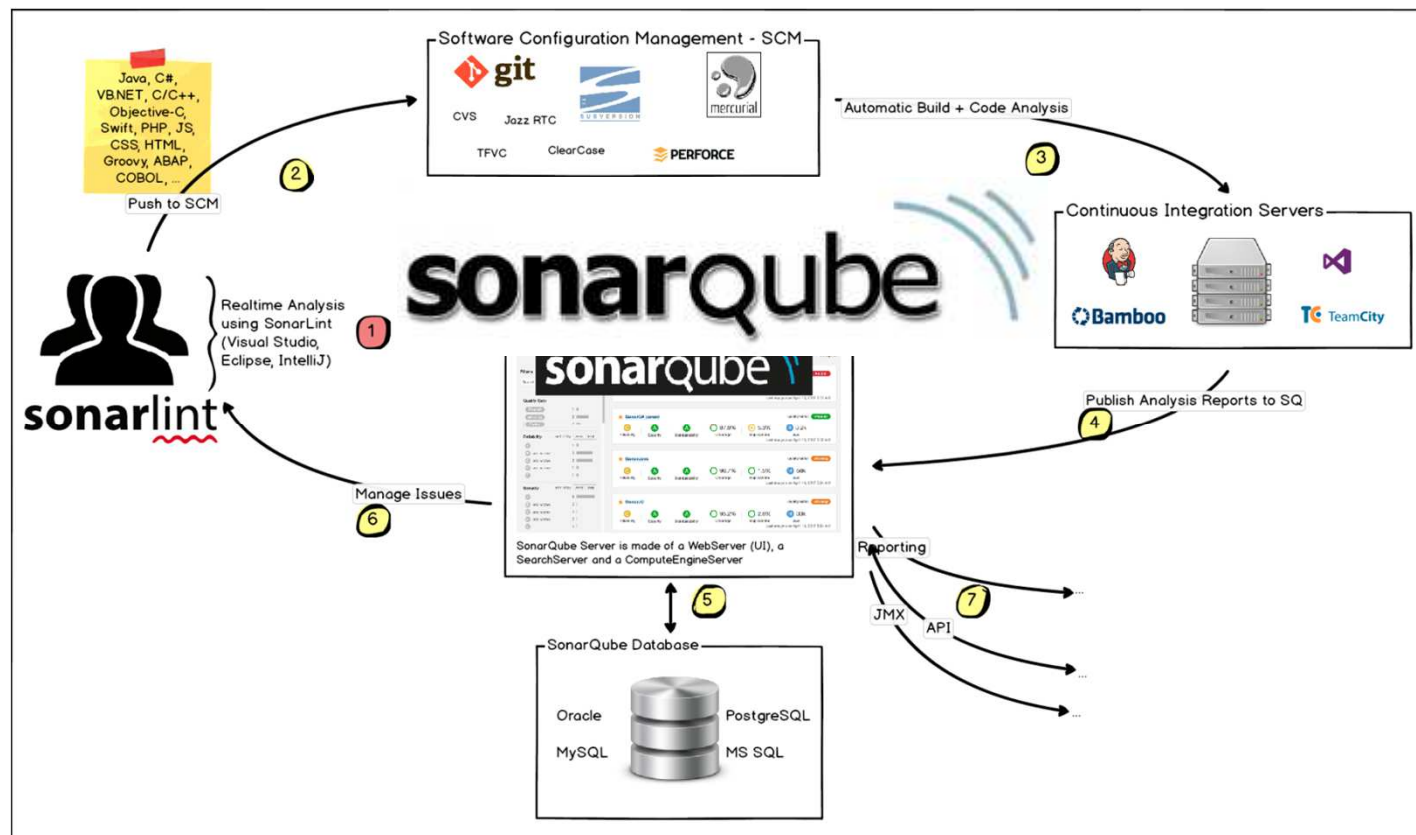
SonarQube

SonarQube is the leading Open Source product for Continuous Code Quality Inspection

- 700 enterprise customers worldwide + 80.000 other installations
- Very popular in the Open Source Community
- Maintained by SonarSource, Switzerland
- IDE Plugins (SonarLint) for Eclipse, IntelliJ and Microsoft Visual Studio for instant feedback
- Batch-scans are often integrated as ant, Maven or MSBuild Tasks and run by a build server like Jenkins

SonarQube - Architecture

The batch-scan reads the sourcefiles from the SCM, analyzes them with language-specific scanners and feeds the results to the SonarQube-Web-UI for review



SonarQube

Demo

SonarQube

The commercial version supports more languages and offers enterprise features

- Open Source version:
 - Supports Java, C#, PHP, Python,...
 - Suitable for lightweight, homogenous projects
- Commercial version:
 - Enterprise Features (Governance, Reporting, ...)
 - Additional languages: C/C++, COBOL, **PL/I**, ...

So, why this session?

SonarQube and PL/I

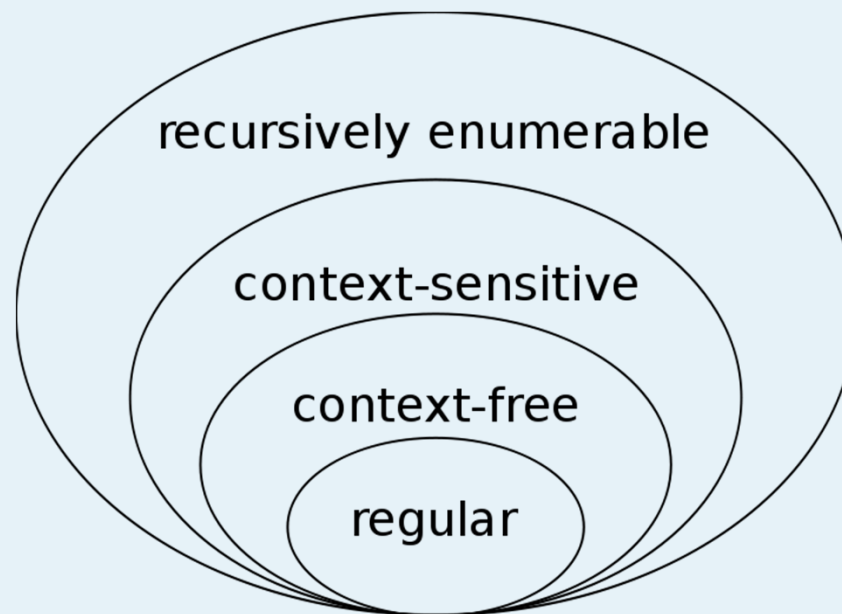
SonarQube's PL/I support is limited and concentrates on distributed PL/I

- PL/I support requires at least SonarQube Professional + PL/I Plugin. => 21.000 € / year
- Scanner is meant for distributed PL/I, not z/OS
- Plugin offers only 21 rules for the PL/I language
- The scanner often fails for valid Enterprise PL/I code
- As is the case with most custom PL/I scanners

Why is it so difficult to analyze PL/I code?

Languages like Fortran and PL/I are less strongly structured than e.g. Java or C#

- In 1956 famous linguist Naom Chomsky published his hierarchy of formal languages



Why is it so difficult to analyze PL/I code?

Most custom scanners are based on tools like lex and yacc which require the language to be at least context-free

- This makes it comparatively easy to write scanners for Java or C# based on tools like lex and yacc
- Fortran and PL/I are known to be context-sensitive and *not* context-free.
- This means no such scanner will ever support PL/I in it's full strength

In 25 years I came to the conclusion:

**The only reliable PL/I
scanner is IBM's
Enterprise PL/I compiler**

Enterprise PL/I

The Enterprise PL/I Compiler provides 2 highly useful features:
RULES and **XINFO**

- The RULES directive supports ~40 suboptions especially for QA, e.g. RULES(NOLAXIF) flags IF conditions which do not evaluate to boolean
- The XINFO(XML) directive exports the compiler messages in XML format

```
<MESSAGE>
  <MSGNUMBER>IBM2418I E</MSGNUMBER>
  <MSGLINE>22</MSGLINE>
  <MSGFILE>21</MSGFILE>
  <MSGTEXT>Variable FORCE is unreferenced.</MSGTEXT>
</MESSAGE>
```

SonarQube

With its custom language support SonarQube can be extended to analyze new languages

- SonarQube is extensible through its plugin concept
- Writing your own plugin is not trivial but there are some good examples
- Documentation is provided here:
<https://docs.sonarqube.org/display/DEV/Extension+Guide>

The Xinfo plugin

The Xinfo plugin combines Enterprise PL/I with SonarQube's custom language support

- The plugin parses the compilers XML output and feeds it to SonarQube's REST interface
- This works for every language that supports a XINFO-like directive: PL/I, COBOL, C/C++, Assembler
- Scans may run on every kind of server: Windows, Unix, z/OS.
- Restriction on z/OS: PL/I source and XML output must reside on OMVS

The Xinfo plugin: Executing scans

A simple shell-, batch- or JCL script executes the scan. It uses the command line client which comes along with SonarQube

- Scan is invoked via SonarQube commandline client
- Successfully tested on Windows, Unix
- Prereq: Sonar Server 6.3 or higher
- On z/OS the scan is invoked using the IBM utility BPXBATCH:

```
//BPXBATCH EXEC PGM=BPXBATCH,  
// PARM='sh /u/y08577/sonar-scanner-2.8/bin/sonar-scanner'  
//STDOUT DD SYSOUT=*  
//STDERR DD SYSOUT=*  
//STDENV DD *  
/*
```

The Xinfo plugin – PL/I Basic

Demo

Exploiting more SonarQube APIs

Integration for Debug Tool and CPD (Copy/Paste Detection)

- A converter transforms Debug Tool Code Coverage files into the corresponding SonarQube format to determine the test coverage
- Split the sourcecode into tokens and let SonarQube compute duplications.

The Xinfo plugin – PL/I extended

Demo

Wanna try it?

The Xinfo plugin is Open Source

- Xinfo plugin source code is published under Eclipse Public License (EPL) v. 2.0
- Clone-URL on last slide
- Every comment is honestly appreciated

Whats next?

There is still a lot of work to do

- Remediation costs per rule to compute the technical debt
- Custom measures (for instance refactor programs with too many INCLUDEs / copybooks)
- Automatically assign a finding to its author by questioning the SCM („blame“)



Thank you very much for your attention!

Thomas Zierer

E-Mail: thomas.zierer@muenchen-mail.de

Fork me on GitHub: <https://github.com/tgmz/sonar-xinfo-plugin>