Software Engineering

Prof. Dr. Bertrand Meyer

March 2007 – June 2007

Automatic code inspection
Automation of code reviewing

Possible bugs
Code convention violations
Dead code
Duplicated code
Suboptimal code

According to a report released by The Standish Group automated code inspection reduced the number of people needed for manual code reviews by 50%.
Main concept

Syntactical matching of rules violation

Rule – is description of set of ASTs that can be reason for the one of the mentioned above problems.
How it works

For expressing some rules representation of AST should contain whitespaces, tabulations, EOLs, comments
Rules represented via Java code or XML XQuery
Existing tools

**Checkstyle 4.1** - Checkstyle

http://checkstyle.sourceforge.net/

- PMD

http://pmd.sourceforge.net/

- FindBugs

http://findbugs.sourceforge.net/

- JCSC (Java Coding Standard Checker)

http://jcsc.sourceforge.net/
Why PMD?

Support user rules creation via both Java code and XML

XQuery

Standard rules cover wide range of rule types

Stable version

Support plug-ins for various IDE
What does ‘PMD’ mean?

Pretty Much Done
Project Mess Detector
Project Monitoring Directives
Project Meets Deadline
Programming Mistake Detector
Pounds Mistakes Dead
PMD Meaning Discovery (recursion, hooray!)
Programs of Mass Destruction
A 'Chaotic Metal' rock band name
   “Pretty Marry Dies”
Application of the PMD

Select standard rules
Write project specific rules
Select rules parameters

Source code → Changing → PMD → Warnings comments writing
New source code → Source code → Warnings

Bug fixing → Comments

Software Engineering, lecture: Automatic code inspection
Future development: automatic bug fixing

Program transformation via rewriting rules. For example StrategoXT.

http://www.program-transformation.org/Stratego/JavaFront

\[
\begin{align*}
R_1 &\rightarrow R'_1 \\
R_2 &\rightarrow R'_2
\end{align*}
\]
Future development: integration with a prover

```java
requires i > 5;
int f(int i){
    if (i < 2) /*UnconditionalIfStatement*/
        return i +1;
    else
        return i -1;
}
```
Future development: empirics for filtering warnings

**Goal:**
- Decrease warnings number
- Remove false warnings

**Problems:**
- Fuzzy warning criteria
- Context depending warnings
Pros and cons

Pros

- Don’t require any additional efforts.
- Users don’t need have any specific knowledge. It’s enough that user understand notion of the AST.

Cons

- Not sound. It’s possible that many from the found warnings are not real errors.
- Not complete. It can miss many real errors.