代码质量管理

静态分析工具

## 概述

在软件开发过程中，开发团队往往要花费大量的时间和精力发现并修改代码缺陷。 静态代码分析（static code analysis）工具能够在代码构建过程中帮助开发人员快速、有效的定位代码缺陷并及时纠正这些问题，从而极大地提高软件可靠性并节省软件开发和测试成本。

## 什么是静态代码分析

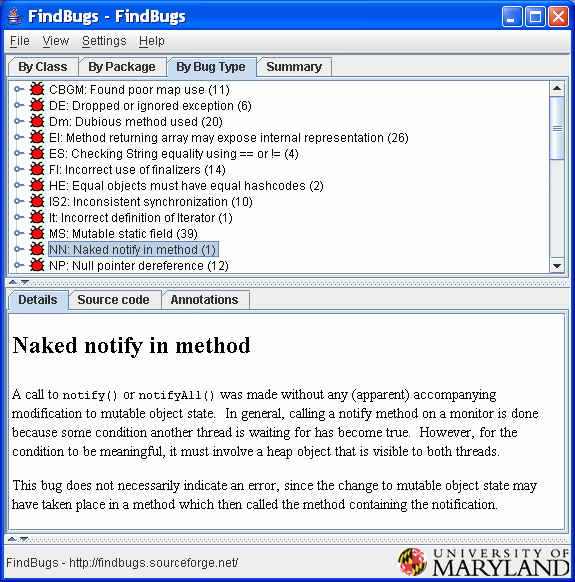
* 静态代码分析是指无需运行被测代码，仅通过分析或检查源程序的语法、结构、过程、接口等来检查程序的正确性，找出代码隐藏的错误和缺陷，如参数不匹配，有歧义的嵌套语句，错误的递归，非法计算，可能出现的空指针引用等等。
* 在软件开发过程中，静态代码分析往往先于动态测试之前进行，同时也可以作为制定动态测试用例的参考。统计证明，在整个软件开发生命周期中，30% 至 70% 的代码逻辑设计和编码缺陷是可以通过静态代码分析来发现和修复的。
* 但是，由于静态代码分析往往要求大量的时间消耗和相关知识的积累，因此对于软件开发团队来说，使用静态代码分析工具自动化执行代码检查和分析，能够极大地提高软件可靠性并节省软件开发和测试成本。

## 主流静态检查工具介绍

* JAVA

1. FindBugs:由马里兰大学提供的一款开源 Java 静态代码分析工具。FindBugs 通过检查类文件或 JAR 文件，将字节码与一组缺陷模式进行对比从而发现代码缺陷，完成静态代码分析。

下图是分析一个匿名项目的结果：



该结果可导出为xml文档。

FindBugs最近的一个版本是3.0.1，于2015年发布。

1. PMD:由 DARPA 在 SourceForge 上发布的开源 Java 代码静态分析工具。PMD 通过其内置的编码规则对 Java 代码进行静态检查，主要包括对潜在的 bug，未使用的代码，重复的代码，循环体创建新对象等问题的检验，侧重于对不良编程习惯进行分析，侧重于重构。

PMD附带了许多可以直接使用的规则，利用这些规则可以找出Java源程序的许多问题，例如：

* 潜在的bug：空的try/catch/finally/switch语句
* 未使用的代码：未使用的局部变量、参数、私有方法等
* 可选的代码：String/StringBuffer的滥用
* 复杂的表达式：不必须的if语句、可以使用while循环完成的for循环
* 重复的代码：拷贝/粘贴代码意味着拷贝/粘贴bugs
* 循环体创建新对象：尽量不要再for或while循环体内实例化一个新对象
* 资源关闭：Connect，Result，Statement等使用之后确保关闭掉

此外，用户还可以自己定义规则，检查Java代码是否符合某些特定的编码规范。例如，你可以编写一个规则，要求PMD找出所有创建Thread和Socket对象的操作。

PMD最近的一个版本是5.6.1，于2017年4月发布。

1. Checkstyle:SourceForge 的开源项目，通过检查对代码编码格式，命名约定，Javadoc，类设计等方面进行代码规范和风格的检查，从而有效约束开发人员更好地遵循代码编写规范。

CheckStyle检验的主要内容

* Javadoc注释
* 命名约定
* 标题
* Import语句
* 体积大小
* 空白
* 修饰符
* 块
* 代码问题
* 类设计
* 混合检查（包括一些有用的比如非必须的System.out和printstackTrace）

以上三个工具都有对应的eclipse插件。

* PHP rips, phpcheckstyle, PHPMD, PHP CodeSniffer
* golint 用于检测go代码

golint会输出一些代码存在的问题：  
比如：

recorder.go:55:5: exported var RecordBind should have comment or be unexported

recorder.go:158:1: exported function Record\_ErrorRecord should have comment or be unexported

recorder.go:173:6: don't use underscores in Go names; type Data\_MemStats should be DataMemStats

recorder.go:179:2: struct field FreeRam should be FreeRAM

上面的输出中文件recorder.go，179行，在struct中字段FreeRam应该是 FreeRAM,输出信息非常的详细。

golint 会检测的方面：

变量名规范

变量的声明，像var str string = "test"，会有警告，应该var str = "test"

大小写问题，大写导出包的要有注释

x += 1 应该 x++

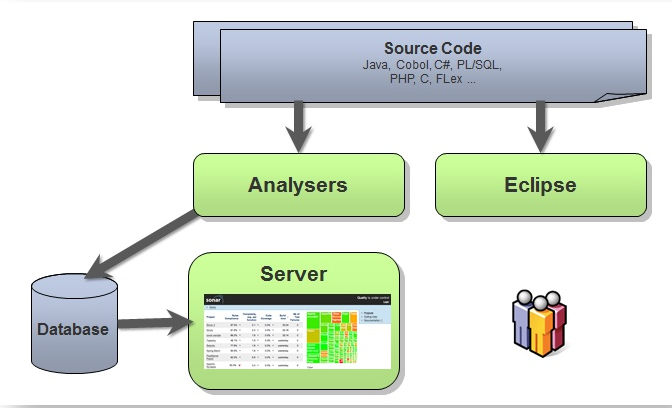
等等

* Android/IOS Android-Lint，infer, godeyes
* 多语言 SonarQube

## SonarQube简介

Sonar (SonarQube)是一个开源平台，用于管理源代码的质量。

#### SonarQube架构



#### SonarQube特性

* 支持多种语言

通过安装相应插件的方式，提供JAVA、PHP、Java Script、HTML、CSS、Object-c等公司几乎所有产品代码的静态分析。

* 项目平台化管理，在线修复问题，在线开展评审

Sonar不仅仅能够展示项目质量数据，更是一个提供问题管理、多维度质量指标、在线代码评审、修复建议的代码质量管理平台。



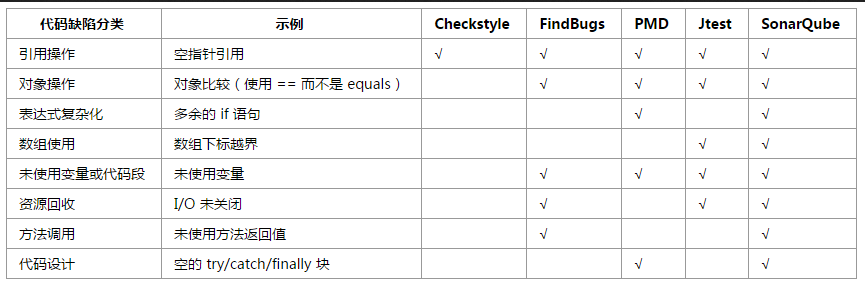
* 界面友好，使用方便

Sonar提供了丰富的本地化语言插件，通过安装中文插件使我们的工作更加方便。

* 检测规则丰富

除了官方语言插件外，Sonar也提供了丰富的基于业界优秀工具检测规则的插件。如CheckStyle, PMD, Findbugs等。除此之外，Sonar支持编写自定义规则。

Java 静态代码分析工具对比



* 支持代码规则、质量配置的修改

不同项目均可以设计符合自身需求的质量配置，通过添加激活不同的代码规则，定制符合公司要求的检测结果。此特性能够很好地检测并着重解决项目真正需要解决的问题。在使用静态检查工具前设置好检查的规则，真正让工具成为控制代码质量的帮手而不是研发的负担。

代码规则与质量配置页面：





* 完善的权限控制

SonarQube平台拥有完善的用户、群组以及项目的权限控制功能。能够实现不同角色和团队之间的权限隔离。



* 支持多种编辑器和持续集成工具

通过安装相应插件，能够实现在eclipse等编辑器中直接开展扫描或查看结果。也可以通过Maven、ANT调用扫描。通过Jenkins持续集成工具更能实现 “研发人员提交代码，系统自动进行（或定时触发）静态检查并发送邮件通知” 的功能，最终实现 “编译->检测->打包->部署->测试->发布” 自动化。

示例：Jenkins每隔15分钟检测SVN是否更新，有更新则触发代码静态分析：



* 参考资源丰富，拥有完善的社区支持

SonarQube是一个成熟的JAVA开源项目，网上参考资料也非常多。 参考网站:

官网链接: <http://www.sonarqube.org/>

官方文档链接: <http://docs.sonarqube.org/display/HOME/SonarQube+Platform>

开源中国: <http://www.oschina.net/p/sonar>

SonarQube最近的一个版本是6.3，于2017年4月发布，最新的长期支持版本是5.6.6，于2017年2月发布。

## 方案

公司当前的开发语言有：c、c++、c#、php、java、javascript、golang、erlang、object-c。SonarQube对上述语言能够全部支持（golang是通过[sonar-golang](https://github.com/uartois/sonar-golang)

支持），而上述其它静态检测工具相对支持语言单一，不同语言不用工具，使用起来分散，不能够像SonarQube一样统一检测，统一管理。综合上述SonarQube的特性，推荐使用SonarQube。