Tarefa 1. Procura de analizadores estáticos de código no mercado e comparativas.

Java

- AgileJ StructureViews
- Checkstyle
- FindBugs
- Hammurapi
- PMD
- Soot
- Squale
- Jtest
- LDRA Testbed
- RIPS
- SemmleCode
- SonarJ
- Kalistick
- Visio

Comparativa

Cada analizador distingue en su salida las siguientes **categorías de bugs**:

Bugs por categoria en FindBugs		
Categ. BAD_PRACTICE	137	89
Categ. CORRECTNESS	49	50
Categ. EXPERIMENTAL	2	11
Categ. MALICIOUS_CODE	150	11
Categ. MT_CORRECTNESS	36	15
Categ. PERFORMANCE	236	89
Categ. STYLE	147	121
Bugs totales (*)	**757**	**386**

Bugs por categoría en PMD		
RuleSet Basic Rules	258	59
RuleSet Braces Rules	1916	69
RuleSet Clone Implementation Rules	51	--
RuleSet Code Size Rules	572	275
RuleSet Controversial Rules	4349	2705
RuleSet Coupling Rules	44	32
RuleSet Design Rules	1425	480
RuleSet Import Statement Rules	--	5
RuleSet Finalizer Rules	6	--
RuleSet J2EE Rules	59	15
RuleSet JUnit Rules	--	26
RuleSet Jakarta Commons Logging Rules	6	--
RuleSet Java Logging Rules	290	7
RuleSet JavaBean Rules	1126	300
RuleSet Migration Rules	115	40
RuleSet Naming Rules	2222	941
RuleSet Optimization Rules	4586	1881
RuleSet Security Code Guidelines	50	3
RuleSet Strict Exception Rules	19	24
RuleSet String and StringBuffer Rules	113	48
RuleSet Type Resolution Rules	919	185
RuleSet Unused Code Rules	215	69
Totales(*)	**18341**	**7164**

Para elegir una herramienta de análisis estático es necesario tener en cuenta características como la usabilidad, la eficiencia, la extensibilidad y la técnica de análisis. En muchos casos será necesario elegir más de un analizador de distinto tipo para poder abarcar más tipos de bugs. De los resultados extraídos en estas pruebas podemos decir que:

Los dos analizadores son fáciles de usar en cuanto a instalación, configuración, actualización y uso posterior. Hay plugins para varias plataformas e IDEs. Desde Eclipse ambos añaden una perspectiva más que permite la ejecución desde un menú de contexto.

PMD resulta más rápido que Findbugs. El consumo de recursos es mayor en FindBugs que en

PMD. FindBugs añade en los ficheros XML atributos de tiempo de ejecución para cada fichero y proyecto. PMD no dispone de este tipo de información.

En cuanto a extensibilidad PMD resulta más sencillo a la hora de añadir nuevas reglas. Trae un editor propio que permite copiar, modificar, añadir y eliminar reglas de una forma intuitiva y sencilla. No pasa lo mismo con FindBugs que dificulta esta tarea por tener que programar desde cero cada regla que se desee añadir a un proyecto.

La documentación de bugs que permite interpretar los resultados aparece más documentada en PMD que en FindBugs. PMD incorpora además una funcionalidad para generar estadísticas resumen.

En los resultados de los dos analizadores, PMD localiza muchos más bugs que FindBugs en los dos proyectos vistos. Los dos muestran una descripción y localización correcta del error, aunque referida en algunos casos al número de línea donde comienza la clase y no al número de línea del bug. Solo en los ficheros XML de PMD se incluyen descripciones de los bugs detectados.

Los dos analizadores se complementan en cuanto a los tipos de bugs que localizan. La técnica de análisis es distinta en las dos herramientas, FindBugs está clasificado como Bug Pattern y PMD como Style-checker y Bug-Checker, esta es una de las razones por las que el porcentaje de bugs comunes no supera el 30% en los dos proyectos vistos.

Con los criterios vistos siempre resultará más eficaz utilizar más de una herramienta para el análisis de código estático. De esta forma aumentamos el número de bugs detectados.