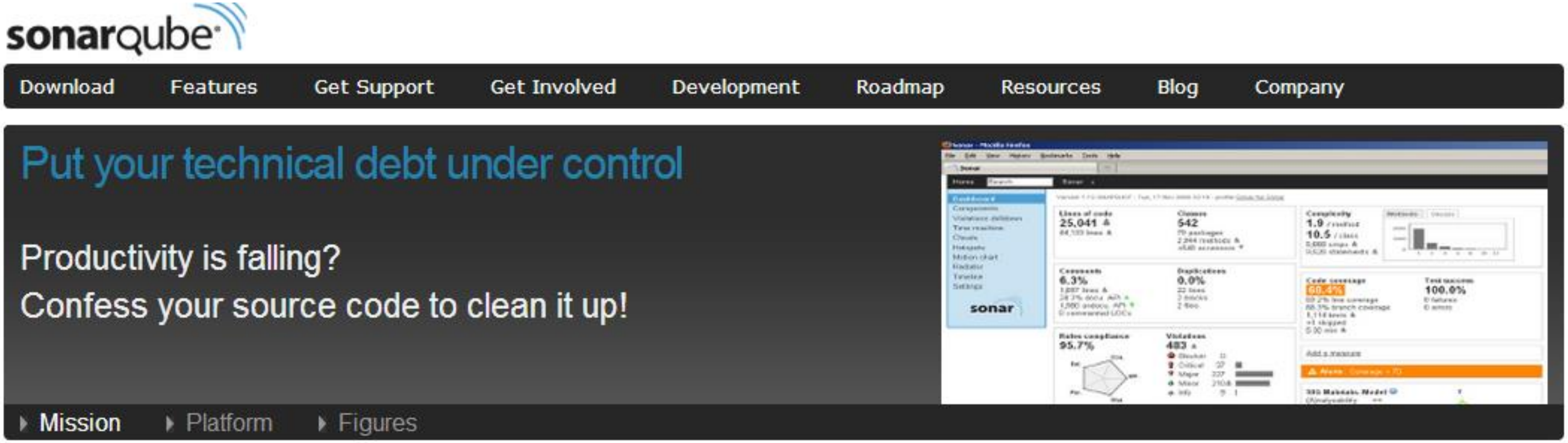
Métricas de calidad de código

Departamento de Sistemas y Computación

Universidad de los Andes, Bogotá

Referencias

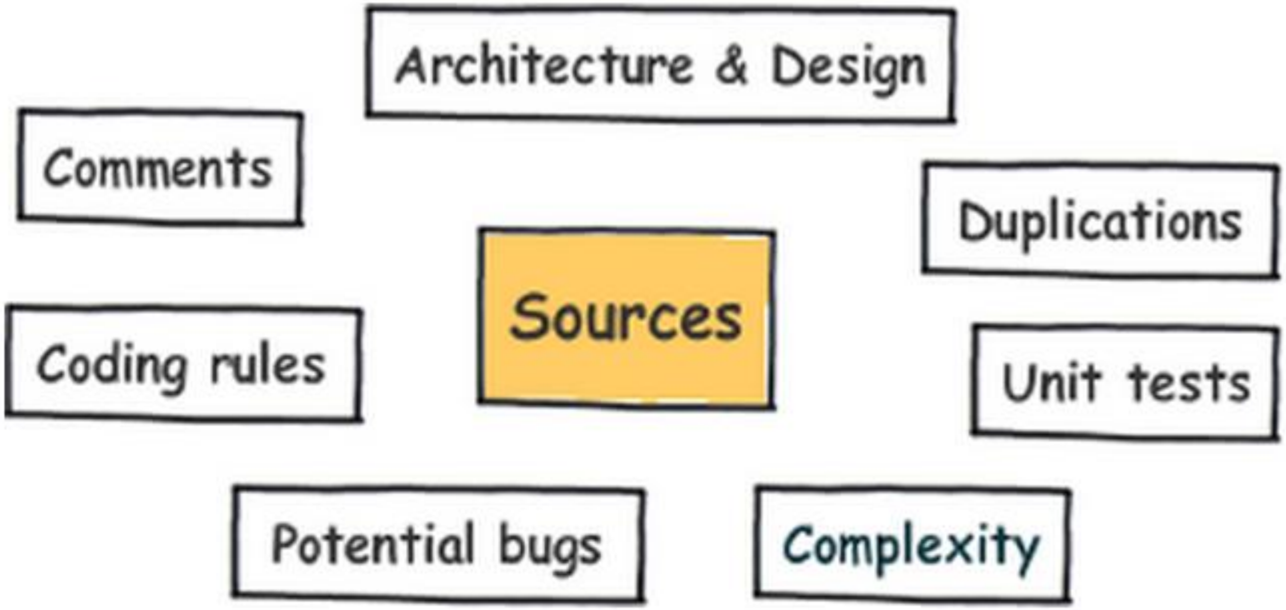


Put your technical debt under control

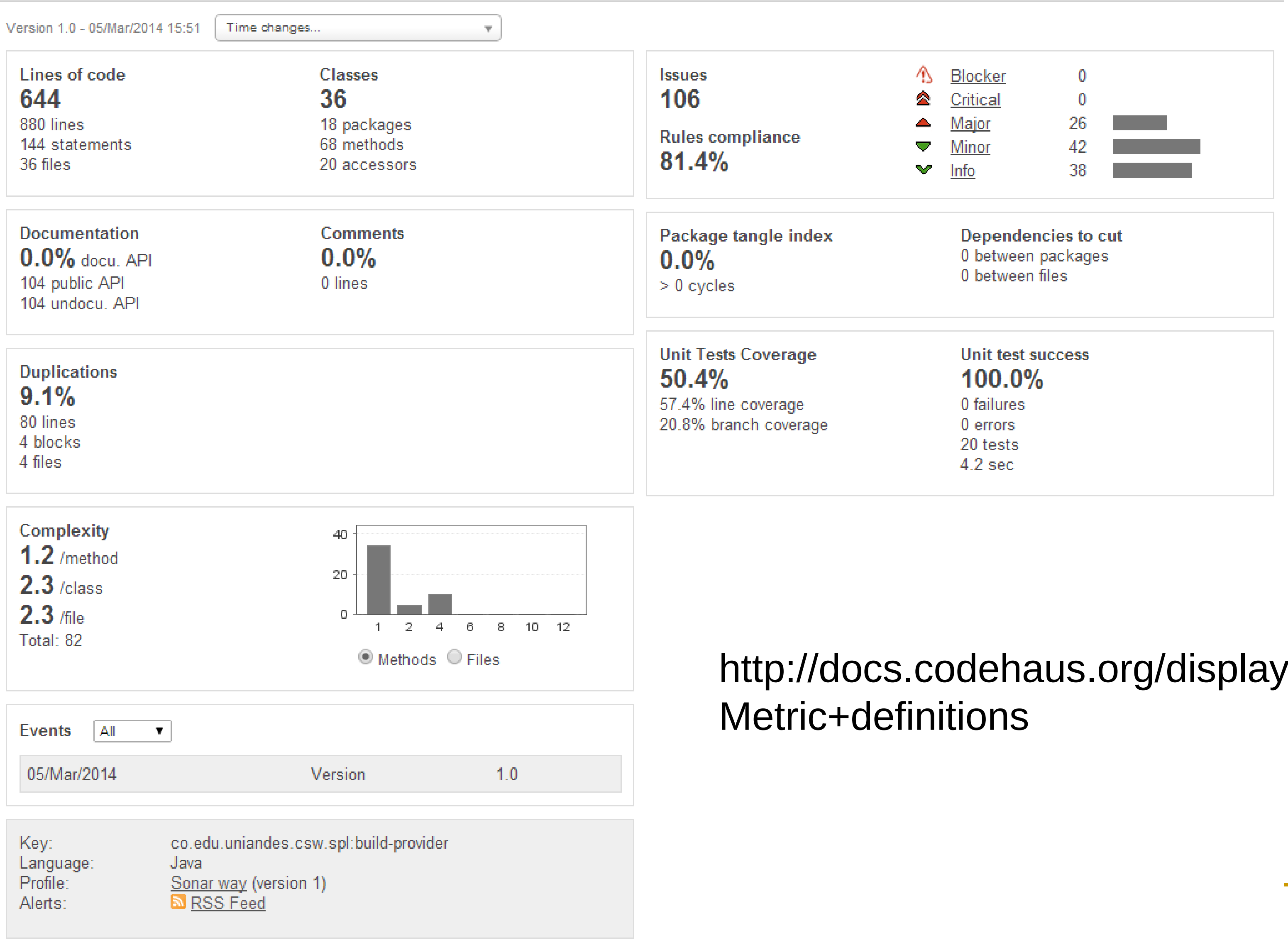
Productivity is falling?

Confess your source code to clean it up!

SonarQube: Calidad del código



Tablero de control básico



Version 1.0 - 05/Mar/2014 15:51 | Time changes... ▾

Lines of code
644
880 lines
144 statements
36 files

Classes
36
18 packages
68 methods
20 accessors

Issues
106

Rules compliance
81.4%

▲	Blocker	0
▲	Critical	0
▲	Major	26
▼	Minor	42
✓	Info	38

Documentation
0.0% docu. API
104 public API
104 undocu. API

Comments
0.0%
0 lines

Package tangle index
0.0%
> 0 cycles

Dependencies to cut
0 between packages
0 between files

Duplications
9.1%
80 lines
4 blocks
4 files

Unit Tests Coverage
50.4%
57.4% line coverage
20.8% branch coverage

Unit test success
100.0%
0 failures
0 errors
20 tests
4.2 sec

Complexity
1.2 /method
2.3 /class
2.3 /file
Total: 82

Events All ▾

05/Mar/2014	Version	1.0

Key:	co.edu.uniandes.csw.spl:build-provider
Language:	Java
Profile:	Sonar way (version 1)
Alerts:	RSS Feed

http://docs.codehaus.org/display/SONAR/Metric+definitions

Tablero de control básico (cont.)

- Complexity
- Documentation
- Duplications
- Issues
- Size
- Tests

Complexity

- Se refiere a la complejidad cyclomática o métrica de McCabe.
- Mide la complejidad de un código en términos del número de flujos de control que encuentre
- Cada función tiene una complejidad mínima de 1

http://docs.codehaus.org/display/SONAR/Metrics+-+Complexity

Complexity

Complexity /class	class_complexity	Average **complexity** by **class**.
Complexity /file	file_complexity	Average **complexity** by **file**.
Complexity /method	function_complexity	Average **complexity** by **function**.

http://docs.codehaus.org/display/SONAR/Metrics+-+Complexity

Complexity

- Keywords incrementing the complexity: `if`, `for`, `while`, `case`, `catch`, `throw`, `return` (that is not the last statement of a method), `&&`, `||`, `?`
- Notes:
  - `else`, `default`, and `finally` keywords do not increment the complexity.
  - simple method with a switch statement and a huge block of case statements can have a surprisingly high complexity value (still it has the same value when converting a switch block to an equivalent sequence of if statements).
  - accessors are not considered as methods and so do not increment the complexity

http://docs.codehaus.org/display/SONAR/
Metrics+-+Complexity

Complexity

Example: the following method has a complexity of 5

```
public void process(Car myCar){          // +1
    if(myCar.isNotMine()){               // +1
         return;                         // +1
    }
    car.paint("red");
    car.changeWheel();
    while(car.hasGazol() &&
car.getDriver().isNotStressed()){   // +2
         car.drive();
    }
    return;
}
```

http://docs.codehaus.org/display/SONAR/Metrics+-+Complexity

Documentación: Líneas de comentarios

```
/**                                        +0 => empty comment line
 *                                         +0 => empty comment line
 * This is my documentation                +1 => significant comment
 * although I don't                        +1 => significant comment
 * have much                               +1 => significant comment
 * to say                                  +1 => significant comment
 *                                         +0 => empty comment line
 ***************************               +0 => non-significant comment
 *                                         +0 => empty comment line
 * blabla...                               +1 => significant comment
 */                                        +0 => empty comment line

/**                                        +0 => empty comment line
 * public String foo() {                   +1 => commented-out code
 *   System.out.println(message);          +1 => commented-out code
 *   return message;                       +1 => commented-out code
 * }                                       +1 => commented-out code
 */
```

http://docs.codehaus.org/display/SONAR/Metric+definitions#Metricdefinitions-Design

Documentación: Densidad de líneas de comentarios

Density of comment lines = **Comment lines** / (**Lines of code** + **Comment lines**) * 100

With such a formula:
50% means that the number of lines of code equals the number of comment lines
100% means that the file only contains comment lines

http://docs.codehaus.org/display/SONAR/Metric+definitions#Metricdefinitions-Design

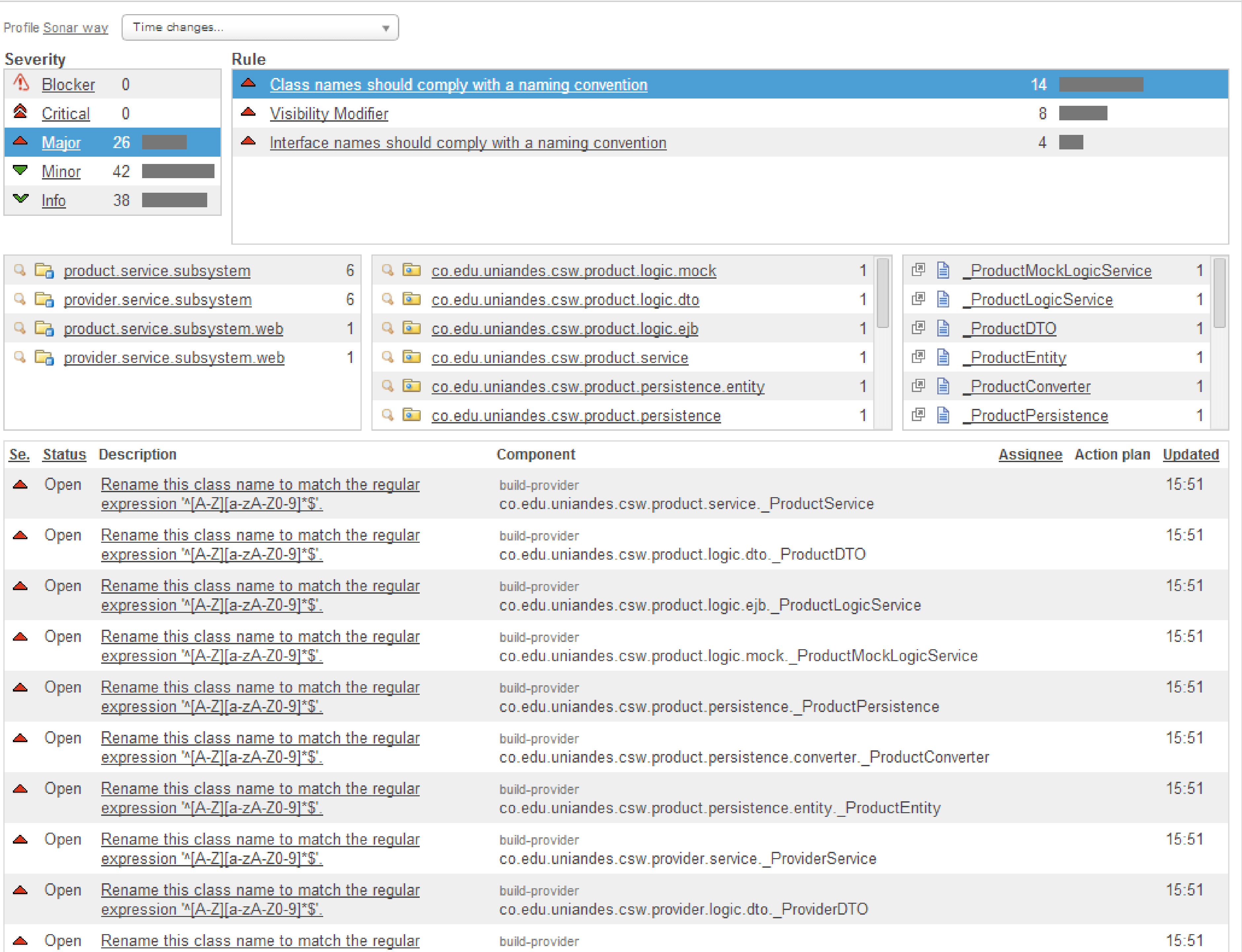
Duplicaciones

Name	Key	Description
Duplicated blocks	duplicated_blocks	Number of duplicated blocks of lines.
Duplicated files	duplicated_files	Number of files involved in a duplication.
Duplicated lines	duplicated_lines	Number of lines involved in a duplication.
Duplicated lines (%)	duplicated_lines_density	Density of duplication = **Duplicated lines** / **Lines** * 100

http://docs.codehaus.org/display/SONAR/Metric+definitions#Metricdefinitions-Design

Issues: Perfil de calidad

- Conjunto de reglas que el código debe cumplir
- Ejemplo:
  - Métodos no deben tener una complejidad mayor que 10
- Los perfiles dependen del lenguaje.
- Hay varios predefinidos que se pueden utilizar y/o modificar



Profile Sonar way Time changes...

Severity	
Blocker	0
Critical	0
Major	26
Minor	42
Info	38

Rule	
Class names should comply with a naming convention	14
Visibility Modifier	8
Interface names should comply with a naming convention	4

product.service.subsystem	6
provider.service.subsystem	6
product.service.subsystem.web	1
provider.service.subsystem.web	1

co.edu.uniandes.csw.product.logic.mock	1
co.edu.uniandes.csw.product.logic.dto	1
co.edu.uniandes.csw.product.logic.ejb	1
co.edu.uniandes.csw.product.service	1
co.edu.uniandes.csw.product.persistence.entity	1
co.edu.uniandes.csw.product.persistence	1

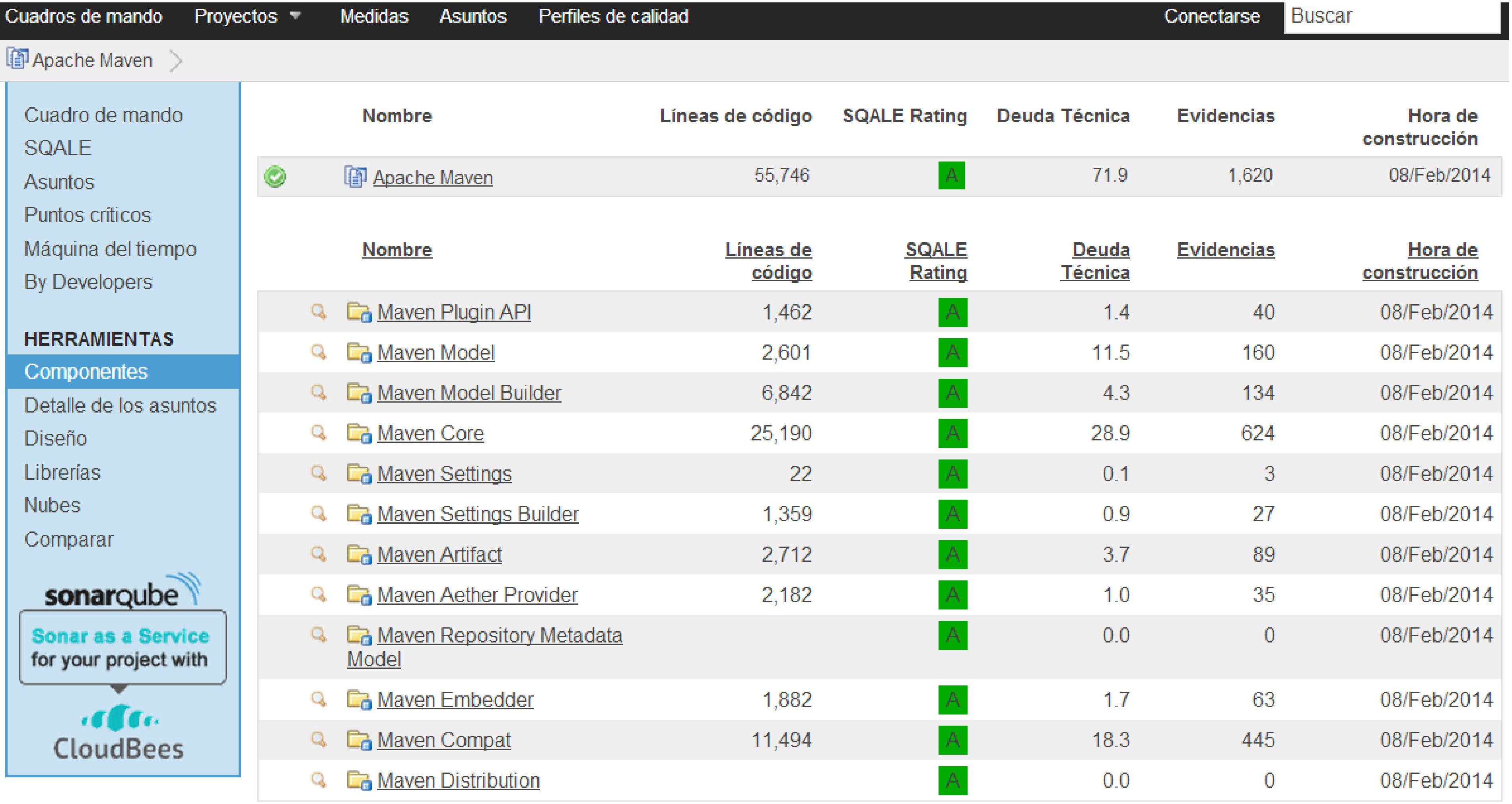
_ProductMockLogicService	1
_ProductLogicService	1
_ProductDTO	1
_ProductEntity	1
_ProductConverter	1
_ProductPersistence	1

Se.	Status	Description	Component	Assignee	Action plan	Updated
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.service._ProductService			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.logic.dto._ProductDTO			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.logic.ejb._ProductLogicService			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.logic.mock._ProductMockLogicService			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.persistence._ProductPersistence			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.persistence.converter._ProductConverter			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.product.persistence.entity._ProductEntity			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.provider.service._ProviderService			15:51
▲	Open	Rename this class name to match the regular expression '^[A-Z][a-zA-Z0-9]*$'.	build-provider co.edu.uniandes.csw.provider.logic.dto._ProviderDTO			15:51
▲	Open	Rename this class name to match the regular	build-provider			15:51

Package tangle index

- Nivel de interdependencia entre los directorios
- Debería valer 0

Otras Métricas



	Nombre	Líneas de código	SQALE Rating	Deuda Técnica	Evidencias	Hora de construcción
✓	Apache Maven	55,746	A	71.9	1,620	08/Feb/2014

Nombre	Líneas de código	SQALE Rating	Deuda Técnica	Evidencias	Hora de construcción
Maven Plugin API	1,462	A	1.4	40	08/Feb/2014
Maven Model	2,601	A	11.5	160	08/Feb/2014
Maven Model Builder	6,842	A	4.3	134	08/Feb/2014
Maven Core	25,190	A	28.9	624	08/Feb/2014
Maven Settings	22	A	0.1	3	08/Feb/2014
Maven Settings Builder	1,359	A	0.9	27	08/Feb/2014
Maven Artifact	2,712	A	3.7	89	08/Feb/2014
Maven Aether Provider	2,182	A	1.0	35	08/Feb/2014
Maven Repository Metadata Model		A	0.0	0	08/Feb/2014
Maven Embedder	1,882	A	1.7	63	08/Feb/2014
Maven Compat	11,494	A	18.3	445	08/Feb/2014
Maven Distribution		A	0.0	0	08/Feb/2014

Deuda Técnica

- Technical debt, design debt, code debt
  - Puede ser interpretada como una medida de la cantidad de trabajo que tocaría hacerle al código para que tenga una calidad aceptable
  - Si la deuda no se corrige, esta genera más intereses haciendo más difícil lograr la calidad

SQALE Rating

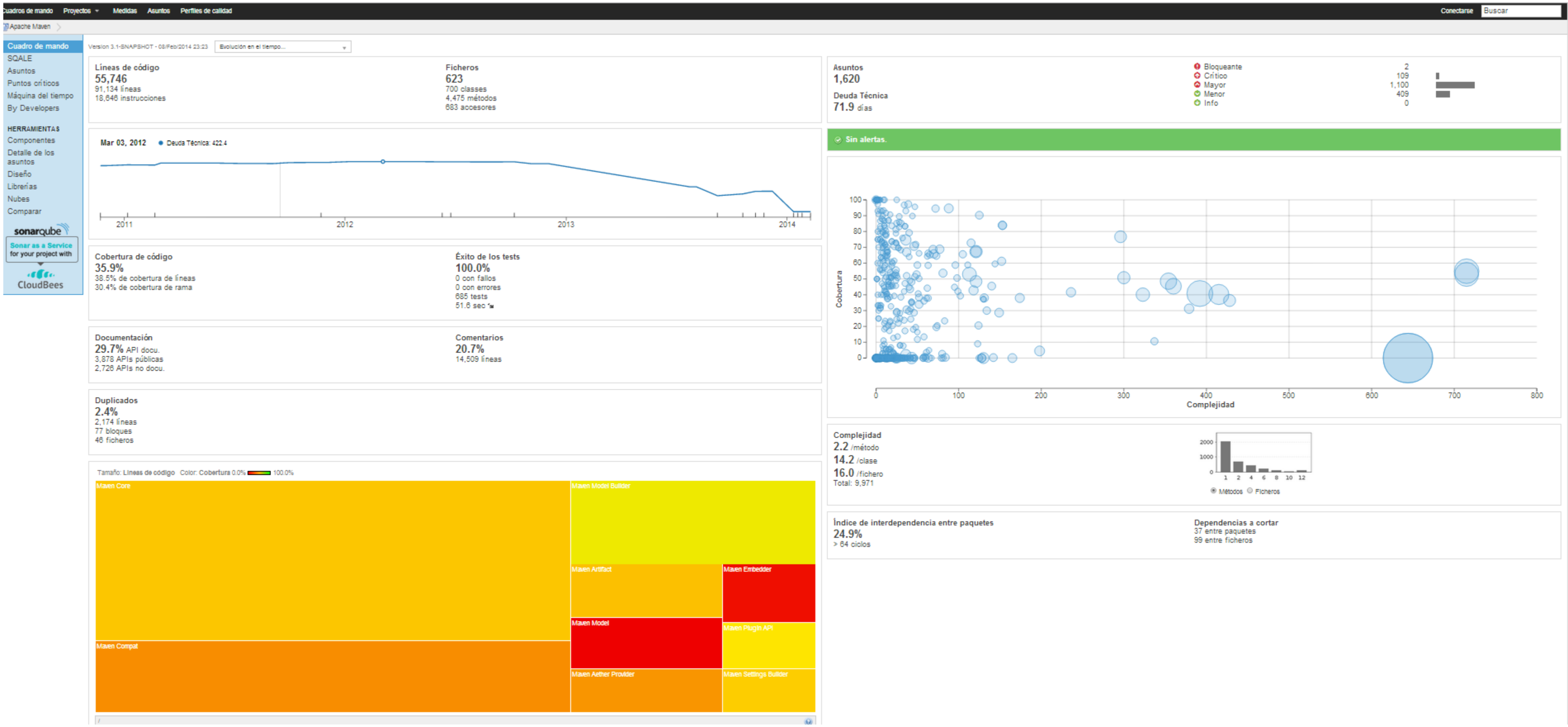
- **SQALE** (Software Quality Assessment based on Lifecycle Expectations)
- Es un método para evaluar el código fuente de un aplicación.
- Es independiente del lenguaje y de las herramientas de análisis de código
- Licencia: Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported license

Sqale y la deuda técnica

- SQALE permite;
  - Definir qué crea la deunda técnica
  - Eestimar correctamente a cuánto asciende la deuda
  - Analizar la deuda con respecto a una perspectiva técnica y de negocio
  - Ofrecer diferentes estrategias de priorización para establecer un plan adecuado.

SQALE Rating

- El método está basado en 4 conceptos:
  - El modelo de calidad
  - El modelo de análisis
  - Los índices
  - Los indicadores



Version 3.1-SNAPSHOT - 08/Feb/2014 23:23

Líneas de código
55,746
91,134 líneas
18,646 instrucciones

Ficheros
623
700 classes
4,475 métodos
683 accesores

Cobertura de código
35.9%
38.5% de cobertura de líneas
30.4% de cobertura de rama

Éxito de los tests
100.0%
0 con fallos
0 con errores
685 tests
51.6 sec

Documentación
29.7% API docu.
3,878 APIs públicas
2,726 APIs no docu.

Comentarios
20.7%
14,509 líneas

Duplicados
2.4%
2,174 líneas
77 bloques
46 ficheros

Asuntos
1,620

Deuda Técnica
71.9 días

Bloqueante	2
Crítico	109
Mayor	1,100
Menor	409
Info	0

Sin alertas.

Complejidad
2.2 /método
14.2 /clase
16.0 /fichero
Total: 9,971

Índice de interdependencia entre paquetes
24.9%
> 64 ciclos

Dependencias a cortar
37 entre paquetes
99 entre ficheros