

The background of the slide is a composite image. It shows a person from behind, sitting on a ledge and looking out over a city skyline. The image has a blue and green color cast. Overlaid on the image are white concentric circles and lines, resembling sonar waves or a network map, emanating from the person and connecting to various buildings in the city. In the top left corner, there is a bright green light flare.

sonarqube

Revealed

No es charla para novatos

**DON'T BE
A DUMMY**



**#VLC
TESTING18**
28 y 29 de noviembre

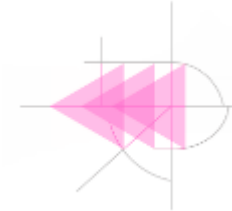
Alejandro Piera Ferrer



**Analista Calidad
Everis**



**+ 10 años
experiencia**



**Of. Calidad
Generalitat
Valenciana**



1

Conceptos básicos de SonarQube

2

Personalización en SonarQube

3

Desarrollo de plugin básico

4

Prerequisitos

5

Clases importantes

6

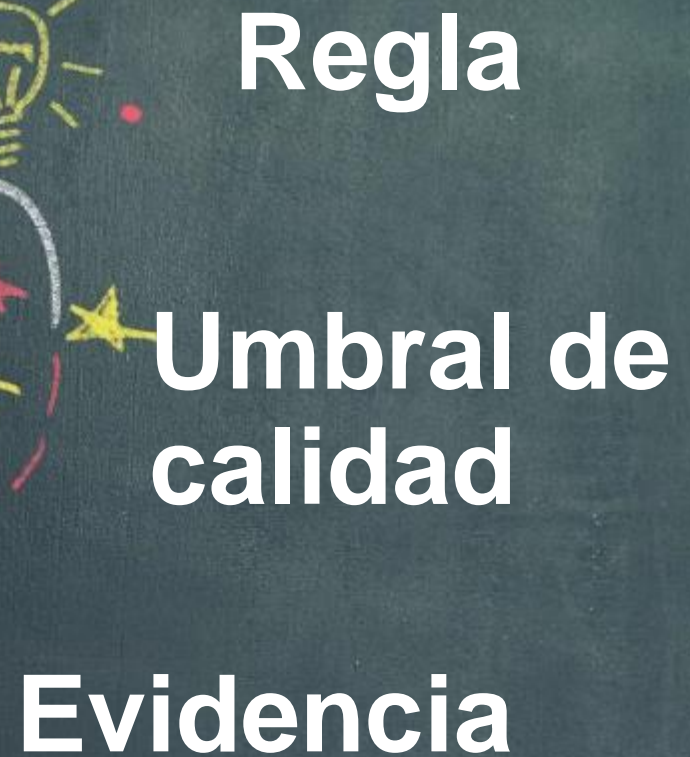
Desarrollo

SonarQube Revealed



Conceptos básicos de SonarQube

Conceptos básicos de SonarQube



Regla

Umbral de
calidad

Evidencia

Métrica



Una métrica es un **tipo de medición**

Las métricas pueden tener **valores o medidas variable a lo largo del tiempo**

Una métrica puede ser:



Cuantitativa: No proporciona una indicación de calidad en el componente

Líneas de código

Complejidad



Cualitativa: Proporciona una indicación de calidad en el componente

Densidad de líneas duplicadas

Cobertura de líneas por prueba

Métrica

Proyecto A	Proyecto B
5.000 líneas duplicadas	5.000 líneas duplicadas
10.000 líneas duplicadas	10.000 líneas duplicadas
50% código duplicado	5% código duplicado

Medida



El valor de una métrica para un componente dado

125 líneas de código para la clase Foo.java

25% de densidad de líneas duplicadas para el proyecto MyProject

Regla



Una **buena práctica de desarrollo**

El **incumplimiento** de una regla, conlleva la **generación de evidencias**

Pueden **verificar la calidad** de los archivos, pruebas unitarias o paquetes

Evidencia



Cuando un componente incumple una regla, se crea una evidencia en ese análisis

Se pueden crear evidencias sobre ficheros de código fuente o de pruebas

Existen 3 tipos



Code Smell: Evidencia que afecta al mantenimiento del código e impide la modificación del código con la misma rapidez que cuando se empieza desde cero



Error: Evidencia que resalta un punto de fallo real o potencial en el software



Vulnerabilidad: Evidencia que destaca un agujero de seguridad que puede usarse para atacar al software

Perfil de calidad



Conjunto de **reglas** de un mismo lenguaje

Cada análisis se basa en un perfil de calidad por lenguaje

Umbral de calidad



Conjunto de condiciones basadas en umbrales sobre las medidas contra las que se miden los proyectos

0 evidencias bloqueantes

Cobertura del código en código nuevo superior al 80%

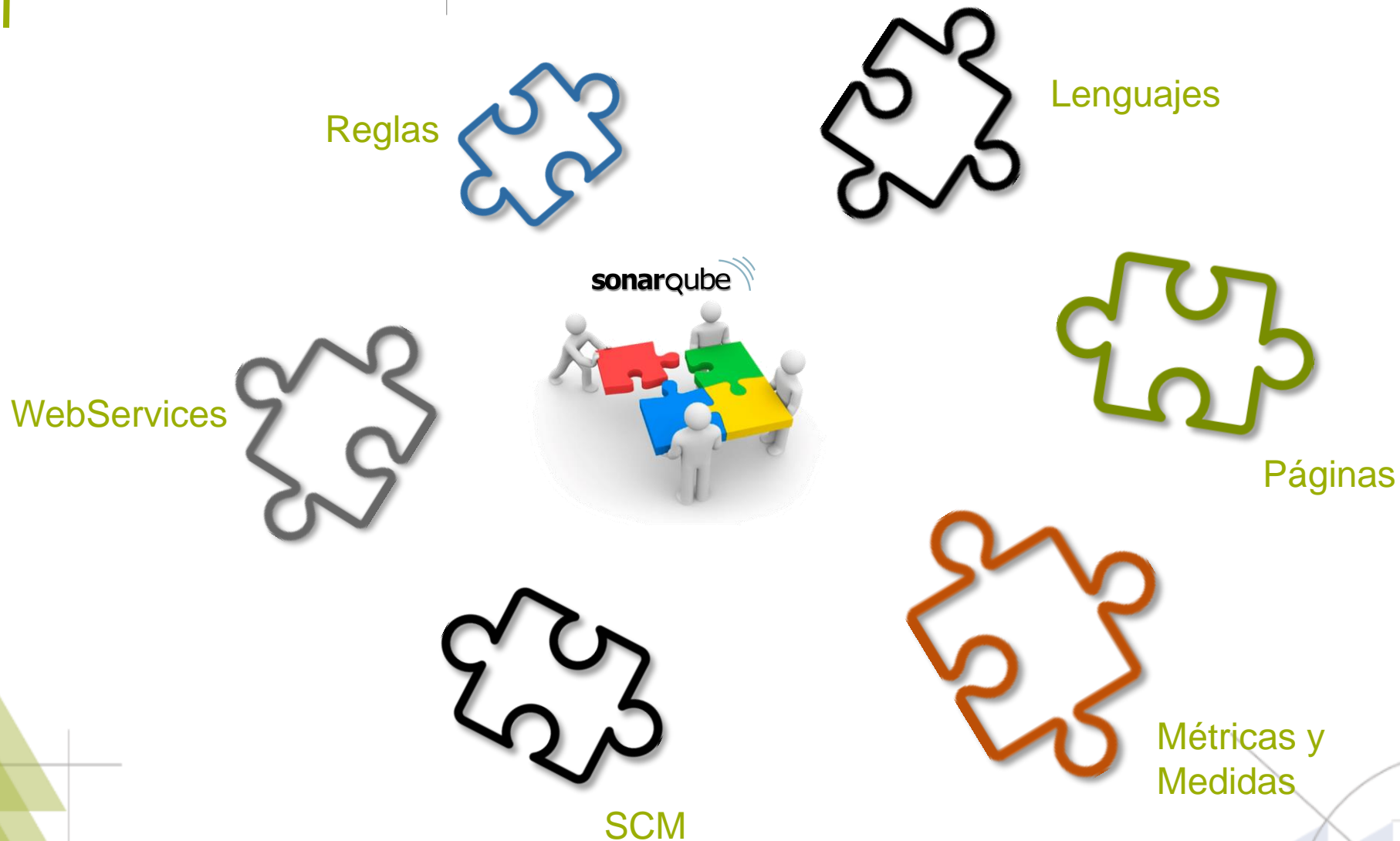
SonarQube Revealed



Personalización de
SonarQube

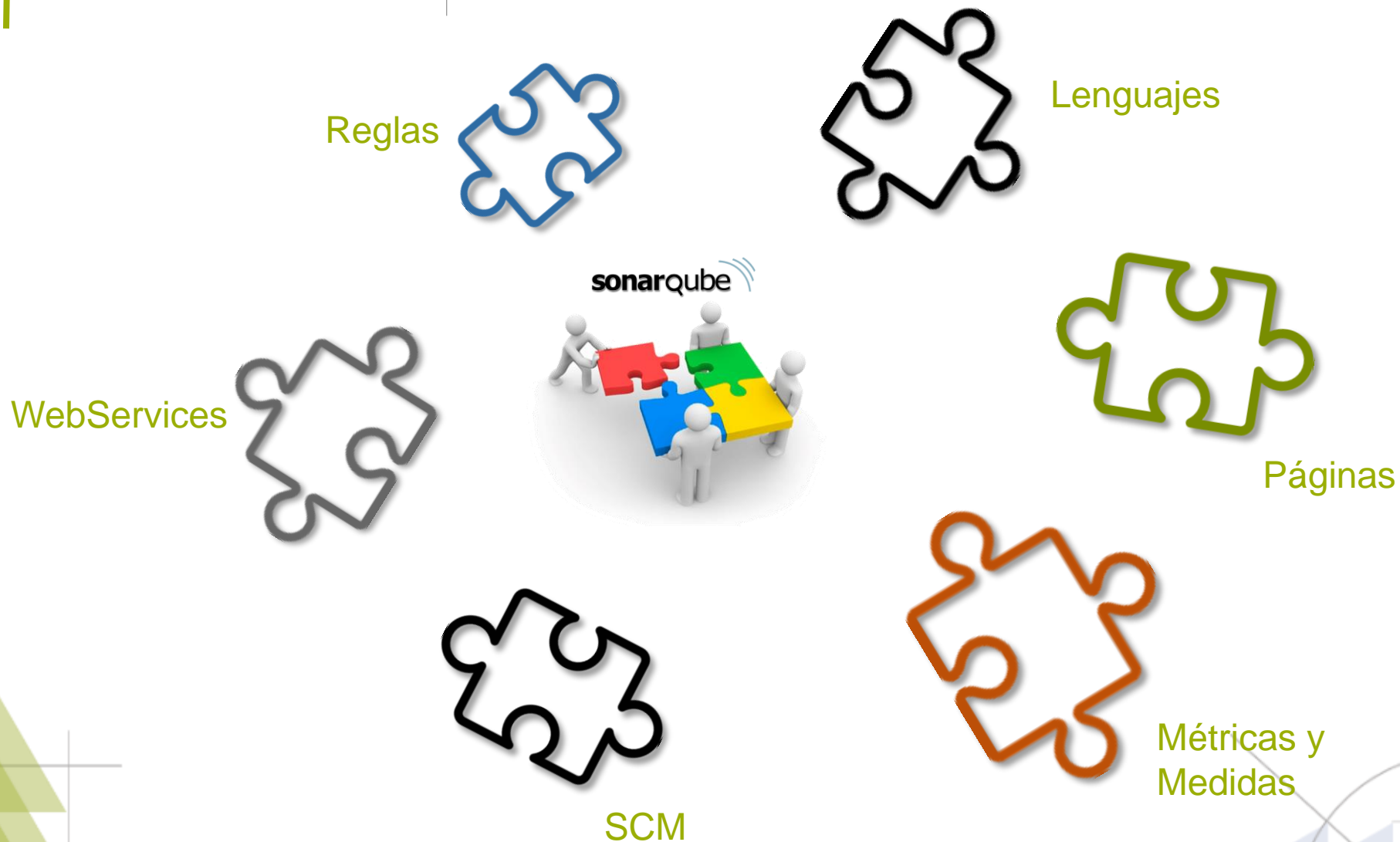
SonarQube Revealed

Personalización de SonarQube



SonarQube Revealed

Personalización de SonarQube



SonarQube Revealed



Desarrollo de un plugin
básico

Índice de cumplimiento de reglas

Calcula una métrica basada en el valor ponderado de las evidencias y el número de líneas de código

Peso de evidencias =

$$\begin{aligned}
 & \text{Nº Ev. Bloqueantes} * \text{Peso Ev. Bloqueantes (10)} \\
 + & \text{Nº Ev. Críticas} * \text{Peso Ev. Críticas (5)} \\
 + & \text{Nº Ev. Mayores} * \text{Peso Ev. Mayores (3)} \\
 + & \text{Nº Ev. Menores} * \text{Peso Ev. Menores (1)} \\
 + & \text{Nº Ev. Informativas} * \text{Peso Ev. Informativas (0)}
 \end{aligned}$$

$$\text{Índice Cumplimiento de Reglas} = \text{Max} (1 - (\text{Peso de evidencias} / \text{Líneas de código}) * 100, 0)$$

SonarQube Revealed

Desarrollo de un plugin básico

Prerequisitos



SonarQube Revealed

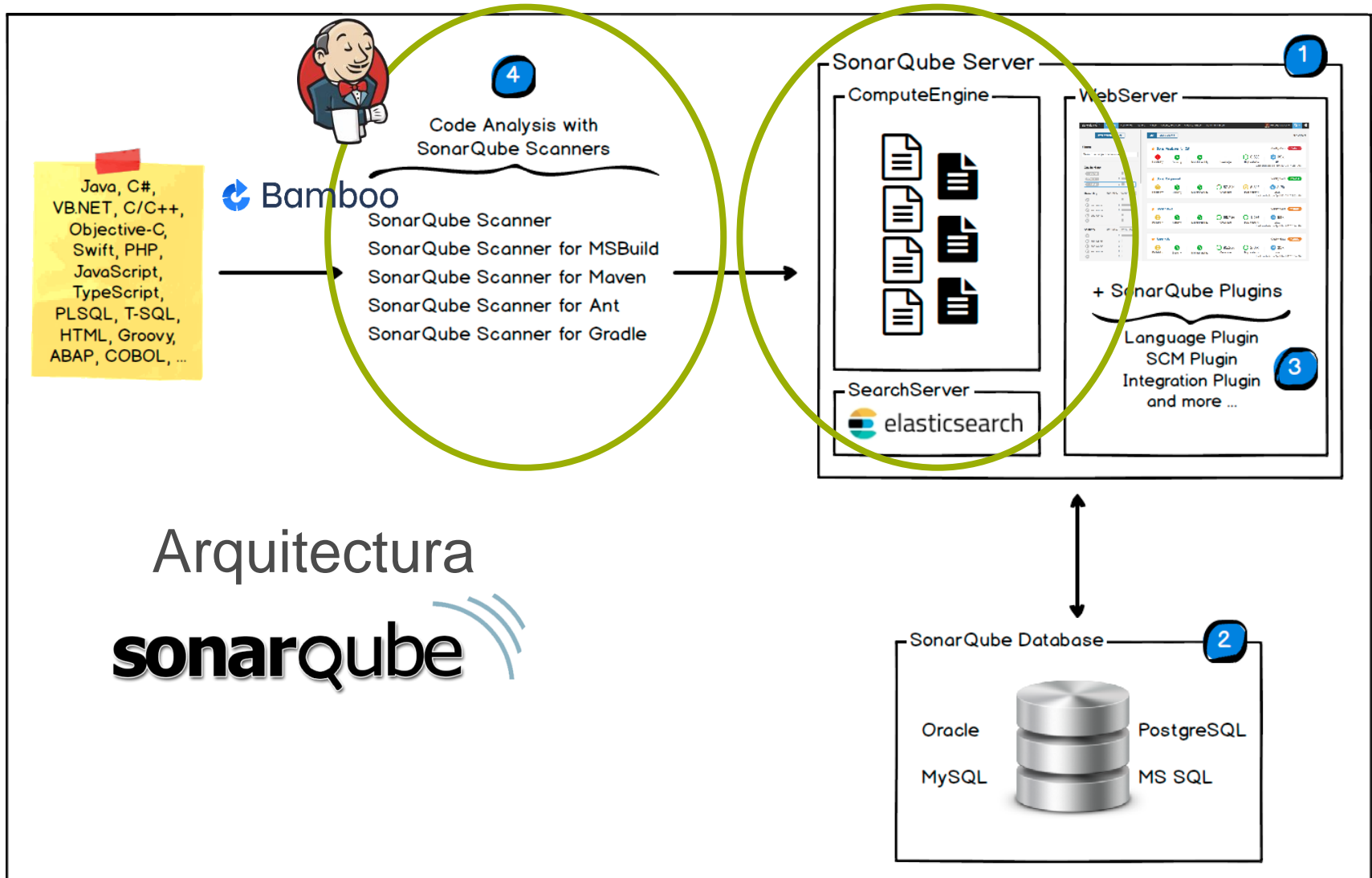


Clases importantes



SonarQube Revealed

Clases importantes



Plugin



Punto de entrada para que los plugins inyecten extensiones en SonarQube

Sensor



Se invoca una vez por cada módulo de un proyecto



El sensor puede analizar un archivo plano, conectarse con un servidor web...



Se utilizan para añadir métricas y evidencias a nivel de archivo

El sensor de cobertura parsea el informe de cobertura y guarda la medida a nivel de fichero

Metrics



Clase utilizada por los plugins para definir nuevas métricas

MeasureComputer



Define como calcular nuevas medidas sobre las métricas declaradas por la clase Metrics

Settings



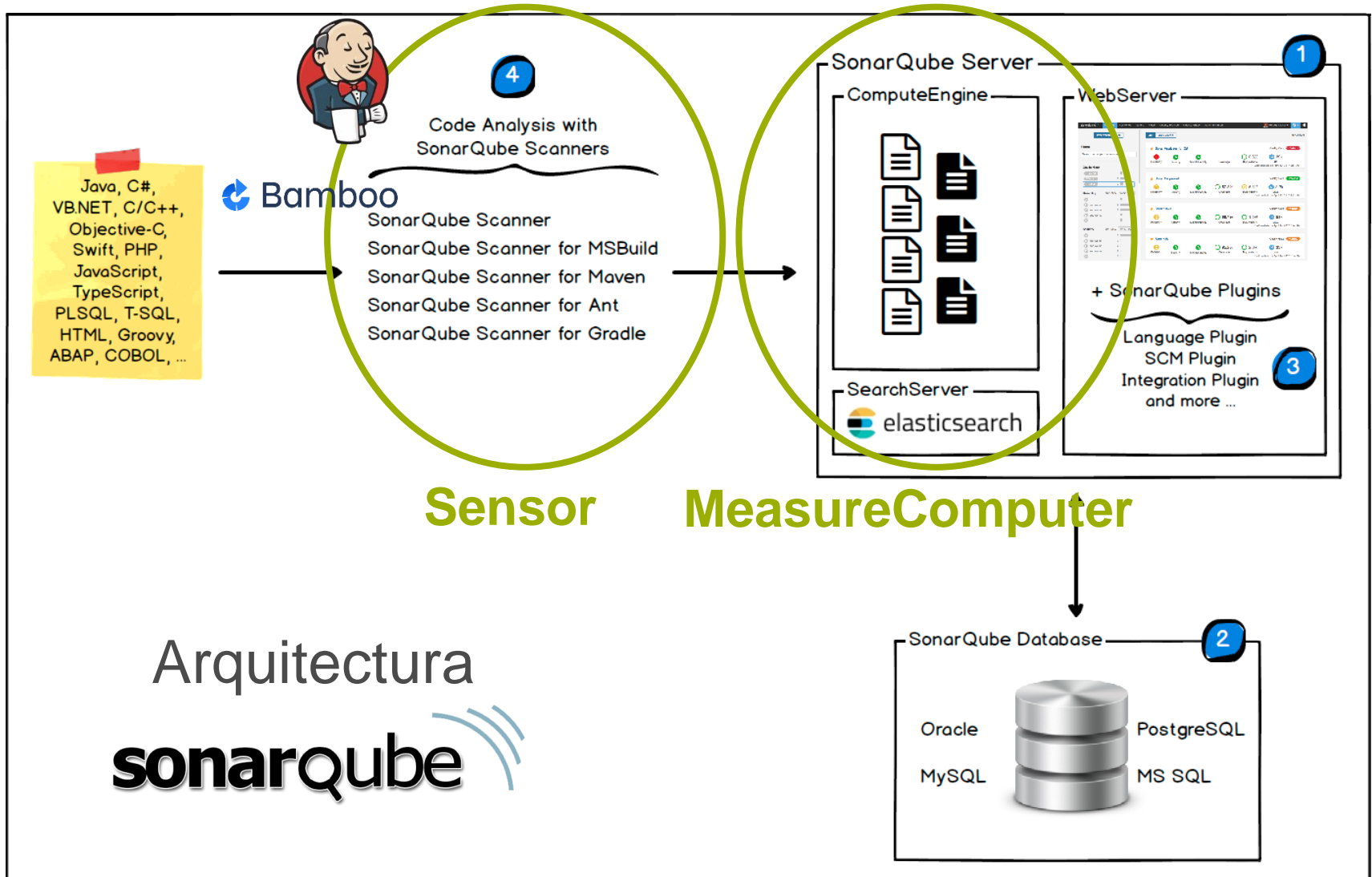
Ofrece acceso a la configuración



Permite crear nuevos parámetros de configuración

SonarQube Revealed

Clases importantes



SonarQube Revealed



Desarrollo

SonarQube Revealed

Desarrollo

Demo

- Pom.xml
- RciPlugin.java
- RciProperties.java
- RciMetrics.java
- RciMeasureComputer.java
- RciPageDefinition.java
- RciWebService.java

```
private static final Logger LOGGER = Loggers.get(RciMeasureComputer.  
@Override  
public MeasureComputerDefinition define(MeasureComputerDefinitionCon  
    LOGGER.debug("Building Rci MeasureComputerDefinition... ");  
    return defContext.newDefinitionBuilder()  
        .setInputMetrics(RciUtils.METRIC_BLOCKER_VIOLATIONS, Rci  
            RciUtils.METRIC_MAJOR_VIOLATIONS, RciUtils.METRIC  
            RciUtils.METRIC_INFO_VIOLATIONS, RciUtils.METRIC  
        .setOutputMetrics(RciUtils.RULES_COMPLIANCE_INDEX_KEY, R  
    }  
@Override  
public void compute(MeasureComputerContext context) {  
    LOGGER.debug("Running Rci Plugin...");  
public class RciWebService implements WebService {  
    private static final Logger LOGGER = Loggers.get(RciWebService.  
    private final Configuration configuration;  
    public RciWebService(Configuration configuration) {  
        this.configuration = configuration;  
    }  
    @Override  
    public void define(Context context) {  
        LOGGER.debug("Create controller to Rci Plugin");  
        final NewController controller = context.createController("R  
        controller.setDescription("Rules Compliance Index Web Servi  
        defineRciPage(controller);  
        controller.done();  
    }  
    private void defineRciPage(final NewController controller) {  
        LOGGER.debug("Create action to render Rci Page");  
        NewAction action = controller.createAction("rciPage").setDe  
            .setSince("1.0").setHandler(new RciRenderPageHandle  
        action.createParam("key").setDescription("Component key").s  
    }  
}  
</plugins>
```



Enlaces de interés

Recurso	URL
Documentación Oficial	https://docs.sonarqube.org/ https://docs.sonarqube.org/display/DEV
Comunidad de usuarios	https://community.sonarsource.com/
Documentación API	http://javadocs.sonarsource.org/ http://javadocs.sonarsource.org/6.7.5/apidocs/
Plugin ejemplo	https://github.com/SonarSource/sonar-custom-plugin-example
Rules Compliance Index	https://github.com/willemsrb/sonar-rci-plugin
Plugin VlcTesting Lite	https://github.com/apieraf/sonarqube-rci-vlctesting-lite-plugin





SonarQube Revealed

Gracias

