Configuración y uso de SonarLint

Oficina de Calidad

Versión: 1.1.0

Control de versiones

Versión	Fecha	Autor	Descripción / Comentarios
V01_v00	30-10-2018	Oficina de Calidad	Primera versión
1.1.0	12-06-2019	Oficina de Calidad	Se añade la nota3

Índice

1. INTRODUCCIÓN 4
1.1. OBJETO 4
1.2. AUDIENCIA 4
1.3. RESOLUCIÓN DE DUDAS 4
1.4. GLOSARIO DE TÉRMINOS 4
1.5. DOCUMENTOS RELACIONADOS 4
2. DESCRIPCIÓN GENERAL 5
3. RECOMENDACIONES GENERALES 5
4. PASOS PARA SINCRONIZAR SONARLINT CON SONARQUBE 6

1. INTRODUCCIÓN

1.1. OBJETO

Este documento pretende servir como guía para configurar SonarLint en todos los IDE (Eclipse en el caso de la SGSICS) utilizados para los proyectos Java desarrollados para el Ministerio del Interior, en adelante MIR.

1.2. AUDIENCIA

Está dirigido a todas las personas que colaboren en labores relacionadas con el desarrollo de los sistemas de información del Área de Desarrollo de la Administración Digital de la SGSICS.

1.3. RESOLUCIÓN DE DUDAS

En caso de requerir cualquier aclaración, pueden ponerse en contacto con la oficina de Calidad del MIR mediante correo electrónico a sgsics.calidadsw@interior.es con la etiqueta “[sonarlint]”.

1.4. GLOSARIO DE TÉRMINOS

Acrónimo	Descripción
SGSICS	Subdirección General de Sistemas de Información y Comunicaciones para la Seguridad.
MIR	Ministerio del Interior.
Plugin	Componente de software que añade una funcionalidad concreta a una herramienta de uso más genérico.
IDE	Entorno de Desarrollo Integrado

1.5. DOCUMENTOS RELACIONADOS

/Ubicación/Documento	Descripción
PublicacionIntranet/20-ANEXOS/MIR-INT-INTEGRACION-GRADLE-JENKINS-SONAR.docx	*Documento que describe los requisitos para la integración de proyectos de desarrollo con las herramientas corporativas de automatización de tareas (Gradle), plataforma de integración continua (Jenkins) y evaluación continua de la calidad del código (SonarQube).*

2. DESCRIPCIÓN GENERAL

Sonarqube ayuda al desarrollo en la detección, de forma rápida y automática, de fallos en el código, a través de las fases de revisión automática de la rama principal del código fuente, construcción del proyecto, pruebas unitarias y, adicionalmente, en la revisión de código estático y dinámico, existencia de código duplicado o cobertura de pruebas unitarias, entre otras.

Todo este proceso se inicia cuando se realiza un *"commit"* en la rama monitorizada, que suele ser el *"trunk"* y afecta, a parte de la infraestructura del MIR, a la aplicación y base de datos de Jenkins, máquina donde está alojada, aplicación y base de datos de Sonarqube, máquina donde está alojada, más las comunicaciones a través de la red del MIR.

SonarLint es una herramienta que se integra en el IDE (Eclipse) para poder ver los errores del código antes de subirlo a subversión y ser analizarlo con SonarQube.

Utilizando SonarLint se evitan fallos en el build de integración continua de SonarQube. Se trata de un plugin que informa, al mismo tiempo que se escribe el código, de la **deuda técnica, code smells y otras malas prácticas** que se pueda estar incurriendo. A los pocos minutos de empezar a usarlo, se empieza a escribir código más limpio, ordenado, y menos propenso a los bugs.

Haciendo uso de SonarLint se ahorra tiempo y recursos. No es necesario realizar un "commit" para que el equipo de desarrollo obtenga el análisis de Sonarqube con las reglas configuradas. Serán necesarios menos análisis de Sonarqube (tras compilación con Jenkins y paso al servidor de Sonarqube) por lo que el ahorro de recursos es considerable.

3. RECOMENDACIONES GENERALES

Se recomienda hacer un uso proporcional a las necesidades de las herramientas de compilación y análisis del código para que puedan aportar utilidad al desarrollo.

Es **obligatorio** utilizar todas las herramientas disponibles para el proceso de IC:

1. Plugin de **SonarLint** que se sincronizará con Sonarqube del MIR para tener las mismas reglas y obtener los mismos resultados.
2. Un IDE donde se integrará el plugin de **SonarLint** y se realizarán las compilaciones en local (Eclipse).

Se debe integrar el código del "trunk" que haya sido compilado y analizado previamente con SonarLint en local.

La frecuencia de "commit" en el "trunk" es variable, no se puede establecer un periodo fijo, deberá ser según las necesidades del desarrollo, pero los aportes deberán ser significativos y después de la compilación y análisis en local con **SonarLint**. Es decir, no debería haber repetidos "commits" intermedios con 2 minutos de intervalo entre ellos, donde hay cambios pequeños y poco significativos en una misma clase.

Para ver información referenciada, leer el apartado: "2.3. INTEGRACIÓN SONARQUBE" del siguiente documento:

PublicacionIntranet/20-ANEXOS/MIR-INT-INTEGRACION-GRADLE-JENKINS-SONAR.docx

4. PASOS PARA SINCRONIZAR SONARLINT CON SONARQUBE

Para sincronizar Sonarlint con Sonarqube, hay que configurar el servidor de SonarQube con los siguientes datos:

- URL: https://sqa-sonarq.mir.es
- Username/token: d30f82e87d35ffb74c7ee336a6650fbe9102d5f3

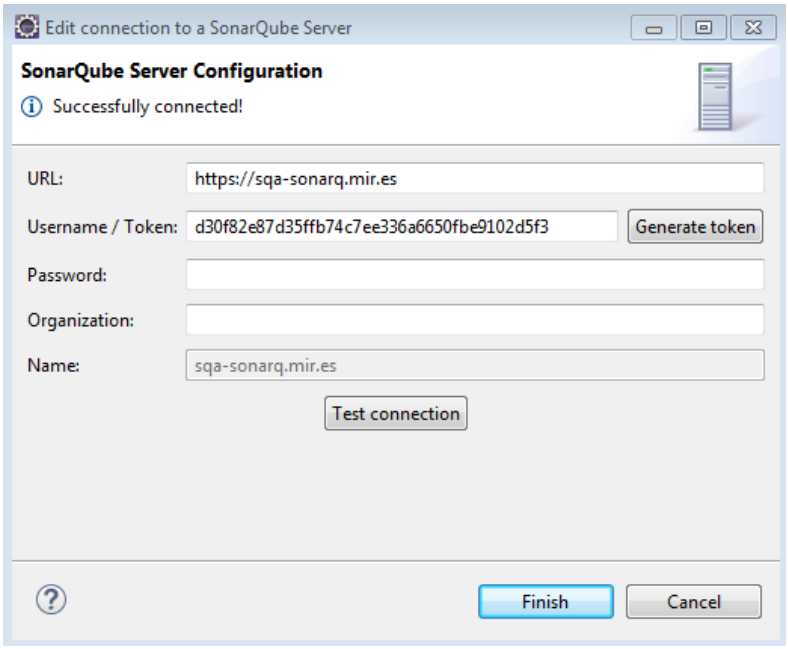

Ilustración 1 - Conexión con SonarQube

Con esos valores, el plugin se conectará al SonarQube del MIR, después sólo hay que hacer *autobind* del proyecto.

El siguiente ejemplo corresponde al proyecto “notif”. Se deberá hacer para el proyecto que corresponda en cada caso:

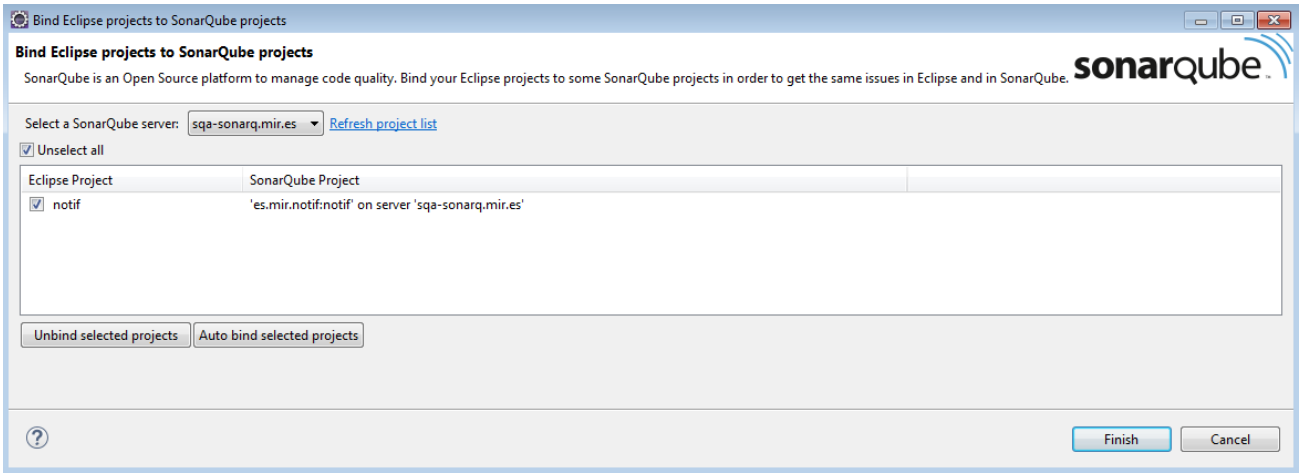

Ilustración 2 – Autobind del proyecto

Nota1: No olvidar añadir la clave pública de SonarQube en la JDK que ejecute el eclipse de cada equipo. Se recuerda a continuación:

- Descargar la clave pública accediendo a la URL https://sqa-sonarq-mir.es.
- Una vez accedido a la web de SonarQube del Ministerio con el explorador Chrome, se debe acceder a las "herramientas para desarrolladores", y en la pestaña "security" se encuentra la opción "View certificate". Esta opción dará la posibilidad de descargarlo en formato ".cer". Se almacena en el equipo local para poder instalarlo después.

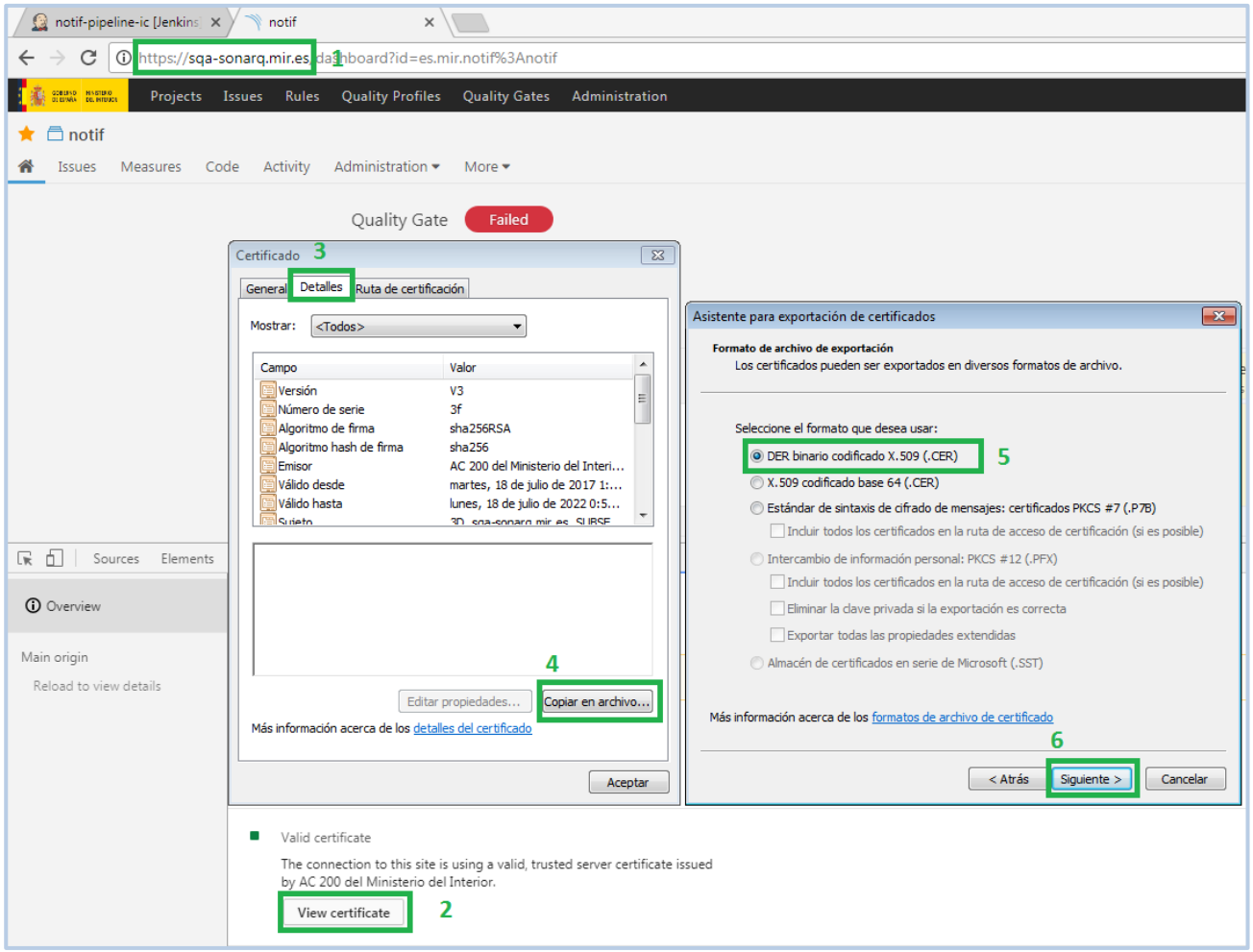

Ilustración 3 – Descargar Clave Pública

- Una vez descargado, por ejemplo con el nombre sonarqubeCert-sonarlint.cer, se debe importar en el *truststore* de la JDK que ejecute eclipse (se puede verificar o modificar la JDK utilizada en el fichero eclipse.ini). Para ello se debe utilizar la herramienta keytool como se indica a continuación:

  keytool -import -v -trustcacerts -alias ***sqa-sonarq.mir.es*** *-file C:****sonarqubeCert-sonarlint.cer*** *-keystore* ***C:/{ECLIPSE_JDK}/jre/lib/security/cacerts*** *-keypass changeit -storepass changeit*

  FAQ:

- Normalmente la keypass y el storepass viene por defecto con “changeit”.
- En caso de que ya tengas una clave con el alias “**sqa-sonarq.mir.es**”, impórtalo con otro alias.

- Para verificar que la instalación ha sido satisfactoria puedes ejecutar la herramienta keytool con las siguientes instrucciones:

  *keytool -list -noprompt -keystore "C**:/{ECLIPSE_JDK}**/jre/lib/security/cacerts" > c:/truststoreData.txt.*

- En el fichero truststoreData.txt debe encontrarse la entrada importada con anterioridad: sqa-sonarq.mir.es, 04-dic-2017, trustedCertEntry, Huella Digital de Certificado (SHA1): **D1:94:AB:26:20:D7:F0:F6:6A:65:90:DB:7F:96:49:0B:CF:FC:DC:96**
- Una vez instalada, reiniciar eclipse y probar de nuevo la conexión, que ya debería realizarse sin problema.

Nota2: Si hay algún problema con el plugin, se pueden ver los errores en la consola. Para que aparezca el log de SonarLint en la pestaña Console, seleccionar “SonarLint Console” en el icono que se muestra en la captura de pantalla:

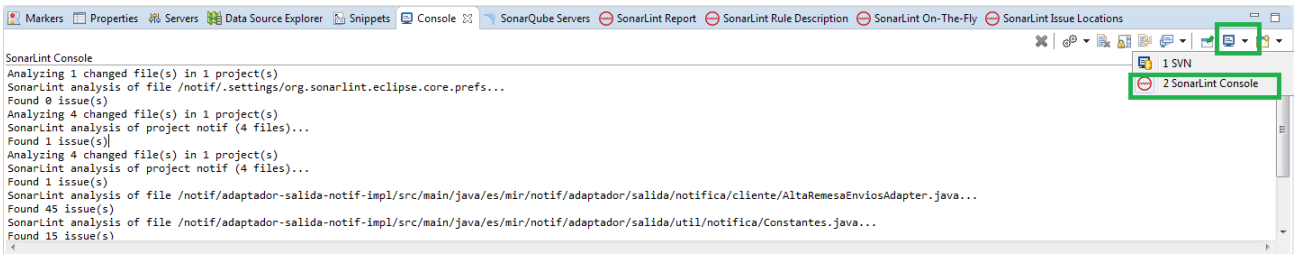

Ilustración 4 – Log de SonarLint

Nota3: Se ha detectado que en algunos equipos hay varias instalaciones de Java. Ya que algunas aplicaciones incorporan su versión java, que incluye el comando keytool. Por ejemplo, Visual Paradigm.

Hay que tenerlo en cuenta, ya que hay que asegurarse que el comando keytool que usemos sea el correcto, ya que nos daría problemas el certificado.

Es recomendable usar rutas absolutas en todo el proceso y sobretodo en el comando keytool para evitar que se use otro keytool de otras versiones.

Además, hay que asegurarse que versión de java usa el eclipse, que debe ser igual al keytool que usemos.

Aunque esto es una mera comprobación, tenemos que asegurarnos al 100% con que máquina virtual de java se está ejecutando el eclipse, ya que normalmente tenemos instaladas varias versiones de java, como hemos comentado anteriormente.

Si es necesario, modificaremos el fichero eclipse.ini añadiendo:

-vm

C:/progra~1/Java/jre1.8.0_181/bin/javaw.exe

La ruta puede cambiar según donde se tenga instalado la máquina virtual java.