**SERVICIO DE CONSULTORÍA PARA HABILITAR PLATAFORMA SOFTWARE RELACIONADA AL MODELO DE INTEGRACIÓN CONTINUA PARA REALIZAR ANÁLISIS ESTÁTICO DEL CÓDIGO FUENTE DE LOS SISTEMAS DEL OSCE BASADO EN TECNOLOGÍA JAVA**

**Orden de Servicio Nº 0000849-2017-OSCE**

**MANUAL DE USO DE LA PLATAFORMA SOFTWARE HABILITADA**

**Elaborado por: Ing. Yhan Fransua Mandros Poblet**

**13 de diciembre del 2017**

**HISTORIAL DE REVISIONES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Autor** | **Descripción** |
| 13/12/2017 | 1.0 | Yhan Fransua Mandros Poblet | Versión inicial |

**Tabla de Contenido**

[**1.** **LINEAMIENTOS DE USO DE PLATAFORMA SOFTWARE HABILITADA** 4](#_Toc501071668)

[**1.1.** **LINEAMIENTOS DE USO PARA SVN** 4](#_Toc501071669)

[**1.2.** **LINEAMIENTOS DE USO PARA NEXUS** 4](#_Toc501071670)

[**1.3.** **LINEAMIENTOS DE USO PARA SONARQUBE** 5](#_Toc501071671)

[**2.** **ENTORNO DE DESARROLLADOR CON PLATAFORMA SOFTWARE HABILITADA** 7](#_Toc501071672)

[**2.1.** **INTEGRACIÓN CON SVN** 7](#_Toc501071673)

[**2.1.1.** **Instalar Tortoise SVN** 7](#_Toc501071674)

[**2.1.2.** **Instalar plugin Eclipse** 8](#_Toc501071675)

[**2.1.3.** **Administrar repositorios en servidor** 10](#_Toc501071676)

[**2.2.** **INTEGRACIÓN CON NEXUS** 11](#_Toc501071677)

[**2.2.1.** **Instalación de maven** 11](#_Toc501071678)

[**2.2.2.** **Configurar maven Eclipse** 13](#_Toc501071679)

[**2.3.** **INTEGRACIÓN CON SONARQUBE** 15](#_Toc501071680)

[**2.3.1.** **Instalar SonarQube Scanner** 15](#_Toc501071681)

[**2.3.2.** **Configurar Eclipse con SonarQube** 16](#_Toc501071682)

[**2.3.3.** **Crear proyectos SonarQube** 17](#_Toc501071683)

[**2.3.4.** **Analizar proyectos con SonarQube Scanner** 19](#_Toc501071684)

[**2.3.5.** **Analizar proyectos con Eclipse** 19](#_Toc501071685)

[**3.** **ASPECTOS TÉCNICOS EN MIGRACIÓN DE REPOSITORIOS** 23](#_Toc501071686)

[**3.1.** **MIGRACION REPOSITORIO SVN 1.7.4 HACIA 1.9.7** 23](#_Toc501071687)

[**3.1.1.** **Migración por repositorio** 23](#_Toc501071688)

[**3.1.2.** **Migración por copia física** 23](#_Toc501071689)

[**3.1.3.** **Copia de trabajo en entorno de desarrollador** 24](#_Toc501071690)

[**3.2.** **MIGRACION REPOSITORIO NEXUS 2.1.2 HACIA 3.6.2** 24](#_Toc501071691)

[**3.2.1.** **Habilitar instalación de Nexus 2.14.x** 24](#_Toc501071692)

[**3.2.2.** **Configurar Sonatype Nexus 2.14.x** 25](#_Toc501071693)

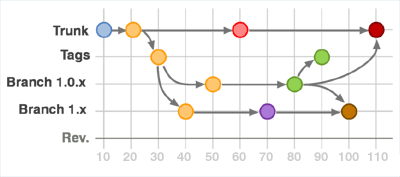
[**3.2.3.** **Realizar migración desde Sonatype Nexus 3.6.x** 26](#_Toc501071694)

[**4.** **RELACIÓN DE LA PLATAFORMA SOFTWARE HABILITADA CON LA NTP ISO/IEC 12207** 35](#_Toc501071695)

**MANUAL DE USO DE LA PLATAFORMA SOFTWARE HABILITADA**

1. **LINEAMIENTOS DE USO DE PLATAFORMA SOFTWARE HABILITADA**
   1. **LINEAMIENTOS DE USO PARA SVN**

* Los sistemas de control de versiones de código fuente como SVN deben usarse para gestionar código fuente de los sistemas del OSCE, evitar usar para la gestión documental de los proyectos de desarrollo de sistemas.
* La gestión de accesos a los repositorios deben ser gestionados a través de grupos, los usuarios deben integrarse con el repositorio de identidad del OSCE.
* La estructura a utilizar será la TTB:
* **Trunk**: Rama de desarrollo principal.
* **Tags:** Rama de gestión de versiones. Reservado para versiones cerradas, por tanto no se desarrollará sobre esta rama.
* **Branches:** Rama con evoluciones paralelas al Trunk.
* La dinámica de trabajo será la siguiente
* Antes de comenzar con la resolución de una tarea, se deberá asegurar la sincronización con el repositorio, bien con un Update o bien con un Checkout dependiendo de si se dispone previamente del código en el entorno local o no.
* Una vez resuelta la tarea, se deberá hacer otro Update para traer al entorno local los cambios que hayan podido ser realizados en paralelo al desarrollo actual. Subversion sabrá integrar los cambios del repositorio con los del entorno local en la mayoría de los casos, pero existirán situaciones que requieran de intervención humana para la integración. Estos casos, se deberán resolver de forma manual procurando mantener las modificaciones propias y las realizadas por los otros desarrolladores en paralelo.
* Finalmente se deberá hacer el Commit para hacer público al resto del equipo el código desarrollado. El alcance del Commit deberá limitarse al código relevante a la resolución de la tarea, y no mezclar desarrollos de distintas tareas en un mismo Commit.
* Flujo de desarrollo recomendado se muestra en la siguiente figura



* 1. **LINEAMIENTOS DE USO PARA NEXUS**
* Debe almacenar artefactos para tecnologías Maven, NPM y Docker.
* Nexus proporcionará el servicio proxy a repositorios externos y servicio de almacenamiento de los artefactos generados por los sistemas del OSCE.
* Los proyectos que pueden hacer uso de los artefactos gestionados por Nexus son:
* Proyectos java con dependencias maven.
* Proyectos front-end como los son angular, react, etc.
* Proyectos back-end con NodeJS.
* Para los artefactos relacionados a Maven se debe gestionar con la semántica descrita a continuación:
* El versionamiento debe hacerse a 03 niveles (A.B.C) e iniciar con 1.0.0.
* El 3er Nivel (C): corrección de errores y otros cambios menores: liberación de parche, incremento del número de tercer nivel, ejemplo: 1.0.1.
* El 2do Nivel (B): nuevas funciones que no rompen las funciones existentes: versiones menor, incrementar el número del segundo nivel, ejemplo: 1.1.0.
* El 1er Nivel (A): cambios que rompen la compatibilidad con versiones anteriores: versiones mayores, incremente el primer nivel, ejemplo: 2.0.0.
* Para los artefactos relacionados NPM se debe gestionar con la semántica descrita a continuación:
* El versionamiento debe hacerse a 03 niveles (A.B.C) e iniciar con 1.0.0
* El 3er Nivel (C): corrección de errores y otros cambios menores: liberación de parche, incremento del número de tercer nivel, ejemplo: 1.0.1.
* El 2do Nivel (B): nuevas funciones que no rompen las funciones existentes: versiones menor, incrementar el número del segundo nivel, ejemplo: 1.1.0.
* El 1er Nivel (A): cambios que rompen la compatibilidad con versiones anteriores: versiones mayores, incremente el primer nivel, ejemplo: 2.0.0.
* Como consumidor, puede especificar qué tipo de actualizaciones puede aceptar su aplicación en el archivo package.json. Si estuviera comenzando con un paquete 1.0.4, así es como debería especificar los rangos:
  + Versiones de parches: 1.0 o 1.0.x o ~ 1.0.4
  + Versiones menores: 1 o 1.x o ^ 1.0.4
  + Versiones mayores: \* o x
* Para los artefactos relacionados a Docker se debe realizar con la semántica descrita a continuación:
* El versionamiento debe hacerse a 03 niveles (A.B.C) e iniciar con 1.0.0.
* El 3er Nivel (C): corrección de errores y otros cambios menores: liberación de parche, incremento del número de tercer nivel, ejemplo: 1.0.1.
* El 2do Nivel (B): nuevas funciones que no rompen las funciones existentes: versiones menor, incrementar el número del segundo nivel, ejemplo: 1.1.0.
* El 1er Nivel (A): cambios que rompen la compatibilidad con versiones anteriores: versiones mayores, incremente el primer nivel, ejemplo: 2.0.0.
* La creación de nuevas imágenes docker deben basarse principalmente en la imagen de Linux Alpine que es una distribución Linux de características mínimas requeridas de menor tamaño.
  1. **LINEAMIENTOS DE USO PARA SONARQUBE**
* SonarQube debe ser configurado con plugins que permita analizar los siguientes lenguajes: java, javascript, xml.
* Los proyectos SonarQube del OSCE debe crearse con un versionamiento de 02 niveles y corresponder a unidades de despliegue independientes.
* Los proyectos SonarQube del OSCE debe considerar la siguiente semántica:
* **Name:** el nombre descriptivo de la unidad de despliegue acompañada de la versión, ejemplo: Sistema 1.0
* **Key:** la clave del proyecto debe ser un código o palabra corta que identifique al proyecto, ejemplo sistema
* **Branch:** la rama debe ser representada por la versión de 02 niveles, ejemplo: 1.0
* La ejecución del análisis de código fuente de los proyectos SonarQube del OSCE en un entorno de desarrollador debe realizarse a través de plugins como SonarLint.
* La ejecución y registro del resultado del análisis de código fuente de los proyectos SonarQube del OSCE deben realizarse a través de sonar-scanner.
* Se debe establecer el rol de analista de calidad para que realice la actividad de ejecutar y registrar el resultado del análisis de código fuente de los proyectos SonarQube del OSCE.
* Para la complejidad ciclomática se debe usar la métrica de perspectiva de riesgo que planteo el Software Engineering Institute (SEI) en su artículo Software Technology Reference Guide en (<http://www.sei.cmu.edu/reports/97hb001.pdf>)

|  |  |
| --- | --- |
| **Complejidad Ciclomática** | **Evaluación de Riesgo** |
| 1-10 | Programa simple, sin mucho riesgo. |
| 11-20 | Más Complejo, riesgo moderado. |
| 21-50 | Complejo, programa de alto riesgo. |
| Mayor que 50 | Programa imposible de probar muy alto riesgo. |

* La compuerta de calidad que usarán por defecto los proyectos SonarQube del OSCE tendrá la siguiente configuración:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Over leak Period** | **Operator** | **Warning** | **Error** |
| Complexity / Class |  | is less than |  | 20 |
| Coverage on New Code | Always | is less than |  | 80 |
| Maintainability Rating | Never | is worse than |  | A x |
| Maintainability Rating on new code | Always | is worse than |  | A x |
| Reliability Rating | Never | is worse than |  | A x |
| Reliability Rating on New Code | Always | is worse than |  | A x |
| Security Rating | Never | is worse than |  | A x |
| Security Rating on New Code | Always | is worse than |  | A x |

* Se deben crear los perfiles de calidad del siguiente cuadro:

|  |  |
| --- | --- |
| **Nombre** | **Descripción** |
| OSCE-Java | Usado para el lenguaje Java, contiene reglas de codificación de severidad: bloqueante, crítica y mayor. |
| OSCE-JavaScript | Usado para el lenguaje JavaScript, contiene reglas de codificación de severidad: bloqueante, crítica y mayor. |
| OSCE-XML | Usado para el lenguaje XML, contiene reglas de codificación de severidad: bloqueante, crítica y mayor. |

* Los umbrales de calidad para el análisis estático de código fuente de los proyectos SonarQube del OSCE deben ser comunicados con el siguiente cuadro:

|  |  |  |
| --- | --- | --- |
| **Medida** | **Valor esperado** | **Comentario** |
| Complejidad ciclomática media por clase | **debe ser menor igual a 20** | Se acepta el riesgo moderado. |
| Deuda técnica permitida | La deuda técnica máxima permitida es de **20 días** | Con 01 desarrollador se puede mejorar la codificación realizada, a más desarrolladores asignados a esta mejora mayor deuda técnica permitida. |
| Cantidad permitida de defectos del tipo Bug | Debe ser igual **00** | Este tipo de defectos afecta la confiabilidad del sistema |
| Cantidad permitida de defectos del tipo vulnerabilidad | Debe ser igual **00** | Este tipo de defectos afecta la seguridad del sistema |
| Porcentaje de código duplicado permitido | Debe ser menor igual a **25%** | Se acepta que algunas partes del código estarán duplicadas. |

* Los resultados del análisis de código estático de los proyectos de SonarQube del OSCE debe ser presentado en los siguientes cuadro:
* Las evidencias de tipo “bug” debe incluir la siguiente información

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TOTAL** **OCURRENCIAS:** | | 5,591 | **CALIFICACIÓN DE** **CONFIABLIDAD:** | E |
| **SEVERIDAD** | Bloqueante | 107 | **ESFUERZO PARA REMEDIAR CONFIABILIDAD:** | 164 días |
| Critica | 0 |
| Mayor | 5,484 |

* Las evidencias de tipo “vulnerabilidad” debe incluir la siguiente información

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TOTAL OCURRENCIAS:** | | 20 | **CALIFICACIÓN DE SEGURIDAD:** | E |
| **SEVERIDAD** | Bloqueante | 10 | **ESFUERZO PARA REMEDIAR SEGURIDAD:** | 3 días |
| Critica | 10 |
| Mayor | 0 |

* Las evidencias de tipo “código que huele mal” debe incluir la siguiente información

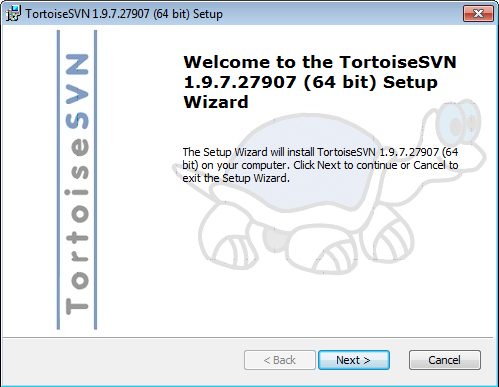
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TOTAL OCURRENCIAS:** | | 19,666 | **CALIFICACIÓN DE MANTENIBILIDAD:** | B |
| **SEVERIDAD** | Bloqueante | 80 | **DEUDA TÉCNICA:** | 1,166 días |
| Critica | 8,040 |
| Mayor | 11,546 |

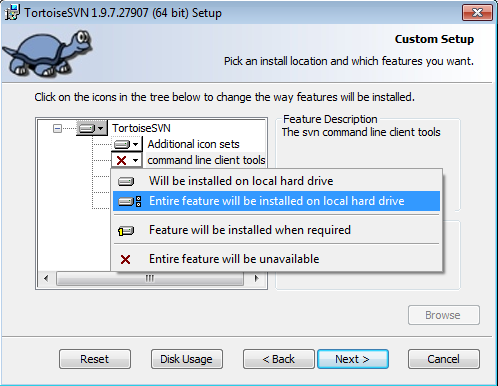
* Las conclusiones del análisis de código estático de los proyectos SonarQube del OSCE deben ser comunicados de la siguiente manera:
* El Proyecto XYZ 1.5 supero/no supero las pruebas de análisis estático de código de fuente.
* Existen 5,591 defectos del tipo bug que es superior al valor umbral de 00.
* Existen 20 defectos del tipo vulnerabilidad que es superior al valor umbral de 00.
* La deuda técnica es de 1,166 días que es superior al valor umbral de 20 días.
* La complejidad ciclomática tiene un valor de 25 que es superior al valor umbral de 20.
* Existe un 30% de líneas de código duplicado que es superior al valor umbral de 25%.

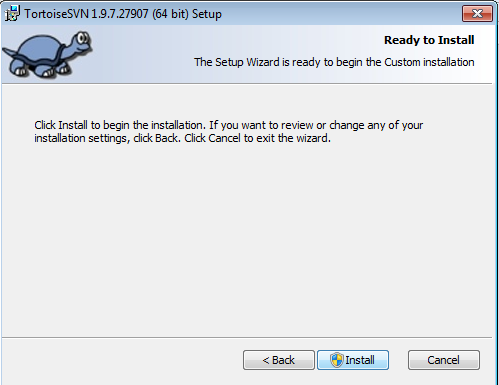
1. **ENTORNO DE DESARROLLADOR CON PLATAFORMA SOFTWARE HABILITADA**
   1. **INTEGRACIÓN CON SVN**

El servicio SVN 1.9.7 se encuentra publicado en la URL [**http://192.168.32.40/svn**](http://192.168.32.40/svn)

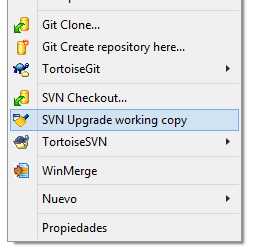
* + 1. **Instalar Tortoise SVN**
* Descargar el instalador Tortoise SVN 1.9.7 desde <https://tortoisesvn.net/downloads.html> y realizar la instalación en la estación de desarrollador:







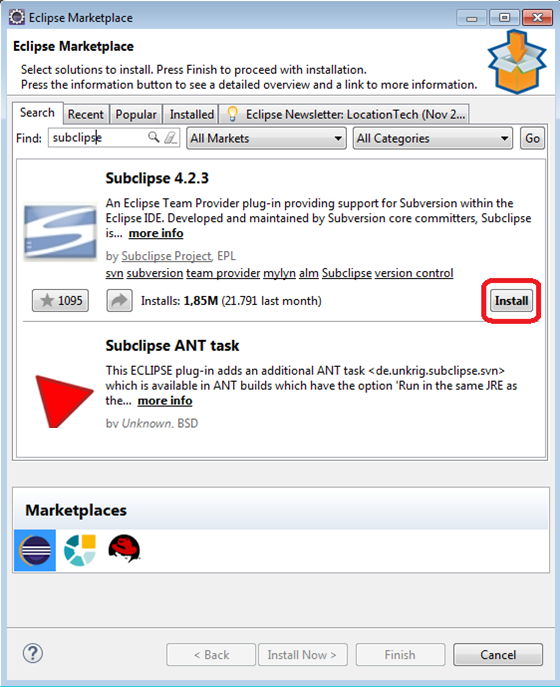
* Para los casos donde exista un repositorio con versión anterior de subversión deberá realizarse una actualización de la copia de trabajo

****

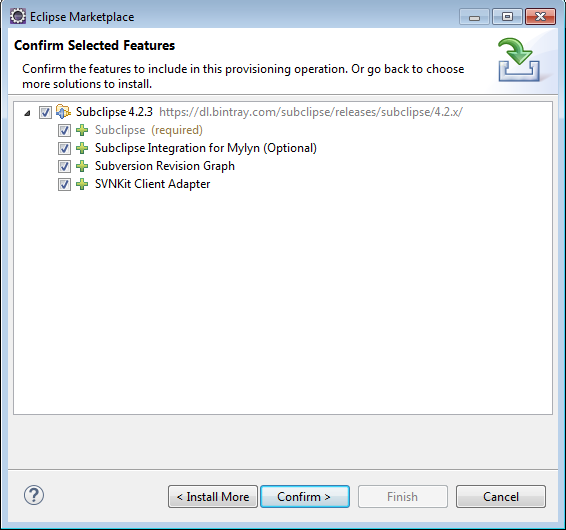
* + 1. **Instalar plugin Eclipse**
* Como plugin usar Sublcipse 4.2.0 el cual requiere como mínimo las siguientes versiones de IDE:

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Fecha de lanzamiento** | **Versión de plataforma** | **Proyecto** |
| Juno | 27 de junio de 2012 | 4.2 | [Juno projects](http://wiki.eclipse.org/Juno) |
| Kepler | 26 de junio de 2013 | 4.3 | [Kepler projects](http://wiki.eclipse.org/Kepler) |
| Luna | 25 de junio de 2014 | 4.4 | [Luna projects](http://wiki.eclipse.org/Luna) |
| Mars | 24 de junio de 2015 | 4.5 | [Mars projects](http://wiki.eclipse.org/Mars) |
| Neon | 22 de junio de 2016 | 4.6 | [Neon projects](http://wiki.eclipse.org/Neon) |
| Oxigen | 28 de junio de 2017 | 4.7 | [Oxigen projects](http://wiki.eclipse.org/Oxygen) |

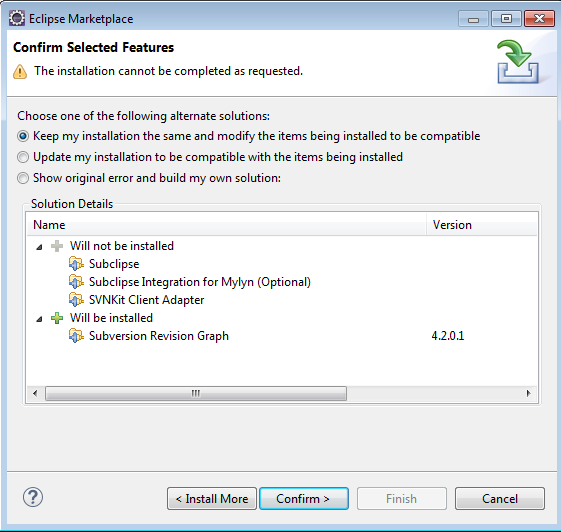
* En Eclipse ir a la opción “Help > Eclipse Marketplace …” y buscar “Subclipse”



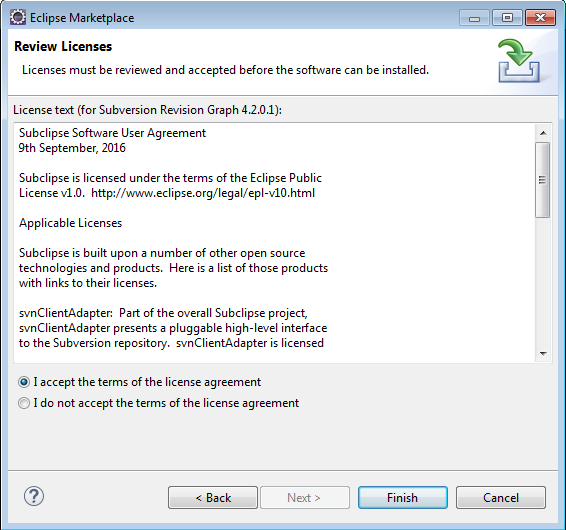
* Dar la opción “Install” y dar clic en la opción “Confirm >”

****

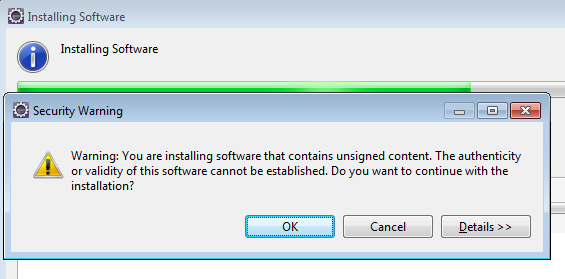
* En caso de ser Eclipse anterior a versión Oxygen usar la compatibilidad del plugin el proceso de instalación preguntará cuál de las alternativas de solución usará, se recomienda usar el de la imagen y dar clic en “Confirm >”:



* Aceptar los términos y condiciones y dar clic en “Finish”



* Aceptar el continuar con la instalación y al terminar reiniciar el IDE:

****

* + 1. **Administrar repositorios en servidor**
* Acceder por SSH al servidor que proporciona el servicio SVN (192.168.32.40)
* Ingresar a la consola de comandos del contenedor docker que ejecuta el SVN:

$ docker exec --it repfsw-svn1 /bin/sh

* Dentro del contenedor ejecutar los siguientes comandos, ejemplo para CREAR REPOSITORIO SVN de nombre new-repo:

$ cd /home/svn/data

$ svnadmin create **new-repo**

$ chown -R root:root /home/svn/data/**new-repo**

* 1. **INTEGRACIÓN CON NEXUS**

El servicio NEXUS 3.6.1 se encuentra publicado en la URL [**http://192.168.32.40/**](http://192.168.32.40/)**nexus3**

* + 1. **Instalación de maven**
* Descomprimir el recurso maven apache-maven-3.5.2.7z en el directorio **D:\sw\app** de manera que se pueda ejecutar **D:\sw\app\apache-maven-3.5.2\bin\mvn.cmd**
* Agregar las variables de sistema:

M2\_HOME = D:\sw\app\apache-maven-3.5.2

* Modificar el PATH del sistema operativo para agregar una ruta con la variable creada:

PATH = …;%M2\_HOME%\bin;

* Verificar el acceso al binario de Maven ejecutando:

c:\ mvn -version

* Editar el archivo %M2\_HOME%\conf\ settings.xml para que contenga lo siguiente:

<?xml version="1.0" encoding="UTF-8"?>

<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">

<localRepository>d:/svr/maven352/repositorio</localRepository>

<pluginGroups>

</pluginGroups>

<proxies>

<proxy>

<id>proxyOsce</id>

<active>true</active>

<protocol>http</protocol>

<host>172.16.2.3</host>

<port>3128</port>

<nonProxyHosts>localhost|172.\*|192.\*|&lt;local&gt;|localhost.localdomain|127.0.0.1|::1|172.16.51.26|[::1]</nonProxyHosts>

</proxy>

</proxies>

<servers>

<server>

<id>osce-releases</id>

<username>**usuario**</username>

<password>123456</password>

</server>

<server>

<id>osce-snapshots</id>

<username>usuario</username>

<password>123456</password>

</server>

<server>

<id>libs3-repository</id>

<username>usuario</username>

<password>123456</password>

</server>

</servers>

<mirrors>

<mirror>

<id>nexus</id>

<mirrorOf>\*</mirrorOf>

<url>http://192.168.32.40/nexus3/repository/public</url>

</mirror>

</mirrors>

<profiles>

<profile>

<id>osce</id>

<repositories>

<repository>

<id>osce-releases</id>

<url>http://192.168.32.40/nexus3/repository/osce-releases</url>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>false</enabled>

</snapshots>

</repository>

<repository>

<id>osce-snapshots</id>

<url>http://192.168.32.40/nexus3/repository/osce-snapshots</url>

<releases>

<enabled>false</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>osce-releases</id>

<url>http://192.168.32.40/nexus3/repository/osce-releases</url>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>false</enabled>

</snapshots>

</pluginRepository>

<pluginRepository>

<id>osce-snapshots</id>

<url>http://192.168.32.40/nexus3/repository/osce-snapshots</url>

<releases>

<enabled>false</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</pluginRepository>

</pluginRepositories>

</profile>

<profile>

<id>libs3</id>

<repositories>

<repository>

<id>libs3-repository</id>

<url>>http://192.168.32.40/nexus3/repository/libs3-repository</url>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>libs3-repository</id>

<url>http://192.168.32.40/nexus3/repository/libs3-repository</url>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</pluginRepository>

</pluginRepositories>

</profile>

<profile>

<activation>

<activeByDefault>true</activeByDefault>

</activation>

<repositories/>

</profile>

</profiles>

<activeProfiles>

<activeProfile>osce</activeProfile>

<activeProfile>libs3</activeProfile>

</activeProfiles>

</settings>

* En la siguiente sección se debe realizar las modificaciones de usuarios que configure en el servicio Nexus3 para poder realizar actualizaciones de artefactos en Nexus

**<servers>**

**<**server>

<id>libs3-repository</id>

<username>**usuario**</username>

<password>123456</password>

</server>

….

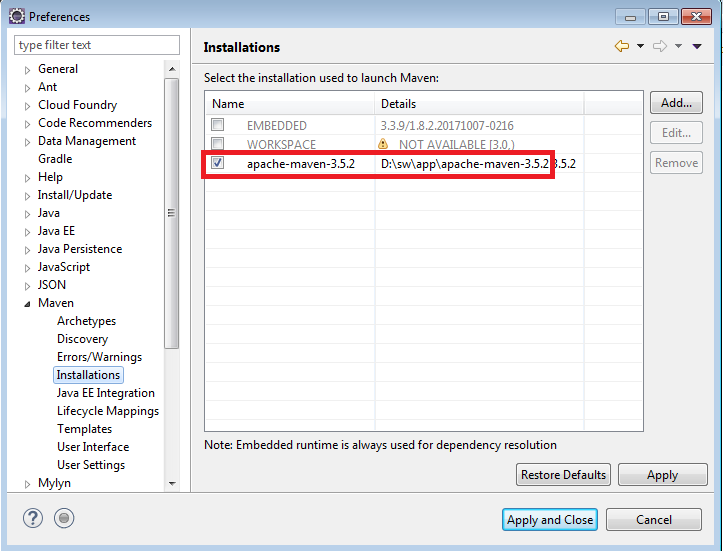
* + 1. **Configurar maven Eclipse**
* Descomprimir el recurso maven proporcionado **apache-maven-3.5.2.7z** de modo que se pueda ejecutar **D:\sw\app\apache-maven-3.5.2\bin\mvn.cmd**
* Editar las variables de entorno del sistema operativo para agregar la variable:

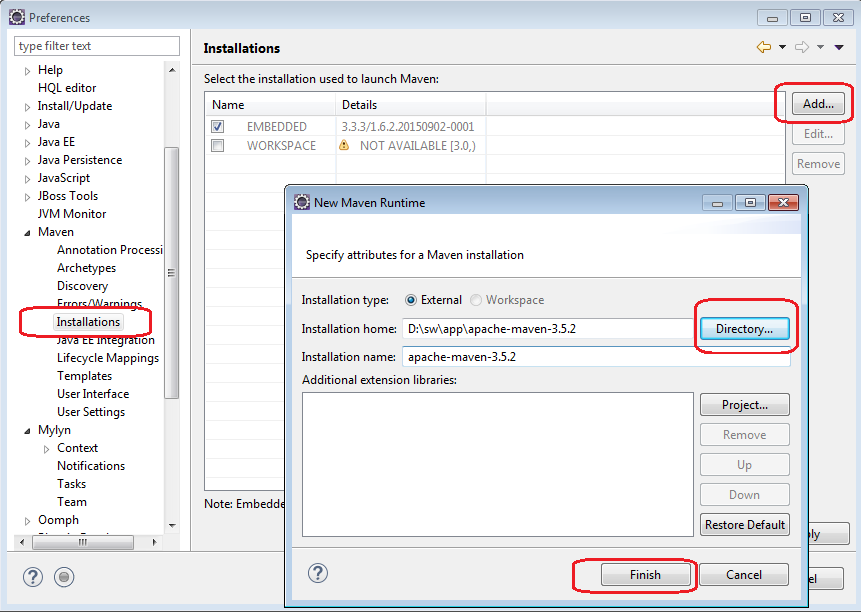
M2\_HOME = D:\sw\app\apache-maven-3.5.2

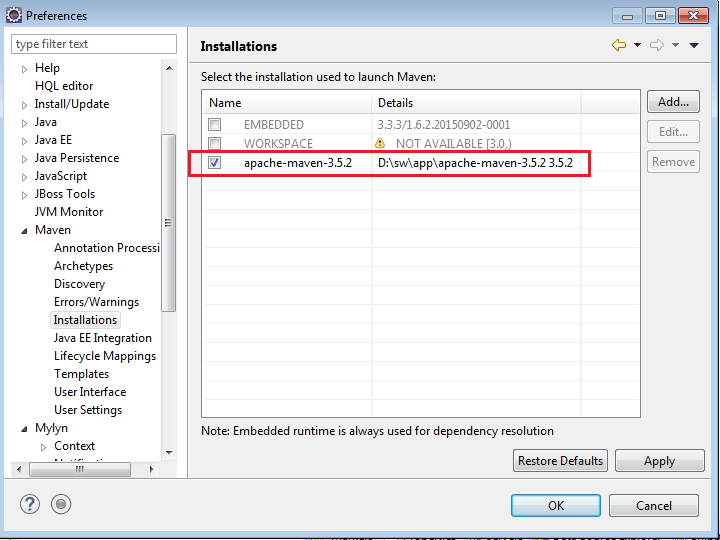
* Modificar el PATH del sistema operativo para agregar una ruta con la variable creada:

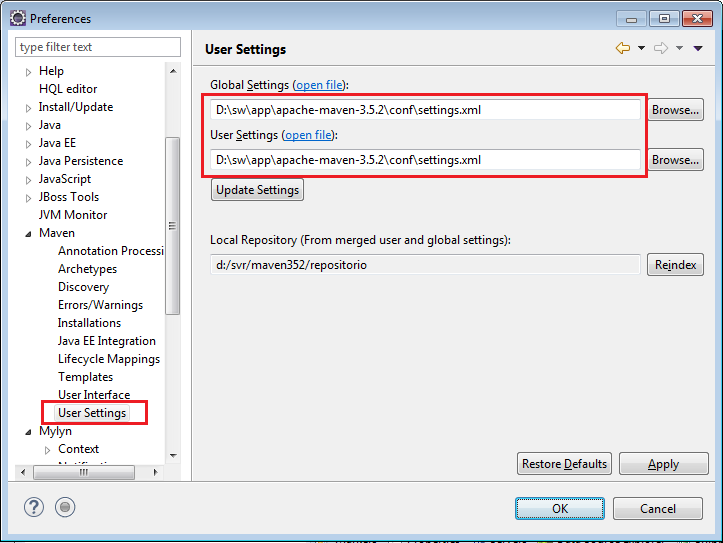
PATH = …;%M2\_HOME%\bin;

* En Eclipse **Windows > Preferences > Maven**, agregar instalación externa descomprimida en el punto anterior y establecer la configuración maven que se utilizara









* 1. **INTEGRACIÓN CON SONARQUBE**

El servicio SonarQube 6.7 se encuentra publicado en la URL [**http://192.168.32.41/sonarqube67**](http://192.168.32.41/sonarqube67)

* + 1. **Instalar SonarQube Scanner**

SonarQube Scanner permite analizar proyectos con SonarQube directamente en el directorio donde se encuentre el código fuente.

* SonarQube Scanner requiere de Java 1.8 o superior.
* Según el sistema operativo descargar del sonar-scanner-cli de la siguiente URL <https://docs.sonarqube.org/display/SCAN/Analyzing+with+SonarQube+Scanner>
* Descomprimir el contenido del archivo descargado y dejar en el nombre la versión a 3 niveles, ejemplo: D:\sw\app\sonar-scanner-3.0.3
* Editar las variables de entorno del sistema operativo para agregar la variable:

SQSCANNER\_HOME = D:\sw\app\sonar-scanner-3.0.3

* Modificar el PATH del sistema operativo para agregar una ruta con la variable creada:

PATH = …;%SQSCANNER\_HOME%\bin;

* Verificar el acceso al binario del SonarQube Scanner ejecutando:

c:\ sonar-scanner --version

* Actualizar la configuración global del servidor SonarQube que se utilizará, para ello editar el archivo %SQSCANNER\_HOME%\conf\sonar-scanner.properties al valor que le corresponderá

#Configure here general information about the environment, such as SonarQube DB details for example

#No information about specific project should appear here

#----- Default SonarQube server

sonar.host.url=http://192.168.32.41/sonarqube67

#----- Default source code encoding

sonar.sourceEncoding=UTF-8

#------ Authentication

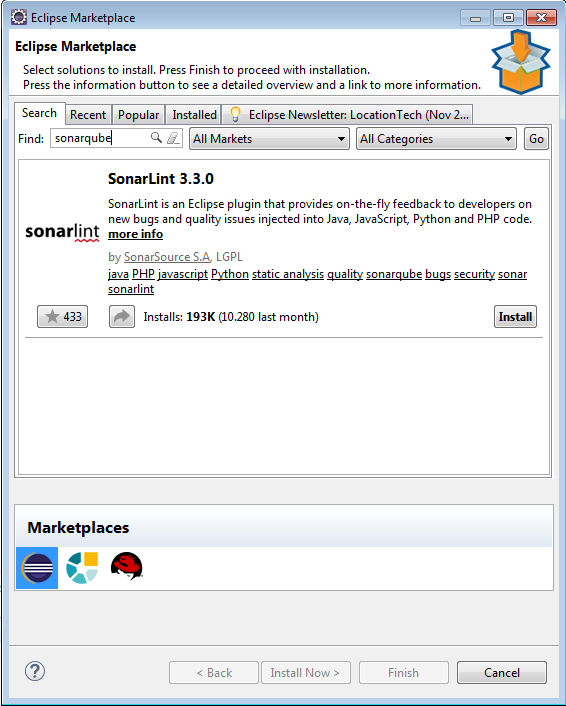
sonar.login=usuario

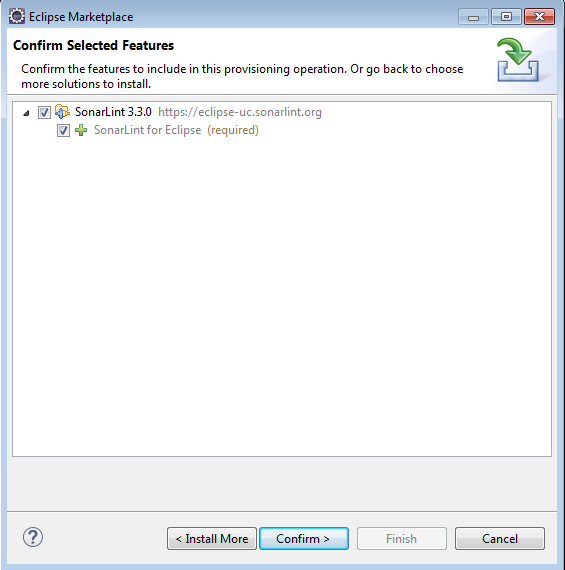
sonar.password=123456

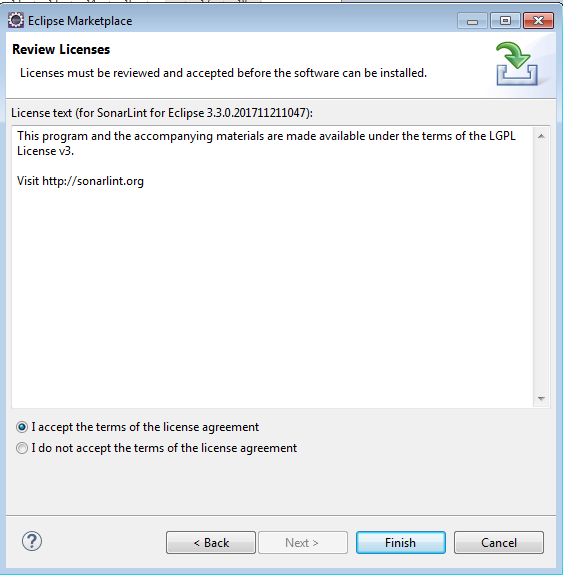
* + 1. **Configurar Eclipse con SonarQube**

El plugin a utilizar en Eclipse es SonarLint el permite al desarrollador codificar y ver los issues en el reportes de SonarLint basados en la configuración remota de SonarQube.

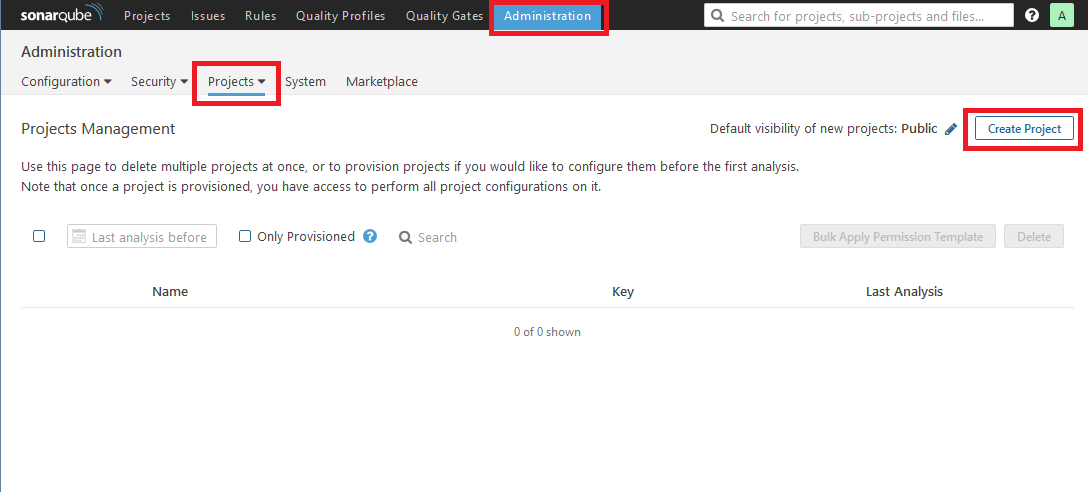
* Instalar el plugin desde Marketplace:





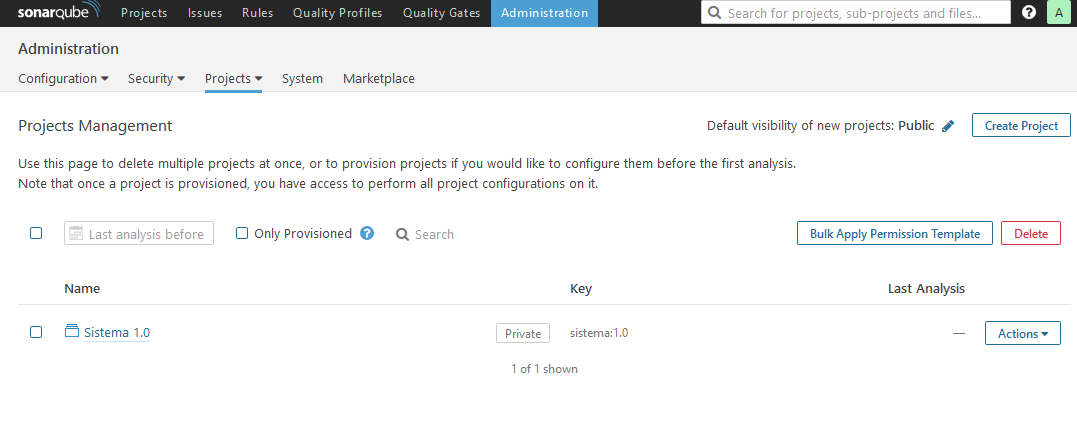


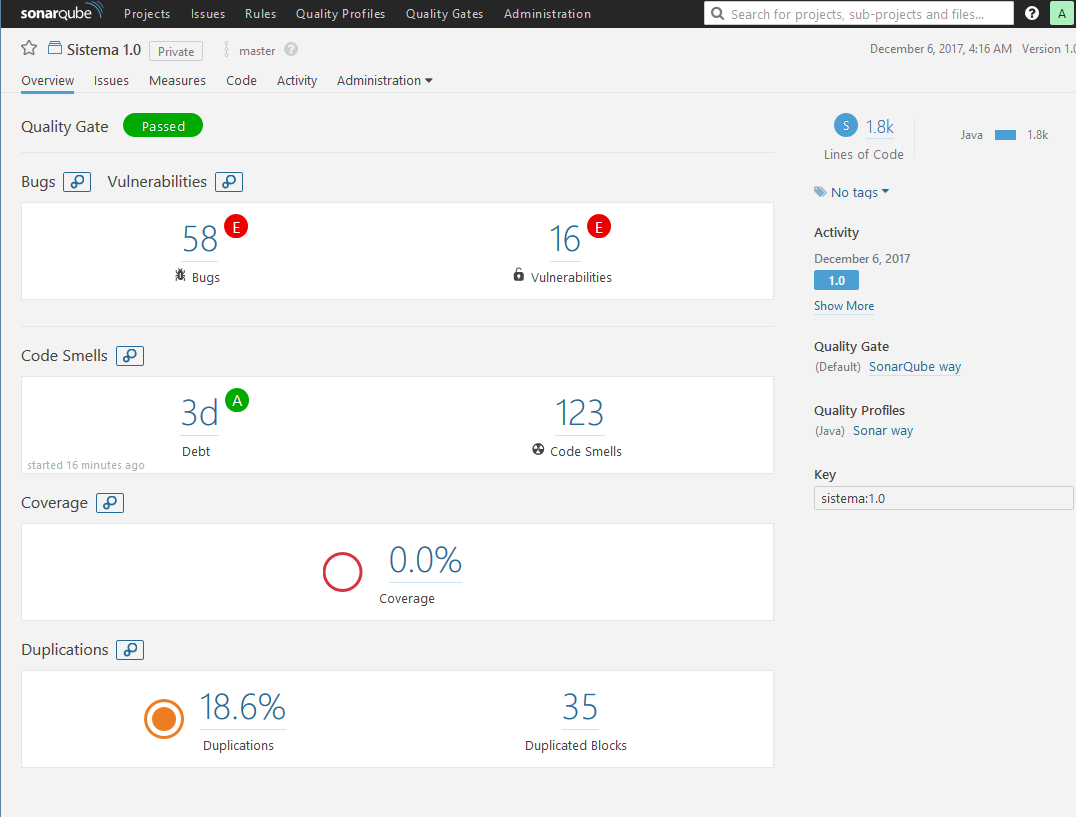
* + 1. **Crear proyectos SonarQube**
* Iniciar sesión con usuario que tenga privilegios de creación de proyectos



* Ingresar valores del proyecto SonarQube que contendrá la información de la revisión que se realice:







* + 1. **Analizar proyectos con SonarQube Scanner**
* Crear archivo de configuración de nombre **sonar-project.properties** en el directorio raíz donde se encuentre el proyecto con el siguiente contenido ejemplo:

*# must be unique in a given SonarQube instance*

*sonar.projectKey=sistema:1.0*

*# this is the name and version displayed in the SonarQube UI. Was mandatory prior to SonarQube 6.1.*

*sonar.projectName=Sistema 1.0*

*sonar.projectVersion=1.0*

*# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.*

*# This property is optional if sonar.modules is set.*

*sonar.sources=.*

*# Encoding of the source code. Default is default system encoding*

*sonar.sourceEncoding=UTF-8*

*# Configuracion de proyectos a escanear*

*sonar.java.source=1.8*

*sonar.sources=./company\_mailer/src/main/java,./ELibrary/src/main/java,./pharmacy/src/main/java*

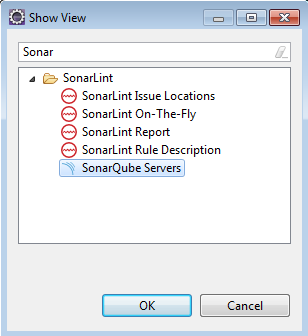
*sonar.java.binaries=./company\_mailer/target/classes,./ELibrary/target/classes,./pharmacy/target/classes*

* En el directorio que contiene el proyecto y el archivo sonar-project.properties ejecutar:

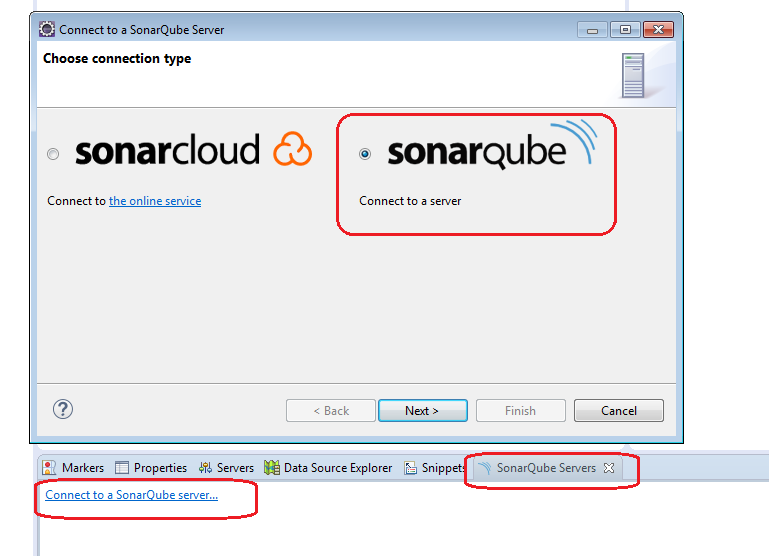
c:\mi-proyecto\sonar-scanner

**Nota:** el usuario que realiza el escaneo debe tener privilegios en el servidor para que sonar-scanner pueda registrar la información en el proyecto objetivo.

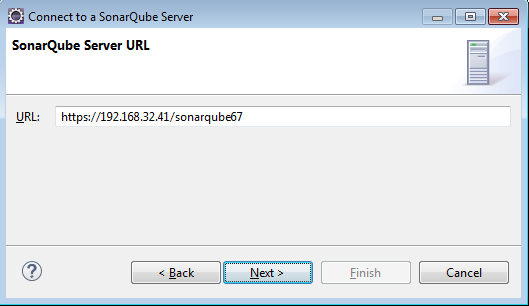
* La ejecución anterior realizara la carga del análisis en el servidor SonarQube configurado durante el proceso de instalación
  + 1. **Analizar proyectos con Eclipse**
* Para crear conexión a servidor SonarQube buscar en “Windows > Show view > Others” la palabra Sonar y seleccionar SonarQube Servers:



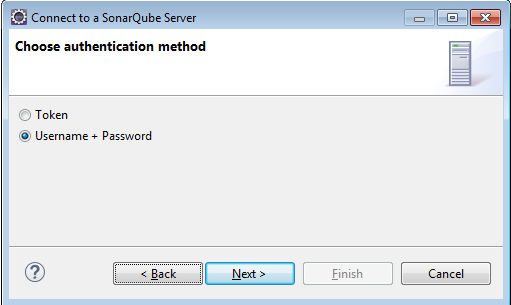
* En la vista apertura dar clic en la opción “Connect to a SonarQube Server”

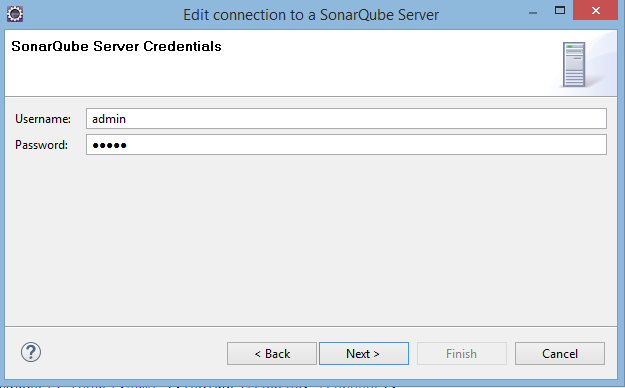


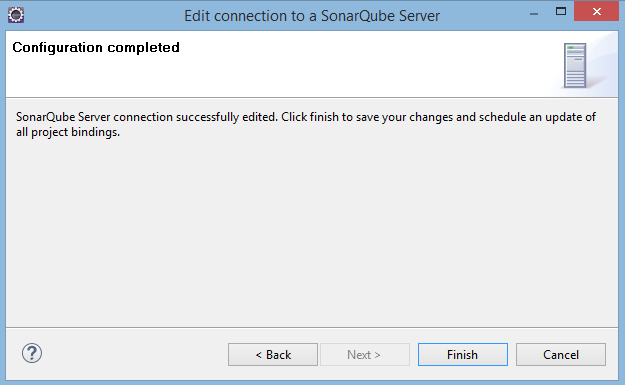
* Ingresar la URL del servidor que se usará en el proyecto



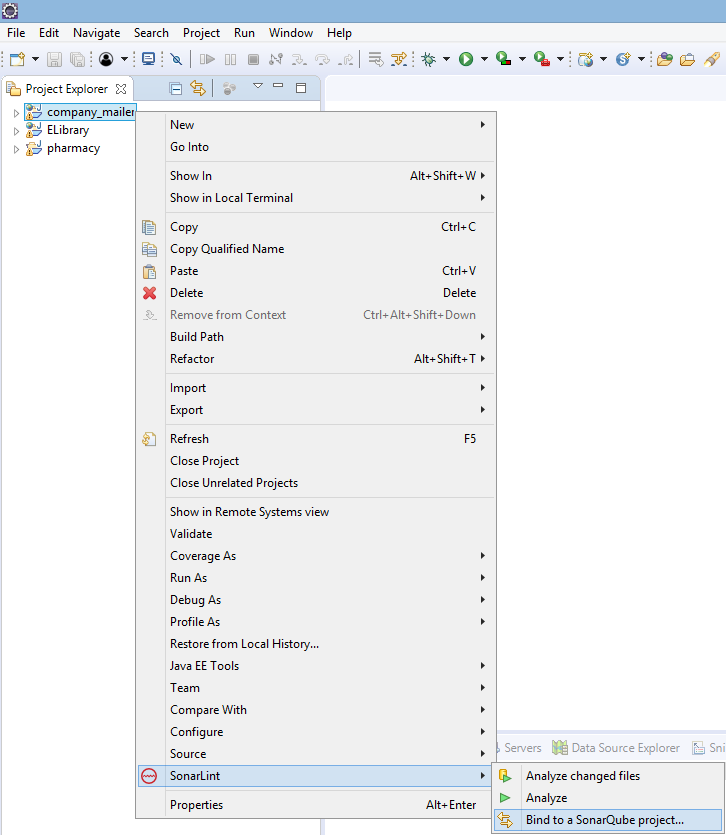
* Seleccione el método de autenticación al servidor SonarQube

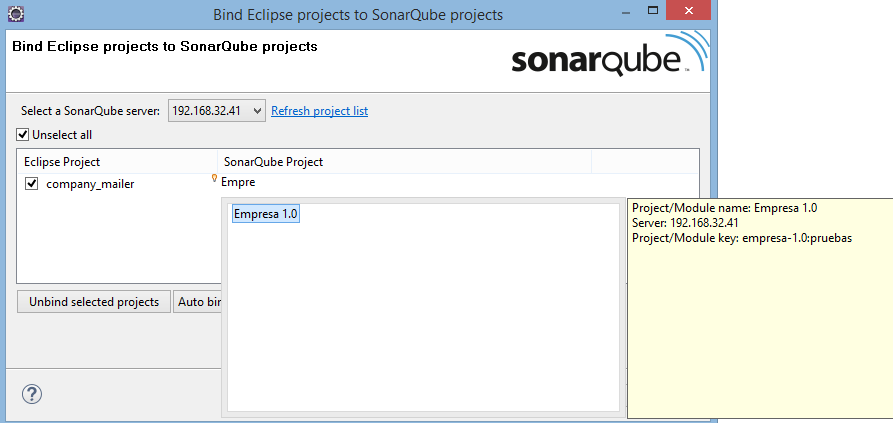




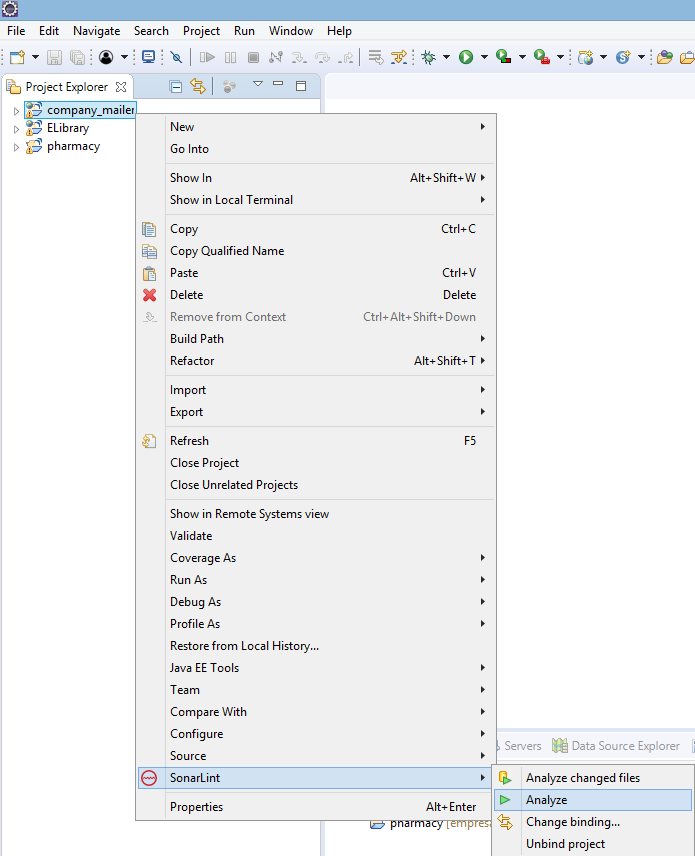


* Vincular el proyecto con repositorio SonarQube, para lo cual se deberá escribir el nombre del repositorio con el cual se desea vincular y aparecerá en el listado los repositorios disponibles en el servidor SonarQube:

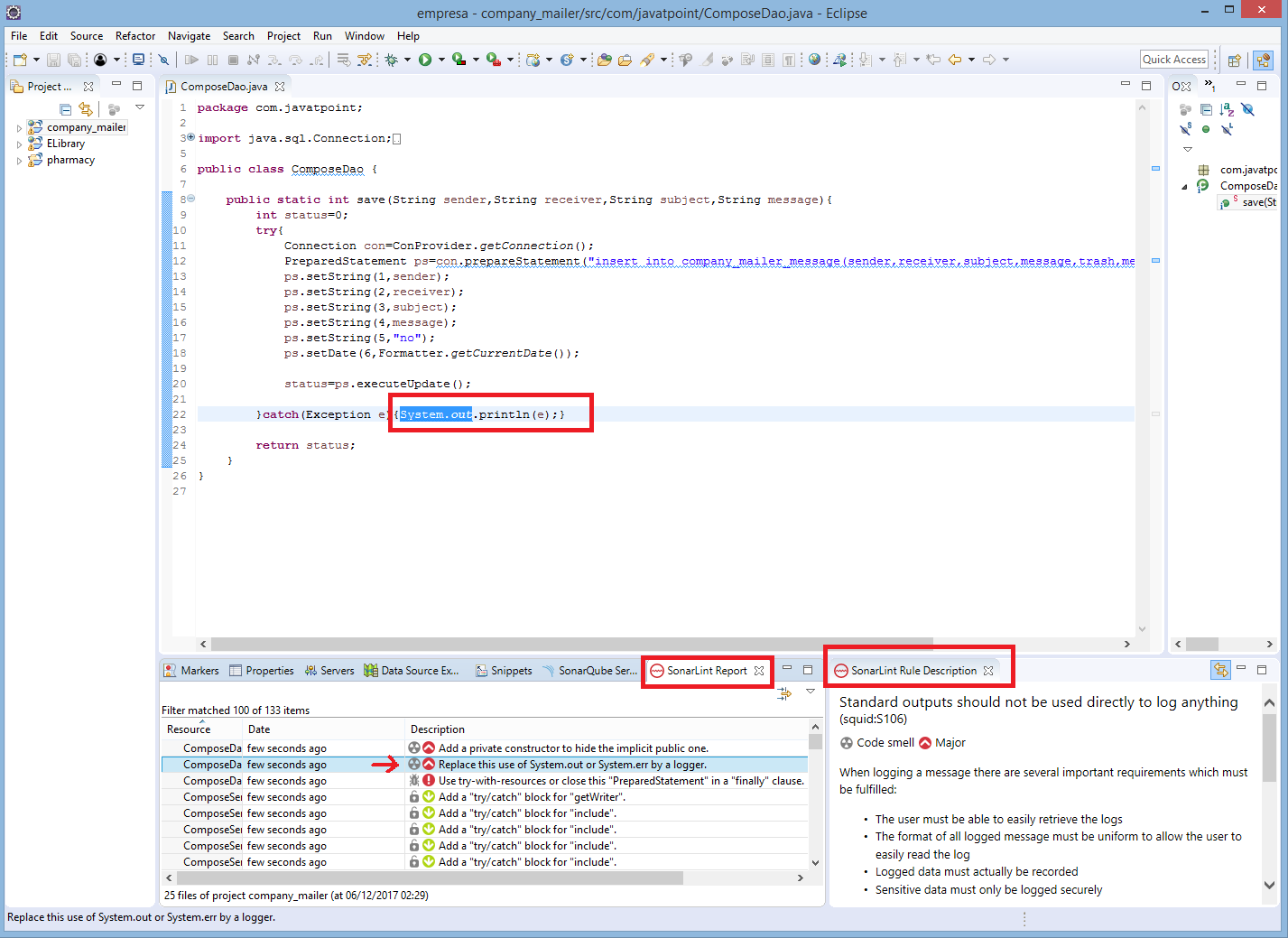




* Analizar proyecto



* Ver resultados



1. **ASPECTOS TÉCNICOS EN MIGRACIÓN DE REPOSITORIOS**
   1. **MIGRACION REPOSITORIO SVN 1.7.4 HACIA 1.9.7**
      1. **Migración por repositorio**

* En el servidor que contiene los repositorios a migrar crear el directorio donde se almacenara los dump’s de los repositorios:

$ mkdir -p /home/svrsvn/dumps

$ chown apache:apache -R /home/svrsvn/dumps

$ cd /home/svrsvn/dumps

* Generar dump de repositorio con el siguiente comando:

$ svnadmin dump /home/svrsvn/rep/repo-migrar/ > \

/home/svrsvn/dumps/repo-migrar\_dump

* Copiar archivo dump en nuevo servidor:

$ scp /home/svrsvn/dumps/repo-migrar\_dump \

operadorsvc@192.168.32.40:/fssvc/svn/1.9.7/svn-data/temp/repo-migrar\_dump

* En el nuevo servidor SVN dentro del contenedor crear el repositorio destino y proceder a importar:

$ docker exec -it repfsw-svn1 /bin/sh

$ cd /home/svn/data

$ svnadmin create **repo-migrar**

$ svnadmin load /home/svn/data/**repo-migrar** < /home/svn/data/temp/repo-migrar\_dump

* + 1. **Migración por copia física**
* Para realizar copia masiva de los repositorios existentes realizar lo siguiente:

$ nohup scp -pr /home/svrsvn/rep/ \

operadorsvc@192.168.32.40:/fssvc/svn/1.9.7/svn-data/

* En el nuevo servidor SVN dentro del contenedor crear el repositorio destino:

$ docker exec -it repfsw-svn1 /bin/sh

$ chown -R root:root /home/svn/data

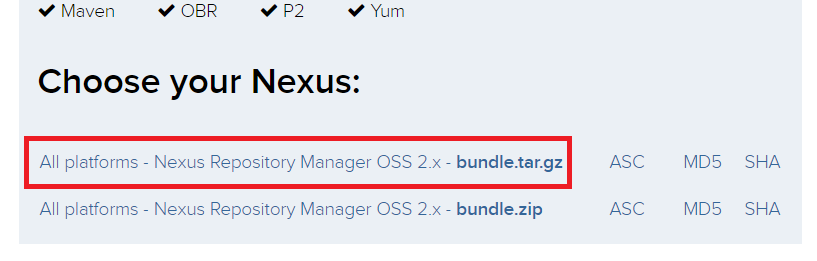
* + 1. **Copia de trabajo en entorno de desarrollador**
* En las estaciones clientes por cada repositorio se debe realizar 02 tareas:
  + svn cleanup
  + svn upgrade

**Nota:** De presentarse inconvenientes en la copia de trabajo es preferible crear una nueva copia ejecutando **svn checkout**

* 1. **MIGRACION REPOSITORIO NEXUS 2.1.2 HACIA 3.6.2**

Para realizar una migración de información hacia Nexus 3.6.x se requiere un servicio Nexus 2.14.x, dado que el OSCE dispone del servicio Nexus 2.1.2 se debe realizar las siguientes actividades

* + 1. **Habilitar instalación de Nexus 2.14.x**
* Instalar el jdk 1.8 en el sistema operativo destino, las instalaciones por defecto usan el directorio /**usr/java/jdk1.8.0\_151/** como **JAVA\_HOME**
* Copiar todo el contenido de la carpeta “/usr/local/sonatype-work” en la carpeta “/usr/local/sonatype-work-old”
* Descargar el binario “nexus-2.14.5” desde la siguiente URL <https://www.sonatype.com/download-oss-sonatype>

****

* Descomprimir el recurso archivo descargado (nexus-2.14.5-02-bundle.tar.gz) en la carpeta **“/usr/local/nexus-2.14.5-02**”
* Modificar el archivo “/usr/local/nexus-2.14.5-02/conf/ nexus.properties” con los siguientes valores

application-port=**8082**

nexus-webapp-context-path=/**nexus2145**

nexus-work=${bundleBasedir}/../**sonatype-work-old**/nexus

* Modificar el archive “/usr/local/nexus-2.14.5-02/bin/jsw/conf/wrapper.conf” con el siguiente valor

wrapper.java.command=**/****usr/java/jdk1.8.0\_151/bin/**java

* Iniciar el servicio instalado

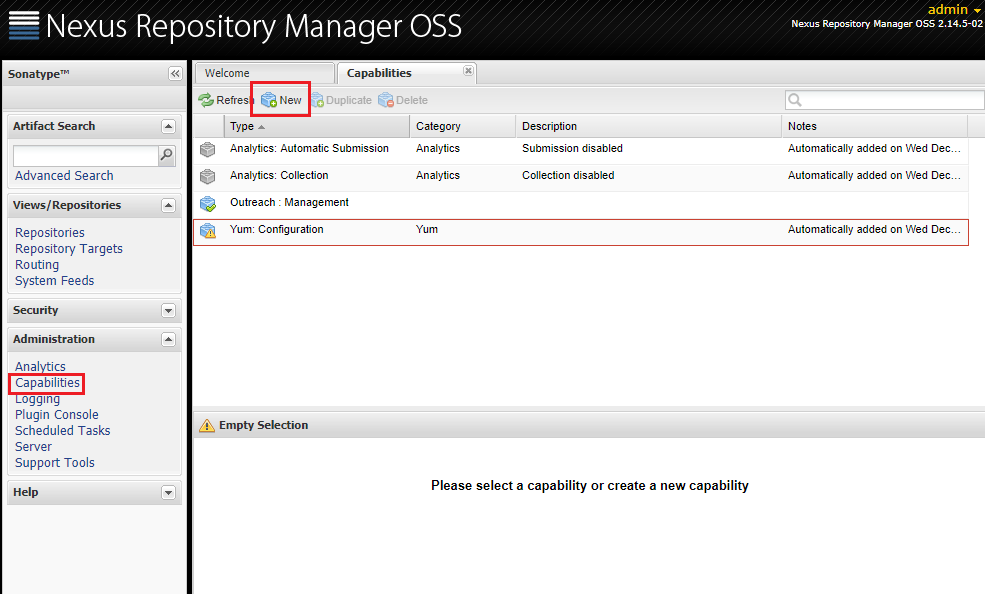
$ nohup /usr/local/nexus-2.14.5-02/bin/nexus start &

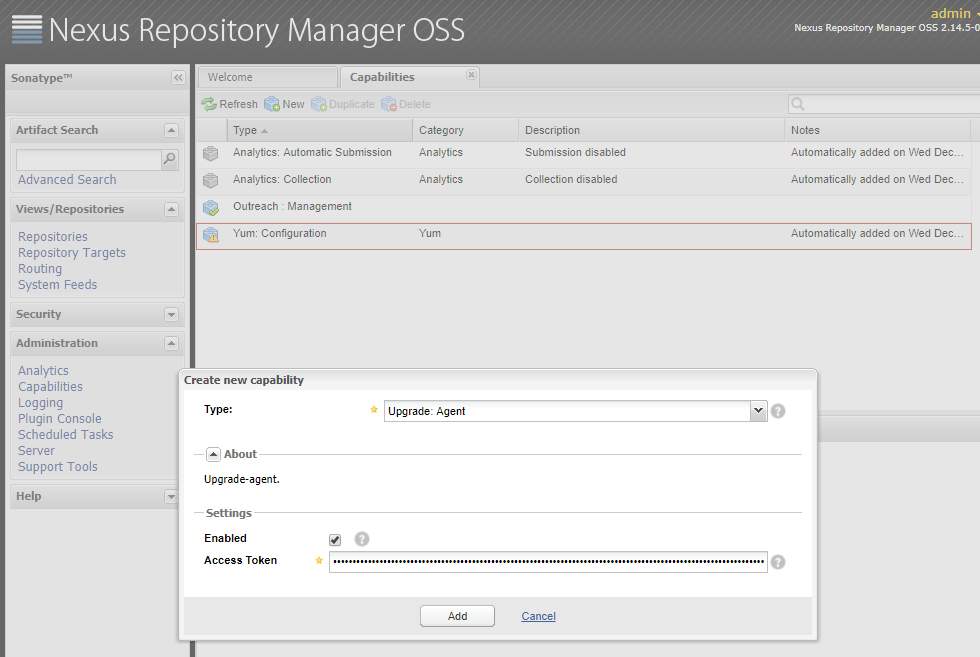
* Validar el acceso el servicio accediendo a la URL http://{host]:8082/nexus2145

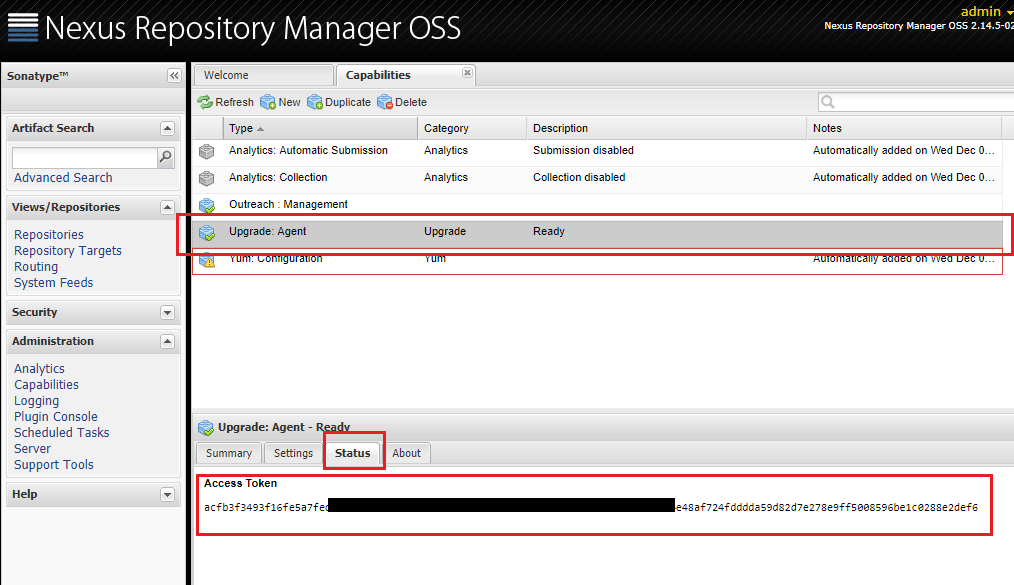
**Nota:** debido que la nueva instalación estará usando copia de data de la instalación anterior (/usr/local/sonatype-work-old) las credenciales serán las mismas que la anterior.

* + 1. **Configurar Sonatype Nexus 2.14.x**

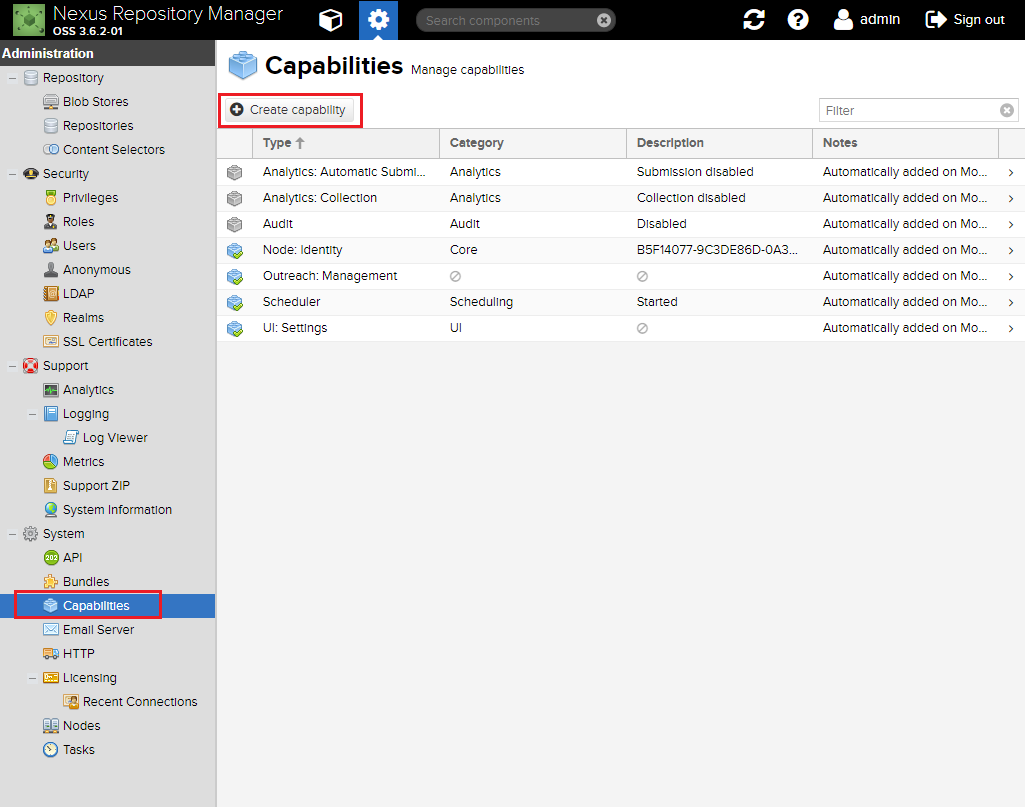
Habilitar una instalación Sonatype Nexus 2.14.x y configurar el agente de migración para exponer la información de este servicio:

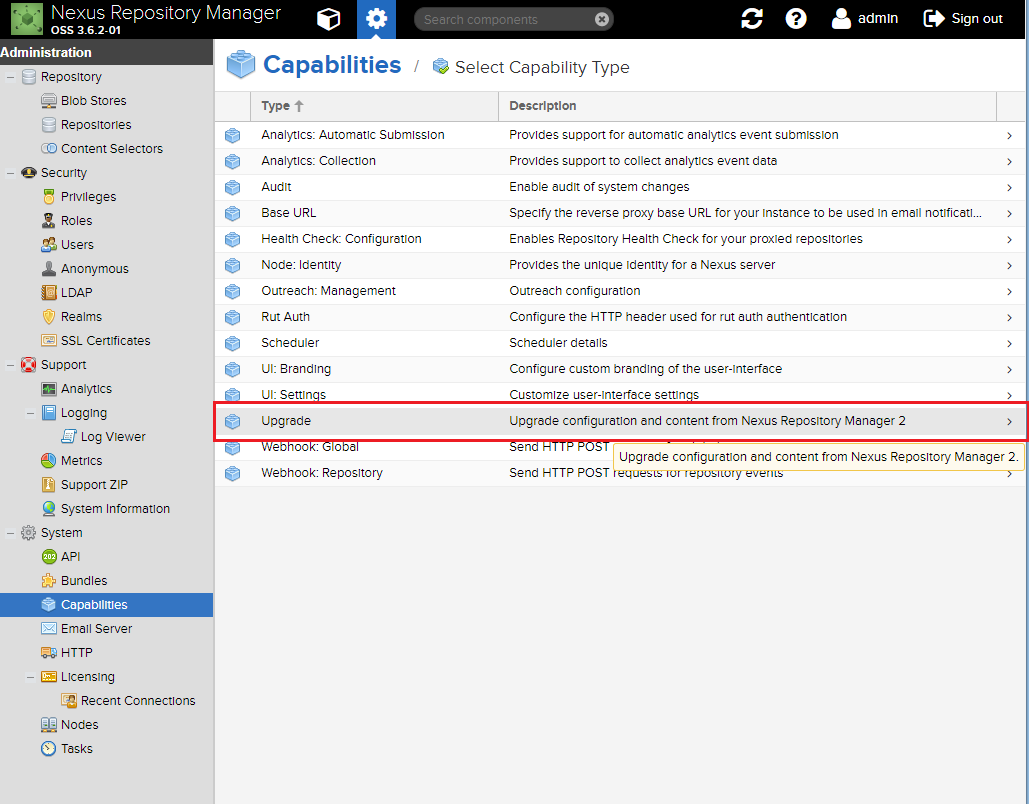
****

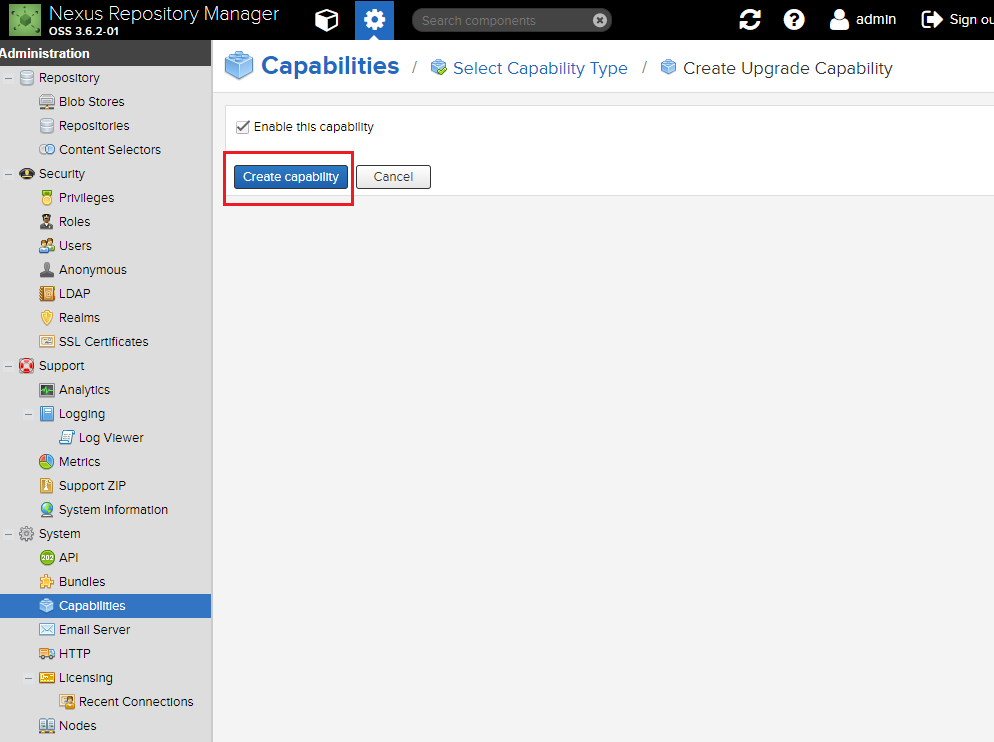
****

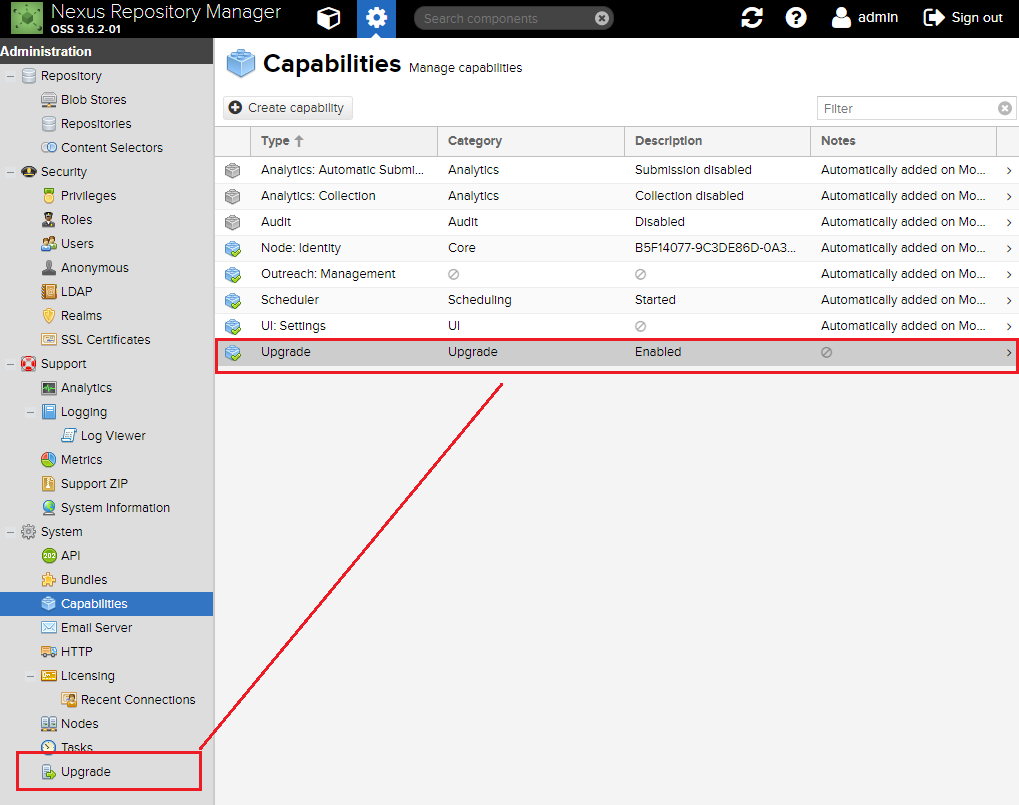
****

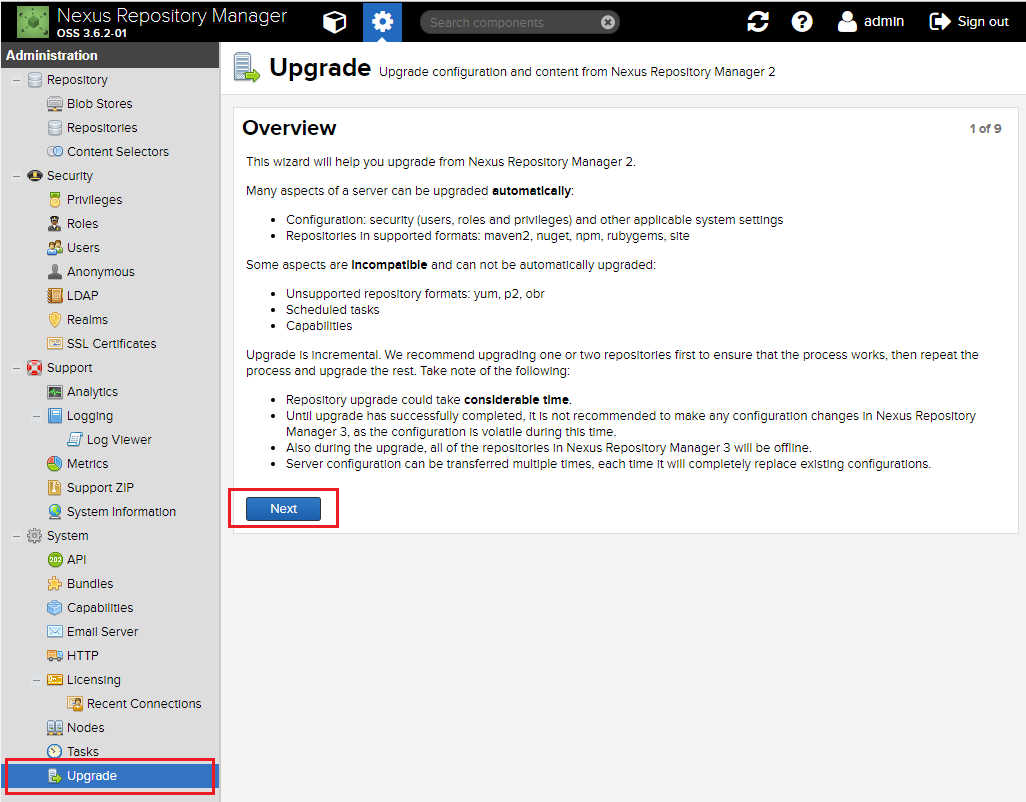
* + 1. **Realizar migración desde Sonatype Nexus 3.6.x**
* Habilitar una actualización (migración desde) de información desde una instalación Sonatype Nexus 2.14.x se debe configurar la capacidad de migración en Sonatype Nexus 3.6.x según se indica en las siguientes figuras:

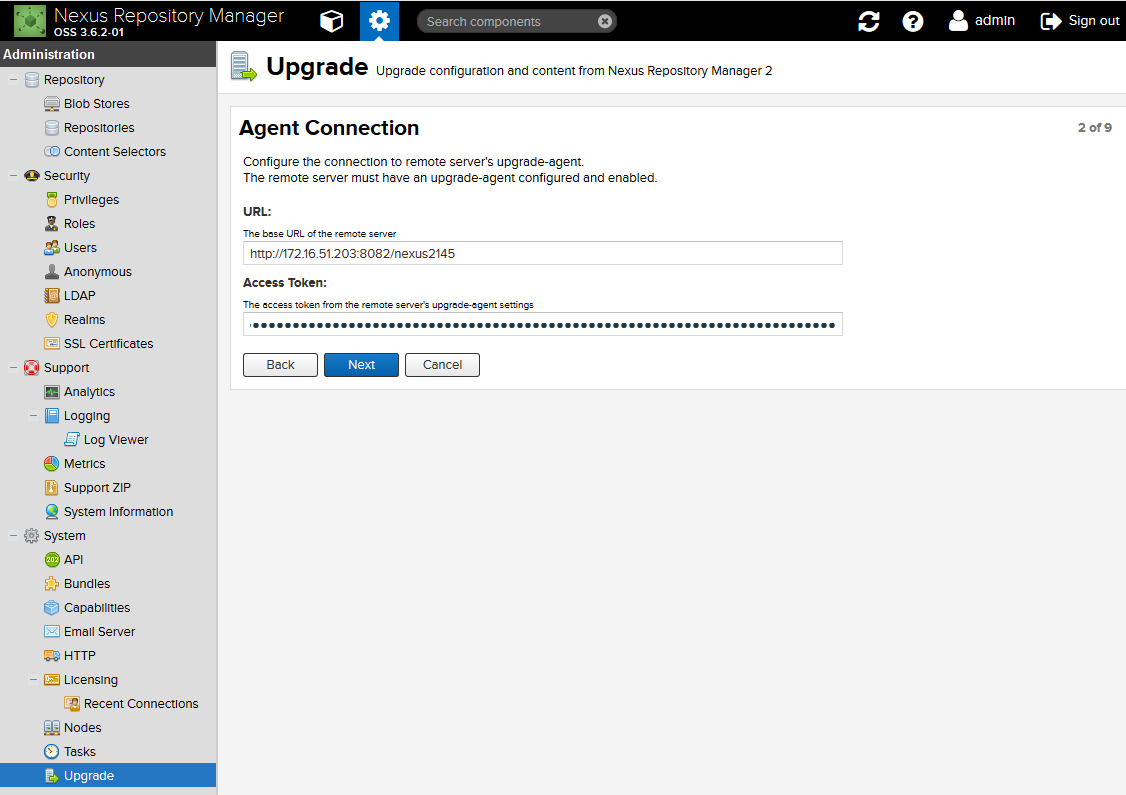
****

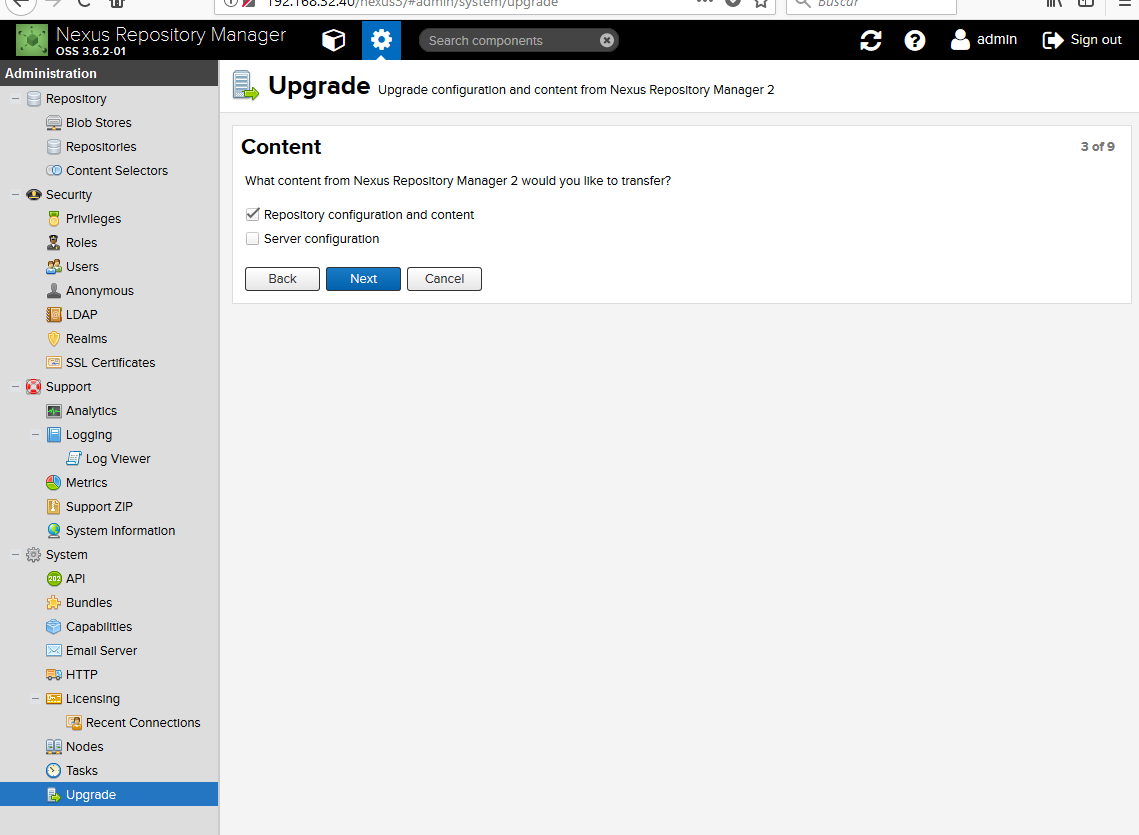
****

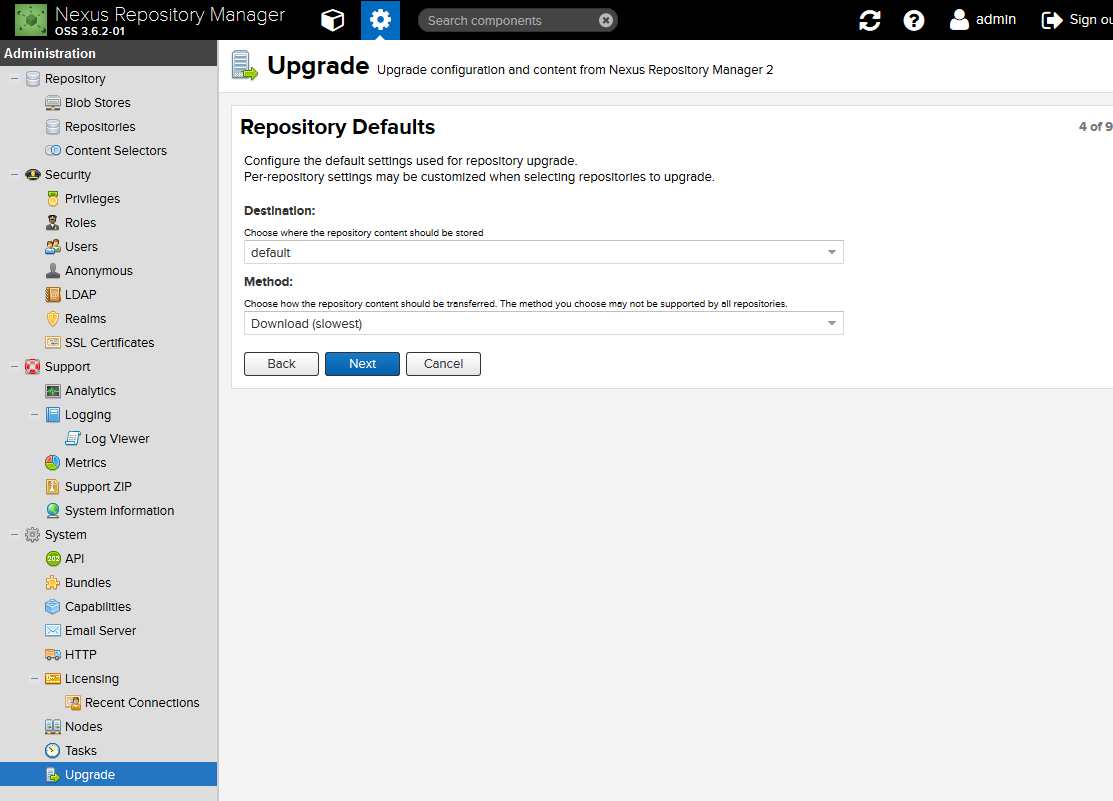
****

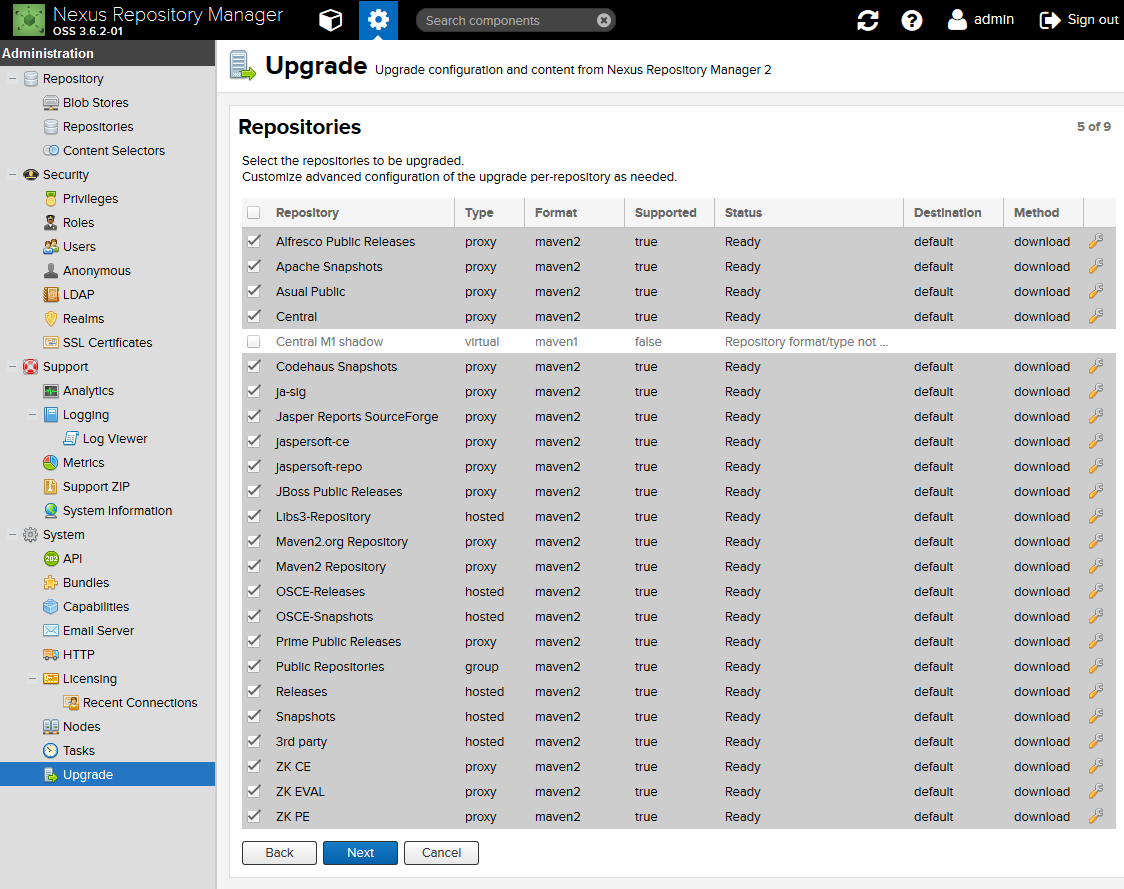
****

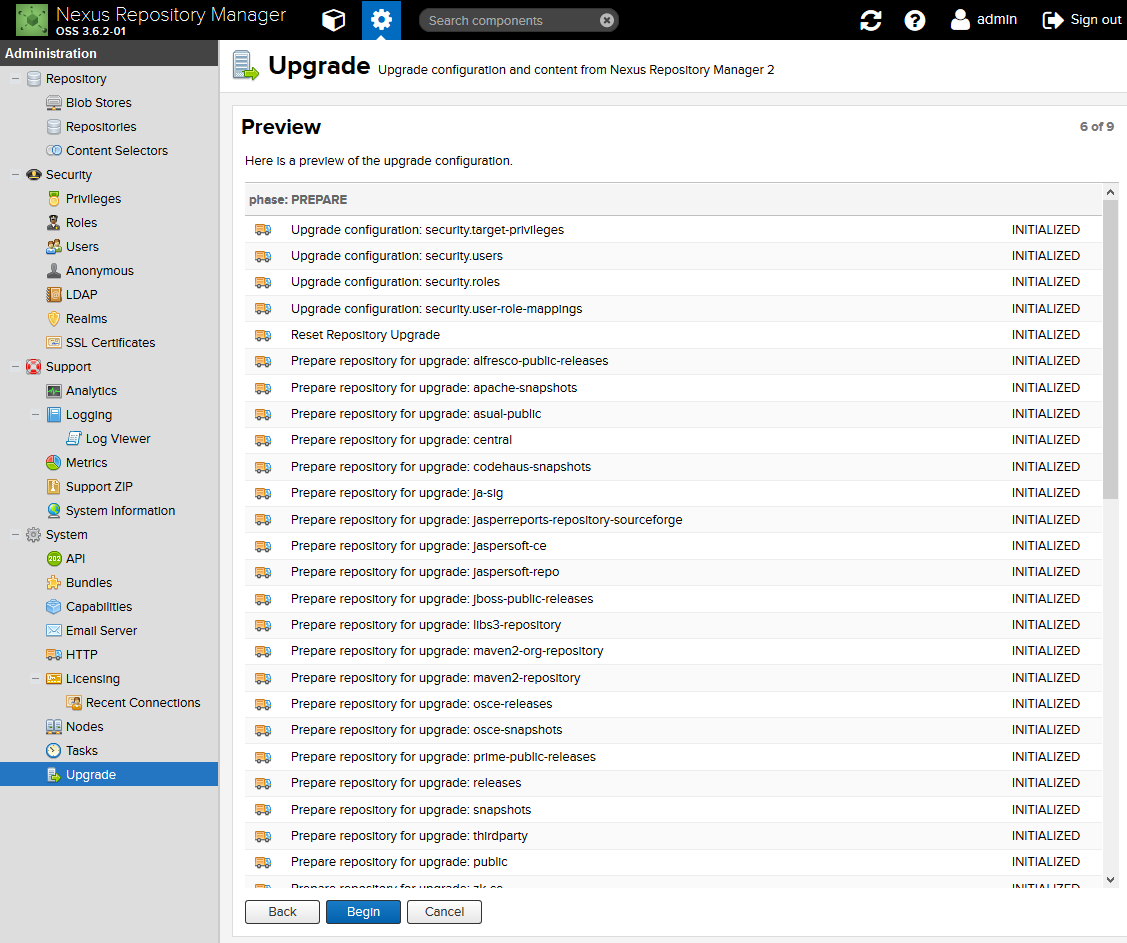
****

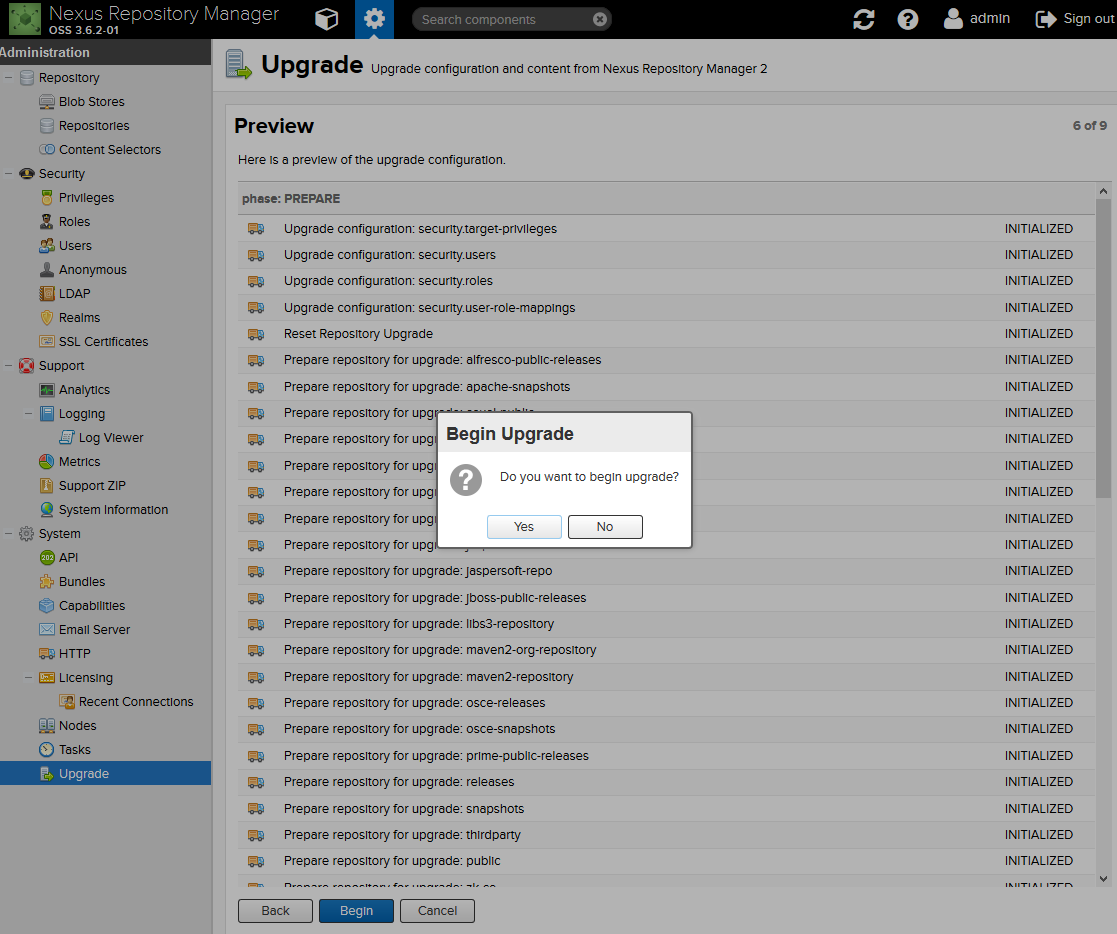
****

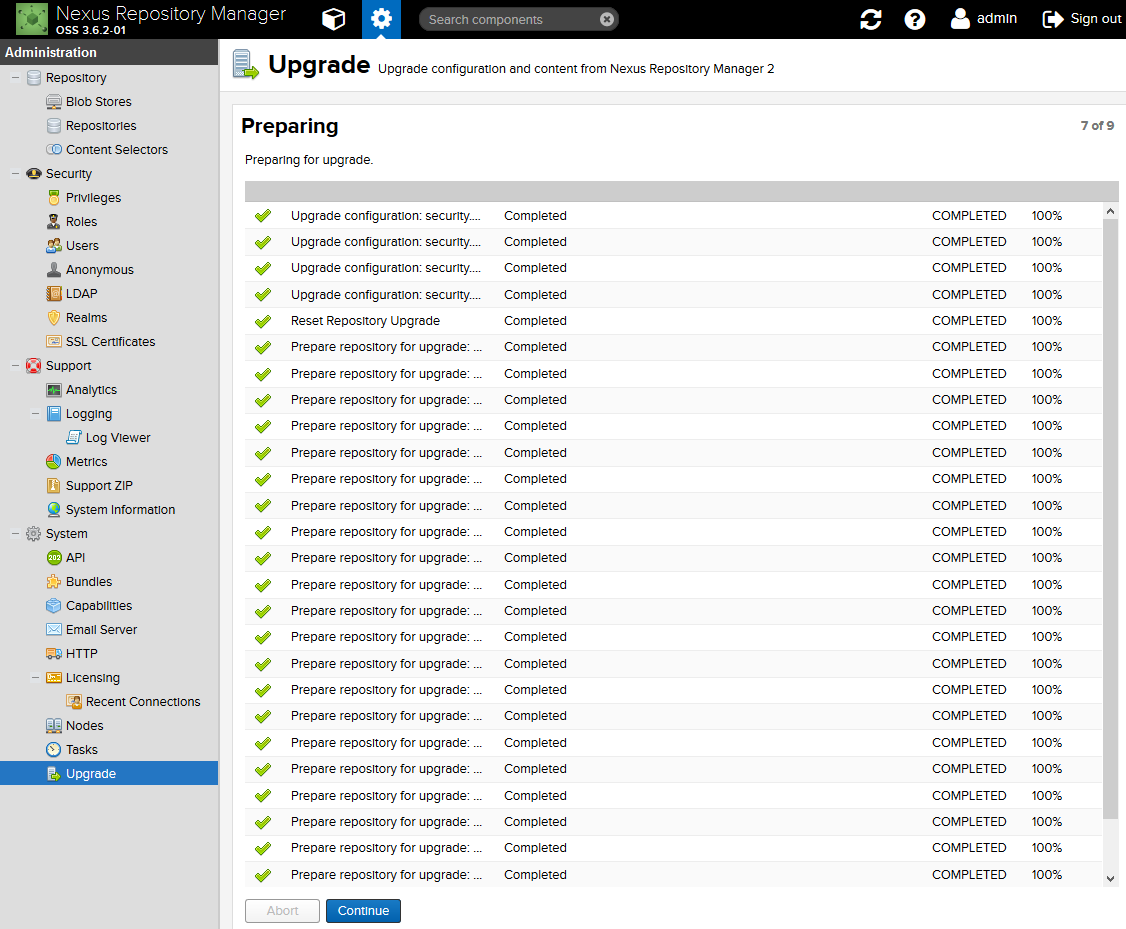
****

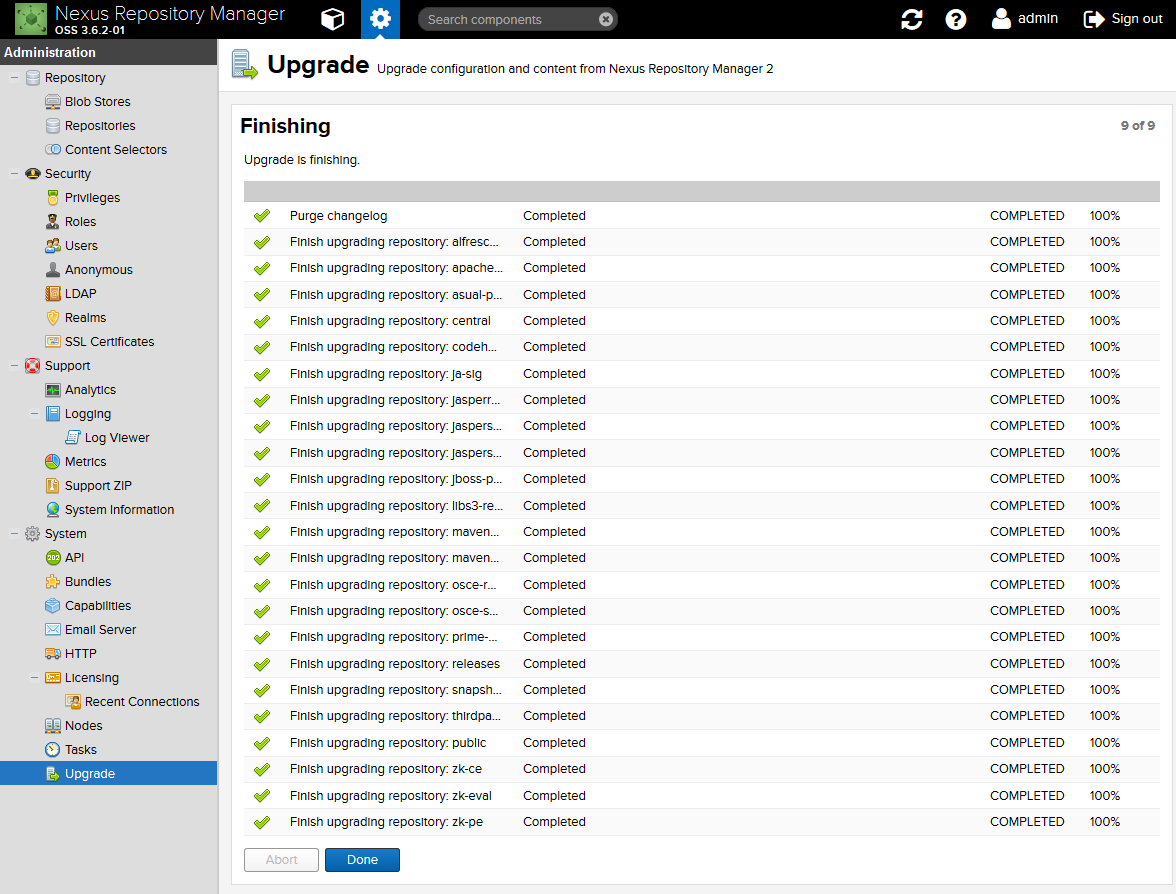
****

****

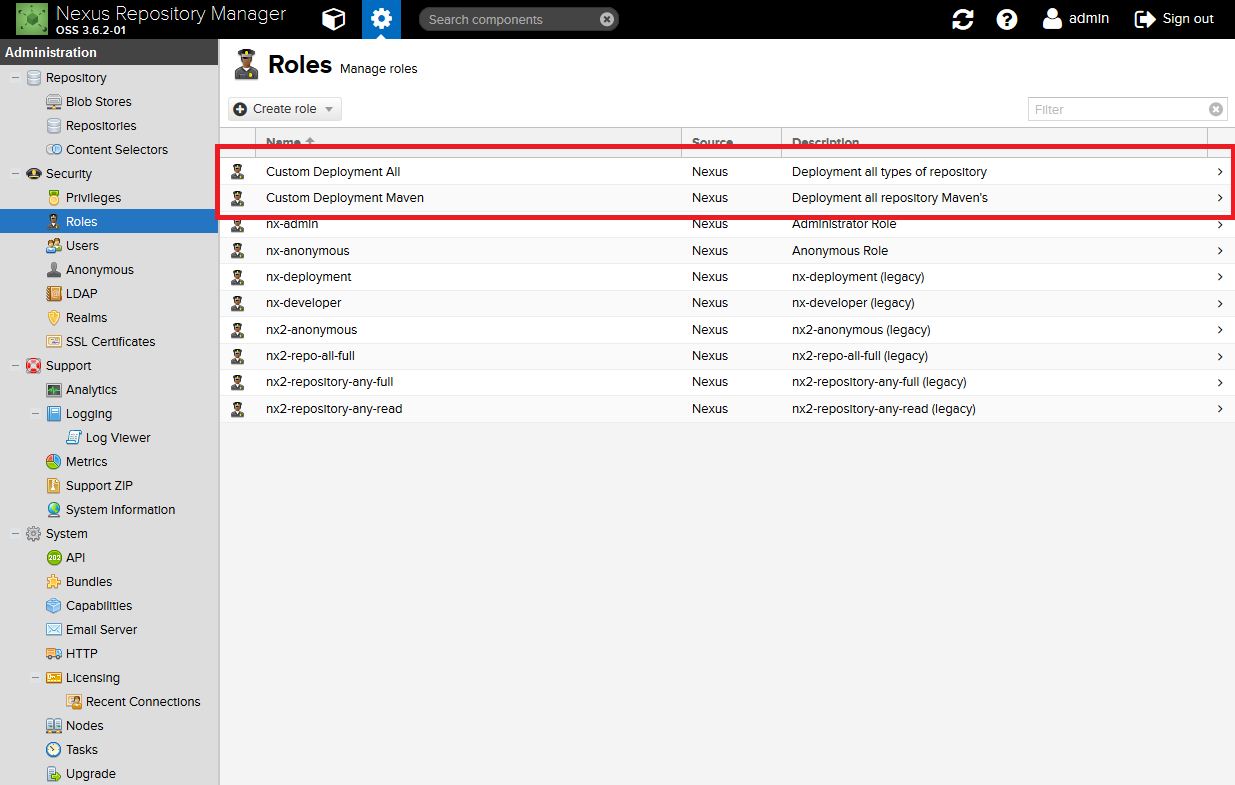
****

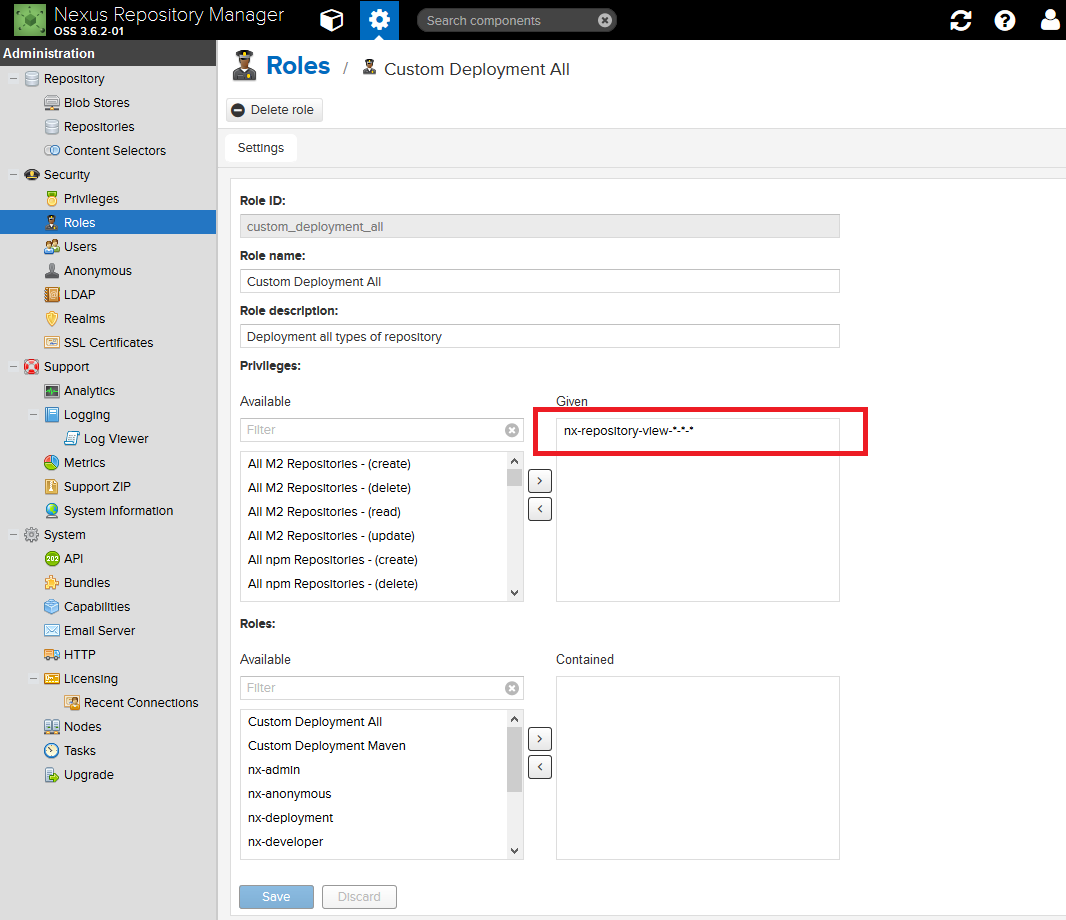
****

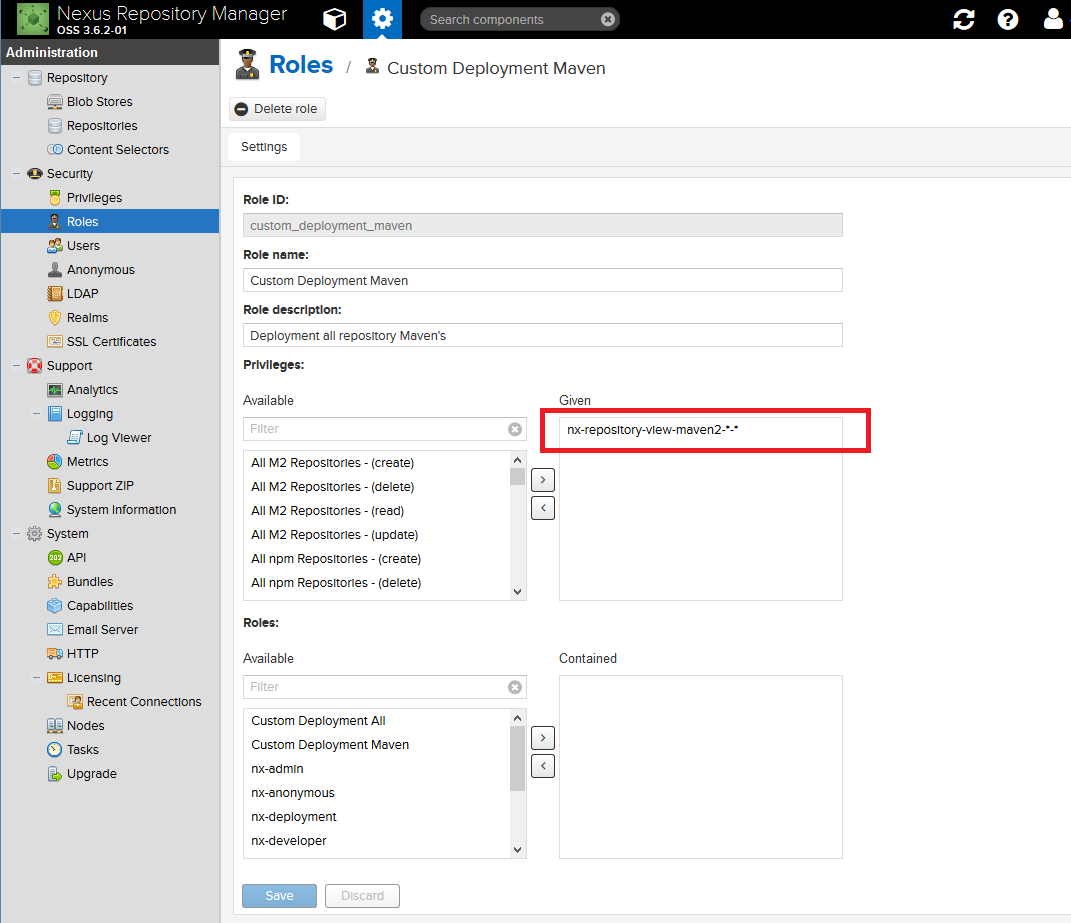
****

****

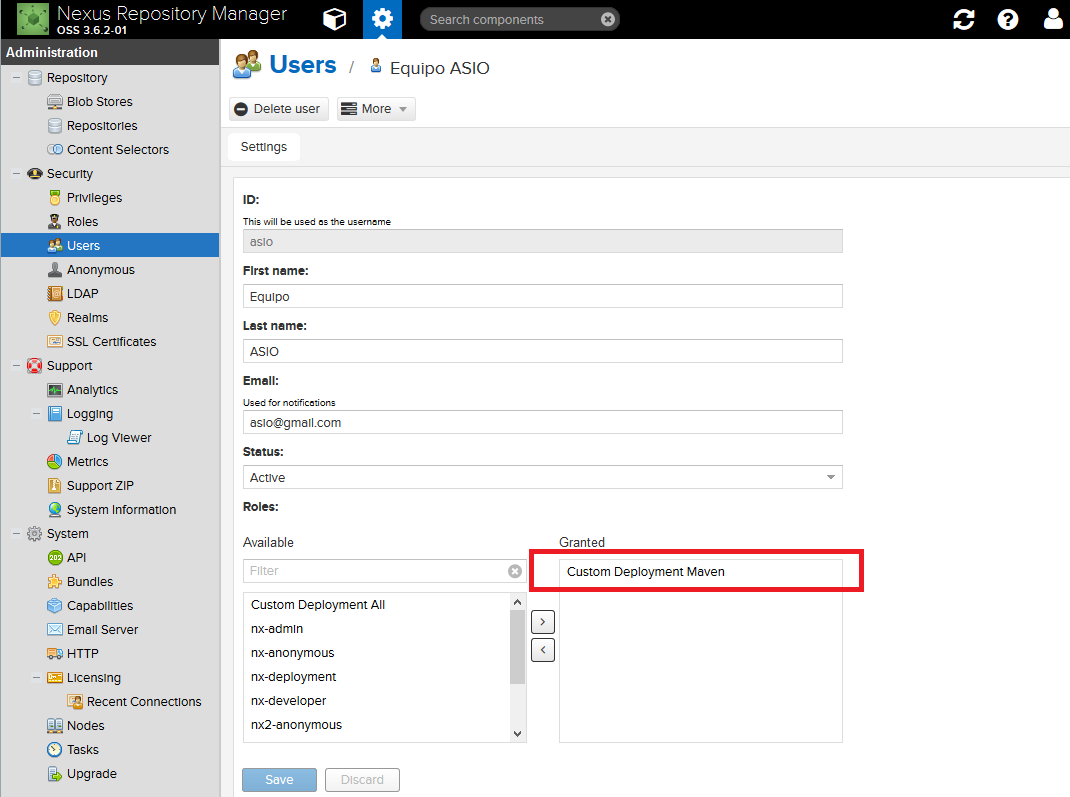
* La gestión de privilegios debe realizarse a través de roles y las siguientes figuras muestran un ejemplo:

****

****

****

* Asignar a usuario rol creado

****

1. **RELACIÓN DE LA PLATAFORMA SOFTWARE HABILITADA CON LA NTP ISO/IEC 12207**

El siguiente cuadro contiene la contiene información de cómo los entregables de la presente consultoría dan soporte a diferentes actividades de los procesos de la norma NTP ISO/IEC 12207-2016

| **NTP ISO/IEC 12207** | **ENTREGABLES DE CONSUTORIA** |
| --- | --- |
| **6 PROCESOS DE CICLO DE VIDA DEL SISTEMA**  **6.2 Procesos Organizacionales de Habilitación del Proyecto**  **6.2.2 Proceso de Gestión de la Infraestructura**  **6.2.2.3.2 Establecimiento de la infraestructura**   * La infraestructura se debe instalar a tiempo para la ejecución de los procesos relevantes. | **INFORME DE INSTALACIÓN DE LA PLATAFORMA SOFTWARE HABILITADA**  **Instrucciones de instalación y/o configuración de los productos software’s habilitados**   * La arquitectura tecnológica utilizada debe servir de guía para replicar instalaciones y documentación. |
| **6 PROCESOS DE CICLO DE VIDA DEL SISTEMA**  **6.2 Procesos Organizacionales de Habilitación del Proyecto**  **6.2.2 Proceso de Gestión de la Infraestructura**  **6.2.2.3.3 Mantenimiento de la infraestructura**   * La infraestructura se debe mantener, verificar y modificar según sea necesario para asegurar que se continúa cumpliendo con los requisitos de los procesos, que emplean este proceso. Como parte del mantenimiento de la infraestructura, se debe identificar el alcance hasta el cual la infraestructura está bajo la gestión de la configuración. | **INFORME DE INSTALACIÓN DE LA PLATAFORMA SOFTWARE HABILITADA**  **Instrucciones de operaciones de los productos software’s habilitados**   * La información contenida en la sección de operaciones puede usarse de guía para establecer el contenido mínimo a solicitar en instructivos de operaciones de servicios middleware. |
| **7 PROCESOS ESPECÍFICOS DEL SOFTWARE**  **7.1 Procesos de Implementación del Software**  **7.1.3 Proceso de Diseño Arquitectural del Software**  **7.1.3.3.1 Diseño arquitectural del software**   * El implementador debe definir y documentar los requisitos de prueba preliminar y el cronograma para la integración del software. | **MANUAL DE USO DE LA PLATAFORMA SOFTWARE HABILITADA**  **Lineamientos de uso de plataforma software habilitada**   * La información contenida en esta sección debe ser comunicada a los diferentes equipos de desarrollo internos y externos para que incorporen estos lineamientos en las actividades relacionados a desarrollo de software. |
| **7 PROCESOS ESPECÍFICOS DEL SOFTWARE**  **7.1 Procesos de Implementación del Software**  **7.1.5 Proceso de Construcción del Software**  **7.1.5.3.1 Construcción del software**   * El implementador debe evaluar el código del software y los resultados de prueba considerando los criterios que se enumeran a continuación. Los resultados de las evaluaciones deben estar documentados.   a) Trazabilidad hasta los requisitos y el diseño del elemento de software.  b) Consistencia externa con los requisitos y el diseño del elemento de software.  c) Consistencia interna entre los requisitos de las unidades.  d) Cobertura de la prueba de las unidades.  e) Pertinencia de los métodos de codificación y estándares utilizados.  f) Factibilidad de la integración del software y la prueba.  g) Factibilidad de la operación y el mantenimiento. | **MANUAL DE USO DE LA PLATAFORMA SOFTWARE HABILITADA**  **Entorno de desarrollador con plataforma software habilitada**   * Proporciona una guía para integrar los entornos de desarrollo con las plataformas de servicios para análisis estático de código fuente y los sistemas de control de versión de código fuente. |
| **7 PROCESOS ESPECÍFICOS DEL SOFTWARE**  **7.1 Procesos de Implementación del Software**  **7.1.7 Proceso de Pruebas de Calificación del Software**  **7.1.7.3.1 Prueba de calificación del software**   * El implementador debe realizar la prueba de calificación según los requisitos de calificación para el elemento de software. Se debe asegurar que se somete a prueba de conformidad la implementación de cada requisito. Los resultados de la prueba de calificación deben estar documentados * El implementador debe evaluar el diseño, el código, las pruebas, los resultados de prueba y la documentación del usuario considerando los criterios que se enumeran a continuación. Los resultados de las evaluaciones deben estar documentados.   a) Cobertura de prueba de los requisitos del elemento de software.  b) Conformidad con los resultados esperados.  c) Factibilidad de la integración del sistema y de la prueba, si se realiza.  d) Factibilidad de la operación y el mantenimiento | **MANUAL DE USO DE LA PLATAFORMA SOFTWARE HABILITADA**  **Lineamientos de uso de plataforma software habilitada**   * Proporciona una propuesta de cómo se debe realizar pruebas de análisis estático de código fuente con SonarQube.   **Entorno de desarrollador con plataforma software habilitada**   * Proporciona guía para integrar los entornos de desarrollo con las plataformas de servicios para análisis estático de código fuente y los sistemas de control de versión de código fuente. |