Servicios Web SOAP:

Para la realización de esta práctica necesitaremos las siguientes herramientas:

- JDK 7 (o superior).
- Eclipse for Java EE Developers.
- Apache Tomcat 8.5.29.
- Apache CXF 2.7.18.
- SOAP UI.

Instalamos dichas herramientas en nuestro sistema, en mi caso con sistema operativo Kali Linux (que está basado en Debian).

Mediante la instalación no encontramos problema alguno.

Mediante la creación del servidor Apache Tomcat me di cuenta de que mi versión del JDK era la 9.0.4 y empezó a darme problemas en el servidor. Por lo tanto, me descargué el JDK con versión 8 update 161 y, en las opciones de eclipse, cambiamos a que use este nuevo JDK, para que no haya problema.

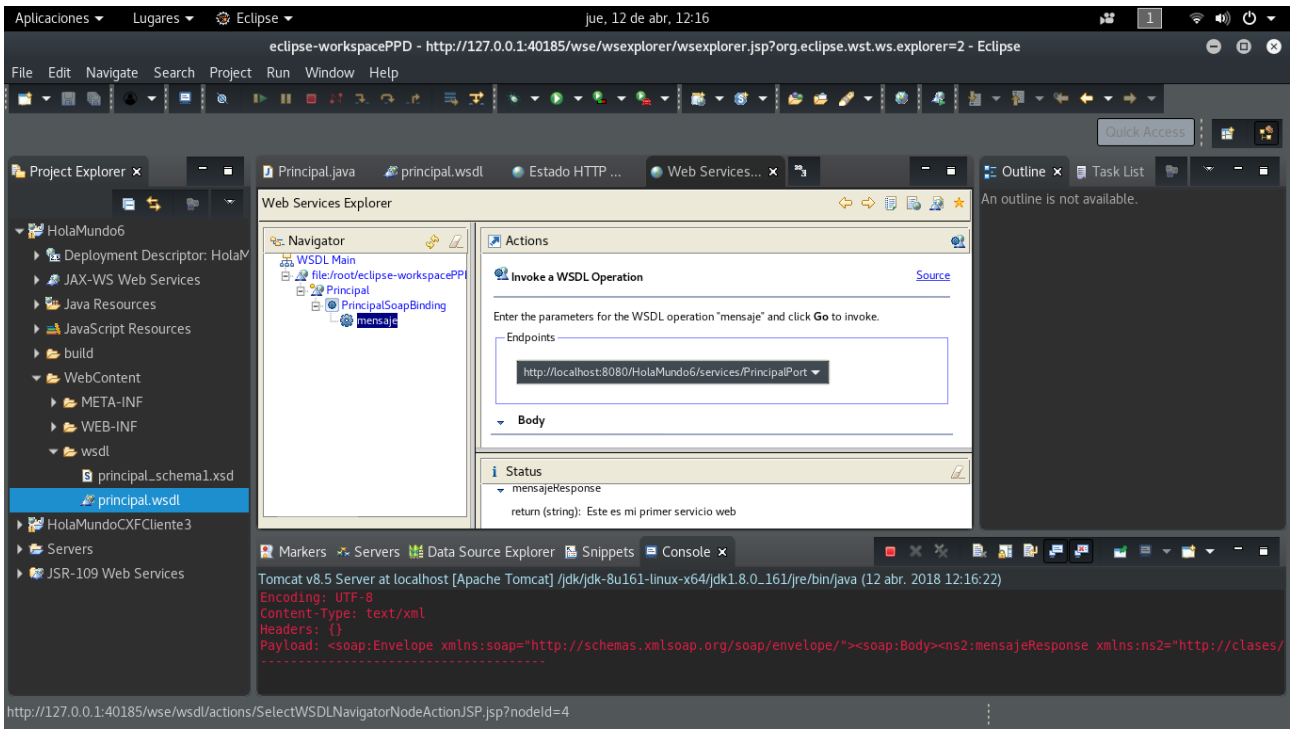
Luego configuramos Eclipse para CXF, que es instantáneo.

A continuación, creamos el servicio web SOAP, tal y como se indica en las transparencias de la práctica e introducimos el código que viene en la diapositiva 16.

Después de configurar el servicio web en “Web Service runtime Apache CXF 2.x” como se dice en las diapositivas, se me generaban los ficheros, pero en la clase “Principal.java” me decía que el “import javax.jws.WebService;” no me dejaba llegar a la ruta establecida del JDK. Para solucionar este problema tuve que seguir las instrucciones de esta página (https://code.i-harness.com/es/q/d201b), en la cual se indica, que hay que cambiar lo que sucede cuando nos encontramos con un paquete “deprecated”, que se encuentra en error, y lo cambiamos a ignorar. Después de esto, ya tenemos solucionado ese problema.

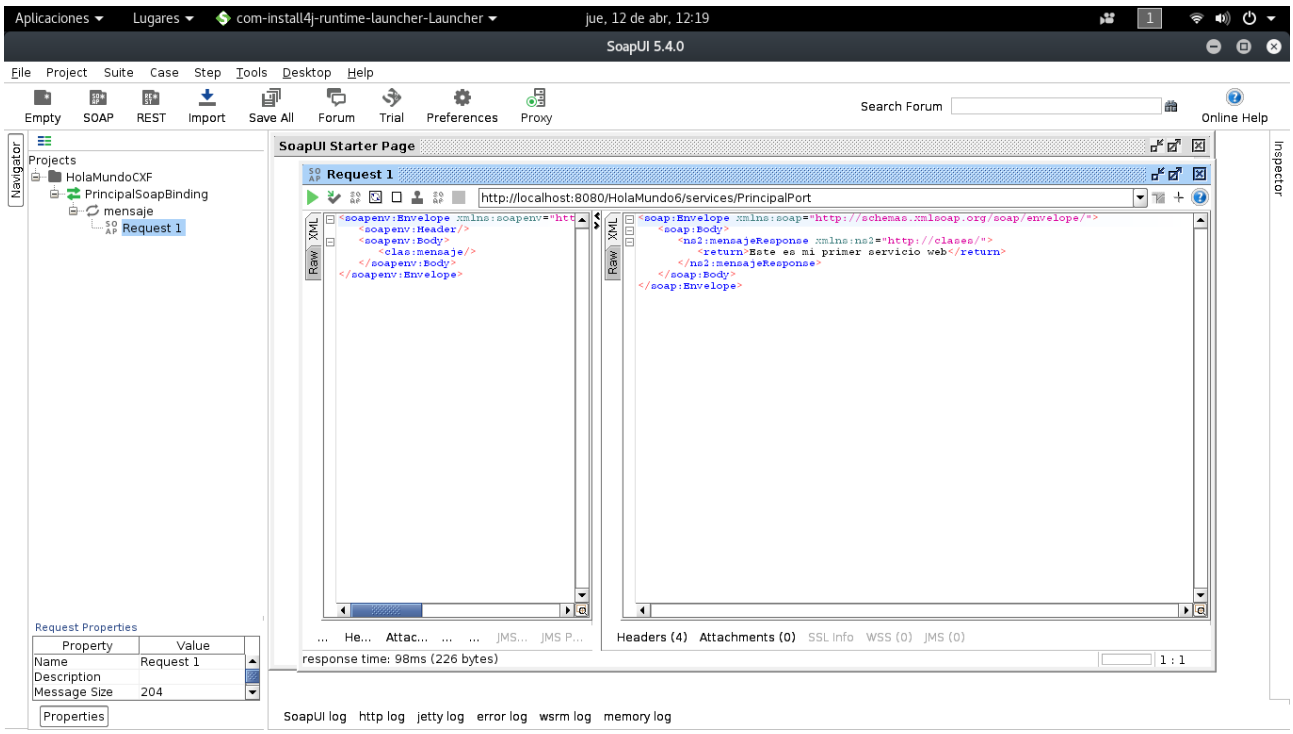
Luego, entramos en el archivo “principal.wsdl” en la vista de “source” y nos dirigimos al final, donde podemos encontrar la dirección del wsld: http://localhost:8080/HolaMundo6/services/PrincipalPort?wsdl.

Una vez probada dicha dirección en el navegador, damos en botón derecho sobre “principal.wsdl” y seleccionamos “Web Services” y luego “Test with web service explorer”, clicamos en “mensaje” y luego en “Go”. Entonces, más abajo, nos tiene que salir el mensaje de “return (String): Este es nmi primer servicio web”.



En caso de que no nos salga nada en “operations” puede ser debido al nombre de la clase o al del paquete, por lo tanto, si rehacemos todo desde cero con otro nombre, debería funcionar, ya que me pasó en uno de los intentos que tuve que hacer para que llegara a funcionar.

Una vez que funciona el “Test with web service explorer”, invocamos el servicio con SOAP UI tal como se muestra en las prácticas ya que en esta etapa, no da ningún problema.

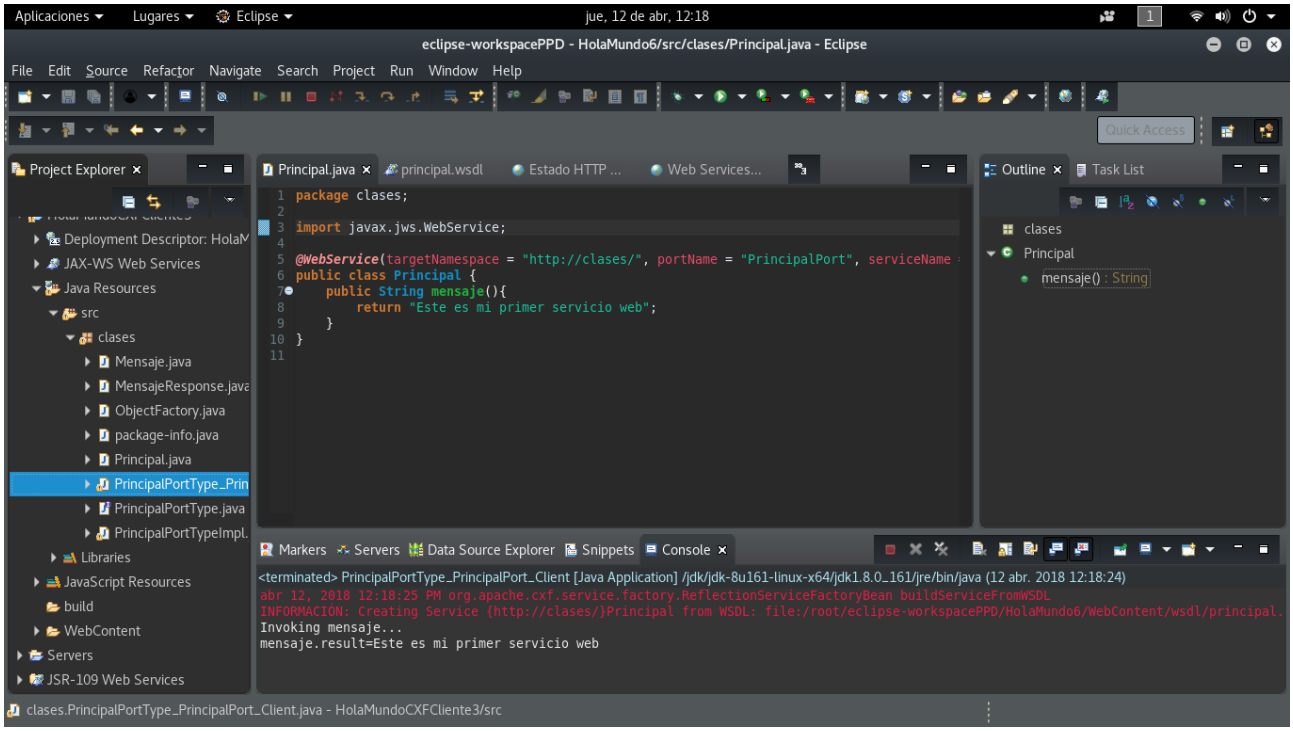


Ahora toca crear el cliente de servicios web SOAP cambiando, como en la diapositiva 36, el Axis, por “Web service runtime Apache CXF 2.x”. Si hemos llegado hasta aquí, no deberíamos tener

ningún problema adicional, salvo que, al intentarlo, no me di cuenta de lo que se dice en la práctica 6 y tuve que buscar el error que daba por el archivo “javaxp.properties” el cual encontré en la siguiente en la siguiente página (https://stackoverflow.com/questions/23011547/webservice-client-generation-error-with-jdk8?utm_medium=organic&utm_source=google_rich_qa&utm_campaign=google_rich_qa).

Luego, nos generará algunos archivos en “Java Resources/src”.

Para probar el cliente, damos botón derecho sobre el archivo “PrincipalPortType_PrincipalPort_Client.java” que se nos ha generado, y seleccionamos “Run as..” y luego “Java Application”. Entonces, en la consola de Eclipse, debe salir el mensaje de “Invoking mensaje… mensaje.result=Este es mi primer servicio web”. Por lo que podemos comprobar que todo está correcto.

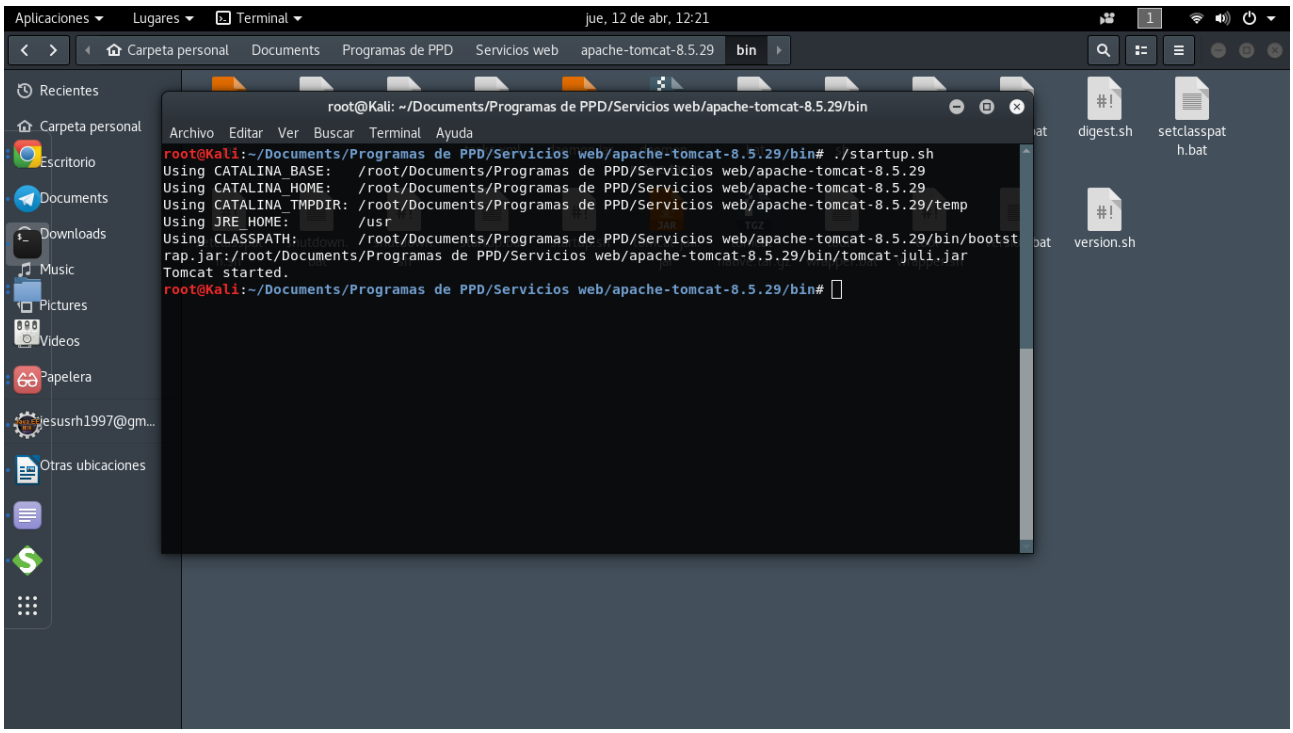


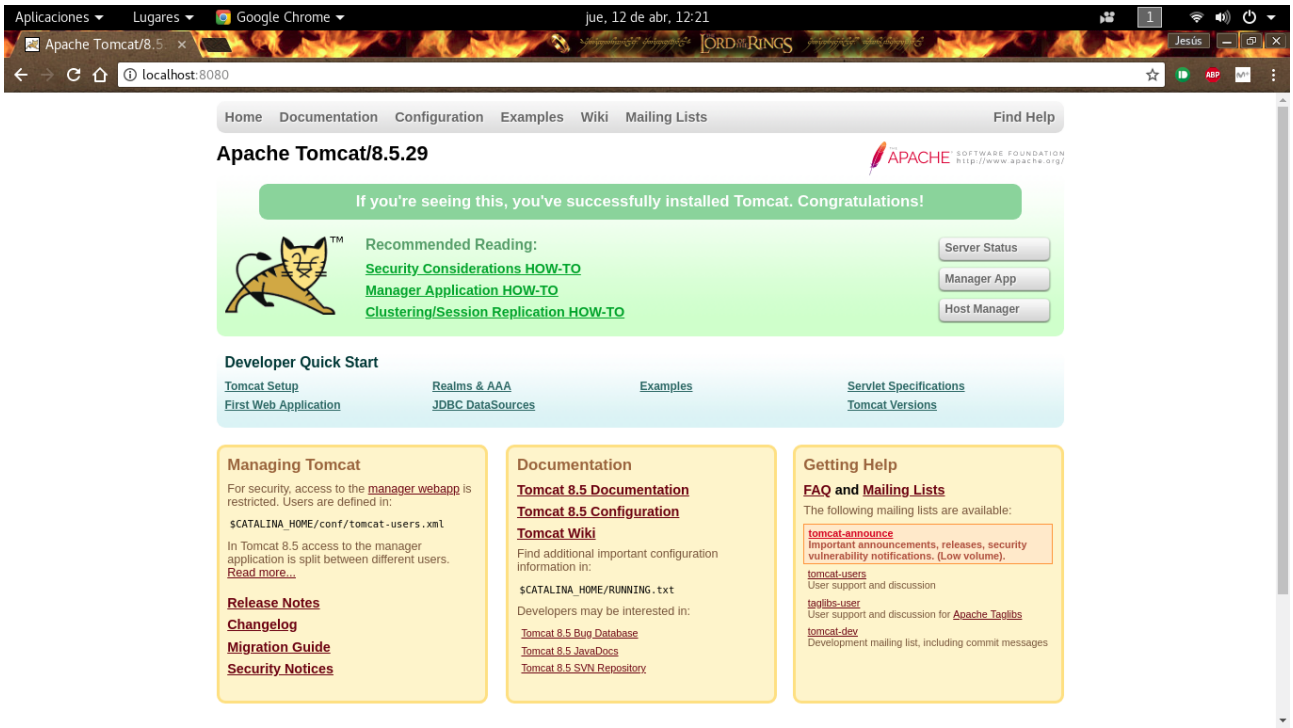
Por último, exportamos el servidor como WAR en “export>war file” y lo exportamos tal y como se dice en las diapositivas de la práctica.

Configuramos los permisos del TomCat en el fichero “conf/tomcat-users.xml” dejándolo así:

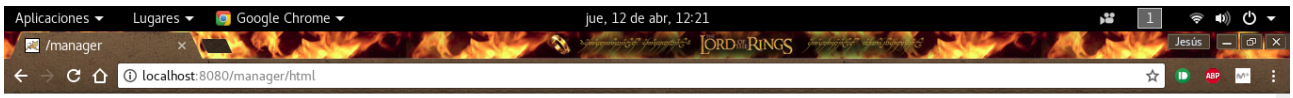
```
<tomcat-users>
  <role rolename="manager-gui"/>
  <role rolename="admin-gui"/>
  <user username="tomcat" password="s3cret" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

Luego, entramos en la carpeta bin del apache tomcat, y, al estar en linux, me vi obligado a cambiar los permisos del archivo “startup.sh” porque no me dejaba ejecutarlo. Al ejecutar el “startup.sh” me pide cambiar los permisos del archivo “catalina.sh”. Una vez hecho esto, volvemos a ejecutar el “startup.sh” (con el servidor tomcat apagado en Eclipse), buscamos “http://localhost:8080/” y nos sale la interfaz del tomcat. Cuando accedemos al “manager webapp” nos pide el usuario y contraseñas que hemos establecido en el archivo “tomcat-users.xml”.





Una vez estamos logeados en el “manager webapp” podemos gestionar todas las aplicaciones web que estén ejecutándose en nuestro servidor Tomcat.



Gestor de Aplicaciones Web de Tomcat

Mensaje:	OK

Gestor			
Listar Aplicaciones	Ayuda HTML de Gestor	Ayuda de Gestor	Estado de Servidor

Aplicaciones

Trayectoria	Versión	Nombre a Mostrar	Ejecutándose	Sesiones	Comandos
/	*Ninguno especificado*	Welcome to Tomcat	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/HolaMundo6	*Ninguno especificado*	HolaMundo6	false	0	Arrancar Parar Recargar Replegar
/docs	*Ninguno especificado*	Tomcat Documentation	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/examples	*Ninguno especificado*	Servlet and JSP Examples	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/host-manager	*Ninguno especificado*	Tomcat Host Manager Application	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/manager	*Ninguno especificado*	Tomcat Manager Application	true	1	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos

Desplegar

Desplegar directorio o archivo WAR localizado en servidor

Trayectoria de Contexto (opcional):