Memoria
Trabajo
Servicios Web
SOAP

Aplicaciones Distribuidas

Victor Paz Rodríguez DNI 42187180M

vpaz14@alumno.uned.es

Tlfo: 616048996

Arquitectura SOAP Desarrollada:

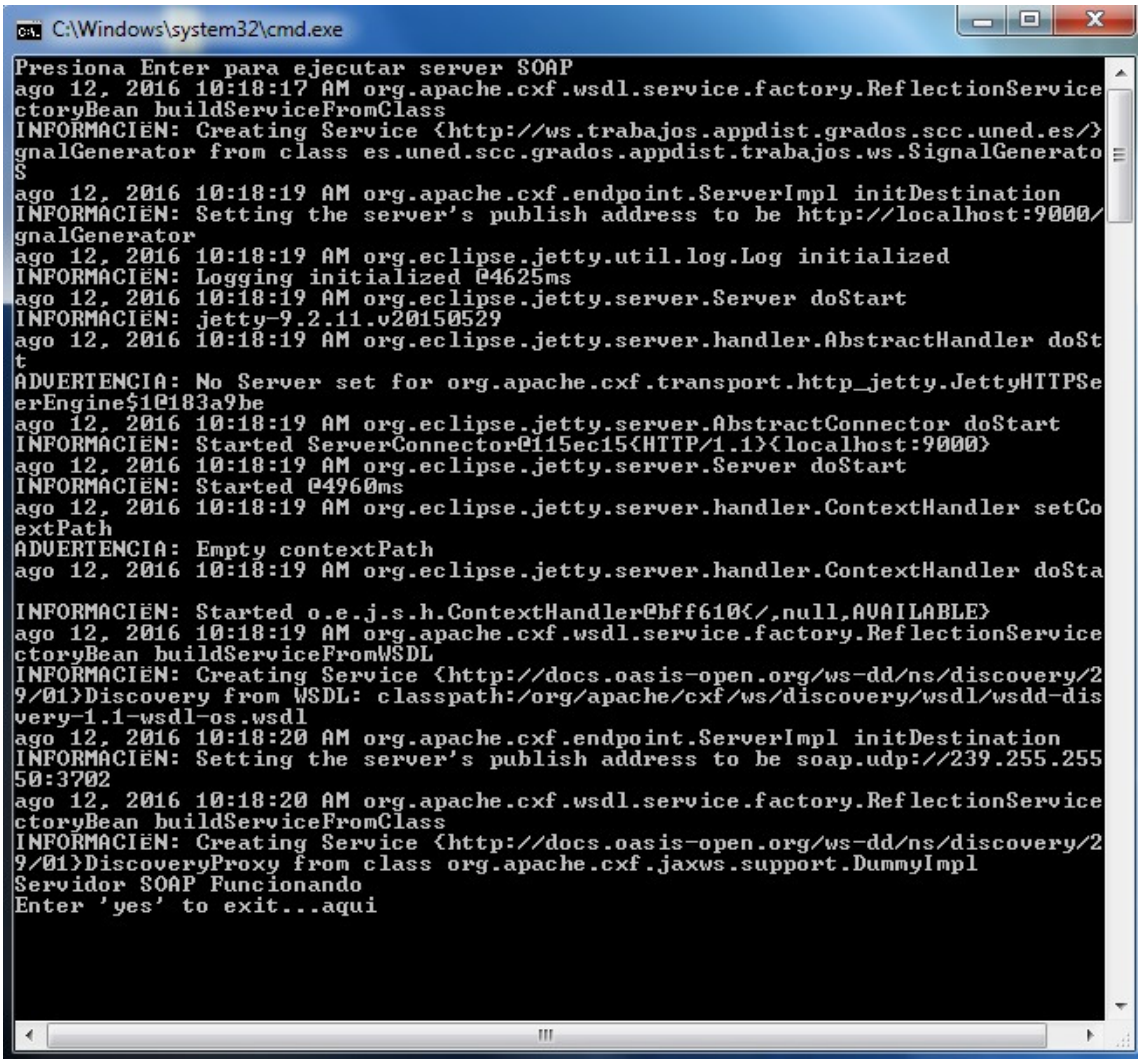
La arquitectura de este trabajo ha sido desarrollada según se indica en el enunciado propuesto por el equipo docente. En primer lugar configuré el eclipse según el vídeo del equipo docente para poder desarrollar esta parte.

Luego he creado la interfaz del servicio SOAP copiando el código que se nos proporciona en el enunciado del trabajo, esta interfaz es muy parecida a la del trabajo RMI, pero cabe destacar la aparición de la anotación @WebService para indicar que es una interfaz de servicio web y la anotación @WebParam en el método setSignalParameters para nombrar el argumento de la operación de este método.

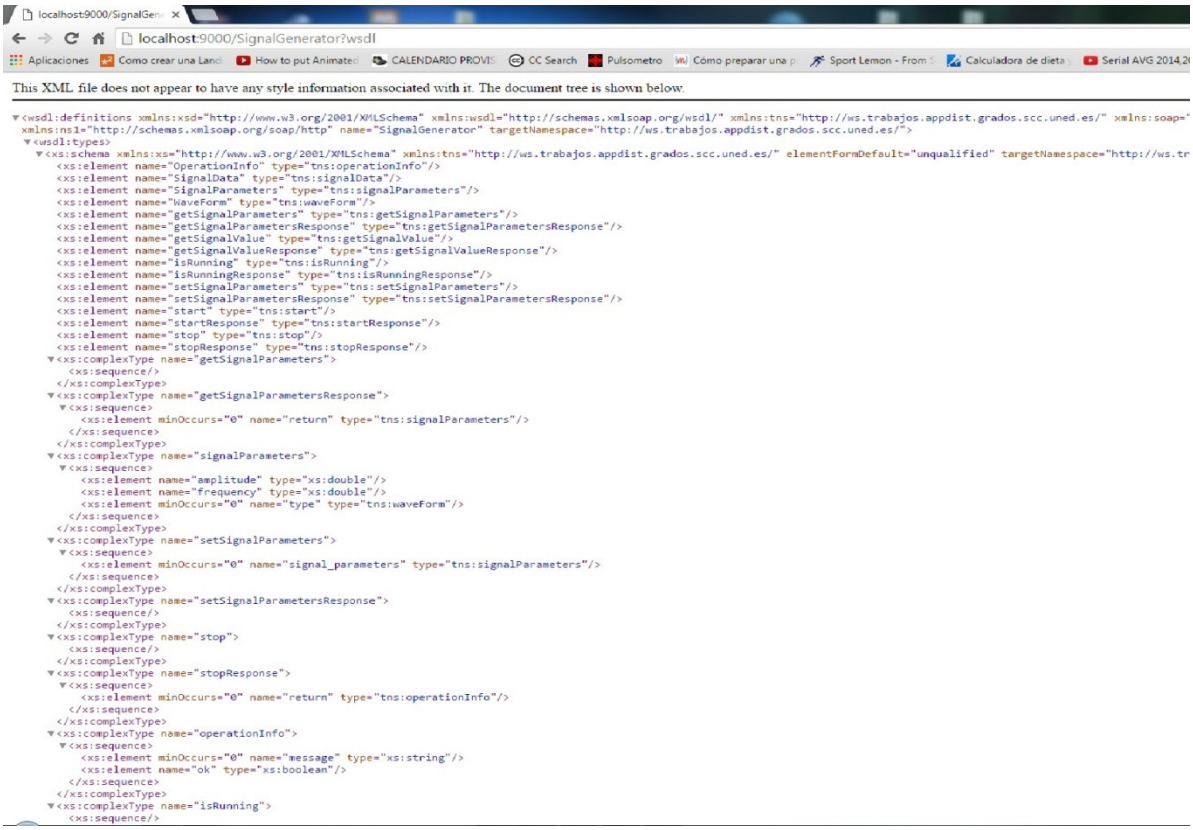
Posteriormente he creado la clase SignalGeneratorWSImpl que implementa los métodos de la interface del servicio como indica el enunciado. La implementación de los métodos me resultó sencilla al ser prácticamente igual que la realizada en el trabajo de RMI. La Clase SignalGeneratorWSImpl implementa los siguientes métodos. **Start** que es el encargado de poner a funcionar el generador de señales, **Stop** encargado de parar el generador de señales, **getSignalValue** que lee los valores de tiempo y los valores de salida del generador, **getSignalParameters** que lee los valores los parámetros (amplitud, frecuencia y tipo de señal) que le pasamos al generador y el método **setSignalParameters** que se encarga de fijar el valor de los distintos parámetros que le pasamos al generador para su funcionamiento. También está implementado el método **isrunning** que comprueba si el generador está en funcionamiento o no.

Una vez implementada la interfaz remota procedí a crear la clase WSServer. Para la implementación de esta clase usé la ayuda proporcionada en el enunciado por el equipo docente. La clase WSServer consta de dos métodos; un método llamado **waitForKey** que es el encargado de mantener el servidor SOAP funcionando hasta que el usuario decida pararlo**,** básicamente es el mismo que proporciona el equipo docente en el enunciado del trabajo RMI salvo porque cuando se introduce la palabra yes para salir del servidor además de salir del bucle de espera, también realiza la acción parar el servidor. Finalmente hay un método principal que contiene las sentencias que vienen en el enunciado para usar las funcionalidades CFX e invoca al método citado anteriormente. La creación del servidor no me dio problemas, pero al crear el archivo bat del servidor si me creó problemas al referenciar el classpath de las librerías CFX, ya que al poner el path del enunciado ./lib/apache-cxf-3.1.3/lib no se me mostraba la información del servidor, que si me mostraba el eclipse, para solucionarlo y que todo me funcione correctamente he puesto el path **./lib/apache-cxf-3.1.3/*.**

A continuación se muestra una captura de pantalla del servidor SOAP funcionando.



En la siguiente imagen se puede ver una captura del documento WDSL de definición del servicio que se puede visualizar una vez el servidor está en funcionamiento en el enlace: http://localhost:9000/SignalGenerator?wsdl.



Posteriormente procedí a configurar el eclipse para crear el cliente SOAP como se indica en el vídeo que suministra el equipo docente, lo que genera nuevas clases. Luego implementé la clase WSClient, esta clase sólo consta de un método principal, este método simplemente crea una instancia de la clase SignalGeneratorLocator y con esta instancia obtenemos la referencia del servicio que se le pasa como parámetro a la instancia que creamos de la clase auxiliar PlottingFrame que describiremos a continuación.

A continuación se muestra una captura de pantalla del cliente SOAP funcionando.

Finalmente implementé la clase PlottingFrame, esta clase implementa la interface ClientPlot suministrada por el equipo docente. En el constructor de la clase le asignamos al objeto soapClient la referencia del servicio que hemos pasado por parámetro al instanciar la clase.

A continuación implemente las funciones de la interface. El método start que hace que cuando se pulsa el botón start de la interface gráfica se ponga a funcionar el generador, el método stop que es análogo al start pero para parar el generador.

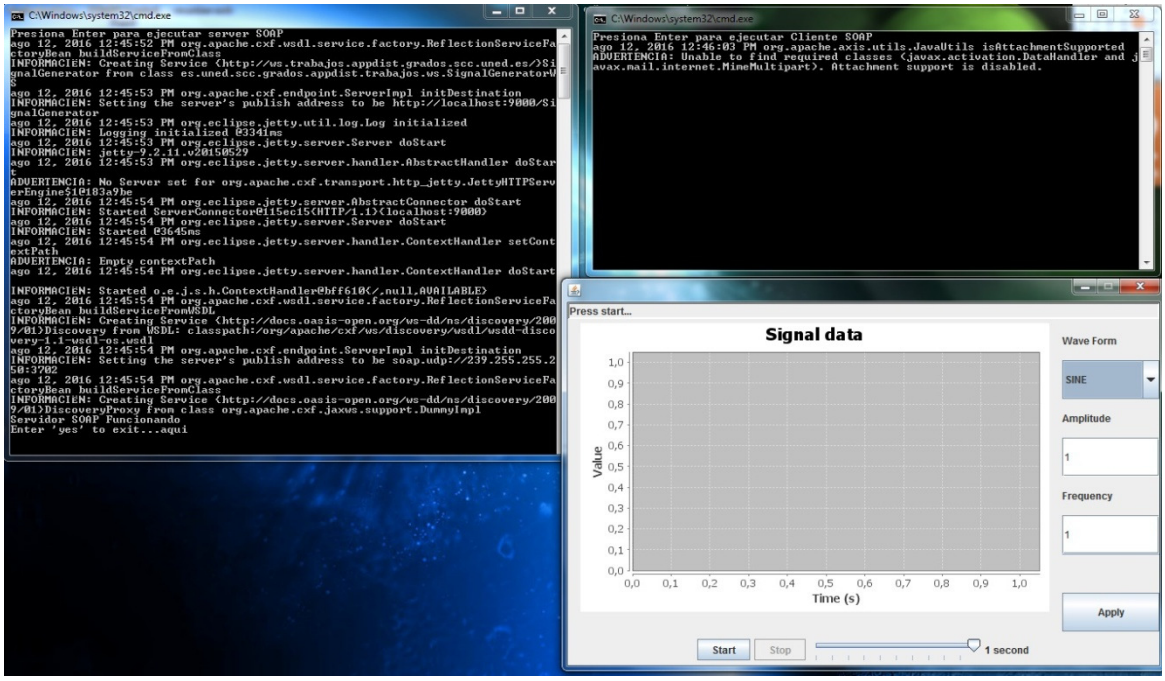
A la hora de implementar los siguientes métodos fue donde aparecieron los mayores problemas de la práctica, ya que el formato de los datos del cliente SOAP diferían del formato de que utiliza nuestra interface gráfica, lo que provocaba un error fatal. Por lo que tuve que crear, como aconsejó el equipo docente en el foro, una clase auxiliar ConvertData que consta de tres métodos para pasar los datos del formato de nuestro cliente a nuestra interface gráfica y viceversa.

El método getSignalValue que es el encargado de que se muestren los distintos valores de salida y de tiempo en la pantalla, este método usa una llamada al método tosenalDataOriginal de la clase ConvertData para pasar los datos del nuestro cliente al formato de la interface gráfica, el método getSignalParameters que nos muestra los parámetros con los que está funcionando el generador (inicialmente la frecuencia y la amplitud se inicializan a 1 y el tipo de señal a sine), este método usa una llamada al método tosenalParametersOriginal de la clase ConvertData para pasar los datos del nuestro cliente al formato de la interface gráfica, y el método setSignalParameters que es el encargado de fijar los valores de los parámetros, al principio con los valores iniciales y luego cuando pulsamos el botón apply para aplicar los cambios realizados a los parámetros. Este último método usa una llamada al método tosenalParametersGenerados de la clase ConvertData para pasar los datos del nuestra interface gráfica al formato de nuestro cliente.

Todos estos métodos hacen uso del servicio SOAP y de la implementación de esta que hemos realizado en la clase SignalGeneratorWSImpl.

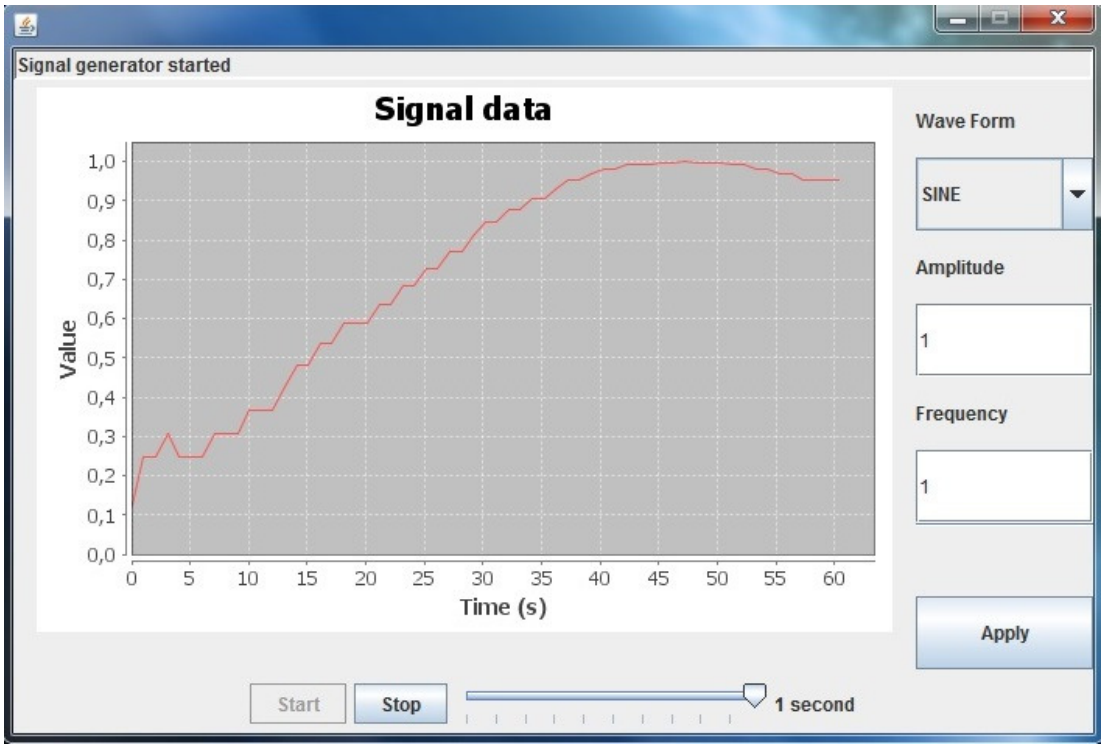
Pruebas SOAP:

En la imagen que se muestra a continuación se puede observar la interface gráfica que aparece después de ejecutar el servidor y el cliente SOAP, tal como se pide en el enunciado.

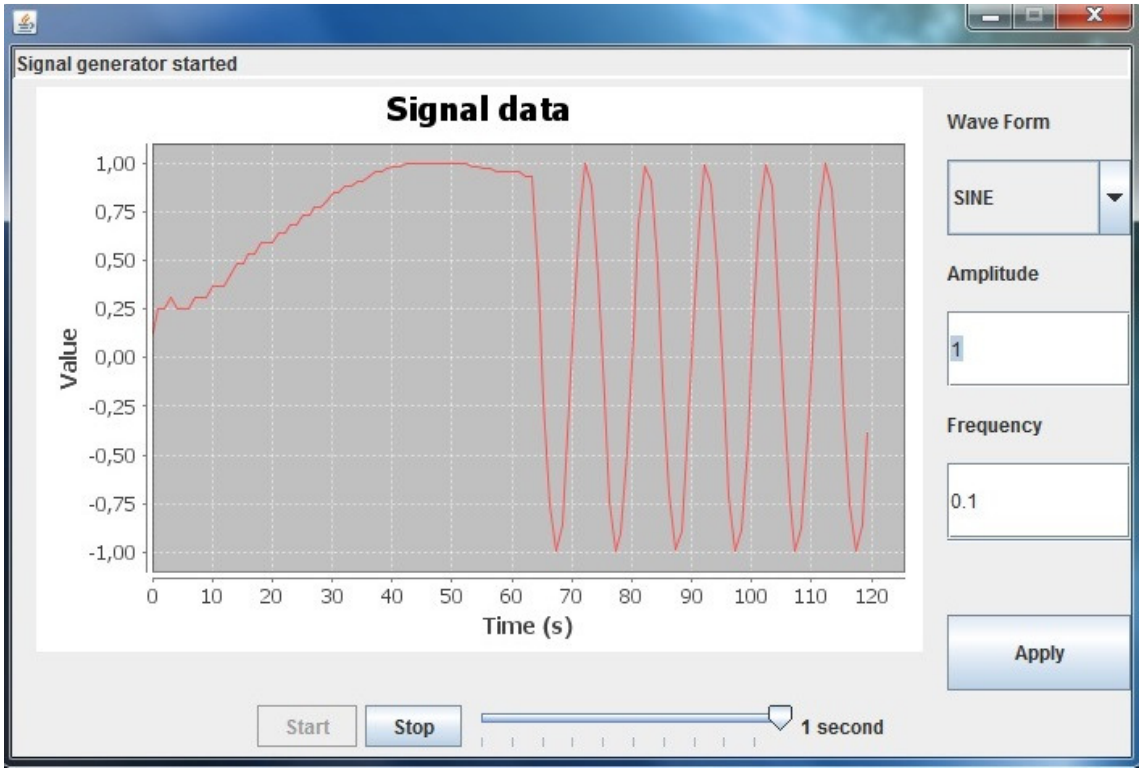


En la siguiente imagen podemos observar la GUI después de un minuto. Se puede observar que la imagen con los parámetros iniciales (Amplitud=1, Frecuencia=1, Señal=sine) dibuja la curva ascendente de un seno desde 0 hasta 1. También se puede observar como entorno al segundo 3 se produce una anomalía que puede ser debida a algún posible fallo en la comunicación o como me parece más probable a algún problema en el proceso serialización o recomposición de los parámetros.

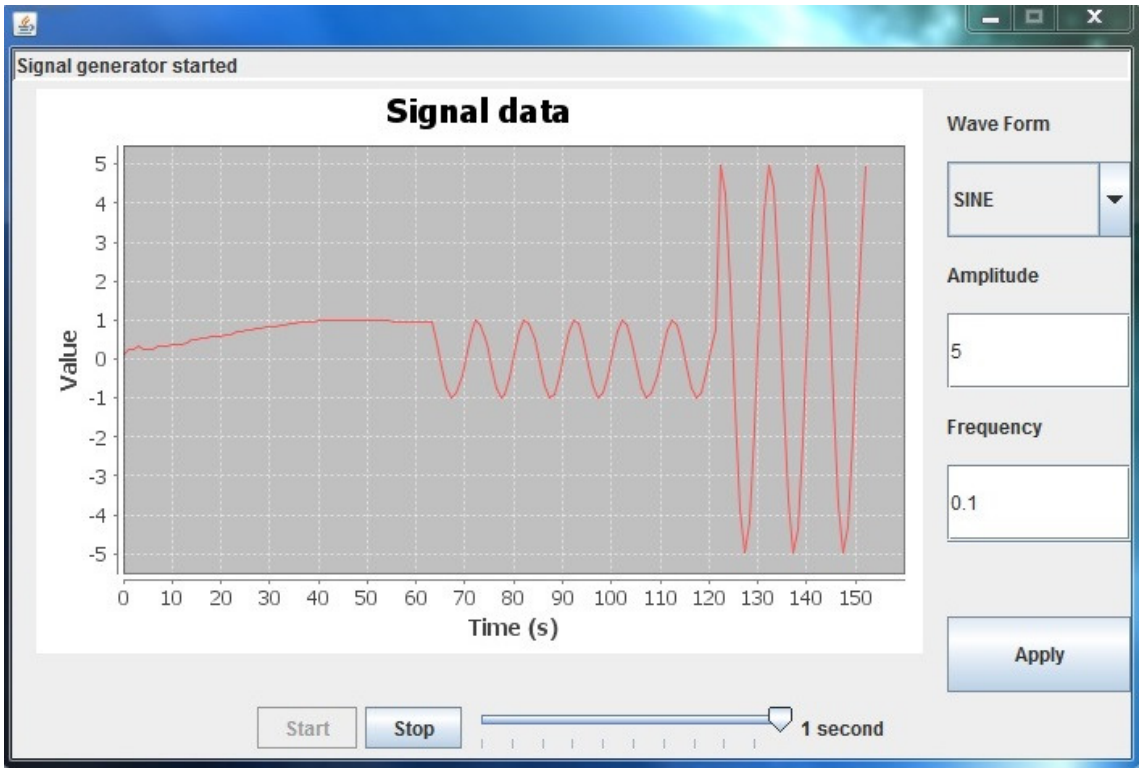
Aunque el comportamiento habitual de la señal es el que se muestra en las siguientes imágenes, hay ocasiones en que la señal presenta un comportamiento totalmente aleatorio desde que comienza su ejecución. No sé a que se debe este comportamiento extraño de la señal en algunas de las ejecuciones del servicio.



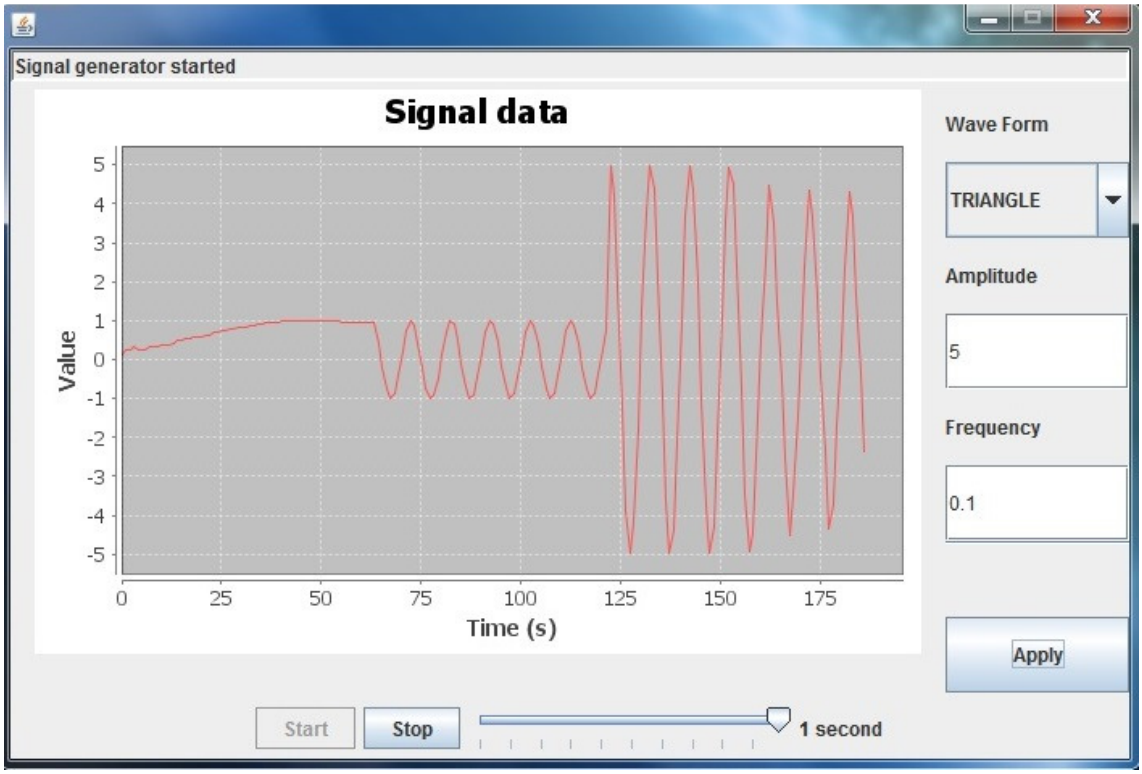
En la imagen que se muestra a continuación podemos ver la imagen después de dos minutos con una frecuencia de 0.1 después del primer minuto. En esta imagen se observa ya claramente la señal sinusoidal al ser la frecuencia mucho más pequeña.



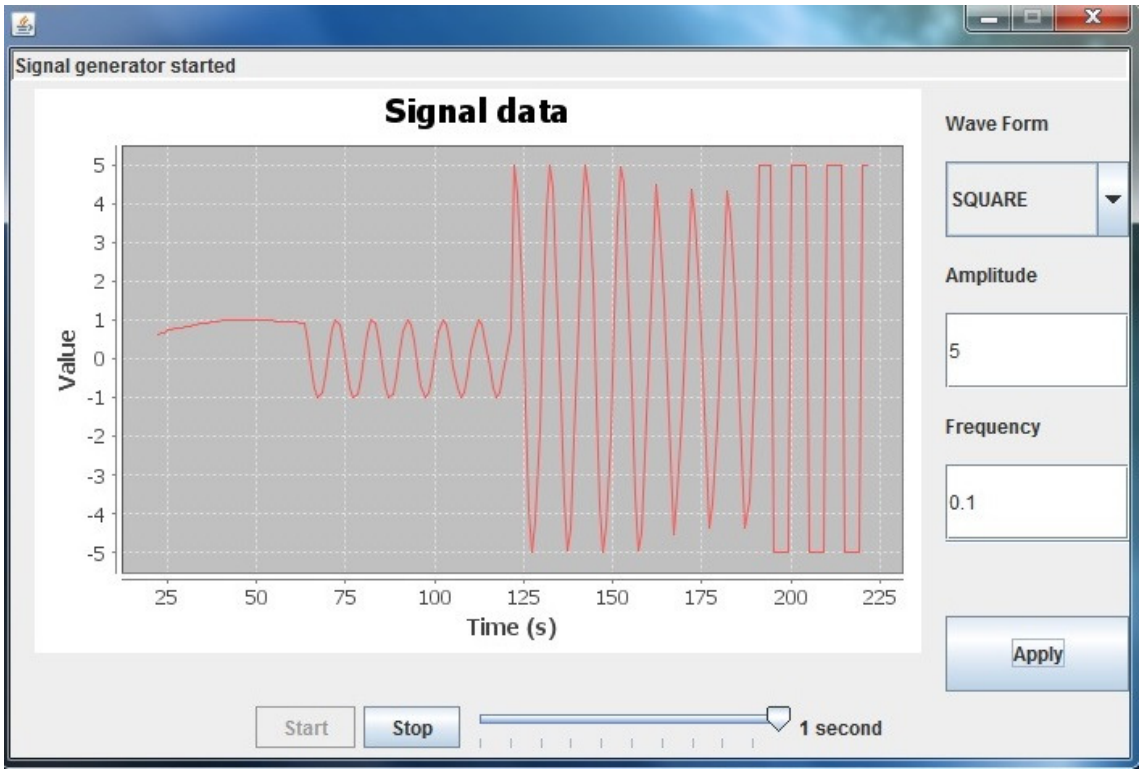
En la siguente imagen podemos observar la señal despues de dos minutos y medio en la que vemos claramente como despues de dos minutos hemos aumentado la amplitud de la señal sinusoidal a 5.



En la siguente imagen podemos observar la señal despues de tres minutos en la que vemos claramente como después de dos minutos minutos y medio la forma de la señal cambia y se vuelve triangular.



A continuación vemos la imagen después de tres minutos y medio, donde se puede observar claramente que la señal ha pasado a ser cuadrada.



En esta última imagen podemos ver el efecto del slider cuando vamos cambiando los valores de este. Vemos que al principio de la gráfica cuando el slider tiene un valor de 1s, da la sensación de que los puntos están más espaciados y la velocidad a la que se cogen los puntos del generador y se muestran en pantalla es mucho más lenta que en la siguiente parte de la gráfica cuando ponemos el slider a 0.1, aquí los puntos están mucho más próximos y la velocidad a la que se muestra la gráfica en pantalla es mucho mayor. Por último cuando ponemos el slider a 0.5 nos encontramos en un punto intermedio de lo comentado anteriormente.