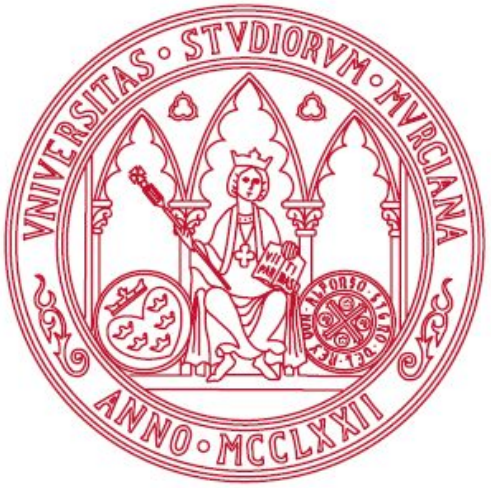

Universidad de Murcia

Grado en Ingeniería Informática

Curso 2020/2021

Programación para las comunicaciones

Profesor: Humberto Martínez Barberá

Práctica V

Jaime Ortiz Aragón - 49196689B - jaime.ortiza@um.es

- **ESCENARIO**

Para el desarrollo de esta práctica, partimos del código que hemos ido desarrollando para las prácticas anteriores. El objetivo es ser capaces de acceder a los datos de los servicios ofrecidos por los sensores, pero a través de un servicio POP/SMTP.

Para realizar esto adaptamos el código del cliente de las estaciones, para que ahora se utilice como servicio de información del sistema. Seguiremos contando con la funcionalidad de las peticiones HTTP/HTTPS, pero añadiendo la funcionalidad de la solicitud de datos por medio del correo electrónico.

La cuenta de correo proporcionada para la petición de datos es de Google y es *jaimeortizappc@gmail.com*, con la contraseña *correoppc*.

- **DESPLIEGUE HTTPS/HTTPS**

La funcionalidad hasta ahora de los servicios HTTP y HTTPS había sido la de hacer la función de carrito de compra, con la mejora que incorporamos de llevar la cuenta de cuántas veces se ha accedido a una cookie. Ahora, la respuesta será un código HTML que contendrá los últimos datos de los sensores. Estos son accedidos mediante un *get*, ya que se ha incorporado al código una nueva propiedad que contiene estos datos actualizados cada vez que se recibe un paquete del servidor.

Esta funcionalidad es atendida por 2 hilos. Uno relativo a la conexión sin TLS y otro a la que sí utiliza TLS, que son lanzados desde la sección *main* del servicio de información.

- **DESPLIEGUE SMTP**

La funcionalidad relativa al correo se ha desplegado en el fichero *Smtp.java*. Este hilo atenderá todas las peticiones realizadas por el cliente de correo. El funcionamiento es el siguiente :

Desde una cuenta de correo, se hace uso del servicio SMTP para enviar un mensaje a la cuenta proporcionada, que actúa de servidor. Lo importante que debe contener este mensaje es el asunto **Request**.

Para la recepción de mensajes, se ha hecho uso del protocolo POP. De este modo, se irá obteniendo una lista de todos los mensajes y se comprobará su asunto. Si este resulta ser *Request*, se enviará a la dirección de correo cliente un mensaje con el asunto *Data.* Es importante destacar que se marcará como borrado con el fin de que sólo se procese una vez cada mensaje. El mensaje de respuesta contendrá en su cuerpo la información de los servidores en texto plano y la misma información en formato XML o JSON, dependiendo de en qué formato se indicó en el lanzamiento de la API.

- **FUNCIONAMIENTO GENERAL**

El lanzamiento de la aplicación comienza introduciendo el formato en el que se van a codificar los datos de los sensores. Dado que en esta práctica no se tienen

en cuenta los mensajes *unicast*, los únicos mensajes que debemos serializar y deserializar son los mensajes de *broadcast* con la información que se genere en los sensores.

Como se ha comentado anteriormente, se cuenta con 3 hilos para atender a los 3 tipos de clientes que soporta la aplicación. Para esto, se ha tenido que adaptar tanto el servidor HTTP (*MultithreadServer.java*) como el servidor HTTPS (*ServerHTTPS.java*), para que en su constructor se les pase un objeto *InformationService*, que contendrá los últimos valores de los sensores. Este objeto también se le pasa a la clase *Smtp.java*. Esta clase establece una conexión con la cuenta de correo proporcionada por medio de la clase *Store* del paquete *mail*. Por defecto es jaimeortizappc@gmail.com, pero se podría actualizar a cualquier cuenta de correo siempre que sepamos la contraseña. Es importante que en la cuenta activemos el reenvío de POP y que permitamos el acceso a la misma de aplicaciones que no sean confiables por Google en este caso. El siguiente paso es obtener los mensajes pendientes de la carpeta de la bandeja de entrada haciendo uso de la clase *Folder.* Del mensaje obtenemos la cuenta desde la que se mandó para saber adonde responder y se hace el tratamiento del asunto y de borrado como se ha comentado en la sección del despliegue.

Como ya se ha comentado, el mensaje de respuesta será multiparte. Los datos en texto plano se obtienen con un get *getLastData* del servicio de información y el fichero adjunto serializado se consigue comprobando el formato del mensaje y tratándolo por medio de una instancia de *TranslatorBroadcast* implementada para la práctica 3.