Lab6 Sistemas Operativos II

Ingeniería en Compuatación - FCEFyN - UNC

Sistemas Embebidos

Introducción

Los *sistemas embebidos* suelen ser accedidos de manera remota. Existen distintas técnicas para hacerlo, una forma muy utilizada suelen ser las *RESTful APIs*. Estas, brindan una interfaz definida y robusta para la comunicación y manipulación del *sistema embebido* de manera remota. Definidas para un esquema *Cliente-Servidor* se utilizan en todas las verticales de la industria tecnológica, desde aplicaciones de *IoT* hasta juegos multijugador.

Objetivo

El objetivo del presente trabajo práctico es que el estudiante tenga un visión *end to end* de una implementación básica de una *RESTful API* sobre un *sistema embedido*. El estudiante deberá implementarlo interactuando con todas las capas del procesos. Desde el *testing* funcional (alto nivel) hasta el código en C del servicio (bajo nivel).

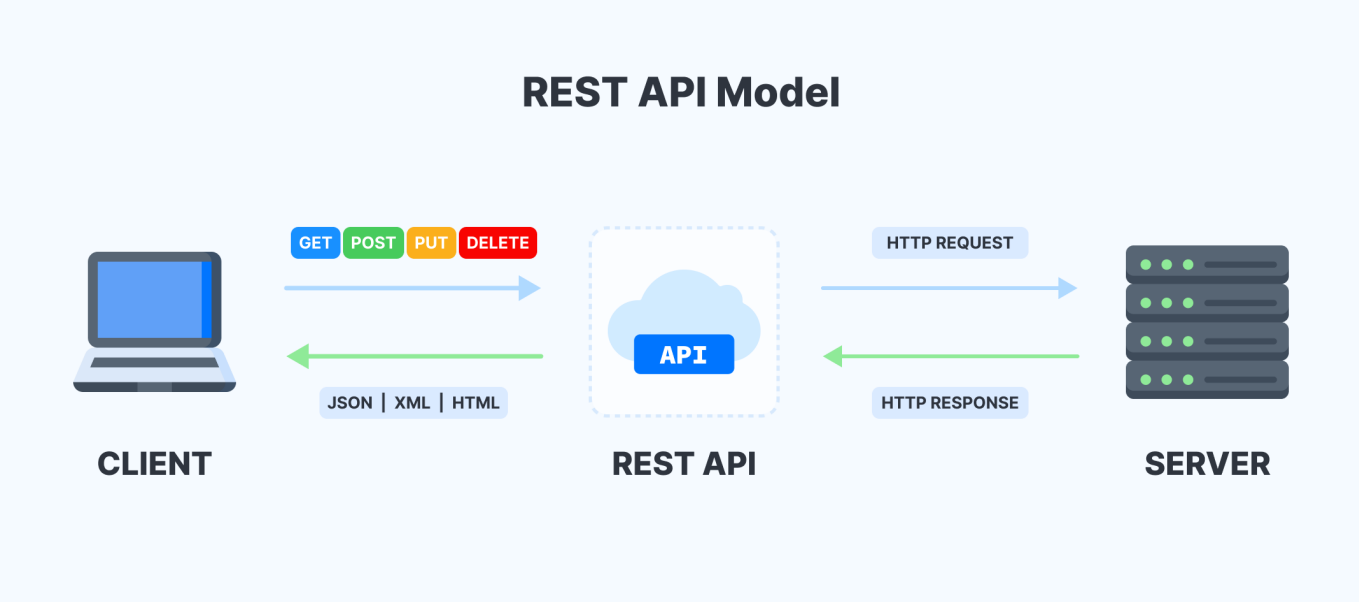

Figura 1: API Rest. (*propiedad intelectual de https://appmaster.io/*)

Desarrollo

Ulfius

En primer lugar se implementó un ejemplo simple obtenido del repositorio oficial de **Ulfius**, agregando un contador, con 2 endpoints para incrementarlo y para mostrarlo. Inicialmente se hicieron consultas mediante la herramienta **curl** pero también es una alternativa con interfaz gráfica usar **Postman**, que facilita hacer las consultas.

Jansson

Luego se buscó la manera de formatear la información que ofrecen estos servicios para brindarla en formato **json**, utilizando la librería **Jansson**. De esta manera el intercambio de información entre cliente y servidor es independiente del lenguaje de programación de cada uno, ya que json es un formato popular.

systemd

A continuación se investigó sobre **systemd**. Este es un proceso que inicia después del kernel para inicializar todos los demás procesos. Es posible declarar **servicios**, que son procesos que gestiona systemd para que se ejecuten ordenadamente en el sistema operativo, con la posibilidad de que estos inicien con el sistema operativo de forma automática.

Se dispone de 2 **servicios de systemd**:

- contador.service: Es el servicio que lleva la cuenta de los usuarios del sistema, que se debe consultar mediante un http request.
- usuarios.service: Este servicio muestra la lista de usuarios del sistema y permite crear nuevos usuarios, por ejemplo para acceder por conexión SSH. Ambos servicios inician con el sistema operativo, y en caso que tengan un error crítico, esperan 5 segundos y luego se reinician automáticamente, guardando esta inforamción en syslog.

nginx

Una vez que los servicios están listos, se emplea el web server **nginx** para servir los endpoints de cada servicio, ofreciendo un web server para cada servicio, que serán accedidos con sus respectivas direcciones, permitiendo formular consultas HTTP como las siguientes:

- GET http://www.contadordeusuarios.com/contador/value
- GET http://www.lab6.com/api/users
- POST http://www.lab6.com/api/users Además, ambos servidores tienen un endpoint llamado /helloworld para verificar la conectividad de una manera sencilla.

yder (Logs)

Se registran los eventos importantes mediante la librería **yder** para registrar logs. Sólo se registran logs del servicio usuarios, cuyo archivo log se encuentra en **/var/log/usuarios/usuarios.log**. Además se implementó **logrotate** para archivar los logs una vez que el archivo se torna muy grande o muy viejo, para prevenir que el sistema se colapse por demasiados mensajes de logs.

makefile

Se brinda un **makefile** con varios targets para poder facilitar la configuración en un nuevo dispositivo o servidor. Los targets disponibles son:

- **make requirements**: Instala las librerías necesarias para el correcto funcionamiento de los servicios.
- **make install**: Compila e instala los servicios, configurando todo lo necesario para que todo funcione solo con este comando.
- **make uninstall**: Detiene los servicios y elimina todos los archivos que estos generaron en el sistema, **excepto los logs**.
- **make delete_logs**: Elimina los logs ubicados en **/var/log/usuarios/**.

- **make config_nginx**: Configura el web server nginx usando los archivos que se encuentran en el repositorio, **no es necesario ejecutarlo porque se ejecuta junto con make install, solo se ofrece en caso de necesitar reiniciar nginx y no los servicios**.

Referencias y ayudas

- System D
- System D en Freedesktop
- nginx web server
- Ulfius HTTP Framework
- Kore Web PLataform
- Logrotate
- Jansson
- Using Jansson
- systemd
- getpwent
- yder Logs
- Regex regcomp
- time & ctime

Autor: Federico Coronati