Trabajo práctico 2

Sistemas Operativos de Propósito General. CESE

Objetivo:

Se deberá desarrollar un sistema que permite mostrar el estado de la red de subtes y controlar la señalización. Para ello el sistema cuenta con un acceso mediante un sitio web que permite modificar y leer el estado de cada línea, y por otro lado se dispone de una placa EDUCIAA que simula mediante sus salidas las lámparas que indican el estado de cada línea en un cartel real y además permiten mediante sus pulsadores cambiar el estado manualmente.

Partes del sistema:

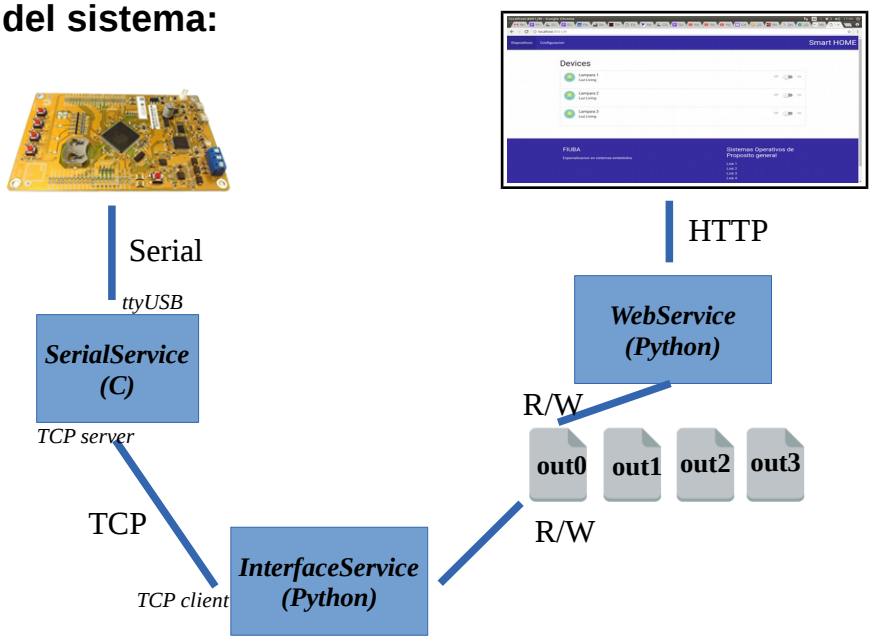
EDU-CIAA:

Mediante una placa EDU-CIAA se simulará el estado de 4 líneas de subte mediante 4 leds.
Si el led esta apagado significa que la línea funciona con normalidad, si parpadea significa que funciona con demoras y si esta encendido significa que la línea esta interrumpida.
La placa contendrá un firmware provisto, que permite mediante el puerto serie del conector usb de debug, enviar tramas a la placa para controlar el estado de las salidas y recibir tramas cuando se presionen los pulsadores.
El firmware provisto está escrito utilizando la biblioteca firmwareV2 y se provee funcionando.

PC:

Mediante la pc se alojará un sitio web también provisto, el cual mostrará el estado de las líneas y haciendo click sobre ellas, se podrá ir alternando su estado entre los tres mencionados previamente.
La PC estará conectada a la placa mediante el puerto serie-usb de debug de la EDU-CIAA.

Arquitectura del sistema:



El servidor web leerá y escribirá los archivos : **/tmp/outXX.txt**

- El servidor web lee el contenido de los archivos para mostrar el estado de las líneas en la página web.
- El servidor web escribe el contenido de los archivos el presionar sobre la imagen de cada línea para cambiar su estado.
- El servicio InterfaceService escribirá en los archivos cuando reciba por socket una trama de que se presionó un pulsador.
- El servicio InterfaceService leerá los archivos y al detectar un cambio enviará por socket una trama indicando el nuevo estado de las líneas.

Servicio "Serial Service"

La EDU-CIAA se comunicará con el servicio *SerialService* mediante el puerto serie, que deberá recibir las tramas de la placa indicando que se presionó un pulsador. Cuando esto ocurra, deberá informarlo al servicio *InterfaceService* mediante socket TCP.
También deberá enviar por el puerto serie la trama que le indica el estado de las salidas, información que le provee el servicio *InterfaceService*. (Cuando se modifican los archivos desde el sitio web). Este servicio iniciará un servidor TCP para que el servicio *InterfaceService* se pueda conectar y comunicar.

Servicio "Interface Service"

Este servicio creará un cliente TCP que se conectará al servicio *SerialService*. Se encargará de leer y escribir en los archivos mencionados previamente para que la información se vea reflejada en la página web.

Funcionamiento

Cuando se presione un pulsador en la placa, ésta lo informará por puerto serie al *SerialService*, el cual deberá enviarlo mediante un paquete TCP al *InterfaceService* para que éste modifique el estado de la línea y escriba el nuevo valor en el archivo correspondiente.

Cuando se cambie desde la página web el estado de una línea, el servidor web escribirá en el archivo correspondiente el nuevo valor, el cual lo detectará el *InterfaceService* y lo informará mediante un paquete TCP al *SerialService*, el cual enviará la trama por el puerto serie a la placa, para que ésta modifique el estado de sus salidas.

Lo que se provee

1) Firmware de la EDU-CIAA
2) Página web
3) Servidor web
4) Servicio "InterfaceService"
5) Biblioteca para el puerto serial

Lo que se debe desarrollar

1) Servicio “SerialService”

Protocolo serie entre EDU-CIAA y SerialService

Seteo estado de salidas (hacia la edu-ciaa)
“>OUTS:X,Y,W,Z\r\n”
Siendo X,Y,W,Z el estado de cada salida. Los estados posibles son ““0” (apagado),”1”(encendido) o “2” (blink)

Evento pulsador (desde la edu-ciaa)
“>TOGGLE STATE:X\r\n”
Siendo X el número de pulsador (0,1, 2 o 3)

Protocolo TCP entre SerialService e InterfaceService

Seteo estado de salidas (desde InterfaceService a serialService)
“:STATESXYWZ\n”
Siendo X,Y,W,Z el estado de cada salida. Los estados posibles son ““0” (apagado),”1”(encendido) o “2” (blink)

Evento pulsador (desde serialService a InterfaceService)
“:LINEXTG\n”
Siendo X el número de pulsador (0,1, 2 o 3)

Puesta en marcha del sistema

1) Inicializamos el programa *InterfaceService*. Abrir una terminal.

```
cd TP2
cd InterfaceService
python Main.py
```

2) Inicializamos el servidor web. Abrir una terminal.

```
cd TP2
cd web
./runServer.sh
```

3) Abrimos un navegador y en la url ponemos : ***localhost:8001*** Deberá aparecer la página web.

4) Conectamos la EDU-CIAA a la PC. Obtener la ttyUSB que se generó ejecutando:

```
ls /dev/ttyUSB*
```

5) Abrir el programa “SerialService” desarrollado asignándole la ttyUSB de la placa.

Consideraciones a tener en cuenta en el programa a desarrollar

1) La función que lee datos del puerto serie no es bloqueante.
2) La función que lee datos del cliente tcp es bloqueante. No cambiar este comportamiento
3) Dadas las condiciones de los puntos 1 y 2, se recomienda lanzar un thread para manejar la comunicación con el cliente TCP y otro (opcional, no necesario) para la comunicación con el puerto serie.
4) No generar condiciones donde el uso del CPU llegue al 100% (bucles sin ejecución bloqueante).
5) El programa debe soportar que el cliente se desconecte, se vuelva a conectar y siga funcionando el sistema.
6) El programa debe poder terminar correctamente si se le envía la signal SIGINT o SIGTERM.
7) Se debe considerar sobre qué hilo de ejecución se ejecutarán los handlers de las signals.
8) Los temas que deberán implementarse en el desarrollo son los siguientes:

- Sockets
- Threads
- Signals
- Mutexes (De ser necesario)

Datos adicionales

Puerto TCP : 10000

IP local : 127.0.0.1

Velocidad ttyUSB de la placa : 115200

Para grabar el firmware de la EDUCIAA:
Bajar el firmware “firmware_v2_TP2.tar.gz” y descomprimirlo. Luego:

```
cd firmware_v2
make clean
make
make download
```

NOTA: En el caso de que aparezca el error “command not found” se deberá agregar a la variable de entorno PATH, la ruta al compilador arm: export PATH=$PATH:/home/usuario/ruta_compiler....

Condiciones de aprobación

Para considerar el trabajo práctico aprobado, el mismo deberá encontrarse funcionando con todos los controles de error implementados, una mínima documentación con comentarios y el código deberá encontrarse bien indentado y legible, además de cumplir con las consideraciones de desarrollo detalladas previamente.

El trabajo deberá ser terminado y entregado al finalizar la clase 8 y se considerará instancia de recuperatorio si se entrega hasta 3 días después. Luego de esa fecha se considerará al alumno desaprobado en la instancia de recuperación y deberá recursar la materia.