**Pràctica 1.   
PROGRAMACIÓ D’APLICACIONS**

**DE XARXA**

Júlia Nogales López

Xarxes

Grau en Enginyeria Informàtica

2023/2024



**Índex**

[1. Introducció 2](#_Toc164628619)

[2. Implementació Client 2](#_Toc164628620)

[2.1. Subscripció 2](#_Toc164628621)

[2.2. Manteniment de la comunicació 3](#_Toc164628622)

[2.3. Introducció de comandes 4](#_Toc164628623)

[3. Implementació Servidor 4](#_Toc164628624)

[3.1. Subscripció 4](#_Toc164628625)

[3.2. Manteniment de la comunicació 5](#_Toc164628626)

[3.3. Introducció de comandes 6](#_Toc164628627)

**Índex de figures**

[**Figura 1.** Procés de subscripció del client 3](#_Toc164628591)

[**Figura 2.** Manteniment de la comunicació del client 4](#_Toc164628592)

[**Figura 3.** Procés de subscripció del servidor 5](#_Toc164628593)

[**Figura 4.** Manteniment de la comunicació del servidor 6](#_Toc164628594)

# Introducció

Per la realització d'aquesta pràctica s'ha implementat un sistema de comunicació basat en el model client-servidor, on el client actua com a controlador dels sensors i el servidor adquireix les funcionalitats d'un equip central per gestionar i controlar les peticions de tots els clients.

El client ha estat programat en Python 3.9 i el servidor en Ansi C (C99 + la llibreria pthread). Les funcionalitats implementades tant en el **client** com en el **servidor** han estat la subscripció, el manteniment de la comunicació i la introducció de comandes.

# Implementació Client

Inicialment, el client executa la funció *read\_config()*, la qual llegeix el fitxer on està emmagatzemada la configuració d’aquest, ubicat per defecte sota el nom de “client.cfg” (tot i que es pot passar com a paràmetre en executar el programa mitjançant la comanda -c <nom\_fitxer>), i que conté la informació necessària per establir connexió amb el servidor. També permet entrar la comanda debug (-d), que dona informació addicional sobre la comunicació entre client-servidor i l’estat del client.

# Subscripció

Un cop tractats els paràmetres i la configuració inicial, el client, que es troba en estat NOT\_SUBSCRIBED, inicia la comunicació amb el servidor enviant-li una petició de subscripció (SUBS\_REQ) emprant el protocol UDP. Aquesta petició inclou la seva adreça MAC i un nombre aleatori inicialitzat a zeros. Un cop el client fa la petició de registre es manté en espera (WAIT\_ACK\_SUBS) per la resposta del servidor, que pot ser d’acceptació (SUBS\_ACK), de rebuig (SUBS\_REJ) o d'error (SUBS\_NACK).

El client també pot enviar-li paquets de negació al servidor si durant el procés comprova que les dades d’identificació d’aquest són errònies mitjançant la funció *verify\_server\_id().*

Si les dades són correctes i està autoritzat, el client és acceptat, envia un altre paquet (SUBS\_INFO) per verificar les dades (on ha d’adjuntar el número aleatori prèviament generat pel servidor) mitjançant *send\_info()* i actualitza el seu estat a WAIT\_ACK\_INFO. Finalment, si és verificat i acceptat pel servidor mitjançant el paquet INFO\_ACK, el client passa a l’estat SUBSCRIBED i es completa la fase de subscripció.

Si durant aquest procés no rep resposta per part del servidor o és negativa, automàticament retornarà a l’estat NOT\_SUBSCRIBED i reiniciarà el procés de subscripció. El client no repetirà el procés de subscripció si ja l’ha hagut de reiniciar ‘o’ vegades sense ser acceptat.

En la figura següent es mostra l’esquema del procés de subscripció del client amb els respectius canvis d’estat segons els paquets rebuts i enviats al servidor, marcat en vermell les situacions alternatives al procediment correcte/esperat:

Diagrama

Descripción generada automáticamente

**Figura 1.** Procés de subscripció del client

# Manteniment de la comunicació

Quan el client ja ha completat la fase de subscripció, aquest passa a mantenir comunicació periòdica amb el servidor. Per fer-ho, envia cada “v” segons un paquet de manteniment de la comunicació (HELLO) mitjançant un thread, mentre que en el programa principal espera a que el servidor respongui constantment amb el mateix tipus de paquet per verificar que no s’ha perdut la connexió.

Per rebre el primer paquet s’inicia un *timeout* del temps indicat (v·r) on, en cas de no rebre el paquet en el temps desitjat, el programa “fingeix” que ha rebut un paquet desconegut per passar a l’estat NOT\_SUBSCRIBED i reiniciar el procés de subscripció.

Si efectivament aconsegueix rebre el primer paquet HELLO del servidor, el client actualitza el seu estat a SEND\_HELLO, inicialitza un altre thread per escoltar comandes entrades per consola i es manté així fins que hi hagi alguna negativa per part de qualsevol dels dos missatgers. Si qualsevol deixa de transmetre aquest paquet, ja sigui perquè envia el paquet de rebuig (HELLO\_REJ) o perquè el servidor deixa de transmetre durant ‘r’ paquets del client (*strike\_hello\_miss*), es cancel·larà el procés de comunicació, el client passarà a l’estat NOT\_SUBSCRIBED i altre cop iniciarà el procés de subscripció.

En el següent diagrama estan representades totes les etapes possibles que es podrien dur a terme durant el procés de la comunicació periòdica entre client i servidor des del punt de vista del client:

Imagen que contiene Diagrama

Descripción generada automáticamente

**Figura 2.** Manteniment de la comunicació del client

# Introducció de comandes

Tot i que ja s’ha comentat breument en l’inici de l’apartat “Implementació Client”, s’ha gestionat la introducció de comandes per consola, tant en el moment d’executar el programa (-d i -c <nom\_fitxer>) com durant tot el programa. Per poder “escoltar” qualsevol entrada que es pugui dur a terme durant el programa, s’ha creat un thread que enregistra tots els inputs per consola (s*ystem\_input()*) i, segons la comanda entrada (stat, set o quit), mostra les dades del controlador, les modifica o tanca el programa, respectivament.

# Implementació Servidor

Pel que respecta a la implementació del servidor, la diferència principal amb el client, sense tenir en compte el llenguatge de programació, ha estat la utilització de threads per poder gestionar més d’un client que intenta comunicar-se, és a dir, obrir un procés nou per cada petició d’un controlador diferent entrada.

Igual que en el client, inicialment el servidor també ha de llegir els arxius de configuració propis (“server.cfg”) i dels controladors registrats (“controllers.dat”) que té per defecte o els que s’introdueixen per consola al executar el programa amb la comanda -c <nom\_fitxer\_config> i -u <nom\_fitxer\_controladors>, respectivament. A més, si es desitja activar el mode debug per rebre més informació de l’execució del programa es pot utilitzar la comanda -d per visualitzar-ho.

Un cop inicialitzat el programa es creen els sockets UDP i TCP (tot i que el protocol TCP no s’ha pogut implementar) per, posteriorment, poder tenir un canal de comunicació amb els clients.

# Subscripció

El procés de subscripció és individual per cada client, i s’inicia només quan un controlador envia una petició de subscripció al servidor (SUBS\_REQ). El servidor rep totes les peticions perquè des de que s’inicia la seva execució ja es troba en un bucle infinit (*service\_loop()*) per esperar peticions dels clients.

Quan el servidor rep un paquet crea un nou thread que és el que utilitzarà per gestionar tot l’enviament de paquets amb aquest client fins que es tanqui la connexió. En rebre el primer paquet, SUBS\_REQ, comprova que el controlador estigui registrat i en estat DISCONNECTED. Un cop verificat que totes les dades són correctes, crea un nou port UDP (que és el que s’utilitzarà en endavant per la comunicació amb aquest controlador), genera un número aleatori per verificar la identitat del controlador en els futurs paquets rebuts, li envia el paquet d’acceptació i actualitza l’estat del controlador a WAIT\_INFO.

Igual que en el cas del client, el servidor espera rebre el paquet de resposta (SUBS\_INFO) en un màxim de temps determinat (“s” cops el temps inicial d’enviament). Si no es rep aquest paquet, enviarà un SUBS\_REJ i finalitzarà el procés de subscripció d’aquell controlador, actualitzant el seu estat a DISCONNECTED. Si el rep dins del temps estipulat, li retornarà l’acceptació del paquet (INFO\_ACK), canviarà l’estat a SUBSCRIBED i es donarà per acabada la fase de subscripció.

Diagrama

Descripción generada automáticamente

**Figura 3.** Procés de subscripció del servidor

# Manteniment de la comunicació

¿?

Gráfico

Descripción generada automáticamente con confianza media

**Figura 4.** Manteniment de la comunicació del servidor

# Introducció de comandes

Pel cas del servidor, les comandes a implementar que s’introdueixen per consola durant l’execució del programa han estat list i quit, on “list” mostra la llista dels controladors i el seu estat actual i “quit” tanca el programa.