

**XARXES**

**PRÀCTICA 1: L’APLICACIÓ ECO   
AMB *SOCKETS* TCP/IP**

**MEMÒRIA**

**Aniol Juanola Vilalta, u1978893,** [**u1978893@campus.udg.edu**](mailto:u1978893@campus.udg.edu)**, GEINF**

**Jordi Badia Auladell, u1978902,**  [**u1978902@campus.udg.edu**](mailto:u1978881@campus.udg.edu)**, GEINF**

**Girona, octubre de 2023**

**Continguts**

[1. Requisits mínims i millores 1](#_Toc146466364)

[2. L’arquitectura en capes 1](#_Toc146466365)

[3. La interfície aplicació-usuari 1](#_Toc146466366)

[4. Els serveis de la capa d’aplicació i transport 1](#_Toc146466367)

[5. Les interfícies de les capes d’aplicació i transport 2](#_Toc146466368)

[5.1 La interfície de la capa d’aplicació ECO 2](#_Toc146466369)

[5.2 La interfície de la capa de transport TCP 2](#_Toc146466370)

[6. Els protocols de les capes d’aplicació i transport i els *sockets* TCP (estudi amb *Wireshark* i “ss”) 2](#_Toc146466371)

[6.1 El protocol de la capa d’aplicació d’ECO (estudi amb *Wireshark*) 2](#_Toc146466372)

[6.2 El protocol de la capa de transport TCP (estudi amb *Wireshark*) 2](#_Toc146466373)

[6.3 L’encapsulació de protocols (estudi amb *Wireshark*) 3](#_Toc146466374)

[6.4 Els *sockets* TCP de l’aplicació (estudi amb “ss”) 3](#_Toc146466375)

[7. Les millores 3](#_Toc146466376)

[7.1 Poseu-hi el nom/títol de la millora 4](#_Toc146466377)

[7.2 Poseu-hi el nom/títol de la millora 4](#_Toc146466378)

[8 Problemes i suggeriments 4](#_Toc146466379)

[9 Treball en parella i dedicació 4](#_Toc146466380)

[Bibliografia 4](#_Toc146466381)

En aquesta pràctica [1] s’ha dissenyat i construït l’aplicació ECO, una aplicació en xarxa (o aplicació distribuïda) inspirada en l’*Echo Protocol* [2]. Per a construir-la s’ha fet servir la interfície de *sockets* TCP/IP [3].

# 1. Requisits mínims i millores

Aquí heu de fer una llista dels requisits mínims i millores que heu fet.

Els requisits (R) mínims fets són els següents:

* R1. Construcció d’una aplicació ECO sobre TCP, seguint un model C-S.
* R2. La interfície aplicació-usuari del client (execució en un terminal).
* R3. La interfície aplicació-usuari (administrador) del servidor (execució en un terminal).
* R4. Programació en llenguatge C i estructura del codi en diversos fitxers, per separar “interfície aplicació-usuari + capa d’aplicació d’ECO” de la capa de transport TCP.
* R5. Estudi dels protocols d’aplicació i transport i dels sockets TCP.

Les millores senzilles (MS) fetes són les següents:

* MS1. Comprovació que al S, scon i sesc són dos descriptors d’un mateix socket, i per tant tenen la mateixa @ (@IP i #port TCP)
* MS2. El format del vostre missatge d’ECO
* MS3. Missatges d’error de la interfície de sockets en 3 casos

# 2. L’arquitectura en capes

L’aplicació ECO és Client-Servidor i la seva arquitectura en forma de capes segueix el model de referència TCP/IP d’Internet, és a dir, el model de 3 capes d’Aplicació (A), Transport (T) i Xarxa (X), la darrera formada per les capes d’Interxarxa (I) i de xarxa. A la Fig. 1 es mostren les capes de l’aplicació ECO.



**Figura 1**: L’arquitectura en capes de l’aplicació ECO.

# 3. La interfície aplicació-usuari

A la funció main() del C i del S ECO s’han fet servir dos grups de funcions, i) les relatives a la interacció entre l’usuari i l’aplicació (via teclat, pantalla, etc.) i ii) les de la interfície de la capa de transport TCP. Les del segon grup es descriuen més avall, mentre que les del primer grup es descriuen a continuació.

Les funcions del main() relatives a la interacció entre l’usuari i l’aplicació són les següents:

* **printf()**: Permet mostrar text per pantalla (*stdout*); s’ha utilitzat per a mostrar menús i els diferents missatges i ecos corresponents.
* **scanf()**: Permet llegir de *stdin*; s’ha utilitzat per a llegir input de l’usuari com adreces IP i ports. Llegeix fins a que troba un salt de línia o un espai.
* **strcmp()**: Permet comparar *strings* (*char\**), de forma que retorna 0 si són iguals. S’ha usat per a comprovar l’input i acabar l’enviament de missatges des de l’executable del client.
* **strcpy()**: Permet copiar un string a un altre string.
* **snprintf()**: Similar al *printf()*, permet guardar l’output a un *char\**.

# 4. Els serveis de la capa d’aplicació i transport

Els serveis d’una capa són les tasques que la capa fa, la funcionalitat que proporciona.

Els serveis de la capa d’aplicació d’ECO són:

* Permet escollir a quin servidor i a quin port es vol connectar el client.
* Permet enviar un missatge a un servidor i aquest el reenvia cap al client altra vegada per a mostrar-lo per pantalla.
* Permet tancar la connexió i obrir-ne una de nova sense reiniciar el programa.

Els serveis de la capa de transport TCP són:

* Inicialitzar un *socket* d’escolta (pel servidor).
* Inicialitzar una connexió a un servidor en mode escolta (pel client).
* Tancar la connexió actual.
* Rebre i enviar missatges.
* Donat un *socket*, retornar la IP i port del propi *socket* i del *peer* connectat.

# 5. Les interfícies de les capes d’aplicació i transport

La interfície d’una capa són el conjunt de “mètodes” o “funcions” amb les quals es poden accedir als serveis de la capa.

En aquesta pràctica no s’ha definit una interfície de la capa d’aplicació d’ECO; en canvi, sí que s’ha fet servir una interfície de la capa de transport TCP.

## 5.1 La interfície de la capa d’aplicació ECO

En aquesta pràctica no s’ha definit una interfície de la capa d’aplicació d’ECO.

## 5.2 La interfície de la capa de transport TCP

Aquí heu de fer una llista de les funcions (són les funcions “externes”) de la “nova” interfície de la capa TCP, tTCP, que heu fet servir, acompanyades d’una breu descripció. Si per fer alguna millora heu fet canvis respecte les funcions dissenyades a classe, expliqueu-ho.

En lloc de fer servir la interfície “original” de *sockets* TCP, a sobre seu s’ha construït una “nova” interfície. Les funcions d’aquesta “nova” interfície de la capa TCP, tTCP, són les següents:

* **TCP\_CreaSockClient**: Donada una adreça IP i un port, crea un *socket* i el retorna.
* **TCP\_CreaSockServidor**: Donada una IP i un port, crea un *socket* d’escolta i el retorna.
* **TCP\_DemanaConnexio**: Donat un *socket*, una IP i un port, es demana una connexió mitjançant el socket prèviament creat i es retorna el descriptor de la connexió establerta.
* **TCP\_AcceptaConnexio**: Donat un *socket* d’escolta, el servidor espera a rebre una connexió i, un cop establerta i acceptada, retorna el descriptor d’aquesta nova connexió.
* **TCP\_Envia**:

# 6. Els protocols de les capes d’aplicació i transport i els *sockets* TCP (estudi amb *Wireshark* i “ss”)

El protocol d’una capa és el conjunt de regles que governen la comunicació, regles que defineixen el diàleg de missatges entre les parts (o entitats) d’una capa d’una manera clara i precisa. La definició d’un protocol comprèn 3 aspectes: el nom i significat dels missatges, el seu format i la seva seqüència temporal.

Per cada capa, heu de descriure el seu protocol, és a dir, el nom, el significat, el format i la seqüència temporal dels seus missatges. Quant a les seqüències temporals, feu-les en un cas concret, el corresponent a l’estudi de l’aplicació fet amb l’analitzador de protocols *Wireshark* i la comanda de xarxa “ss” que es va proposar fer a classe; per dibuixar les seqüencies, feu servir la “plantilla” *PowerPoint* que es troba al *Moodle* de l’assignatura (al dibuix indiqueu també la relació entre els missatges i la interfície de la capa).

## 6.1 El protocol de la capa d’aplicació d’ECO (estudi amb *Wireshark*)

En aquesta pràctica no s’ha definit un protocol de la capa d’aplicació d’ECO que totes les aplicacions haguessin de complir, sinó que cadascú ha pogut decidir com fer-ho.

En el protocol d’ECO només hi ha un missatge, el que porta la línia.

Quant a les seqüències temporals de missatges d’ECO, heu de fer la del cas d’estudi (la de la captura de paquets adjunta). Feu servir la “plantilla” *PowerPoint* que es troba al *Moodle*.

La seqüència temporal dels missatges d’ECO corresponent a l’estudi fet amb *Wireshark* (veure la captura adjunta) es mostra a la Fig. 2.

Poseu-hi la figura

**Figura 2**: Seqüència temporal del protocol d’ECO.

## 6.2 El protocol de la capa de transport TCP (estudi amb *Wireshark*)

Aquí heu de dir el nom, significat, format i seqüències temporals dels missatges TCP pel cas d’estudi.

* Quant als noms i significats dels missatges TCP, els teniu a la “plantilla” *PowerPoint* que es troba al *Moodle*.
* Quant al format dels missatges TCP, podeu posar un format aproximat (reduït) com els de classe, o un format exacte que podeu trobar a molts llocs.
* Quant a les seqüències temporals de missatges TCP, heu de fer la del cas d’estudi (la de la captura de paquets adjunta). Als dibuixos de les seqüències temporals indiqueu la relació entre els missatges TCP i la interfície de la capa TCP; indiqueu-hi també la relació entre aquesta seqüència de missatges TCP i la corresponent seqüència de missatges ECO (p.e., dieu “el missatge TCP xxx porta a dins un missatge ECO”, etc.). Feu servir la “plantilla” *PowerPoint* que es troba al *Moodle*.

El nom i significat dels missatges TCP és el següent:

El format dels missatges TCP es mostra a la Fig. 3.

Poseu-hi la figura

**Figura 3**: Format dels missatges del protocol TCP.

La seqüència temporal dels missatges TCP corresponent a l’estudi fet amb *Wireshark* (veure la captura adjunta) es mostra a la Fig. 4.

Poseu-hi la figura

**Figura 4**: Seqüència temporal del protocol TCP.

## 6.3 L’encapsulació de protocols (estudi amb *Wireshark*)

Del cas d’estudi, escolliu un missatge qualsevol d’ECO i estudieu-ne l’encapsulació de protocols. Com que les línies dels missatges d’ECO són “petites”, TCP no fragmenta el missatge d’ECO, de manera que 1 missatge d’ECO va a dins d’1 missatge TCP d’informació, que va a dins d’1 missatge IP, que va a dins d’un missatge *Ethernet*. Aquí heu de dir, per cadascun d’aquests missatges, quina longitud té (escriviu-la com “c+i”, on “c” és la longitud de la capçalera i “i” la de la informació).

De l’estudi fet amb *Wireshark* (veure la captura adjunta) hem escollit un dels missatges d’ECO i hem estudiat l’encapsulació dels seus protocols. Com que les línies dels missatges d’ECO són “petites”, TCP no fragmenta el missatge d’ECO, de manera que només hi ha un paquet TCP INF. La llista de missatges de protocols i la seva longitud és la següent

* ECO: 25 *bytes*
* TCP d’informació: 20+25 (= 45) *bytes*
* IP: 20+45 (= 65) *bytes*
* *Ethernet*: 14+65 (= 79) *bytes*

## 6.4 Els *sockets* TCP de l’aplicació (estudi amb “ss”)

Aquí heu de descriure l’estudi de *sockets* TCP de l’aplicació fet amb al comanda de xarxa “ss” (“ss -natu”). Amb aquesta comanda mostreu les adreces dels *sockets* del C i del S en els 5 instants següents: i) amb el C i S apagats; ii) un cop s’ha engegat el C i el S però encara no s’han connectat; iii) durant l’enviament de les línies; iv) un cop acabada la connexió; v) un cop s’acaba l’execució del C i del S. A l’instant iii), compareu les adreces mostrades per “ss” amb les adreces que mostra l’aplicació per pantalla (i si també captureu els paquets amb *Wireshark*, amb les adreces observades en els paquets).

En el cas d’estudi, els *sockets* TCP de l’aplicació en diferents instants de l’execució van ser els següents:

# 7. Les millores

En aquesta secció es descriuen les millores que s’han fet.

## 7.1 Comprovació que al S, scon i sesc són dos descriptors d’un mateix socket, i per tant tenen la mateixa @

## 7.2 Poseu-hi el nom/títol de la millora

Feu una descripció en la que ... (com a 7.1).

# 8 Problemes i suggeriments

# La pràctica no ha suposat cap problema.

# 9 Treball en parella i dedicació

# Hem aprofitat les hores de classe per a picar el codi i resoldre els petits problemes que ens sorgien amb l’ajuda del professor de pràctiques. L’informe l’hem redactat posteriorment a casa de forma conjunta per videotrucada.

# Bibliografia

[1] Lluís Fàbrega, *Pràctica 1:* *L’aplicació ECO amb sockets TCP/IP*, curs 2023-24, UdG, 2023.

[2] J. Postel, *RFC 862 - Echo Protocol*, 1983. Disponible a: <https://www.rfc-editor.org/rfc/rfc862>.

[3] Lluís Fàbrega, *La interfície de sockets de C a UNIX*, curs 2023-24, UdG, 2023.