Cypher System

(or SYSTEM FAILURE)

```
@echo off
pause
color 0a
mode 1000

:a
echo
%random%%random%%random%%random%%random%
%random%%random%%random%%random%%random%
%random%%random%%random%%random%%random%
%random%%random%%random%%random%%random%
%random%%random%%random%%random%%random%
%random%%random%%random%%random%%random%
%random%
goto a
```

Introducció conceptual a les pràctiques de Sistemes Operatius

Un sistema operatiu està format, bàsicament, per 4 mòduls o subsistemes:

NUCLI. És l'única capa que pot inhibir interrupcions. Les seves funcions bàsiques són la representació dels processos en execució, el control de la concurrència de processos, la gestió i control de les interrupcions, i dotar al sistema de mecanismes de comunicació entre processos, de sincronització i exclusió mútua.

SISTEMA E/S. És l'única capa que pot executar instruccions del tipus entrada i sortida. La seva funció és la comunicació amb els perifèrics. Per tant, les capes superiors, per a dialogar amb el *hardware*, ho han fer a través d'aquesta capa.

GESTIÓ DE MEMÒRIA. Conjuntament amb el *hardware*, crea la possibilitat de disposar de memòria virtual. A més a més, aquesta capa és la responsable de garantir (també conjuntament amb el *hardware*) la protecció de les dades a memòria i també la seva compartició.

SISTEMA DE FITXERS. Dota al sistema d'una visió estructurada de les dades emmagatzemades al disc.

Les aplicacions d'usuari són programes creats per l'usuari. Quan aquests programes necessiten alguna funcionalitat del sistema operatiu, realitzen el que s'anomena una "crida al sistema", que consisteix en fer crides a funcions que ofereixen les diferents capes. És a dir, si l'aplicació vol mostrar un caràcter per pantalla, haurà de cridar a una funció de la capa d'E/S per a aconseguir-ho.

El contingut de la pràctica de l'assignatura de Sistemes Operatius està orientat a l'aprenentatge d'una arquitectura distribuïda mitjançant la diferents mecanismes de comunicació (*sockets* principalment), posant de relleu la problemàtica de la concurrència de processos i la multitasca. L'objectiu principal és establir connectivitat entre els diferents equips i permetre la transmissió d'informació entre els diversos nodes d'un sistema distribuït. A més, cal la utilització de mecanismes del nucli del sistema operatiu (memòria compartida, mètodes d'exclusió mútua, sincronització i creació de processos) per a poder resoldre la pràctica.

NOTA 0. Les quatre pràctiques del curs 2004-2005 rebien els noms de *Atreides*, *Atreides++*, *Harkonnen* i *Fremen*. Tot era en honor a Frank Herbert i la seva famosa novel·la *Dune*. Força alumnes van recordar l'origen i van descobrir d'on provenia el nom.

NOTA 1. Les quatre pràctiques del curs 2005-2006 rebien els noms de *Oedipus Rex*, *Sphinx*, *Antigona* i *Teiresias*. Com totes les pràctiques de SO, no hi ha res a l'atzar. Els noms provenien de la tragèdia d'Èdip de Sòfocles i dotaven de sentit a les pràctiques. S'iniciava el curs amb una pràctica 1 senzilla, Èdip Rei. Èdip viu feliç com a rei mentre ignora que ha matat el seu pare i s'ha casat amb la seva mare. La ignorància el fa feliç. Els alumnes de SO viuen feliços perquè la pràctica 1 és fàcil i no saben que els espera. La pràctica 2 és l'Sphinx. Èdip ha de superar la prova de l'Sphinx ja que, sinó, aquesta el matarà. Òbviament, la pràctica 2 del curs 2005-2006 era la més difícil de totes i si no la superaves no aprovaves l'assignatura. La pràctica 3 es deia Antígona, qui ajuda a Èdip un cop aquest cau en desgràcia i en la desesperació. Antígona era la salvació dels alumnes que volien anar a l'examen de juny i necessitaven una pràctica llarga lliurada. Us estalvio els detalls de Teiresias. Tot i que molts alumnes van seguir els noms i les referències, cap d'ells va acabar de copsar el sentit de la tragèdia...

NOTA 2. Les pràctiques del curs 2006-2007 es van centrar en l'obra mestra de Dante Alighieri, "La Divina Commedia". Les pràctiques es deien "Inferno", "Purgatorio" i "Paradiso". Suposo que no calen gaire comentaris per entendre què passa a la pràctica 1. La segona pràctica es suavitza però la necessites si vols presentar-te a examen. Finalment, la tercera és relativament curta i senzilla, un paradís comptant amb tot el que has fet anteriorment.

NOTA 3. Les pràctiques del curs 2007-2008 es van centrar en l'obra mestra dels germans Wachowski, "Matrix", on els humans (alumnes) han de lluitar contra les màquines per a sobreviure (aprovar l'assignatura). Les pràctiques es deien "Matrix", "El Ferroviario" i "Keymaker". Matrix era el servidor central on s'havien de connectar els clients, com *Nebuchadnezzar*, per a poder intercanviar arxius i comunicar-se mitjançant un xat distribuït. També podien connectar-se amb altres dimonis, com *Oraculo* o *Link*, que els donaven consells. *El Ferroviario* era una pràctica de simulació de càrrega de processos i administració de memòria. Si coneixeu el paper de *El Ferroviario* a la pel·lícula *Matrix* veureu que la relació és directa. Finalment, la codificació de fitxers en EXT2. Vam considerar que un bon nom per descodificar era el personatge de *Matrix* anomenat *Keymaker*.

NOTA 4. Les pràctiques del curs 2008-2009 es van centrar en l'obra mestra de Isaac Asimov, "La fundació". Les pràctiques es deien: "Trantor", el planeta central de la galàxia, tenint, per tant, la mateixa magnitud que la pràctica 1 de Sistemes Operatius; "Trevize, the loader", nom del conseller de la primera fundació, amb una intuïció insòlita, però amb perilloses intencions; i "Pelorat, the file reader", nom del professor d'història que ajudà a Trevize a trobar el planeta Terra, la llar on descansar després d'aprovar les tres pràctiques de Sistemes Operatius.

NOTA 5. Les pràctiques del curs 2009-2010 es van centrar en el grup australià de *hard rock* AC/DC, com a homenatge a la marxa del que va ser professor de pràctiques de l'assignatura els tres darrers anys, Hugo Meza. La primera pràctica, sota el nom de *Highway to shell* (petita modificació del nom de la cançó més popular del grup), conduïa als alumnes per tres fases. La

primera, *Welcome to the Shell*, donava la benvinguda a l'infern als alumnes amb la implementació d'un intèrpret de comandes *shell*, sota el nom de Malcolm (cantant del grup). A la segona fase, per tal de fugir de les tenebres en què es trobaven, els alumnes es van veure pactant amb el diable (*Dealing with the Devil)*, on havien de seguir tot un protocol per a comunicar-se amb el dimoni Angus (guitarrista del grup). Finalment, quan creien que tot havia passat, es topaven amb *Ballbreaker* (disc que van gravar al 1995), nom que no requereix gaires explicacions. La segona pràctica del curs 2009-2010 rebia el nom de *Back in Black*, nom del disc llençat al 1980, on els alumnes havien d'identificar el format d'un volum donat i extreure'n informació.

NOTA 6. Les pràctiques del curs 2010-11 es van centrar en les obres del polèmic *Donatien Alphonse François de Sade*, més conegut com "el Marqués de Sade". Així, la pràctica es titulava "*Les 120 journées de Sodome*". Aquesta estava dividida en tres fases: "*Le Château de Silling"* era la primera d'elles, on els alumnes gestionaven el comportament del client de l'aplicatiu, el qual rebia el nom del personatge de la novel·la "Blangis". Posteriorment els alumnes s'endinsaven al castell en una segona fase anomenada "*Les quatre madames*", on Blangis es connectava a un dimoni anomenat "Thérèse". Finalment, la cosa es desmadrava a la tercera fase, on els alumnes havien de crear el servidor "*Libertinage*", que, com el seu nom indica, és on començava el "festival".

NOTA 7. Les pràctiques del curs 2011-12 es van centrar en la mundialment coneguda sèrie televisiva del Simp(so)ns. Aquesta pràctica estava dividida en cinc fases: "La shell de Homer", "el bar de Mou", "Clancy Wiggum", "Living in Springfield" i "Living in Springfield++", en el seu conjunt aquestes fases formaven un sistema que permetia executar comandes de forma local, algunes pròpies en un servidor remot i l'activació de diversos serveis que mostraven frases mítiques de la sèrie, tot això seguint una arquitectura client - servidor.

NOTA 8. La pràctica del curs 2012-13 es va anomenar LsBox. El motiu va ser força simple: es tractava de dissenyar i implementar un sistema molt similar (de fet simplificat) del conegut Dropbox. L'únic detall curiós era el logotip de la pràctica que portava un mapa d'Austràlia encobert doncs era un petit homenatge a un monitor de SO que deixava aquestes tasques.

NOTA 9. La pràctica del curs 2013-14 es va anomenar LsHangIn. El motiu era que era una versió simplificada del conegut Google Hangouts: es tractava de dissenyar i implementar un sistema de xats amb múltiples sales per a usuaris. A més, cadascuna de les Fases tenien relació amb la pel·lícula Resacón en las Vegas.

NOTA 10. La pràctica del curs 2014-15 es va anomenar Gekko. Gekko és un empresari i principal protagonista de la pel·lícula Wall Street, relació directa amb la Borsa i l'objectiu de la pràctica. Però, a més, hi havia diversos homenatges més: TumblingDice, el generador de fluctuacions, és també una cançó dels Rolling Stones i Dozer, l'operador de Borsa, és un dels personatges de Matrix.

NOTA 11. La pràctica del curs 2015-16 es va anomenar LsTransfer, the force awakens. Era un clar homenatge al retorn de la saga d'Starwars. A més el servidor es deia Naboo (planeta que surt en diferents episodis d'Starwars) i els clients Gungan (habitants d'aquest planeta).

NOTA 12. La pràctica del curs 2016-17 es va anomenar LsTinder, may the Love be with you. Era per la clara similitud a la xarxa social a la qual la pràctica feia referència i el lema un altre homenatge a la saga Starwars. A més, els diferents processos es deien Rick i Morty en referència a una sèrie americana de televisió d'animació per adults.

NOTA 13. La pràctica del curs 2017-18 es va anomenar LsEat, may the Food be with you. Primerament, el nom en referència a la moda del Just Eat. El lema un altre homenatge a la saga Starwars, en la seva imminent estrena de l'Episodi VIII, The Last Jedi. A més, els diferents processos tenien noms d'homenatge a StarTrek. Picard i Data personatges i Enterprise, la nau.

NOTA 14. La pràctica del curs 2018-19 es va anomenar Cosgrove System, Stairway to heaven. Primerament, Cosgrove és el cognom de la família protagonista d'una pel·lícula mítica de Peter Jackson: Braindead, considerada un clàssic de les pel·lícules gore de l'època. El lema, Stairway to Heaven, no era res relacionat amb els observatoris de la pràctica sinó un homenatge a la famosa cançó de Led Zeppelin. A més, els diferents processos tenien noms dels protagonistes de la pel·lícula Braindead: McGruder, McTavish, Paquita i Lionel.

A partir d'aquí, la interpretació dels noms i els conceptes de la pràctica d'aquest any (relacionant la dificultat, el contingut i l'assignatura) us pertoca a vosaltres...i esperem veure'ls a l'informe de la memòria!

Cypher System (or SYSTEM FAILURE)

Amb la reputació que han agafat els equips programadors de Sistemes Operatius de La Salle Campus Barcelona una gran corporació que de moment vol romandre en l'anonimat ens ha encarregat poder crear un sistema, que vol anomenar Cypher System, que té per objectiu maximitzar la connectivitat entre tots els seus elements connectats, essent eficient en rendiment i eficaç en la transmissió d'arxius de veu.

Els socis de la corporació anònima, que anomenarem TM, ens demanen que dissenyem tot el sistema Cypher segons els requisits que ens especificaran detalladament i que aquest software el puguin integrar en el seu entorn de baix nivell. Ja cansats de les ineficiències d'entorns d'alt nivell han optat per la nostra proposta d'utilitzar el llenguatge C com a element d'implementació de tot el sistema.

El sistema Cypher serà una novetat doncs no és un sistema amb un element central com passa a la majoria d'entorns. Cada node o procés connectat haurà de tenir el mateix protagonisme. Es per això que parlem que aquest cop el disseny serà dissenyar i programar tot el sistema en base a un sol programa: Trinity.

Trinity ha de permetre una interconnexió amb els seus similars facilitat un intercanvi de missatges de text elementals i, a més, l'enviament de fitxers d'àudio de manera puntual.

Una primera aproximació funcional que TM vol que tingui el seu sistema Cypher, tindria l'aspecte que es presenta la Figura 1 tot i que, evidentment, l'arquitectura interna serà molt més complexa.

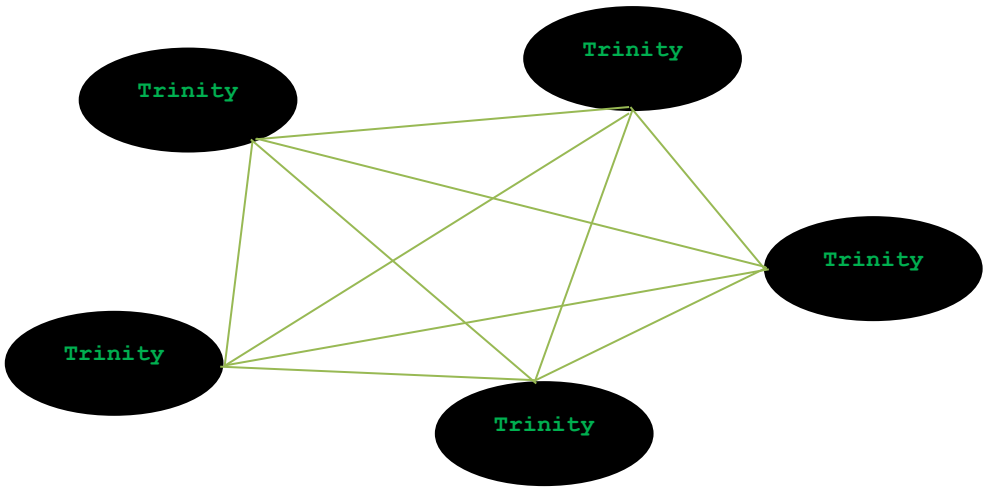

Figura 1. Cypher System: esquema funcional.

Descripció general del funcionament del sistema

El sistema Cypher està format per unitats replicades de Trinity que permeten interconnectar-se entre elles per la transmissió de text o àudio. Cal dissenyar el sistema per a que sigui escalable.

El procés Trinity es manipularà utilitzant un conjunt comandes esteses del sistema, que han de ser *case insensitive*. Per executar-se tindrà un fitxer de configuració associat que contindrà una sèrie de paràmetres importants per la seva execució i funcions.

Per poder intuir una primera execució del sistema la Figura 2 ens mostra un exemple:

```
montserrat$ Trinity Config.dat

Starting Trinity...

$Persefone: SHOW CONNECTIONS
Testing...
3 connections available
8011
8013
8014
$Persefone: CONNECT 8012
Connecting...
8012 connected: Niobe
$Persefone: say Niobe "Hello Niobe"
$Persefone: SHOW AUDIOS Niobe
[Niobe] Audios
JAGGER.MP3
ORBISON.MP3
$Persefone: DOWNLOAD Niobe JAGGER.mp3
[Niobe] JAGGER.MP3 downloaded

^C
Disconnecting Trinity...
montserrat$
```

Figura 2. Execució de Trinity.

Per poder dissenyar i implementar la pràctica es recomana que es faci una **lectura complerta de l'enunciat**. Per facilitar la seva realització s'han planificat 4 fases:

1. A la fase 1 es crearà el procés Trinity. Es processarà el fitxer de configuració i es realitzaran tot el tractament d'interpretació de les comandes esteses així com la seva desconnexió del sistema.
2. A la fase 2 s'implementarà una part de la connectivitat amb altres rèpliques de Trinity i l'enviament de missatges de text entre processos.
3. A la fase 3 s'implementarà la transmissió dels arxius d'àudio.
4. La fase 4 consistirà poder realitzar un broadcast amb els missatges de text.

Fase 1: Starting Trinity

En la fase 1 es començarà la programació de Trinity. Primer de tot caldrà processar el fitxer de configuració, el nom del qual es rep per paràmetre.

Fitxer de configuració:

Aquest fitxer de text tindrà diversos camps:

- El nom de l'usuari que executa Trinity.
- Nom de la carpeta on es troben els arxius d'àudio .
- La seva IP i port (cadascun en una línia que acaba amb un '\n').
- Les connexions que podrien estar disponibles al Sistema Cypher. Aquí hi haurà una IP (o una direcció web) i rang de ports a escanejar (port inicial i port final). Dins d'aquest rang de ports pot estar inclòs el port del propi procés Trinity que s'està executant.

Exemple de fitxer de text de configuració:

```
Persefone
Audios1
192.168.50.51
8010
Montserrat.salle.url.edu
8011
8020
```

Seguidament caldrà dissenyar i implementar tota la lògica de programa que interpreta totes les comandes esteses que permeten el funcionament de Trinity. Aquestes comandes són *case insensitive*.

COMANDA	DESCRIPCIÓ
`SHOW CONNECTIONS`	Escaneja les connexions possibles i mostra les connexions disponibles.
`CONNECT <port>`	Es connecta a un altre usuari com a client.
`SAY <user> "<text>"`	Envia un missatge de text a un altre usuari.
`BROADCAST "<text>"`	Envia un missatge de text a tots els usuaris connectats.
`SHOW AUDIOS <user>`	Mostra per pantalla els àudios d'un determinat usuari connectat.
`DOWNLOAD <user> <audio>`	Descarrega un determinat àudio d'un determinat usuari connectat.
`EXIT`	Surt de Cypher System

Taula 1. Llistat de comandes.

Consideracions Fase 1:

- Un cop finalitzada l'execució del procés Trinity s'ha d'**alliberar** tot tipus de memòria dinàmica si n'hi hagués.
- Podem considerar que el format del fitxer de configuració és correcte.
- **No** es poden utilitzar les funcions printf, scanf, gets, puts, etc. Només es podrà interactuar amb la pantalla i amb fitxers amb les funcions read (lectura) i write (escriptura). Sí es permet fer ús de la funció sprintf i similars.
- **No** es poden utilitzar les funcions system, popen ni cap variant.
- Cal garantir l'estabilitat de l'aplicació i el seu correcte funcionament. En cap cas es poden produir bucles infinits, *core dumpeds*, esperes actives, *warnings* al compilar, etc. També cal controlar tots aquells aspectes susceptibles de donar algun error i, en cas que es produeixin, **informar degudament a l'usuari** i, si és possible, seguir amb el funcionament normal de l'aplicació.
- Des d'ara i fins al final de la pràctica cal considerar que el procés Trinity pot finalitzar la seva execució utilitzant la comanda adequada o prement per CTRL+C. Cal tenir-ho en compte i procurar perquè tot el sistema quedi el més **estable** possible en cas que es produeixi.
- És obligatori la utilització d'un *makefile* per generar l'executable.
- No és suficient que l'arxiu que pugeu al pou tingui extensió .tar, sinó que s'ha de poder desempaquetar amb la comanda tar. Qualsevol pràctica o *checkpoint* que no es pugui "descomprimir" d'aquesta manera no serà corregida.
- Recordeu que el codi ha d'estar degudament modulat, és a dir: no es pot ubicar tot en un sol fitxer .c, i és obligatori que hi hagi un fitxer *makefile*. Tingueu en compte que si en intentar corregir una pràctica aquesta no compila amb la comanda make (pel motiu que sigui), l'entrega serà qualificada com a no apta.

Fase 2: Connecting Trinity

En aquesta segona fase caldrà poder fer un primer pas cap a la interconnectivitat de cadascuna de les Trinity.

Per aconseguir això cal implementar la possible execució de les comandes SHOW CONNECTIONS, CONNECT, SAY i EXIT.

Com que els diversos processos Trinity és evident que poden estar en màquines diferents aquesta connectivitat caldrà dissenyar-la mitjançant sockets. Reviseu el protocol de comunicació al PDF dels Annexos .

Pel que fa a poder resoldre la funcionalitat que demana la comanda SHOW CONNECTIONS es recomana que es faci parsejant la sortida que l'script proporcionat genera (*show_connections.sh*).

Per veure un petit exemple senzill de comportament podeu fixar-vos en les Figures 3 i 4 simulant dos programes Trinity al Cypher System.

```
montserrat$ Trinity Config1.dat
Starting Trinity...
$Persefone: SHOW CONNECTIONS
Testing...
1 connections available
8013
$Persefone: CONNECT 8013
Connecting...
8013 connected: Niobe
$Persefone: say Niobe "Hello Niobe"
$Persefone: SHOW CONNECTIONS
Testing...
2 connections available
8013 Niobe
8014
$Persefone: Say Niobe "Ciao"
$Persefone: Exit
Disconnecting Trinity...
montserrat$
```

Figura 3. Execució de Trinity (Persefone).

```
montserrat$ Trinity Config2.dat
Starting Trinity...
[Persefone]:Hello Niobe
$Niobe: SHOW CONNECTIONS
Testing...
1 connections available
8010
$Niobe: CONNECT 8010
Connecting...
8010 connected: Persefone
$Niobe: SHOW CONNECTIONS
Testing...
1 connections available
8010 Persefone
[Persefone]: Ciao
$Niobe: SHOW CONNECTIONS
Testing...
0 connections available
^C
Disconnecting Trinity...
montserrat$
```

Figura 4. Execució de Trinity (Niobe).

Consideracions Fase 2:

- Les de la Fase 1 segueixen essent vàlides.
- Des de Trinity cal tenir un cert control d'errors en la comunicació (si intenta connectar-se a una altra Trinity que ja no està activa, si s'estableix una connexió i després l'altra es desconnecta, etc.).
- Podem garantir que els missatges de text no seran mai més llargs de 180 caràcters.
- Quan un procés Trinity fa Exit o surt del sistema amb CTRL+C cal desconnectar-se dels altres processos Trinity on s'havia connectat.
- Pel disseny de les connectivitats cal tenir en compte que quan un procés Trinity demana connectar-se amb una altra Trinity, el primer està actuant com a client. Un cop establerta la connexió podrà enviar missatges de text a la segona Trinity.
- La Trinity que actua com a servidor ha de mantenir les connexions estables per, posteriorment, poder enviar àudios a altres Trinity connectades com a clients.
- Finalment, una Trinity que actua com a client, tot i poder estar connectada a altres Trinity, no interactuarà amb més d'una la vegada enviant-li missatges de text. Seria completament ineficient paral·lelitzar aquesta funcionalitat.

Fase 3: Transfering audio!

En aquesta fase, s'implementarà tota la transmissió de dades dels àudios entre les diferents Trinity que hi hagi. Per aconseguir això cal implementar la possible execució de les comandes SHOW AUDIOS i DOWNLOAD.

Els àudios que es volen enviar a un altre procés client que els demana s'han de poder descomposar en diferents trames de dades per poder ser enviades. A més, cal garantir que l'enviament ha estat correcte fen servir algun sistema que garanteixi la integritat de les dades transmeses (MD5SUM).

Un procés Trinity actuant com a client només podrà demanar una petició de descàrrega alhora. Però un procés Trinity actuant com a servidor pot rebre diferents peticions de descàrrega a la vegada i haurà de gestionar-les de manera eficient i concurrent.

El protocol de comunicació està definit en el PDF dels Annexos de la pràctica.

Un possible exemple senzill d'execució seria el que es presenta a la Figura 2.

Consideracions Fase 3:

- Les de les fases anteriors segueixen essent vàlides.
- Tot l'emmagatzemament de les dades ha de ser dinàmic.
- Com en tot sistema el nom dels fitxers és *case sensitive*.
- El conjunt d'àudios que té un usuari en concret només es buscaran a la carpeta que indica el fitxer de configuració.
- No es permès usar codi programat MD5SUM. Cal invocar a la funció que realitza aquesta tasca.
- Podem garantir que no es tancarà cap procés Trinity en mig d'un enviament d'àudio.

Fase 4: Broadcasting

En aquesta fase s'haurà d'implementar una funcionalitat que permetrà a un procés Trinity fer *broadcasting* del missatge de text que vol enviar. Això és que quan enviar un missatge, usant la comanda estesa adequada, el missatge de text s'enviarà a tots els processos Trinity amb els quals està connectat com a client.

Per veure un possible exemple, podeu veure l'execució a les figures 5, 6 i 7.

```
montserrat$ Trinity Config1.dat
Starting Trinity...
$Persefone: SHOW CONNECTIONS
Testing...
2 connections available
8011
8014
$Persefone: CONNECT 8011
Connecting...
8011 connected: Merovingio
$Persefone: CONNECT 8014
Connecting...
8014 connected: Link
$Persefone: SHOW CONNECTIONS
Testing...
2 connections available
8011 Merovingio
8014 Link
$Persefone: BROADCAST "Hello world"
[Link] Cool!
[Merovingio] Cool!
$Persefone: Exit
Disconnecting Trinity...
montserrat$
```

Figura 5. Execució de Trinity (Persefone).

```
montserrat$ Trinity Config2.dat
Starting Trinity...
$Merovingio: SHOW CONNECTIONS
Testing...
2 connections available
8010
8014
[Persefone] Hello World
$Merovingio: Exit
Disconnecting Trinity...
montserrat$
```

Figura 6. Execució de Trinity (Merovingio).

```
montserrat$ Trinity Config3.dat
Starting Trinity...
$Link: SHOW CONNECTIONS
Testing...
2 connections available
8010
8011
[Persefone] Hello World
$Link: Exit
Disconnecting Trinity...
montserrat$
```

Figura 7. Execució de Trinity (Link).

Consideracions Fase 4:

- Les de les Fases anteriors segueixen essent vàlides.
- Com es pot observar, cada cop que es fa un broadcast a les Trinity connectades, aquestes retornen al client que ha fet broadcast un missatge d'ackowledgement ("Cool!") de manera automàtica.
- Per evitar possibles problemes amb la visualització de pantalla (amb textos tallats per múltiples enviaments etc.) hem considerat que, a partir d'aquesta Fase 4, qualsevol operació sortida que impliqui la utilització de la pantalla aquesta es farà amb exclusió mútua individual.

Requeriments de lliurament i planificació

Aquesta pràctica disposarà de diferents punts de control o *checkpoints* per poder fer-ne un seguiment acurat i garantir la consolidació de cadascuna de les fases de manera incremental. Concretament se seguirà el següent calendari de dates límit:

CONCEPTE	**DEADLINE**
Fase 1	31/10/2019
Fase 2	28/11/2019
Fase 3	17/12/2019
Lliurament Final 1 (sobre 10)	09/01/2020
Lliurament Final 2 (sobre 10)	02/02/2020
Lliurament Final 3 (sobre 8)	15/05/2020
Lliurament Final 4 (sobre 6)	13/07/2020

Els *checkpoints* no són obligatoris, tot i ser altament recomanables per poder garantir la robustesa de la pràctica. Aquests *checkpoints* només serviran per incrementar linealment la qualificació de la pràctica. En cap cas penalitzaran la seva nota.

També és recomanable **validar el disseny global de la pràctica** amb els monitors de pràctiques abans d'iniciar les fases 2, 3 i 4. Així podreu garantir que la implementació no patirà d'inconsistències insalvables per fases posteriors. Per això només cal aprofitar els horaris de dubtes i portar els dissenys impresos a validar.

Els lliuraments es realitzaran a l'eStudy en un pou amb un fitxer que inclogui el codi font (fitxers .c, .h i el makefile) de la pràctica, funcionant completament sobre Montserrat, així com els fitxers de dades i configuració utilitzats. El fitxer haurà de ser obligatòriament en format .tar. El podeu generar amb la comanda següent:

```
tar cf Gx_Fn.tar *.c *.h makefile *.ext_fitxers
```

on Fn és el número de Fase a lliurar. Per exemple, el grup 12 enviaria G12_F1.tar per al lliurament del checkpoint de la Fase 1.

En els lliuraments finals també caldrà dipositar-la memòria en format PDF. La memòria ha de constar, obligatòriament, dels punts que s'indiquen més endavant.