Programació de serveis i processos

TEMA 01 – PART I

Programació de processos i fils

Unió Europea
Fons social europeu
L'FSE inverteix en el teu futur

IES DR. LLUÍS SIMARRO

1. Programació multiprocés i paral·lela

El dia a dia demana **cada vegada més potència de càlcul** a les aplicacions informàtiques, més i més càlculs numèrics en enginyeria, ciència, astrofísica, meteorologia… Volem que els ordinadors reaccionen amb un **raonament humà**. Així, tenim ordinadors que juguen partides d'escacs, que parlen com humans o que prenen decisions a partir de dades poc precises…

Malgrat que no és l'únic factor determinant, els **processadors** juguen un **paper fonamental** a l'hora d'assolir aquesta potència. Cada vegada són més ràpids i eficients i els sistemes operatius que permeten coordinar diversos processadors per incrementar el nombre de càlculs per unitat de temps. L'ús de diversos processadors dins d'un sistema informàtic s'anomena multiprocés. Ara bé, la coordinació de diversos processadors pot provocar situacions d'error que haurem d'evitar.

1.1 Programació multiprocés

Un cuiner que tinga intenció de fer un plat molt elaborat, amb salses i guarnicions diferents, probablement utilitzarà diferents paelles per cuinar cada element que necessite utilitzar per elaborar el plat. Tindrà una paella amb la salsa, una altra preparant la guarnició i en una altra cuinarà el plat principal. Tot això ho farà alhora i, finalment, quan acabe totes aquestes tasques les ajuntarà en un únic plat.

Aquesta analogia ens serveix per introduir el terme **multiprocés**. El cuiner està **executant diferents processos a la vegada**: cuina la salsa, la guarnició i el plat principal, seguint un ordre d'execució per arribar a un resultat que esperava, un bon plat.

1.1.1. Procés, sistemes monoprocessador-multiprocessador

Seguint amb l'analogia del cuiner, un procés seria l'activitat realitzada pel cuiner (**processador**) d'elaborar algun dels complements o el plat principal (**resultats**) a partir

d'una recepta (**programa**). Cada procés de creació requereix agafar els ingredients (les **dades**), posar-los a la paella (**recurs** associat) i cuinar-los amb independència de la resta de processos.

- Un **programa** és un element **estàtic**, un conjunt d'instruccions, unes línies de codi escrites en un llenguatge de programació, que descriuen el tractament que cal donar a unes dades inicials (d'entrada) per aconseguir el resultat esperat (una eixida concreta). En canvi, un **procés** és **dinàmic**, és una instància d'un programa en execució, que realitza els canvis indicats pel programa a les dades inicials i obté una eixida concreta. El procés, a més de les instruccions, requerirà també de **recursos** específics per a l'execució com ara el comptador d'instruccions del programa, el contingut dels registres o les dades.

El **sistema operatiu** és l'encarregat de la **gestió de processos**. Els crea, els elimina i els proveeix d'instruments que en permeten l'execució i també la comunicació entre ells. Quan s'executa un **programa**, el sistema operatiu crea una **instància** del programa: el **procés**. Si el programa es tornara a executar es crearia una nova instància totalment independent a l'anterior (un nou procés) amb les seues pròpies variables, piles i registres.

Imaginem un servidor d'aplicacions en el qual tenim instal·lat un programa d'edició de text. Posem per cas que existeixen diversos usuaris que volen escriure els seus textos executant l'editor. Cada instància del programa és un procés totalment independent als altres. Cada procés té unes dades d'entrada i per tant diferents eixides. Cada usuari escriurà a la seua execució de l'editor (el procés), el seu text. A la figura.1 veiem aquest exemple.

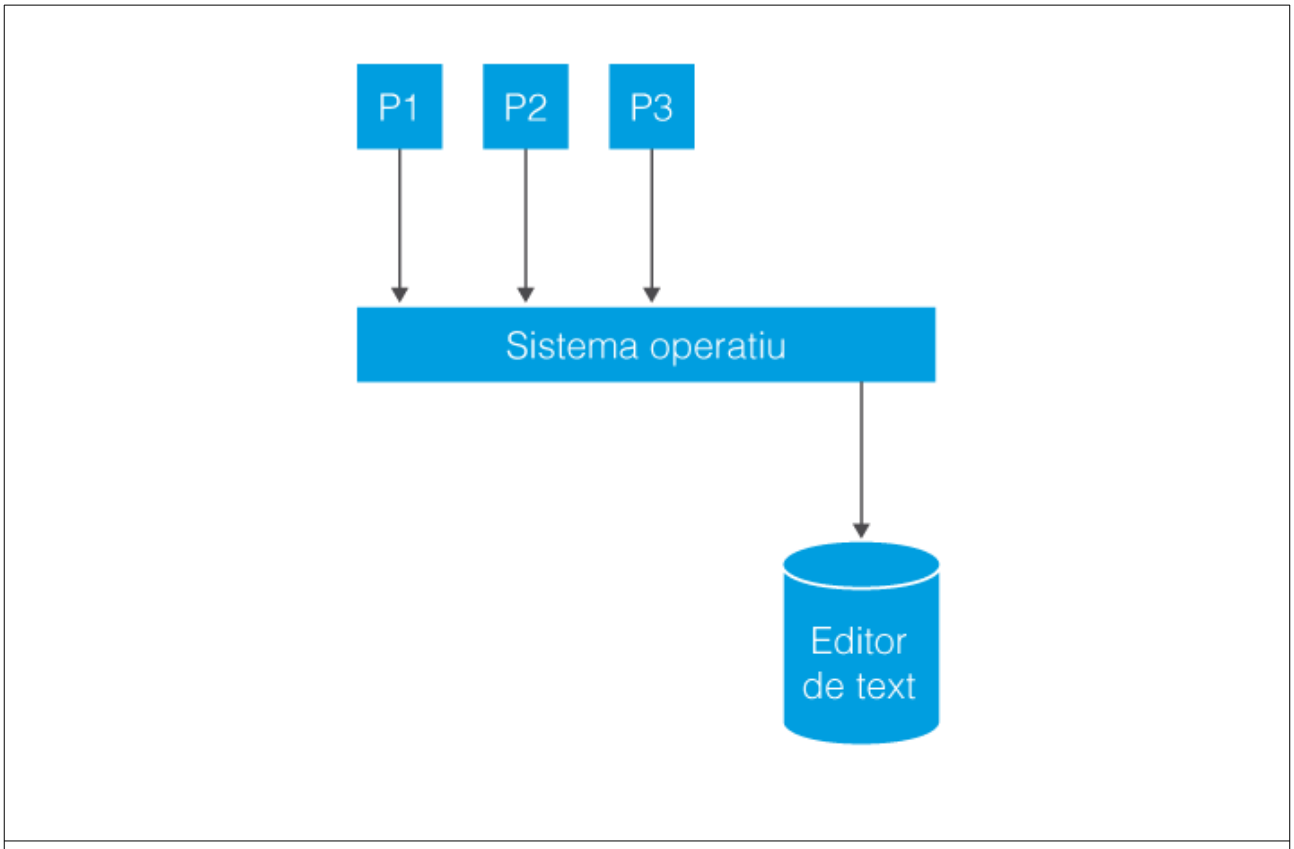

Figura 1. Execució de processos

Quan un **procés** es troba en **execució** es troba completament **en memòria** i té **assignats** els **recursos** que necessita. Un procés no pot escriure en zones de memòria assignada a altres processos, la memòria no és compartida. Cada **procés** té una estructura de dades en la qual es guarda la **informació associada** a l'**execució** del procés. Aquesta zona s'anomena **Bloc de Control de Procés** (BCP). Els processos competeixen amb altres processos executats a la vegada pels recursos del sistema. Opcionalment el sistema operatiu permet també que els processos puguen **comunicar-se** i **col·laborar** entre ells.

Fins ara hem parlat d'elements de programari, com ara programes i processos, però no del maquinari utilitzat per executar-los. L'element de maquinari en el qual s'**executen** els **processos** és el **processador**. Un processador és el component de maquinari d'un sistema informàtic encarregat d'**executar les instruccions** i **processar** les **dades**. Quan un dispositiu informàtic està format per un únic processador parlarem de

sistemes **monoprocessador**, per contra, si està format per més d'un processador parlarem de sistemes **multiprocessador**.

- Un **sistema monoprocessador** és aquell que està format únicament per un processador.
- Un **sistema multiprocessador** està format per més d'un processador.

Placa Base Multiprocessador

Actualment, la majoria dels **sistemes operatius aprofiten** els **temps** de **repòs** dels **processos**, quan esperen, per exemple, alguna dada de l'usuari o es troben pendents d'alguna operació d'entrada/eixida, per **introduir** al processador un **altre procés**, **simulant** així una **execució paral·lela**.

De forma genèrica anomenarem els processos que s'executen a la vegada, ja siga

de forma real o simulada, **processos concurrents**. L'execució de processos concurrents, encara que siga de forma simulada, fa **augmentar** el **rendiment** del sistema informàtic ja que aprofita més el temps del processador. És el sistema operatiu l'encarregat de gestionar l'execució concurrent de diferents **processos** contra un **mateix processador** i sovint s'anomena **multiprogramació**.

■ Parlem de **multiprogramació** quan el sistema operatiu gestiona l'execució de processos concurrents a un sistema monoprocessador.

La figura.2 correspon a una imatge de processos concurrents en la qual hi ha tres processos en execució i es van intercalant repartint-se el temps del processador.

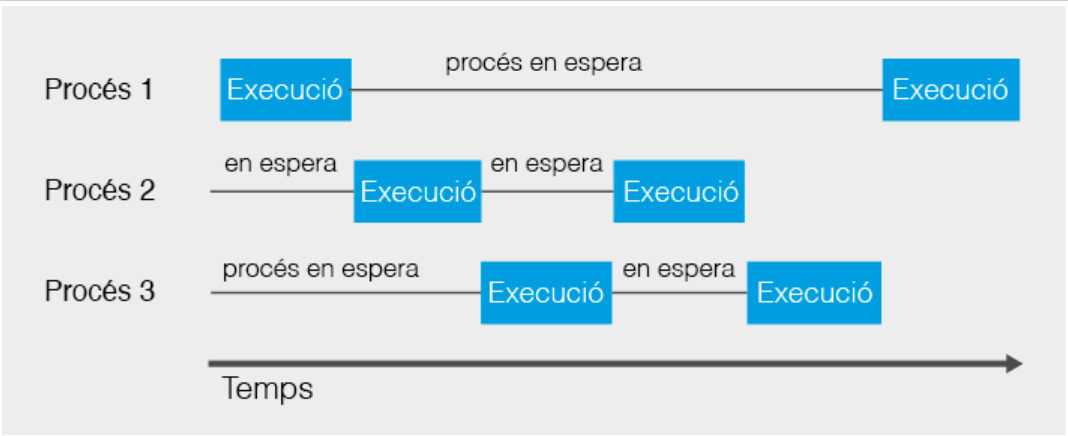

Figura 2. Execució de processos concurrents

Processador Intel amb dos nuclis

Els sistemes informàtics monoprocessadors actuals tenen una gran velocitat i si han d'executar més d'un procés a la vegada aniran alternant la seua execució simulant un multiprocés. Quan en un ordinador monoprocessador s'apliquen tècniques de multiprogramació els beneficis en rendiment no són tan evidents com en sistemes informàtics multiprocessador.

En un sistema informàtic **multiprocessador** existeixen **dos o més processadors**, per tant es poden executar simultàniament diversos processos. També es poden qualificar de multiprocessadors aquells dispositius el **processador** dels quals té **més d'un nucli** (**multicore**) , és a dir, tenen **més d'una CPU al mateix circuit integrat del processador**. Evidentment l'arquitectura és diferent, però aquest sistema, de la mateixa manera que un equip multiprocessador, ens permet **executar** de manera **simultània processos diferents**.

Els sistemes multiprocessadors es poden classificar depenent de la seua arquitectura en sistemes multiprocessador fortament acoblats i sistemes multiprocessador dèbilment acoblats.

Sistemes multiprocessadors fortament acoblats: en aquesta arquitectura els diferents processadors **comparteixen una mateixa memòria** i estan interconnectats a ella a través d'un **bus**. També poden tenir una xicoteta memòria cau (***caché***) a cada processador. Hi ha una forta **col·laboració** entre processadors **compartint variables** a la **memòria comuna** a mode de resultats parcials o també per comunicar-se entre ells. Actualment, la majoria de sistemes operatius suporten aquest tipus de sistemes multiprocessador.

Els sistemes fortament acoblats depenen del pes que té cada processador a l'hora d'executar els processos. Es poden dividir en **sistemes multiprocés simètrics**, en els quals els processadors del sistema són de característiques similars i competeixen entre iguals per executar processos, i **sistemes multiprocés asimètrics** en els quals un dels processadors del sistema, el màster, controla la resta de processadors. Aquest sistema també s'anomena **master-slave**.

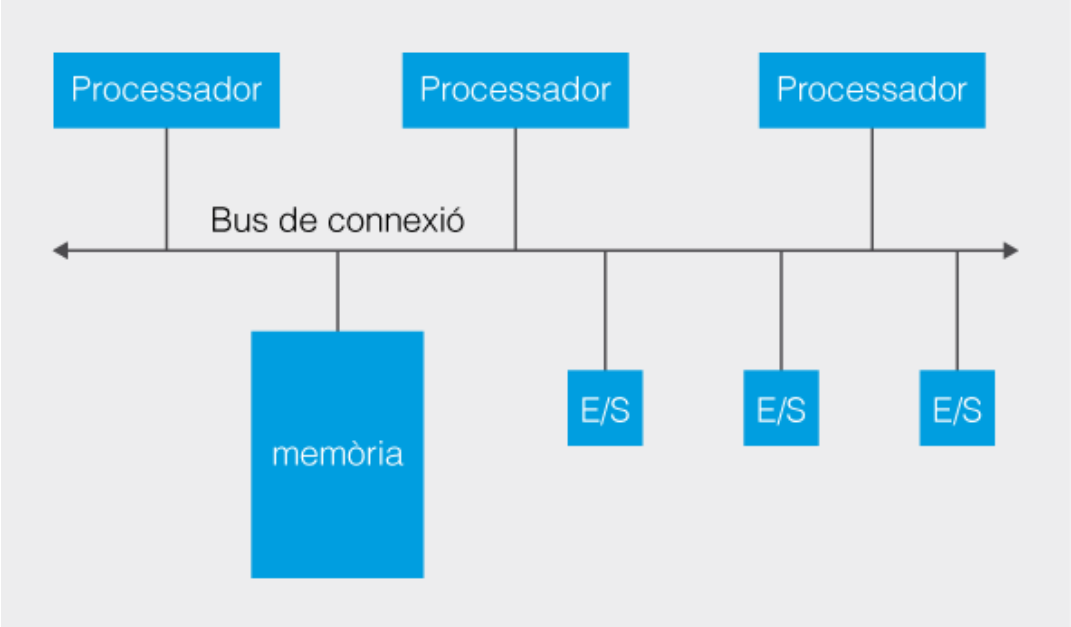

Figura 3. Sistema informàtic multiprocessador fortament acoblat

Sistemes multiprocessadors dèbilment acoblats: aquests sistemes **no comparteixen memòria**, **cada processador** té una **memòria associada**. A les execucions que necessiten **col·laboració** entre processos els cal l'**intercanvi** de **missatges** a través d'enllaços de **comunicacions**, per habilitar la comunicació entre processos. Un tipus d'aquests sistemes poc acoblats són els **sistemes distribuïts**, en els quals cada dispositiu monoprocessador (o multiprocessador) podria estar situat a **llocs físicament distants**. Una xarxa d'ordinadors o Internet podria ser un exemple d'un sistema dèbilment acoblat distribuït.

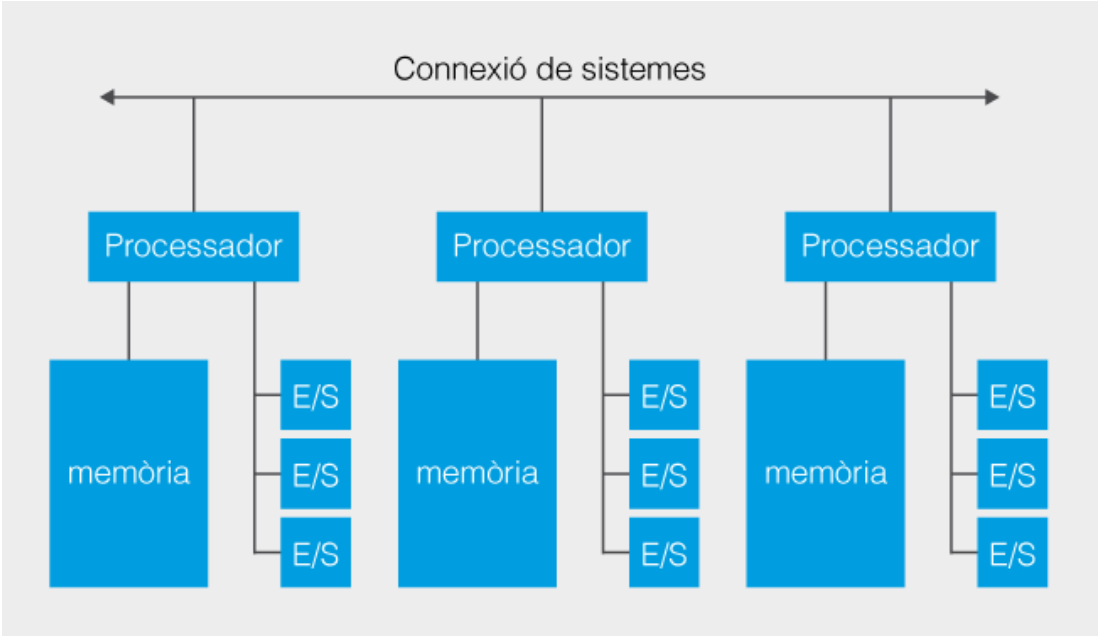

Figura 4. Sistema informàtic multiprocessador dèbilment acoblat

1.1.2. Programació concurrent, paral·lela i distribuïda

Simula, creat al 1966 va ser el primer llenguatge de programació orientat a objectes concurrent.

Actualment, una gran quantitat d'aplicacions utilitzen la programació concurrent. En els sistemes operatius actuals existeixen moltes **aplicacions executant-se al mateix temps i compartint informació**. Per exemple, podem escriure en un editor de text, tenir posat un reproductor de música i descarregar un vídeo al mateix temps. També podem tenir en execució un navegador capaç de carregar en diverses pestanyes diferents pàgines web. Són exemples que il·lustren la idea de programació concurrent.

Deep Blue. Supercomputador d'IBM amb 256 processadors treballant en paral·lel

Gràcies a l'evolució que ha sofert el maquinari durant els últims anys s'han creat **sistemes operatius** que poden **optimitzar** els **recursos** dels processadors i poden **executar** diferents **processos** de forma **simultània**. L'execució simultània de processos s'anomena també **concurrència**. No vol dir que s'hagen d'executar exactament al mateix moment, l'intercalat de processos també es considera execució concurrent. La majoria dels llenguatges de programació actuals poden fer ús d'aquest recurs i dissenyar aplicacions en les quals els processos s'executen de manera concurrent.

■ Parlem de **programació concurrent** quan s'executen en un dispositiu informàtic de forma simultània diferents tasques (processos).

L'execució concurrent pot comportar certs **problemes** a l'hora d'**accedir** a les **dades** que no trobarem mai a l'execució seqüencial i que caldrà tenir en compte. Bàsicament usarem dues tècniques **per evitar**-los: el **boqueig** i la **comunicació**.

Si la programació concurrent es dóna en un computador amb un **únic processador** parlarem de **multiprogramació**. En canvi, si el computador té **més d'un processador** i, per tant, els processos es poden executar de forma realment simultània, parlarem de

programació paral·lela.

En un dispositiu **multiprocessador** la **concurrència** és **real**, els processos són executats de forma simultània en diferents processadors del sistema. És habitual que el número de processos que s'estiga executant de forma concurrent siga major que el número de processadors. Per tant serà obligatori que alguns processos s'executen sobre el mateix processador.

■ Quan la programació concurrent es realitza en un sistema multiprocessador parlem de **programació paral·lela**.

En els **ordinadors multiprocessadors** la concurrència és real, per tant el **temps d'execució** acostuma a ser **menor** que en **dispositius monoprocessadors**.

A la figura.5 veiem un esquema del resultat d'una execució de tres processos en paral·lel en un ordinador multiprocessador amb tres processadors. En contraposició tenim la figura.2 amb l'execució en un ordinador monoprocessador.

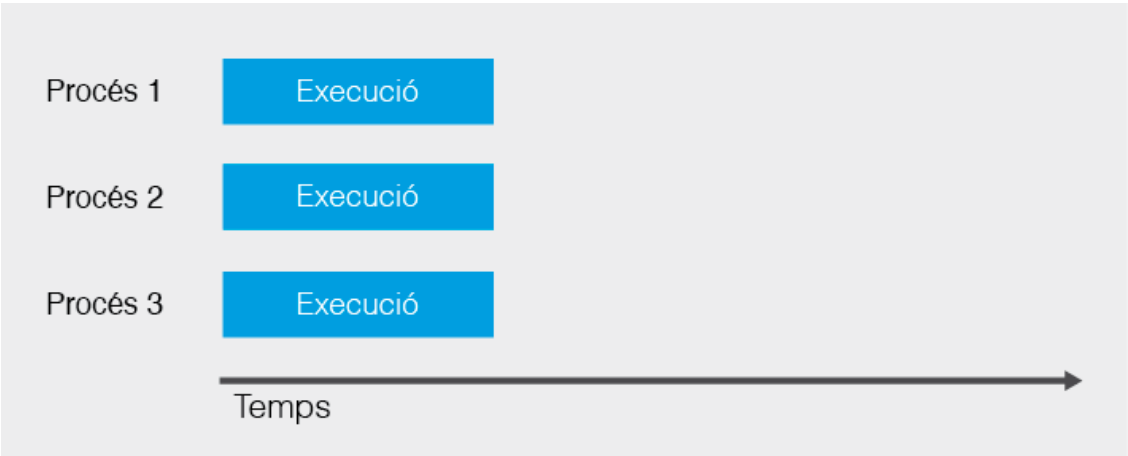

Figura 5. Execució en paral·lel

El principal **desavantatge** de la **programació paral·lela** són els **controls** que hem d'afegir per tal que la **comunicació** i **sincronització** dels **processos** que s'executen

concurrentment siguen correctes. Dos processos s'han de sincronitzar o comunicar quan un **procés necessita** alguna **dada** que està processant l'altre o bé un d'ells ha d'esperar la **finalització** de l'**altre** per **poder continuar** la seua execució.

El llenguatge de programació **Java** té integrat el suport a la concurrència en el propi llenguatge i no a través de llibreries.

La programació paral·lela està molt lligada a l'arquitectura del sistema de computació. Però quan programem en llenguatges d'alt nivell ens hem d'abstreure de la plataforma en la qual s'executarà. Els llenguatges d'alt nivell ens proveeixen de **llibreries** que ens faciliten aquesta abstracció i ens permeten utilitzar les mateixes operacions per **programar** de **forma paral·lela** que acabaran implementades per usar-se un una plataforma concreta.

Un tipus especial de programació paral·lela és l'anomenada **programació distribuïda**. Aquesta programació es dóna en **sistemes informàtics distribuïts**. Un sistema distribuït està format per un conjunt d'ordinadors que poden estar situats en llocs geogràfics diferents units entre ells a través d'una xarxa de comunicacions. Un exemple de sistema distribuït pot ser el d'un banc amb moltes oficines al món, amb un ordinador central per oficina per guardar els comptes locals i fer les transaccions locals. Aquest ordinador es pot comunicar amb els altres ordinadors centrals de la xarxa d'oficines. Quan es fa una transacció no importa on es troba la compte o el client.

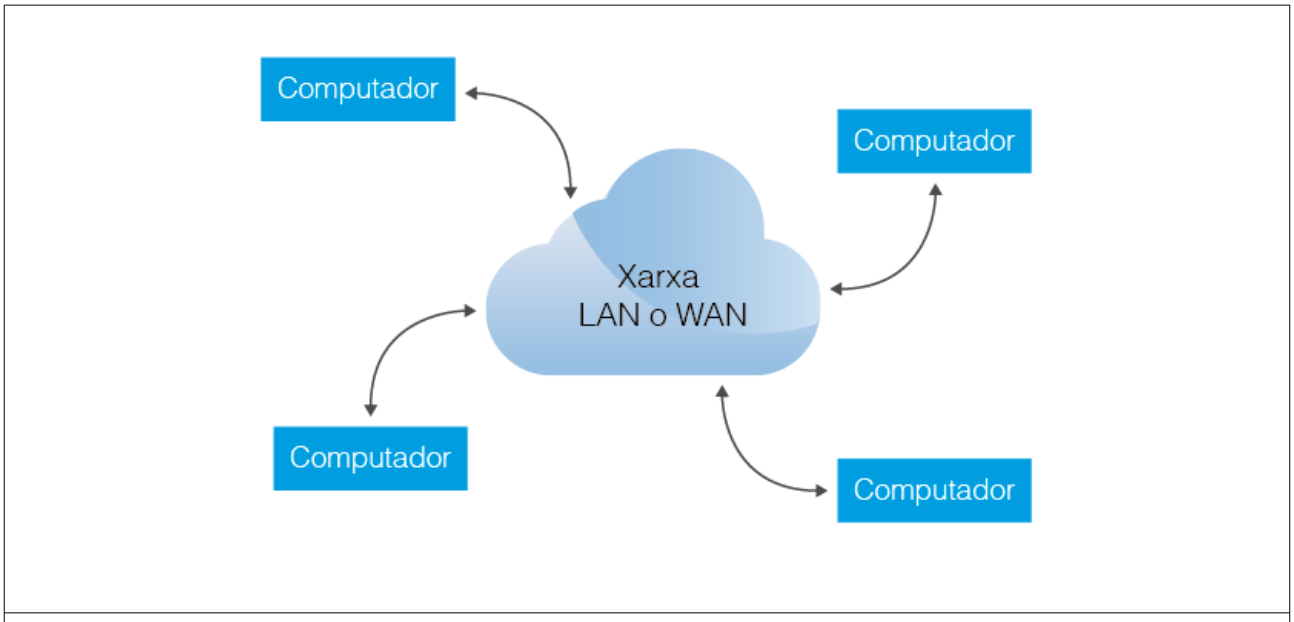

Figura 6. Execució de programació distribuïda

■ La **programació distribuïda** és un tipus de programació concurrent en la qual els processos són executats en una xarxa de processadors autònoms o en un sistema distribuït. És un sistema de computadores independents que des del **punt de vista de l'usuari** del sistema es veu com **una sola computadora**.

En els **sistemes distribuïts**, a diferència dels sistemes paral·lels, **no existeix memòria compartida**, per tant si necessiten la utilització de **variables compartides** es crearan rèpliques d'aquestes variables als diferents **dispositius** de la **xarxa** i caldrà **controlar la coherència de les dades**.

La **comunicació** entre **processos** per intercanviar dades o sincronitzar-se entre ells es fa amb **missatges** que s'envien a través de la **xarxa** de **comunicacions** que comparteixen.

Els principals **avantatges** són que es tracta de sistemes altament **escalables** i per tant **reconfigurables** fàcilment i d'**alta disponibilitat**. Com a **desavantatges** més importants, trobarem que són sistemes **heterogenis**, **complexos** de sincronitzar. Les comunicacions es fan a través de missatges que utilitzen la **xarxa** de **comunicació**

compartida i això pot provocar-ne la **saturació**.

1.1.3. Avantatges i desavantatges del multiprocés

Hem definit **programació concurrent** o **concurrència** com la tècnica per la qual múltiples processos s'executen alhora i poden comunicar-se entre ells. La majoria dels sistemes intenten aprofitar aquesta concurrència per incrementar la capacitat d'execució.

Incrementar la potència de càlcul i el rendiment és un dels principals avantatges. Quan s'executen diversos processos a l'hora, la velocitat d'execució global es pot incrementar. Cal tenir en compte, però, que no sempre és així. A vegades, depenent de la complexitat de l'aplicació, les tècniques de sincronisme o comunicació són més costoses en temps que l'execució dels processos.

Un sistema multiprocessador és **flexible**, ja que, si augmenten el processos que s'estan executant és capaç de distribuir la càrrega de treball dels processadors, i pot també reassignar dinàmicament els recursos de memòria i els dispositius per ser més eficients. És de **fàcil creixement**. Si el sistema ho permet, es poden afegir nous processadors de forma senzilla i així augmenten la seua potència.

Els sistemes multiprocessador poden ser **sistemes redundants**. El fet de disposar de diversos processadors, pot permetre un augment de la disponibilitat dels recursos (més processadors al servei de l'usuari) o bé l'ús de processadors especialitzats en tasques de verificació i control. En el darrer cas parlarem de sistemes amb una **alta tolerància a fallades**. Una fallada d'un processador no fa que el sistema es pare.

Els sistemes multiprocés ens permeten diferenciar processos per la seua **especialització** i, per tant, reservar processadors per operacions complexes, aprofitar-ne d'altres per processaments paral·lels i per avançar l'execució.

Els **inconvenients** del multiprocessament vénen provocats sobretot pel **control**

que s'ha de realitzar quan hi ha diversos processos en execució i han de **compartir informació** o **comunicar-se**. Això **incrementa la complexitat de la programació** i penalitza el temps d'execució.

Si el multiprocessament és en un entorn multiprocessador paral·lel o distribuït, el **trànsit** dels **busos** de **comunicació** s'incrementa de manera paral·lela al número de processadors o computadors que incorporem al sistema, i això pot arribar a ser un crític **coll d'ampolla**.

1.2 Processos i serveis

La majoria de nosaltres tenim a l'ordinador un antivirus instal·lat. Quan posem en marxa el nostre ordinador no arranquem l'antivirus i no interactuem amb ell a no ser que trobe un virus i ens avise. A l'iniciar el sistema, el programa d'antivirus s'executa. Aleshores es crea un procés que es manté en execució fins que apaguem el nostre ordinador. Aquest procés que controla les infeccions de tots els fitxers que entren al nostre ordinador és un servei.

- Un **servei** és un tipus de procés que **no té interfície amb l'usuari**. Poden ser, depenent de la seua configuració, inicialitzats pel sistema de forma automàtica, en el qual van realitzant les seues funcions sense que l'usuari se n'assabente o bé es poden mantenir a l'espera que algú els faça una petició per fer una tasca en concret.

Depenent de com s'estan executant, els **processos** es classificaran com a processos en **primer pla** (**foreground**, en anglès) i processos en **segon pla** (**background**). Els processos en **primer pla** mantenen una **comunicació** amb l'**usuari**, ja siga informant de les accions realitzades o esperant les seues peticions per mitjà d'alguna interfície d'usuari. En canvi, els que s'executen en **segon pla no** es **mostren explícitament** a l'**usuari**, bé perquè no els cal la seua intervenció o bé perquè les dades

requerides no s'incorporen a través d'una interfície d'usuari.

Els **serveis** són processos **executats** en **segon pla**. **Servei** és una nomenclatura utilitzada en **Windows**. En sistemes **Linux** s'anomenen també processos **deamon**.

El nom **deamon** prové de les sigles angleses DAEMON (Disk And Execution Monitor). Sovint a la literatura tècnica s'ha estès aquest concepte usant la paraula "**dimoni**", pel que podeu trobar aquesta paraula fent referència als programes que s'executen en segon pla.

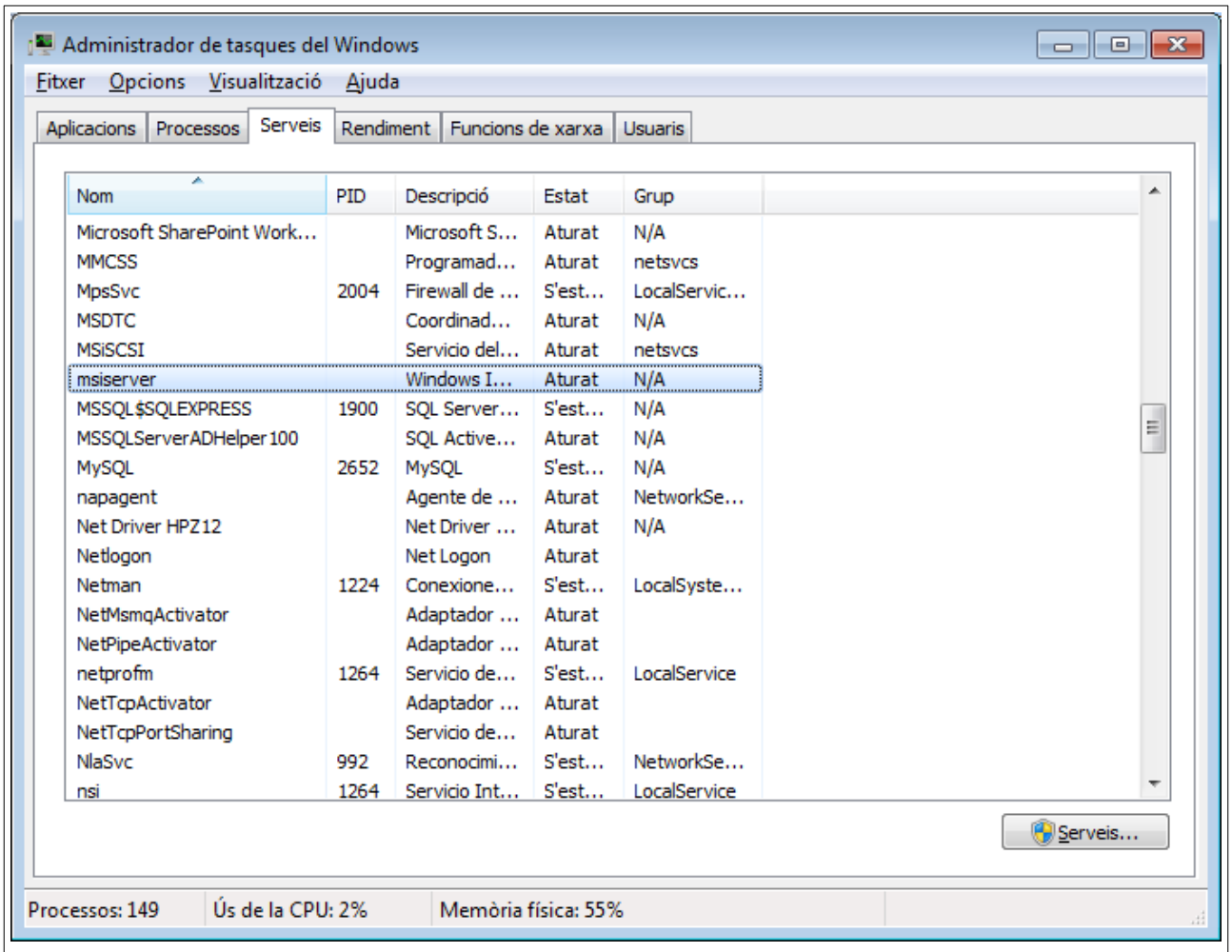

Figura 7. Serveis Windows

Com que els **serveis** no disposen d'interfície d'usuari directa, emmagatzemen la **informació** generada o les possibles errades que vagen produint, en fitxers de registre habitualment coneguts com a **logs**.

A Linux el nom dels **daemons** sempre acaben amb la lletra d, com ara httpd, el daemon de l'http.

Alguns **serveis** poden estar a l'**espera** de ser **cridats** per realitzar les seues funcions. Per exemple, un **servidor web** té un servei actiu. A Linux és un httpd el que està escoltant un port de la xarxa i quan l'usuari fa una petició a través de la xarxa per obtenir una pàgina web, és el dimoni el que s'encarrega de processar la petició i enviar la pàgina de retorn, si té accés autoritzat i la pàgina existeix, o bé fer arribar a través de la xarxa el missatge informatiu indicant la impossibilitat del lliurament.

Els **serveis** poden estar en **execució local**, al nostre ordinador, com ara el Firewall, l'antivirus, la connexió Wi-Fi, o els processos que controlen el nostre ordinador, o bé poden **executar-se** en algun **ordinador** d'un **sistema** informàtic **distribuït** o a Internet, etc. Les crides als serveis distribuïts es realitzen **sobre protocols** de **comunicació**. Per exemple, els serveis d'http, DNS (Domain Name System), FTP (File Transfer Protocol) són serveis que ens ofereixen servidors que són a Internet.

1.2.1. Fils i processos

Recordem el cuiner que treballa simultàniament preparant un plat per diferents comensals. El processador d'un sistema informàtic (cuiner) està executant diferents instàncies del mateix programa (la recepta). A banda, el cuiner va fent diferents parts del plat. Si la recepta és llonganissa amb ous fregits, dividirà la seua tasca en preparar la llonganissa per una banda i els ous fregits per l'altra. Ha dividit el procés en dos subprocessos. Cada un d'aquests subprocessos s'anomenen fils.

El sistema operatiu pot mantenir en execució diversos processos a la vegada fent servir concurrència o paral·lelisme. A més, dins de **cada procés** poden **executar-se diversos fils**, de manera que blocs d'instruccions que presenten certa independència puguen executar-se a la vegada. Els **fils comparteixen els recursos del procés** (dades, codi, memòria, etc). En conseqüència, els fils, a diferència dels processos, comparteixen memòria i si un fil modifica una variable del procés, la resta de fils podran veure la modificació quan accedisquen a la variable.

Els **processos** són anomenats **entitats pesades** perquè estan a **espais d'adreçament** de **memòria independents**, de creació i de comunicació entre processos, cosa que consumeix molts recursos de processador. En canvi, ni la creació de fils ni la comunicació consumeixen molts temps de processador. Per aquest motiu els **fils** s'anomenen **entitats lleugeres**.

Un procés estarà en execució mentre algun dels seus fils estiga actiu. Tot i així, també és cert que si finalitzem un procés de forma forçada, els seus fils també acabaran l'execució.

Podem parlar de dos nivells de fils: els **fils de nivell d'usuari**, que són els que creem quan **programem** amb un llenguatge de programació com ara **Java**; i els **fils de segon nivell** que són els que **crea el sistema operatiu** per donar **suport als primers**. Nosaltres programarem els nostres fils utilitzant unes llibreries que ens proporciona el llenguatge de programació. Són les llibreries amb les instruccions pertinents que indicaran al sistema els fils que han de crear i com gestionar-los.

A la figura.8 es mostren dos processos un amb l'execució d'un fil i l'altre amb l'execució de tres fils concurrents.

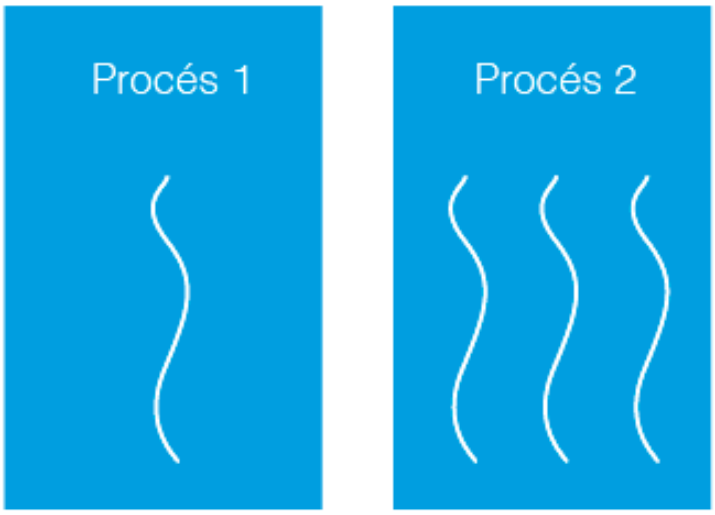

Figura 8. Procés amb un únic fil en execució i procés amb diversos fils en execució

La figura.9 il·lustra el tractament que el sistema operatiu fa dels fils. En referència al processament, **cada fil es processa de forma independent** encara que pertanya o no a un mateix procés. L'única **diferència** entre el tractament de **fils** i **processos** la trobem a nivell de **memòria**. Per a **fils d'un mateix procés** el sistema operatiu ha de mantenir les **mateixes dades** en **memòria**. Per a **fils de diferents processos** en canvi cal disposar de **dades independents**. En el cas que siga necessari que un mateix processador alterne l'execució de diversos processos, caldrà restaurar la memòria específica del procés en cada canvi. D'això s'anomena també **canvi de context**. Si l'alternança de processament es fa entre fils d'un mateix procés, no caldrà restaurar la memòria i per tant el procés serà molt més eficient.

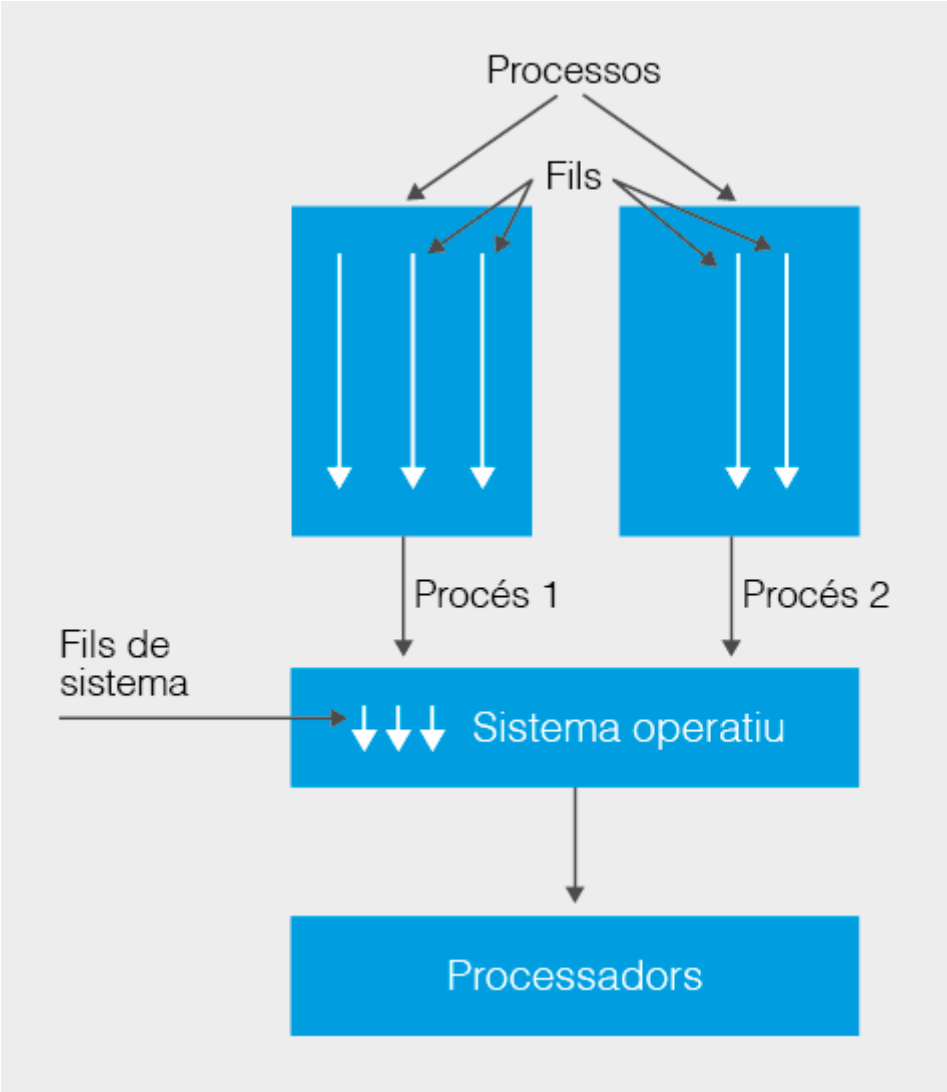

Figura 9. Dos processos amb diferents fils en execució. Cada fil s'executa de forma independent. Els fils del mateix procés comparteixen memòria.