Software Distribuït - T3 - Antecedents: Sistemes Operatius

Eloi Puertas i Prats

Universitat de Barcelona
Grau en Enginyeria Informàtica

14 de febrer de 2024

Aplicacions Distribuïdes

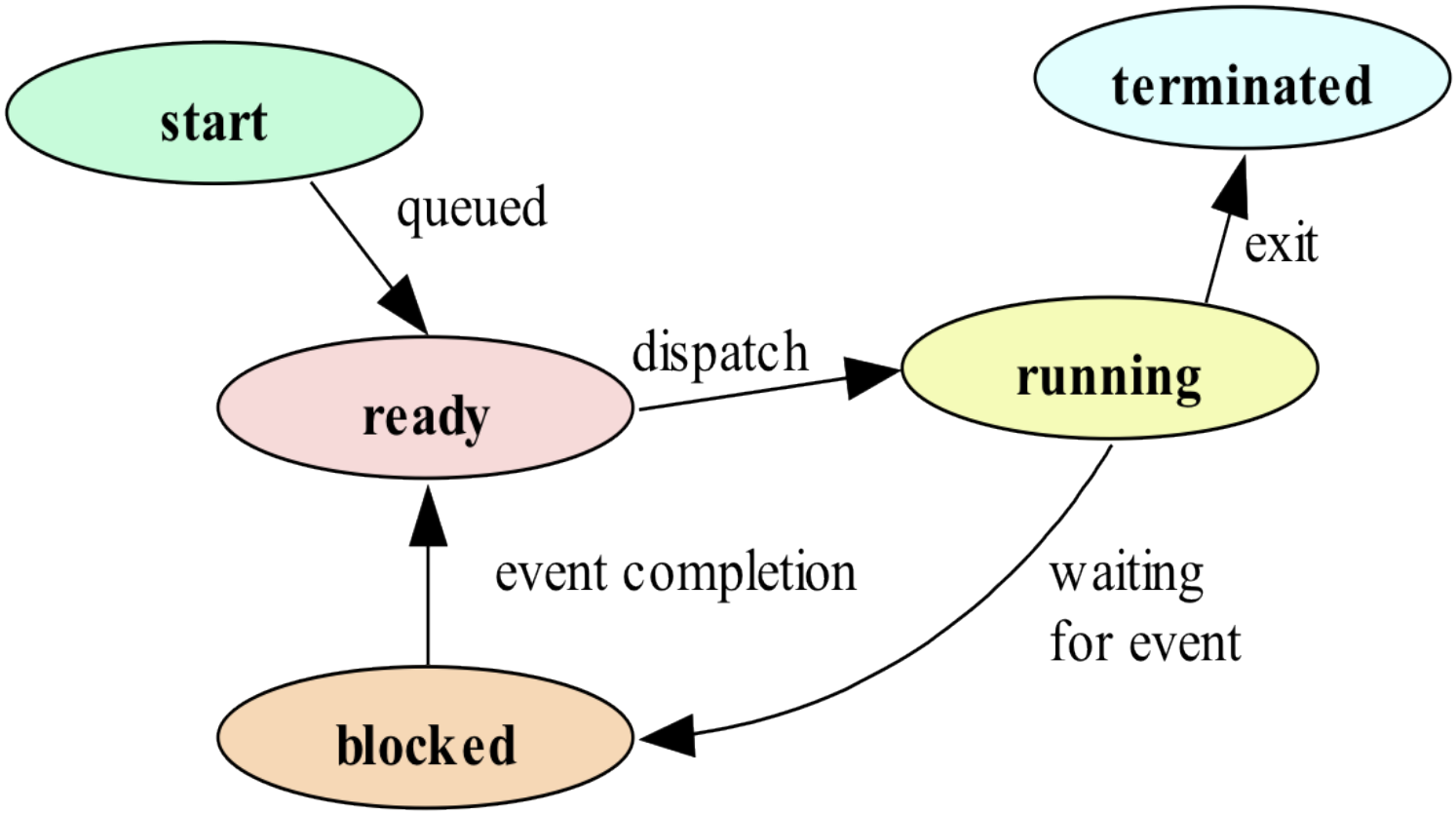
Característiques que diferencien les aplicacions distribuïdes de les convencionals

1. **Comunicació Intaprocessos:** Una aplicació distribuïda necessita la participació de 2 o més entitats independents (processos). Per tal de poder fer això, els processos han de tenir la capacitat d'intercanviar-se dades entre ells.
2. **Sincronització d'esdeveniments:** En una aplicació distribuïda, l'enviament i la recepció de les dades entre el particpants ha de ser sincronitzada.

Definicions bàsiques de S.O.

- **procés:** és un programa en execució, amb els valors actuals, les seves variables d'estat, i tots els recursos usats pel S.O. per gestionar la seva execució. És una entitat dinàmica que només existeix quan el programa està executant-se.
- **programa:** És el conjunt d'ordres per resoldre un problema o funció específica, basades en un llenguatge de programació.

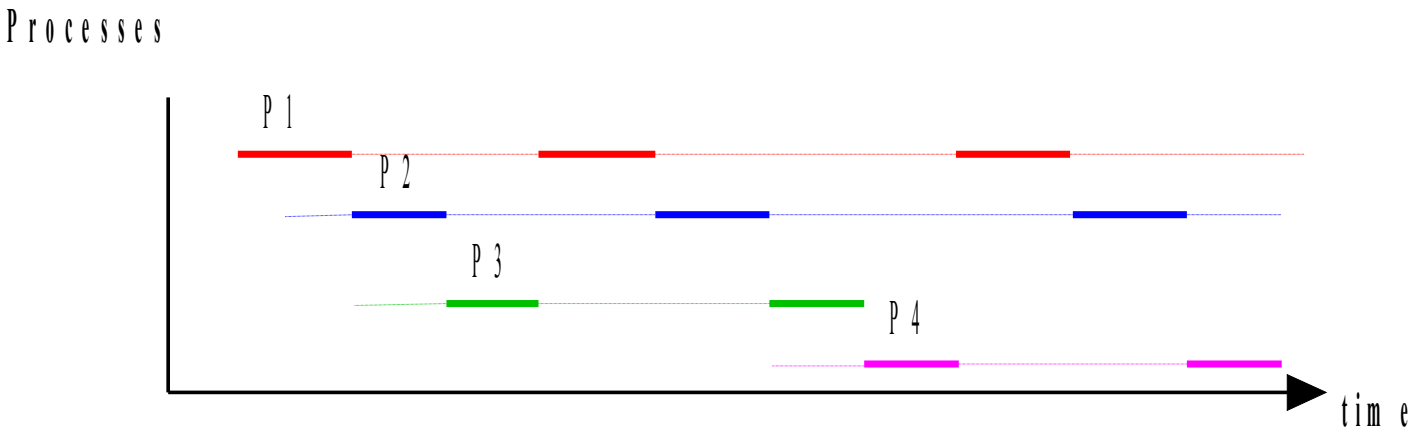
Diagrama de transició d'estats d'un procés



Simplifed finite state diagram for a process's lifetime

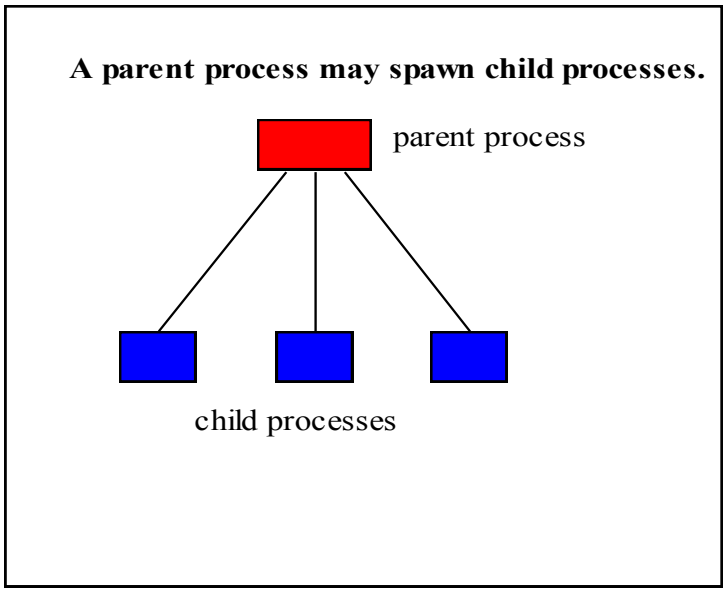
Processos concurrents

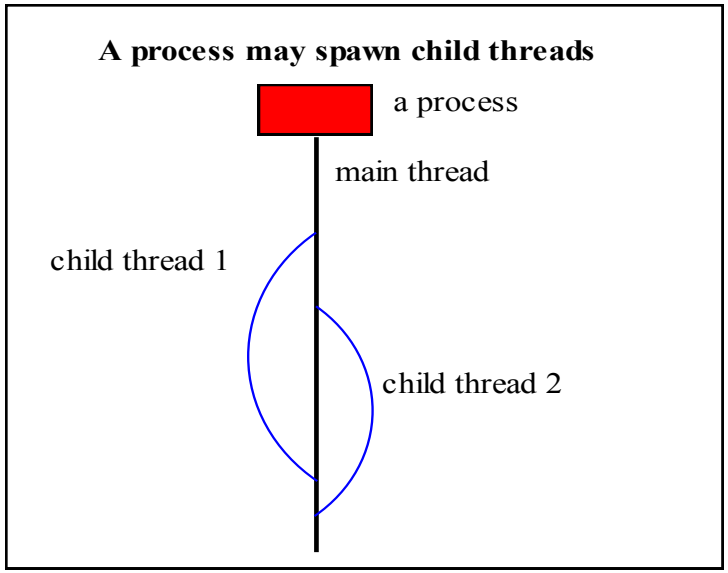
Els Sistemes Operatius d'avui en dia, poden fer com si múltiples processos s'executin concurrentment en una sola CPU, fent que els recursos siguin de temps compartit entre els processos.



Fils d'execució concurrents dins d'un procés. Threads.

Un procés pot tenir fils d'execució paral.lels (Threads). Tots els fils estan compartint els mateixos recursos dins del procés.





Concurrent processing within a process

Comunicació Intra-processos

- Les aplicacions distribuïdes necessiten intercanviar informació entre processos independents sovint de màquines diferents.
- Els sistemes operatius proveeixen eines per a la comunicació intra-processos (IPC), com els **sockets**, cues de missatges, semàfors, i memòria compartida.
- Les aplicacions distribuïdes solen fer servir APIs que utilitzen les eines d'IPC del S.O.

Funcions bàsiques en una API estàndard d'IPC

- Connect (adreça sender, adreça receiver), per comunicacions orientades a la connexió (TCP)
- Send ([receiver], missatge)
- Receive ([sender], buffer per guardar missatge)
- Disconnect (id connexió), per comunicacions orientades a la connexió (TCP)

Sincronització d'esdeveniments

- La comunicació entre processos requereix que els dos processos **sincronitzin les seves operacions**: d'una banda s'envia, i després l'altre rep fins que totes les dades han estat enviades i rebudes.
- Per anar bé, la recepció s'ha d'iniciar abans que comenci l'enviament.
- La definició d'un protocol serà necessari per a garantir la correcta sincronització

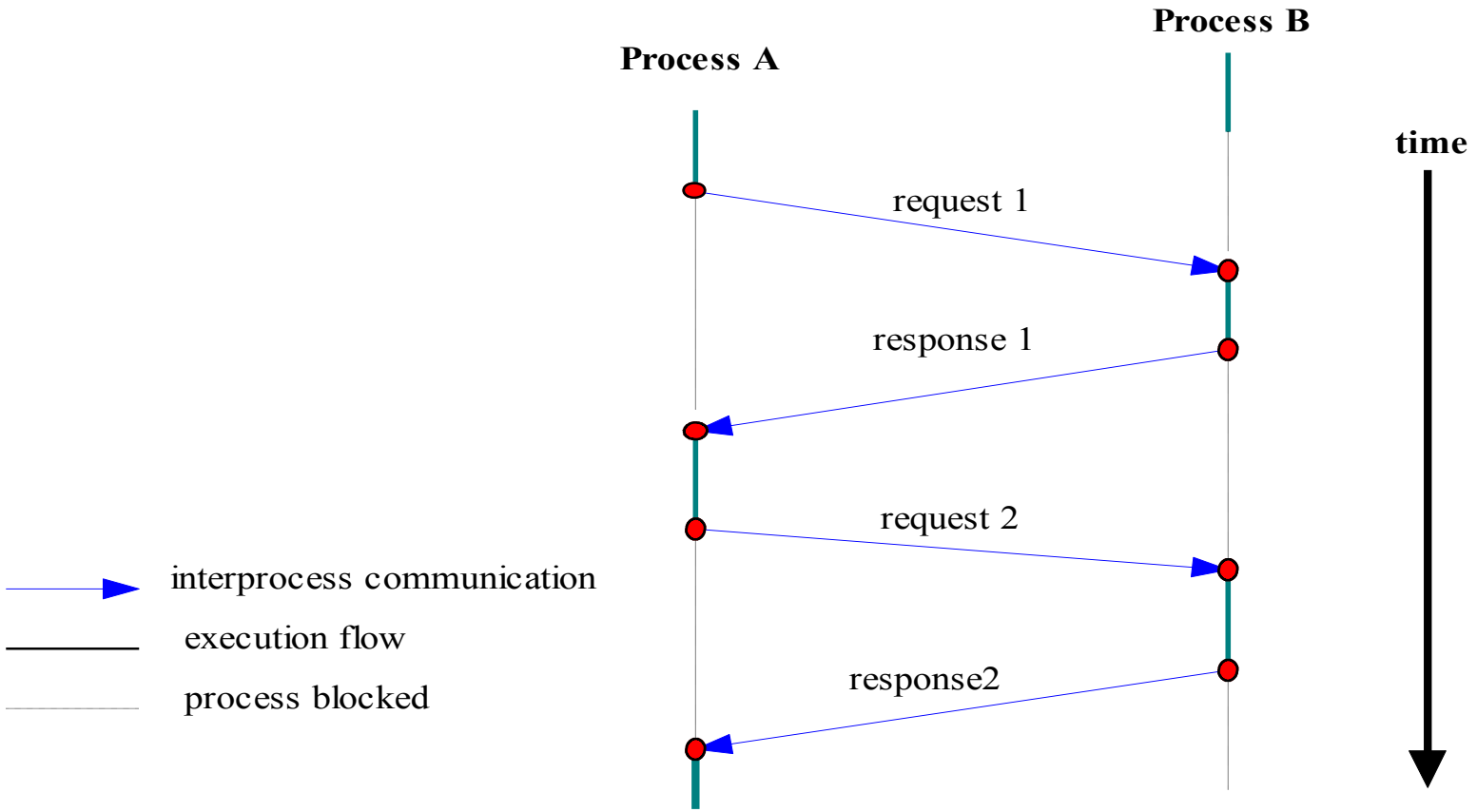
Protocol

En una aplicació distribuïda, dos processos han de comunicar-se basant-se en un protocol d'acord mutu.

L'especificació d'un protocol ha d'incloure:

1. la seqüència d'intercanvi de dades, que pot ser descrita mitjançant un diagrama d'esdeveniments.
   - les funcions d'IPC poden proporcionar la sincronització necessària fent servir operacions bloquejants. Una operació bloquejant llançada per un procés, el deixarà bloquejat fins que aquesta s'hagi completat.
2. l'especificació del format de les dades intercanviades en cada pas.

Diagrama d'esdeveniments



Event diagram for a protocol

Representació de les dades

- Les dades transmesos per a la xarxa són una seqüència binària.
- Un sistema de comunicació entre processos pot imposar la representació de les dades que s'enviaran per la xarxa.
- A causa que diferents equips poden tenir *diferents* formats d'emmagatzematge intern per al mateix tipus de dades, una representació externa de dades és aconsellable.

Es coneix com a *Data Marshalling* el procés :

1. d'allisar una estructura de dades.
2. convertir les dades a una representació exterior.

Representació de les dades

Quan es defineix l'iteracció entre client i servidor amb un protocol, com sap qui llegeix que qui escriu ha acabat el seu missatge?
Definició de les trames de missatges

1. **Missatges de longitud fixa.** Es decideix la longitud que tindrà el missatge, sempre la mateixa.
2. **Missatges amb finalitzador.** El missatge s'acaba quan es troba un símbol o patró concret.
3. **Missatges que codifiquen la seva longitud.** El missatge indica prèviament quina és la longitud del missatge.

És important codificar adequadament els diferents tipus de dades.
És còmode treballar amb cadenes de caràcters, (protocols basats en text)

Protocols basats en text

- En un protocol, quan les dades que s'intercanvien són una seqüència de caràcters, o de text, es té l'avantatge de que les dades poden ser fàcilment analitzades per les aplicacions i per a la lectura humana.
- És una pràctica comuna definir els missatges en forma de cadenes de caràcters (d'un o dos bytes). Aquests protocols se'n diuen basats en text.
- Molts del protocols de xarxa més populars, com FTP (File Transfer Protocol), HTTP i SMTP (Simple Mail Transfer Protocol), estan basats en text.
- A vegades però, es prefereix utilitzar representacions més compactes per reduir la mida de la trama del missatge.

Exercicis de Protocols

- Defineix un protocol per a enviar i rebre un enter per la xarxa.
- Defineix un protocol per a enviar i rebre un nombre en punt flotant per la xarxa.
- Defineix un protocol per a enviar un string per la xarxa.
- Defineix un protocol per a enviar un paquet format per 2 strings de 20 caràcters i dos enters llargs per la xarxa

Operacions Síncrones vs Asíncrones

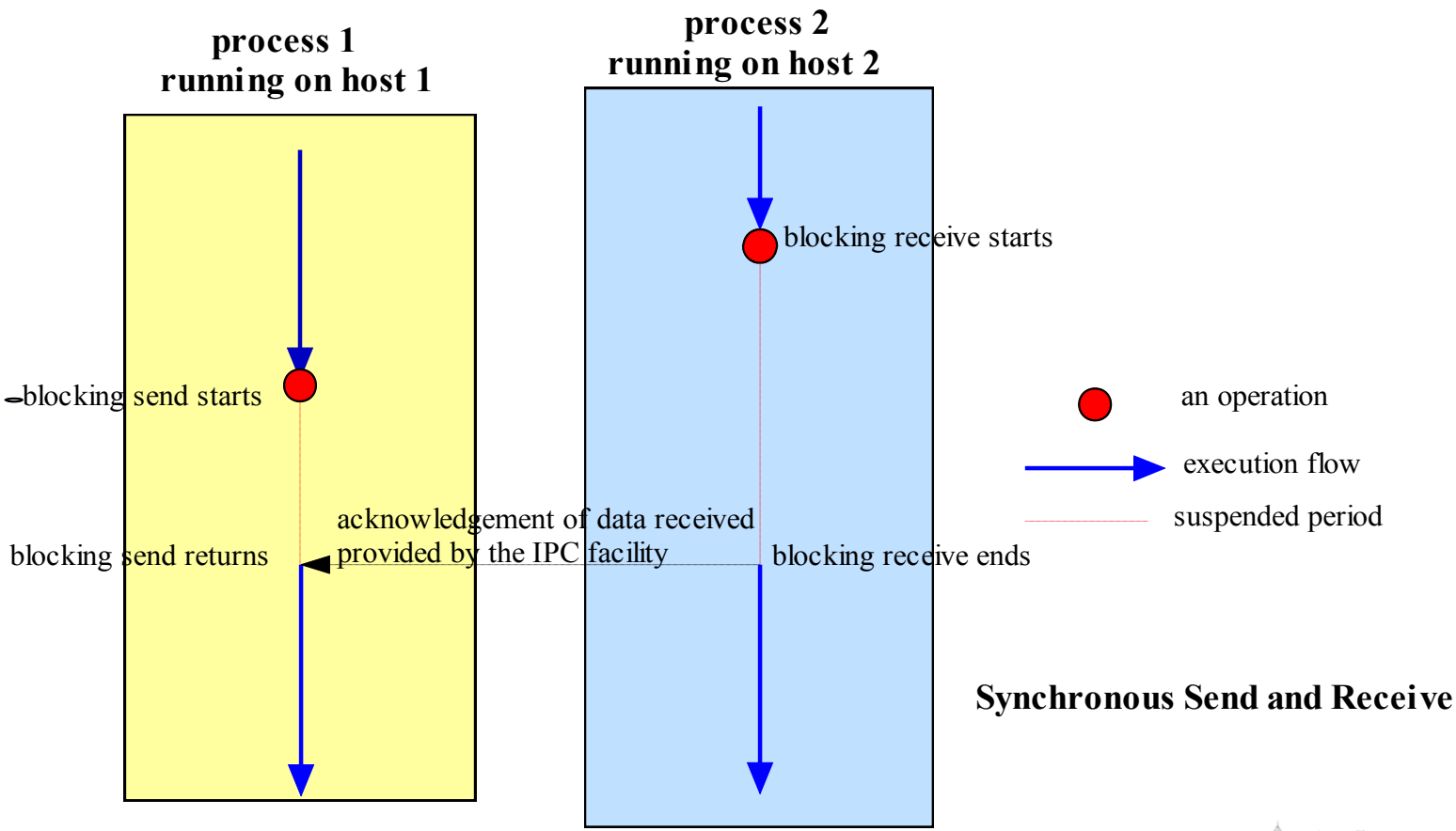
Les operacions d'IPC poden ser **síncrones** (amb bloqueig) o **asíncrones** (sense bloqueig).
Una operació **síncrona** emesa per un prcés el bloquejarà.

- Problema: El procés pot quedar bloquejat per temps indefinit.
- Avantatge: La sincronització d'operacions és senzilla.

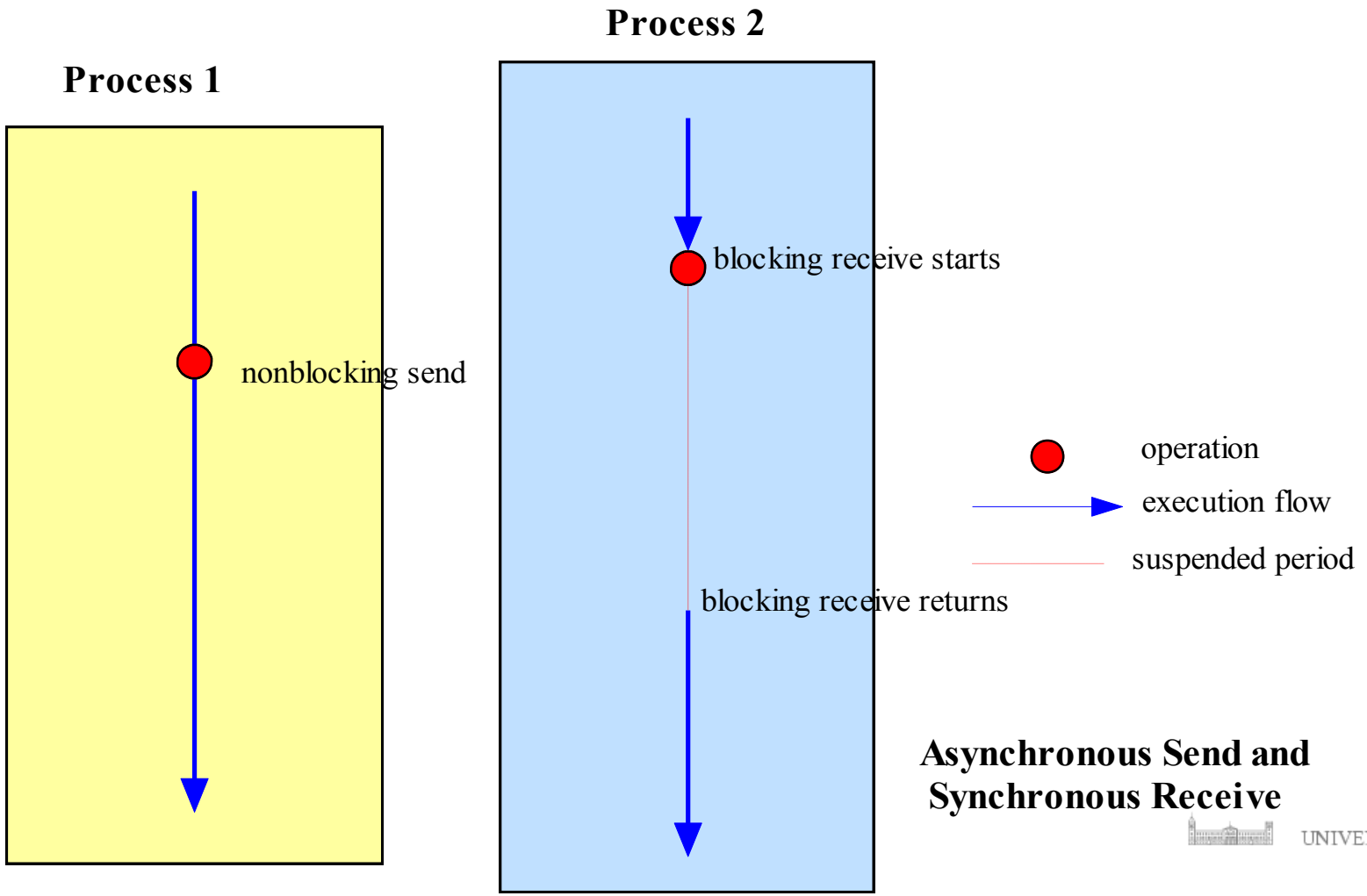
Una operació **asíncrona** emesa per un procés no el bloquejarà. El procés continuarà amb la seva execució i, opcionalment, pot ser notificat pel sistema quan l'operació s'hagi complert.

- Problema: s'ha de garantir que existeixi sincronització entre operacions, per tant si no es controla bé poden *ignorar-se* missatges.
- Avantatge: allibera el processador per fer altres tasques i no es queda bloquejat pendent de sincronitzar-se.

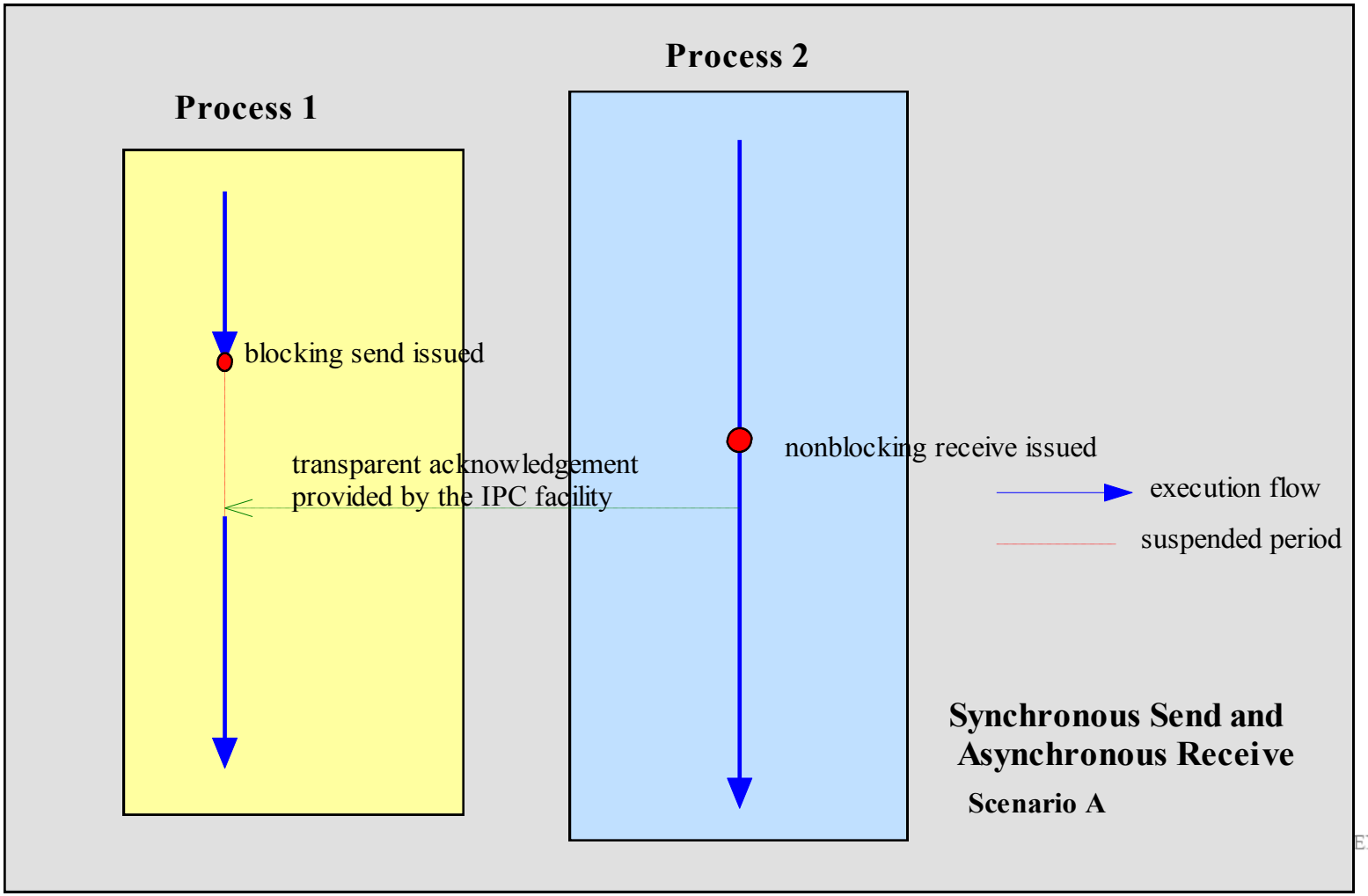
Enviament i Recepció síncrona



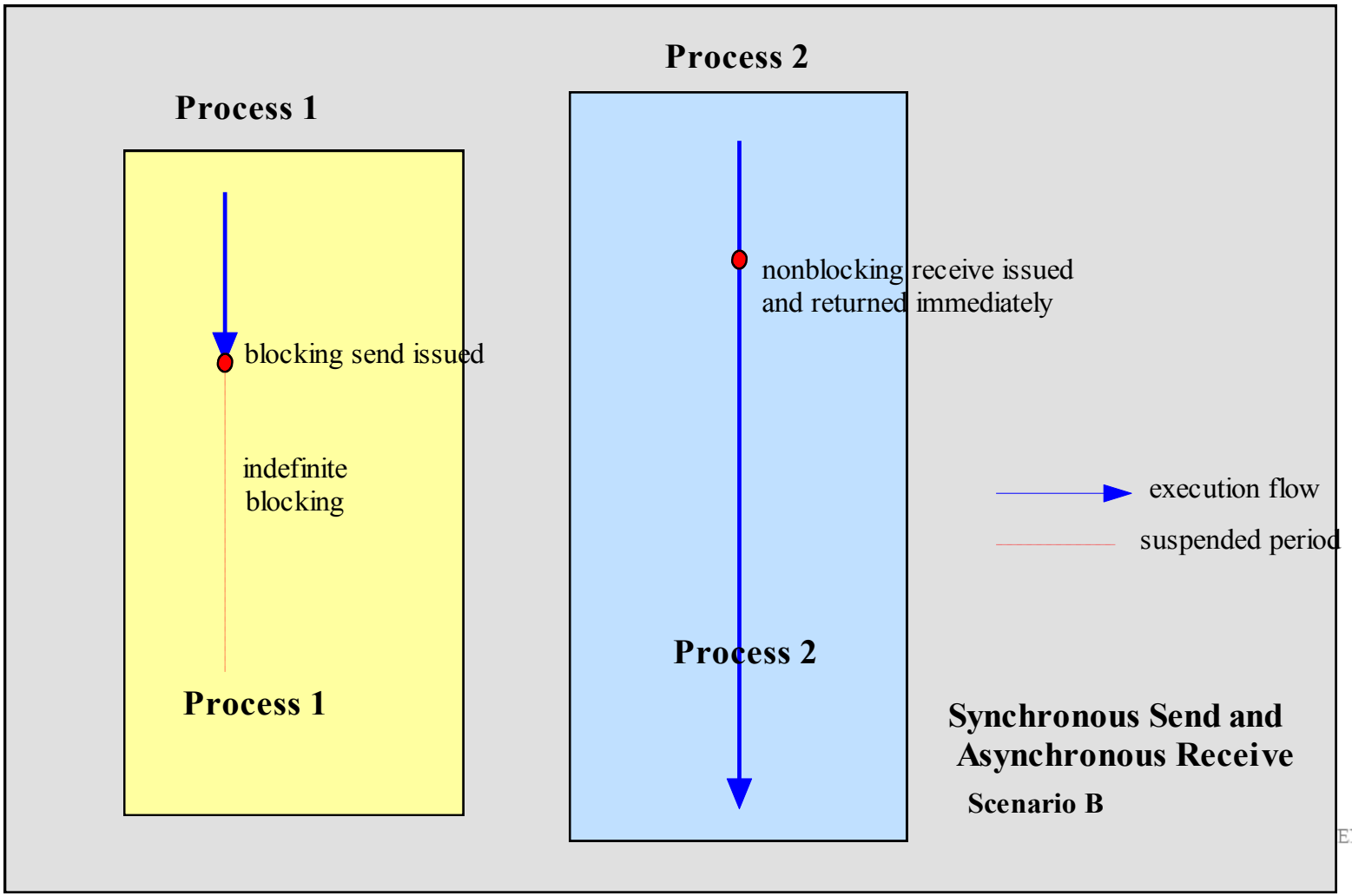
Enviament Asícnron i Recepció Síncrona



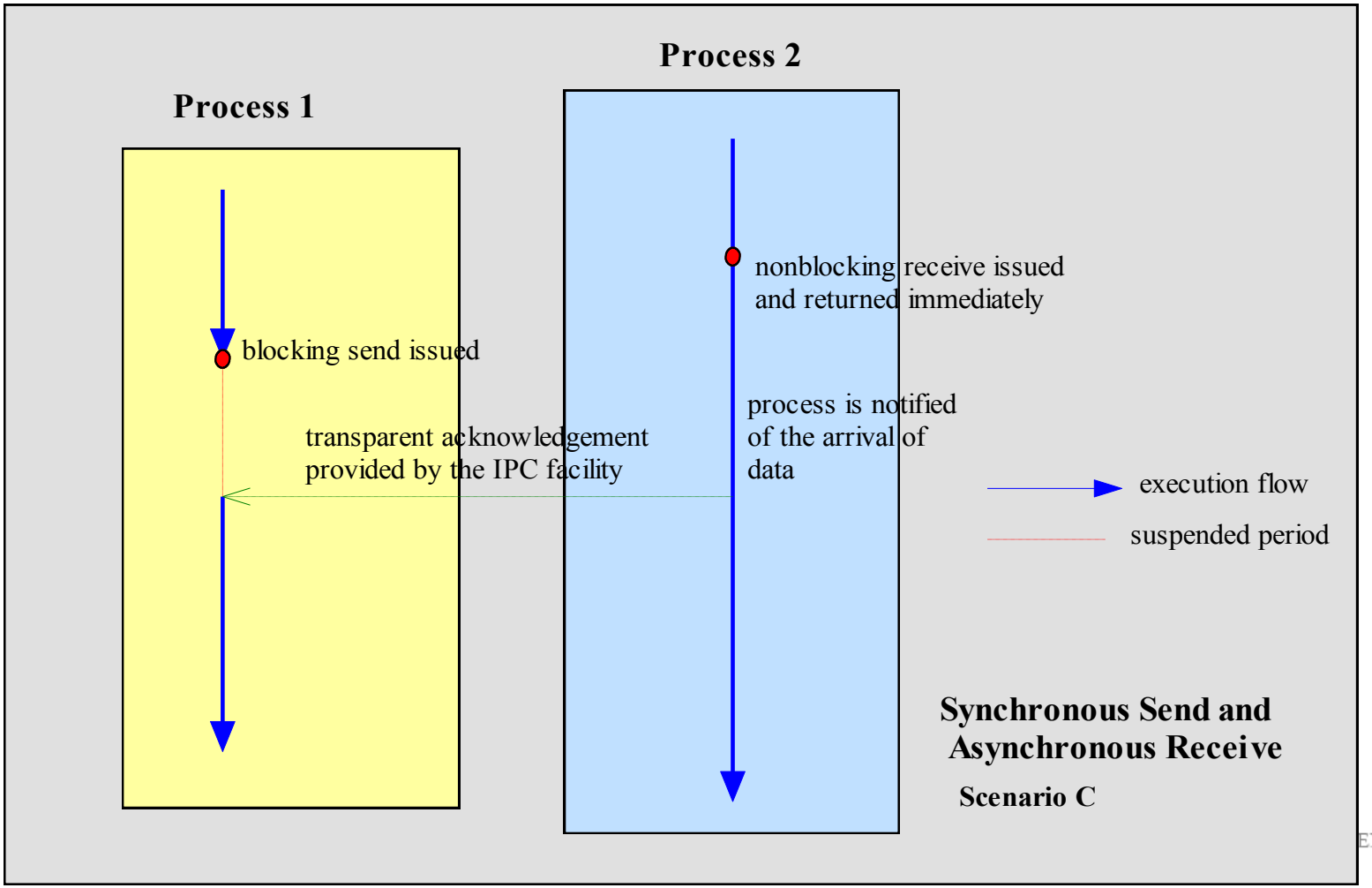
Enviament Síncron i Recepció Asíncrona (1)



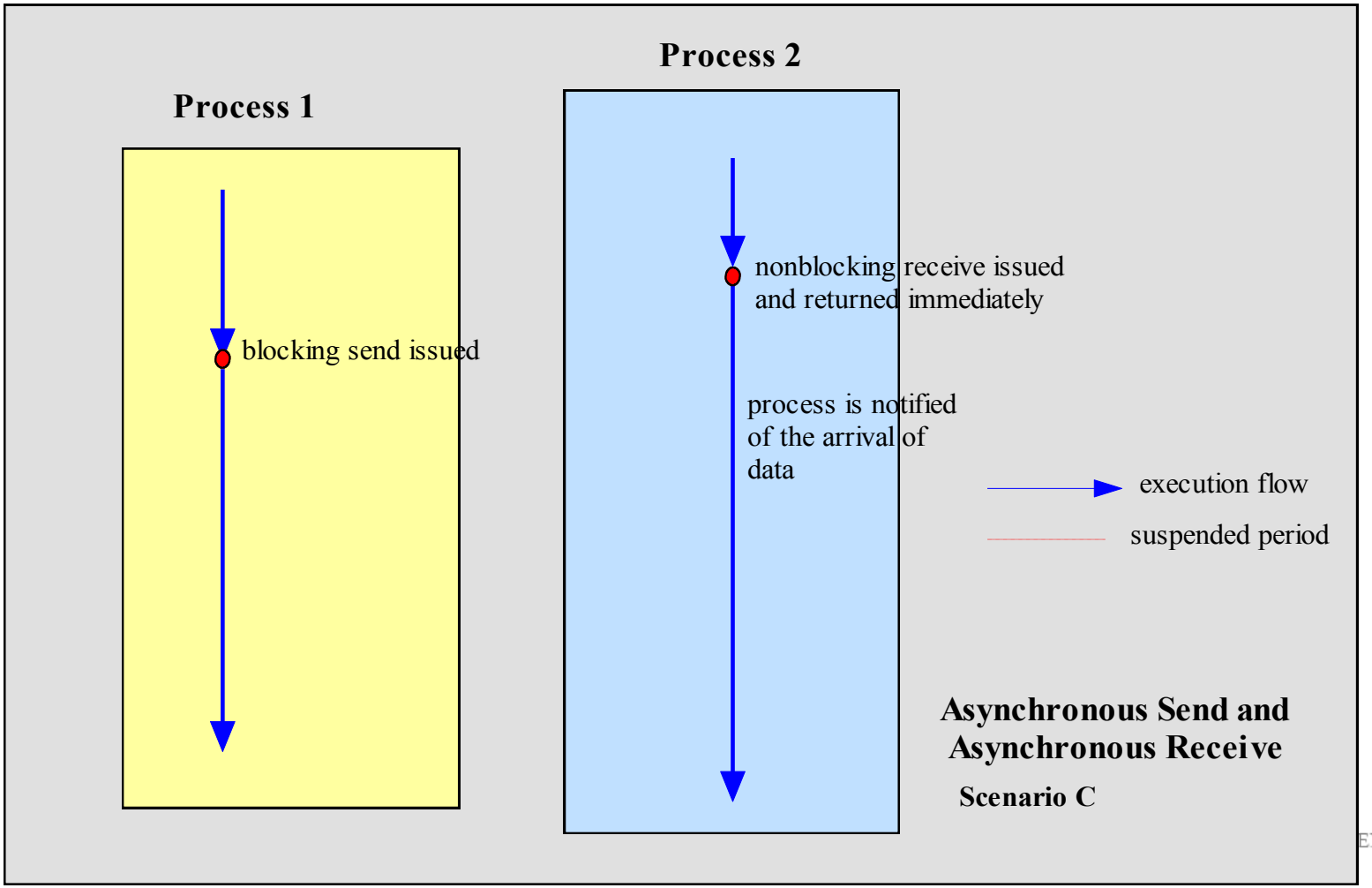
Enviament Síncron i Recepció Asíncrona (2)



Enviament Síncron i Recepció Asíncrona (3)

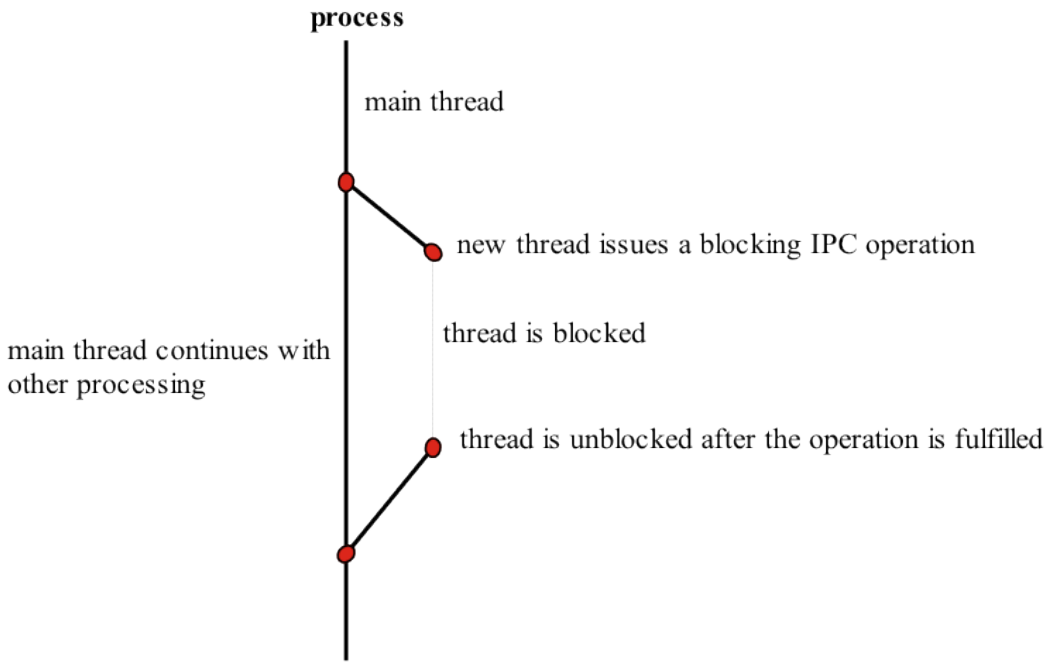


Enviament Asíncron i Recepció Asíncrona



Fent servir Threads per simular una comunicació no bloquejant.

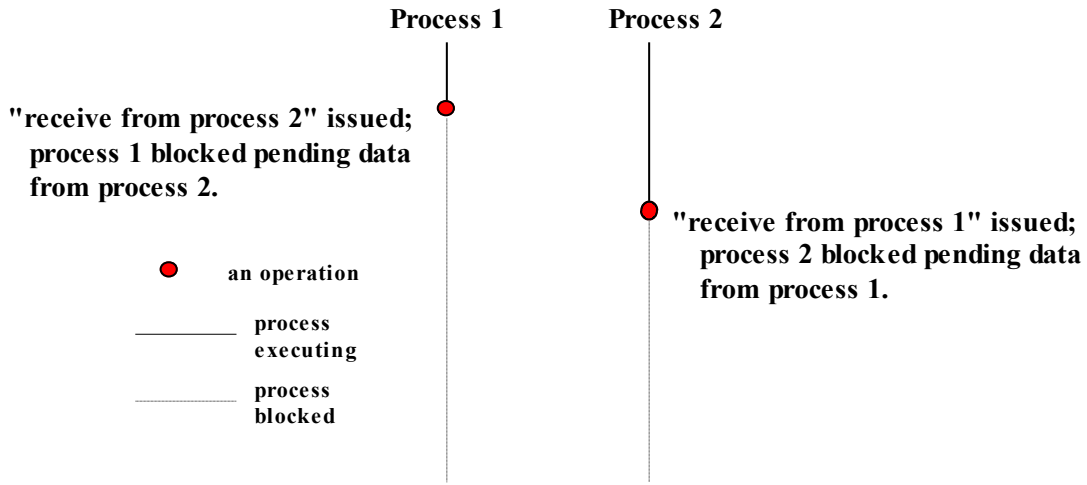
Si tan sols s'ofereixen operacions bloquejants per enviar i / o rebre, llavors el programador ha d'utilitzar processos o fils per a obtenir operacions síncrones que no bloquegin el fluxe principal del programa.



Problemes amb operacions bloquejants: Deadlocks

Les operacions bloquejants llançades en una seqüència incorrecte, poden causar Deadlocks.

Els deadlocks s'han d'evitar.



Problemes amb operacions bloquejants: Bloquejos i Timeouts

Les operacions de connexió i enviament i rebuda síncrona poden acabar en un bloqueig indefinit.
Per exemple, una operació de rebre pot fer que el procés es quedi esperant indefinidament si l'operació no es pot completar o hi ha un tall en la xarxa.
En general, és inacceptable que un procés que fa una operació IPC es pengi per un temps indefinit.
Els bloquejos indefinits es pot evitar mitjançant l'ús de *Timeouts*.

UNIVERSITAT DE BARCELONA