Clase 6

Conceptos generales

- Arquitecturas de memoria distribuidas= procesadores + memoria local + red de comunicaciones + mecanismos de comunicación/ sincronización → Intercambio de mensajes.
- Programa distribuido: programa concurrente comunicado por mensajes. Supone la ejecución sobre arquitectura de memoria distribuida, aunque puedan ejecutarse sobre una de memoria compartida.
- Primitivas de pasaje de mensajes: interfaz con el sistema de comunicaciones → semáforos + datos + sincronización.
- Los procesos SOLO comparten canales.

Características

- Los canales es lo único que comparten los procesos.
- Mecanismos para el procesamiento distribuido
  - Pasaje de mensajes asincrónicos (PMA).
  - Pasaje de mensajes sincrónicos (PMS).
  - Llamado a procedimientos remotos (RPC).
  - Rendezvous.

La sincronización de la comunicación interproceso depende del patrón de interacción como

- Productores y consumidores.
- Clientes y servidores.
- Pares que interactúan.

Relación entre mecanismos de sincronización

Semáforos	Monitores	PM	RPC y rendezvous
Mejora respecto de busy waiting	Combinan exclusión mutua implícita y señalización explítica	Extiende semáforos con datos.	Combina la interface procedural de monitores con PM implícito.

Pasaje de mensajes asincrónicos (PMA)

Los canales son estructuras que permiten el intercambio de mensajes de manera bloqueante o no.

Las declaraciones de canales especifican cómo se definen y se utilizan estos canales en el programa. Los procesos pueden enviar mensajes a través de un canal utiliza la operación `send` y los procesos pueden recibir mensajes a través de la operación `receive`. Estas operaciones son fundamentales para la comunicación y sincronización entre procesos.

En un sistema PMA los canales permiten que los procesos se comuniquen y compartan sin necesidad de que estén sincronizados en tiempo real.

- Declaración de canales: `chan ch (id: tipo.. id: tipo)`
- `Send` un proceso agrega un mensaje al final de la cola de un canal, que no bloquea al emisor.
- `Receive` un proceso recibe un mensaje desde un canal con `receive`, que demora al receptor hasta que en el canal haya al menos un mensaje, luego toma el primero y lo almacena en variables locales `receive ch(var..var)`

Las variables del receive deben tener los mismos tipos que la declaración del canal

Receive es una primitiva bloqueante. El proceso no hará nada hasta recibir un mensaje en la cola correspondiente al canal.

Características de los canales

- Acceso a los contenidos de cada canal: atómico y respeta orden FIFO.
- Canales ilimitados.
- Los mensajes NO se pierden ni modifican y en algún momento serán leídos.

- `empty(ch)` determina si la cola de un canal está vacía. Útil cuando el proceso puede hacer trabajo productivo mientras espera un mensaje.
- Los canales son declarados globales a los procesos, ya que pueden ser compartidos. Según la forma en que se usan podría ser:
  - Cualquier proceso puede enviar o recibir por alguno de los canales declarados.
  - En algunos casos un canal tiene un solo receptos y muchos emisores.
  - Si el canal tiene un único emisor y un único receptor se lo denomina link

Productores y consumidores

`Filtro` proceso que recibe mensajes de uno o más canales de entrada y envía mensajes a uno o más canales de salida. La salida de un filtro es función de su estado inicial y de los valores recibidos.

Esta función del filtro puede especificarse por un predicado que relacione los valores de los mensajes de salida con los de entrada.

En un patrón de productores y consumidores, algunos procesos generan datos y los envían a otros procesos que los usan. El filtro se refiere a procesos que pueden transformar o filtrar los datos entre productores y consumidores.

`Canales de entrada y salida compartidos` los procesos podrán compartir canales de comunicación para recibir y evitar datos. Los canales de entrada son usados por los productores para enviar datos y los canales de salida son usados por los consumidores para recibir datos.

`Static naming y Dynamic namin` enfoques diferentes para organizar los canales de comunicación en al red. En el enfoque static, se utiliza un arreglo global de canales y los canales están predefinidos. En el dynamic se pueden crear canales de manera dinámica y asignarlos a los procesos según necesario.

`Conexión de filtros` los filtros son procesos que transformarán datos,

Clientes y servidores

En esta arquitectura, hay 2 tipos de procesos. Los `clientes` son procesos que realizan peticiones mientras que los `servidores` son procesos que manejan esas solicitudes y proporcionan respuestas.

`Monitores` encapsula variables permanentes que registran el estado y provee un conjunto de procedures. Los simulamos, usando procesos servidores y PM.

- Servidor: proceso que maneja pedidos (requerimientos) de otros procesos clientes.
- Cliente: envía un mensaje a un canal de requerimientos general, luego recibe el resultado desde un canal de respuesta propio.

En un sistema distribuido, lo natural es que el proceso Servidor resida en un procesador físico y M procesos Cliente residan en otros N procesadores.

Para simular Mname, usamos un proceso server Servidor.
➢ Las variables permanentes serán variables locales de Servidor.
➢ Llamado: un proceso cliente envía un mensaje a un canal de requerimiento.
➢ Luego recibe el resultado por un canal de respuesta propio