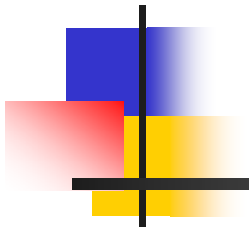
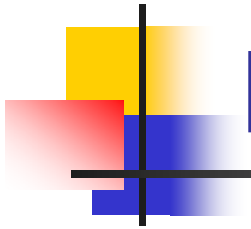
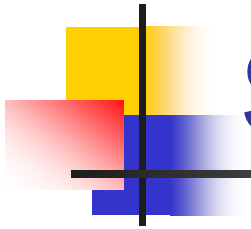
Tema 3. Paso de mensajes





Bibliografía

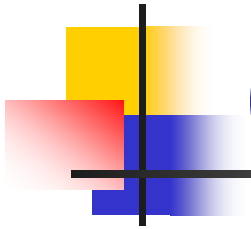
- Programación Concurrente
  - J. Palma, C. Garrido, F. Sánchez, A. Quesada, 2003
  - Capítulo 7
- Principles of Concurrent and Distributed Programming
  - M. Ben-Ari. Prentice Hall, 1990
  - Capítulo 7
- Sistemas Operativos
  - A. Silberschatz, P. Galvin. Addison Wesley Longman, 1999
  - Capítulo 4.6



Sistemas de paso de mensajes

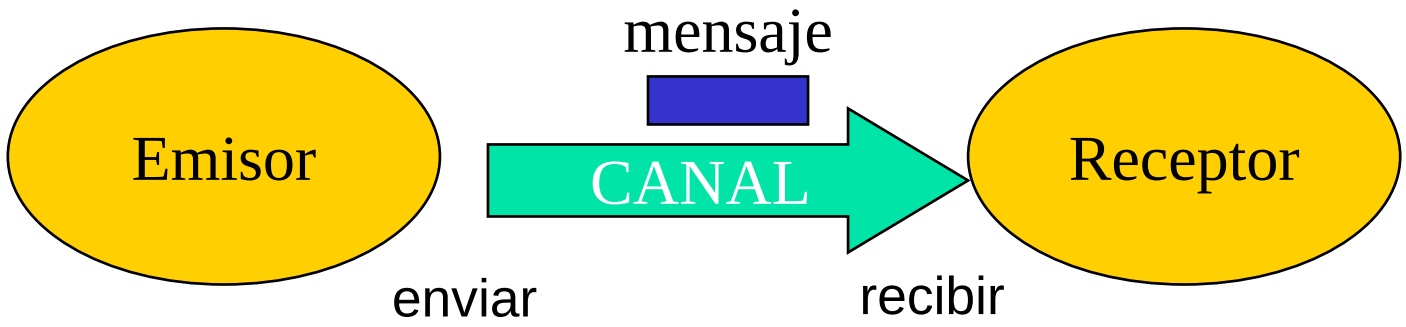
- Los semáforos, monitores, regiones críticas, etc. son todas herramientas que se basan en la existencia de **memoria compartida.**
- El modelo de memoria compartida es difícil o imposible de trasladar a un sistema distribuido, en el que no existe físicamente compartición de memoria.

Sistemas de paso de mensajes (2)

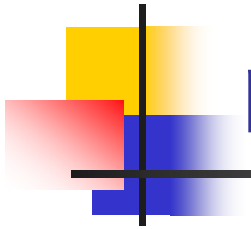


- Como alternativa al modelo de memoria compartida, se define el modelo de **paso de mensajes**:
  - los procesos no comparten memoria (variables, objetos, etc.)
  - la comunicación se hace mediante operaciones explícitas de envío y recepción

Modelo general

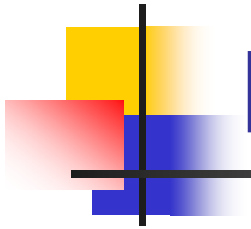


Ventajas del paso de mensajes

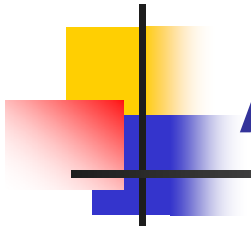


- Válido para cualquier arquitectura de computadores
  - sistemas distribuidos
  - arquitecturas paralelas sin memoria compartida
  - también en sistemas de memoria compartida
- No existe el problema universal del acceso en exclusión mutua a datos compartidos.

Memoria Compartida *OR/XOR* Paso de Mensajes

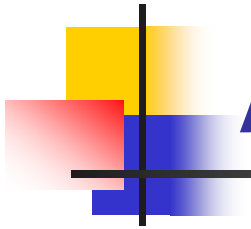


- Ambos esquemas de comunicación NO son mutuamente exclusivos
- Podrían utilizarse simultáneamente dentro de un mismo SO o lenguaje, o incluso dentro de un mismo proceso



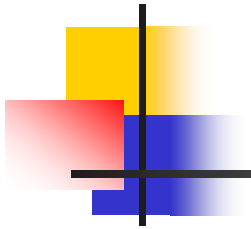
Aspectos de diseño

- Sincronización entre emisor y receptor
  - Comunicación síncrona/asíncrona
- Identificación en el proceso de comunicación
  - Comunicación directa/indirecta
  - Comunicación simétrica/asimétrica
- Características del canal
  - Capacidad, uni/bidireccional, etc…



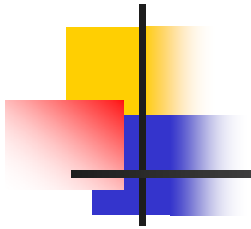
Aspectos de diseño

- Identificación en el proceso de comunicación
  - nombrado, denominación, direccionamiento
- Sincronización
- Características del canal



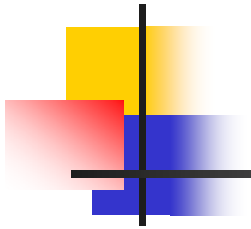
Identificación

- Comunicación directa
  - Cada proceso que desea comunicarse debe nombrar explícitamente el destinatario o el remitente de la comunicación
    - **enviar**(*P*, *mensaje*)
      - Enviar un mensaje al proceso P
    - **recibir**(*Q*, *mensaje*)
      - Recibir un mensaje del proceso Q



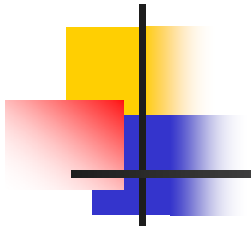
Identificación

- Comunicación indirecta
  - Con la comunicación indirecta, los mensajes se envían a, y se reciben de, buzones (también llamados puertos)
    - **enviar**(*A*, *mensaje*)
      - Enviar un mensaje al buzón A
    - **recibir**(*A*, *mensaje*)
      - Recibir un mensaje del buzón A



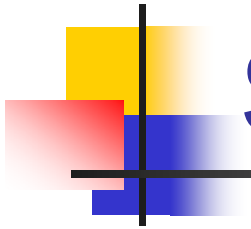
Identificación

- Ventajas/Desventajas
  - Cambio identificación
  - Mensajes distinta naturaleza
  - Aplicaciones cliente/servidor
  - Identificación en sistemas distribuidos



Identificación

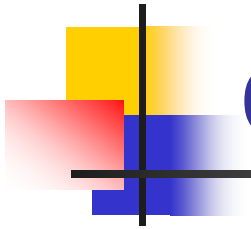
- Comunicación simétrica
  - Los procesos tanto receptor como emisor necesitan nombrar al otro para comunicarse
- Comunicación asimétrica
  - Sólo el emisor nombra al destinatario
  - Resuelve el problema en aplicaciones cliente/servidor



Sincronización

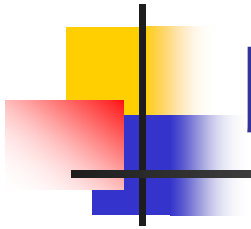
- **Com. síncrona.** El intercambio de un mensaje es una operación atómica que exige la participación simultánea del emisor y el receptor (rendezvous)
  - Extended rendezvous, Encuentro extendido o Invocación remota
- **Com. asíncrona.** El emisor puede enviar un mensaje sin bloquearse; el receptor lo puede recoger más tarde.

Repercusiones de la comunicación asíncrona

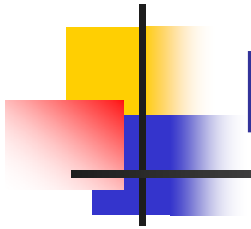


- El emisor puede enviar varios mensajes:
  - necesidad de disponer de **búferes**
- ¿Cuándo sabe el emisor que su mensaje ha llegado/se ha atendido?
  - conveniencia de operaciones de “acuse de recibo” o de respuesta
    (send → receive → send_reply → receive_reply)

Llamadas bloqueantes / no bloqueantes

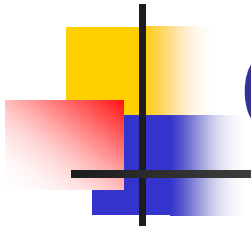


- Las operaciones de envío y recepción pueden estar definidas como bloqueantes o no bloqueantes
- Un envío/recepción con bloqueo es un ejemplo de comunicación síncrona
- Un envió/recepción sin bloqueo es un ejemplo de comunicación asíncrona



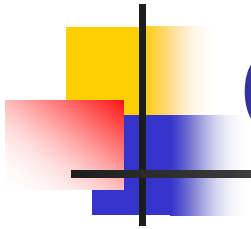
Ejemplos

- Comunicación directa simétrica
  - Enviar(P,mensaje)/Recibir(Q,mensaje)
- Comunicación directa asimétrica
  - Enviar(P,mensaje)/Recibir(mensaje)
  - Enviar(P,mensaje)/Recibir(id,mensaje)



Características del canal

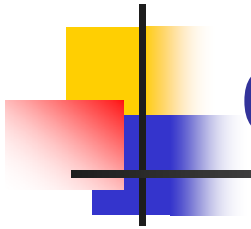
- Punto a punto, multipunto
- Unidireccional, bidireccional
- Capacidad del canal
  - cero
  - limitada
  - infinita (teórico)



Características del canal

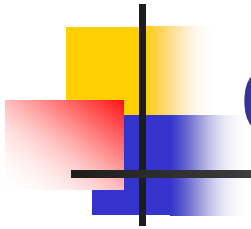
- Flujo de datos
  - Enlaces unidireccionales (síncrona) o bidireccionales (síncrona o asincrona)
- Capacidad del canal
  - cero, limitada, infinita (teórico)
- Tamaño de los mensajes
  - Longitud fija o variable
- Canales con tipo o sin tipo
- Paso por copia o por referencia

Características del canal de comunicación: ejemplos

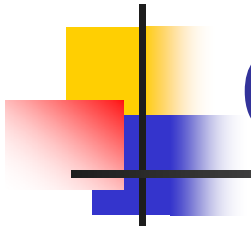


- Comunicación directa
  - Se establece automáticamente
  - Un canal se asocia a exactamente dos procesos
  - Entre cada par de procesos sólo existe un canal
  - El enlace puede ser unidireccional, pero suele ser bidireccional

Características del canal de comunicación: ejemplos

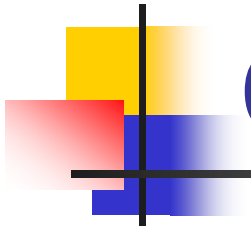


- Comunicación indirecta
  - Se establece un canal entre un par de procesos sólo si tienen un buzón compartido
  - Un canal puede estar asociado a más de dos procesos
  - Entre cada par de procesos en comunicación puede haber varios enlaces distintos, cada un de los cuales corresponderá a un buzón
  - Los enlaces pueden ser unidireccionales o bidireccionales



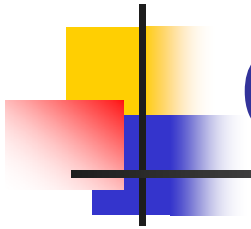
Condiciones de error

- 1 máquina => los mensajes se implementan (generalmente) en memoria compartida
  - Fallo => falla todo el sistema
- Entornos distribuidos => procesos residen en diferentes máquinas
  - Los mensajes se transfieren por líneas de comunicación
  - La probabilidad de que ocurra un error durante la comunicación y el procesamiento es mucho mayor que en un entorno de una sola máquina
  - El fallo de un enlace no causa necesariamente el fallo de todo el sistema



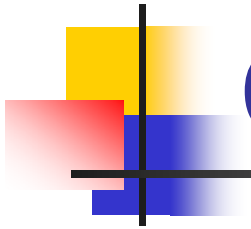
Condiciones de error

- Cuando ocurre un fallo en un sistema, sea centralizado o distribuido, el sistema debe intentar recuperarse del error
- Algunas situaciones de error:
  - El emisor o el receptor podría terminar antes de que se procese un mensaje
    - P espera un mensaje de Q (proceso terminado)
    - P envía un mensaje a Q (proceso terminado)



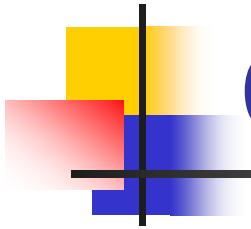
Condiciones de error

- Mensajes perdidos
  - Fallo *hardware* o de la línea de comunicación
- Tres métodos para enfrentar este suceso en función de quien asume la responsabilidad de detectar el fallo:
  - SO
  - Emisor
  - SO/Emisor



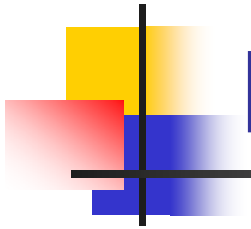
Condiciones de error

- No siempre es necesario detectar los mensajes perdidos => protocolos de red que garantizan la confiabilidad
- ¿Cómo detectar la pérdida de mensajes?
  - El método de detección más común consiste en emplear *tiempos límite* o *plazos*



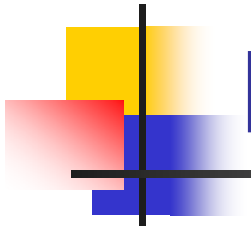
Condiciones de error

- Mensajes alterados
  - es común el uso de códigos de verificación de errores (paridad, etc...)



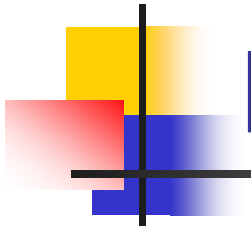
Ejercicio

- Supongamos que partimos de un sistema de comunicación mediante paso de mensaje donde la comunicación es directa y sincrona:
  - enviar(A,mensaje)
  - recibir(B,mensaje)
- Implementar el problema del productor/consumidor asumiendo que sólo existe un proceso de cada tipo



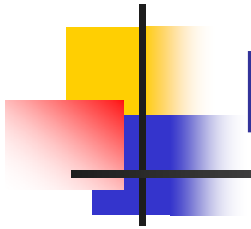
Espera selectiva

```
•select
•   RECEIVE(buzon1, mensaje);
•   sentencias;
•or
•   RECEIVE(buzon2, mensaje);
•   sentencias
•or
•   ...
•or
•   RECEIVE(buzónN, mensaje);
•   sentencias;
•end select;
```



Espera selectiva con guardas

```
•...
•select
•   when condicion1 =>
•   RECEIVE(buzon1, mensaje);
•   sentencias;
•or
•   when condicion2 =>
•   RECEIVE(buzon2, mensaje);
•   sentencias;
•or
•   ...
•or
•   when condicionN =>
•   RECEIVE(buzónN, mensaje);
•   sentencias;
•end select;
•...
```



Ejercicio

- Búfer limitado