code URJC

Programación Concurrente

Tema 3

Paso de Mensajes

Micael Gallego
micael.gallego@urjc.es
@micael_gallego

Universidad Rey Juan Carlos

Concurrencia con Paso de Mensajes

- **Introducción**
- Identificación de procesos
- Sincronización
- Canal de comunicación
- Conclusiones

Introducción

- Las técnicas usadas para la construcción de programas concurrentes se dividen en dos grandes modelos:
  - **Memoria compartida:** Los procesos pueden acceder a una memoria compartida para comunicarse y sincronizarse.
  - **Paso de mensajes:** Los procesos sólo se pueden comunicar y sincronizar con el envío de mensajes de uno a otro.

Introducción

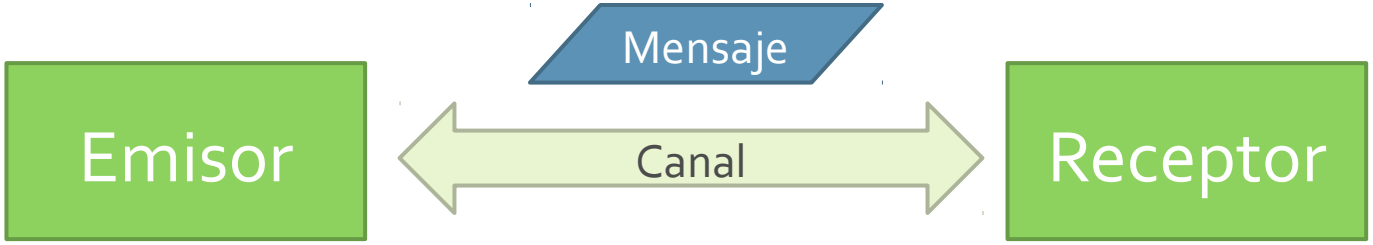
- **¿Cuándo usar paso de mensajes?**
  - Tiene que usarse obligatoriamente en sistemas multiprocesador poco acoplados (sistemas distribuidos)
  - Puede usarse también en sistemas muy acoplados:
    - El programa será más escalable (podrá ser usado en un cluster en el futuro)
    - Cuando el modelo de concurrencia sea más adecuado para el programa concreto
    - Cuestión de gustos.

Introducción

- **Modelos híbridos**
  - Es habitual que un sistema informático distribuido combine ambos modelos.
    - Modelo de paso de mensajes para comunicación entre nodos.
    - Modelo de memoria compartida para comunicación entre procesos dentro del mismo nodo.
  - Pero cada vez se usan más modelos puros basado en **paso de mensajes** en ambos niveles

Introducción

- **Primitivas básicas** en un modelo de paso de mensajes:
  - **Send:** Envía información (mensaje) a otro proceso
  - **Receive:** Recibe información (mensaje) de otro proceso



Introducción

- Estas primitivas pueden ser:
  - **Explícitas:** Cuando se usan directamente por el programador
  - **Implícitas:** Cuando el modelo las oculta dentro de otras primitivas de más alto nivel

Introducción

- Existen muchos tipos concretos de modelos de paso de mensajes dependiendo de:
  - **Identificación de procesos** al enviar o recibir (nombrado, direccionamiento, denominación…)
  - **Sincronización:** Envío y recepción bloqueantes (síncronas) o no bloqueantes (asíncronas)
  - **Características del canal:** Capacidad, tipo de datos, etc…

Concurrencia con Paso de Mensajes

- Introducción
- **Identificación de procesos**
- Sincronización
- Canal de comunicación
- Conclusiones

Identificación de procesos

- **Identificación de procesos**
  - La forma en la que el **proceso emisor** indica a qué **proceso receptor** va dirigido el mensaje y viceversa
  - Operaciones:

```
void send(Id receiverId, Message message)
Message receive(Id senderId)
```

Identificación de procesos

- **Comunicación directa simétrica:**
  - Tanto el proceso emisor como el proceso receptor indican el id del otro proceso.
    - Proceso A ejecuta: **send("B", Message)**
    - Proceso B ejecuta: **m = receive("A")**
  - **Desventajas:**
    - La identificación explícita de el receptor y el emisor puede hacer el sistema poco flexible a futuras ampliaciones

Identificación de procesos

- **Comunicación directa asimétrica:**
  - Normalmente el proceso emisor indica el id del receptor, pero no al contrario
    - Proceso A ejecuta: **send("B", Message)**
    - Proceso B ejecuta: **m = receive()**
  - **Ventajas:**
    - Es más flexible que el modelo simétrico
    - Es el modelo usado en arquitectura cliente / servidor en sistemas distribuidos
    - Habitualmente el receptor puede conocer qué emisor concreto le ha enviado cada mensaje

Identificación de procesos

- **Comunicación directa vs comunicación indirecta**
  - **Comunicación directa:** Los procesos identifican a otros procesos
  - **Comunicación indirecta:** Los procesos emisores envían los mensajes a almacenes intermedios y los receptores los recogen de esos almacenes intermedios

Identificación de procesos

- **Comunicación indirecta**
  - La comunicación se realiza con almacenes intermedios
  - Esos almacenes se denominan **buzones** o **colas**

```
void send(Id queueId, Message message)
Message receive(Id queueId)
```

Identificación de procesos

- **Comunicación indirecta**
  - Dependiendo de las restricciones en las colas y los mensajes existen diversos modelos:
    - Varios procesos pueden recibir mensajes de una misma cola o cada cola puede ser exclusiva por proceso receptor
    - Un mensaje enviado a una cola puede ser recibido por un único proceso (unicast) o por todos los procesos (multicast)
    - Un mismo proceso puede enviar mensajes a varias colas
    - La creación de colas puede ser estática o dinámica (en tiempo de ejecución)

Concurrencia con Paso de Mensajes

- Introducción
- Identificación de procesos
- **Sincronización**
- Canal de comunicación
- Conclusiones

Sincronización

- Tipos de comunicaciones en función de la **sincronización**:
  - **Comunicación síncrona:** Emisor y receptor coinciden en el tiempo en el momento de la comunicación (ejemplo: **teléfono**)
  - **Comunicación asíncrona:** Emisor y receptor tienen que coincidir en el tiempo para llevar a cabo la comunicación (ejemplo: **mail**)

Sincronización

- **Bloqueo de envío y recepción**
  - **Comunicación síncrona:**
    - El **emisor** queda **bloqueado** en la operación de envío hasta que el receptor ejecute la operación de recepción
    - El **receptor** queda **bloqueado** en la operación de recepción

Cita simple (*rendez-vous*)

Sincronización

- **Bloqueo de envío y recepción**
  - **Comunicación síncrona (caso especial):**
    - El **emisor** queda **bloqueado** en la operación de envío hasta que el receptor ejecute la operación de **recepción**, **procese el mensaje** y envíe un mensaje de **vuelta**, que será recibido por el emisor.

Cita extendida (*extended rendezvous*)

Sincronización

- **Bloqueo de envío y recepción**
  - **Comunicación síncrona (tiempo de espera):**
    - En algunos sistemas, las operaciones de envío y recepción bloqueantes disponen de versiones en las que se puede especificar un **timeout**.

Sincronización

- **Bloqueo de envío y recepción**
  - **Comunicación asíncrona:**
    - El **emisor** no queda **bloqueado** en la operación de envío, los mensajes se guardan en un **buffer** hasta que el receptor los recoge
    - El **receptor** puede ejecutar la operación de recepción **sin bloquearse**

Sincronización

- **Bloqueo de envío y recepción**
  - **Comunicación asíncrona (casos especiales):**
    - El **emisor** podría quedarse **bloqueado** en la operación de envío si el buffer se llena de mensajes pendientes de recibir.
    - El **receptor** podría querer **bloquearse** en la operación de recepción **si no hay mensajes**

Concurrencia con Paso de Mensajes

code URJC

- Introducción
- Identificación de procesos
- Sincronización
- **Canal de comunicación**
- Conclusiones

Canal de comunicación

- **Características del canal de comunicación**
  - **Flujo de datos:** bidireccional o unidireccional. La comunicación asíncrona suele ser unidireccional y la síncrona bidireccional.
  - **Capacidad del canal:** Relevante en comunicación asíncrona (limitada o infinita dependiendo de los recursos HW)

Canal de comunicación

- **Características del canal de comunicación**
  - **Tipo y tamaño de los mensajes:** Algunos sistemas pueden poner restricciones de todos los mensajes del mismo tipo o del mismo tamaño.
  - **Ordenación y fiabilidad del envío:** Los enlaces de red pueden ser menos fiables que los canales internos en una máquina

Concurrencia con Paso de Mensajes

- Introducción
- Identificación de procesos
- Sincronización
- Canal de comunicación
- **Conclusiones**

Conclusiones

- El modelo de **paso de mensajes** es interesante porque permite construir programas escalables (desde un **nodo** a un **cluster**)
- Existen una gran **variedad de detalles** en este modelo, y cada tecnología de desarrollo implementa modelos ligeramente **diferentes**