.UBAfiuba
FACULTAD DE INGENIERÍA

Técnicas de Programación Concurrente I

Pasaje de Mensajes

Ing. Pablo A. Deymonnaz
pdeymon@fi.uba.ar

Facultad de Ingeniería
Universidad de Buenos Aires

Índice

Modelos de Comunicación

- Comunicación
  - Sincrónica
  - Asincrónica - Buffer
- Direccionamiento ¿Cómo se determina a quién dirigir un mensaje?
  - Simétrico
  - Asimétrico
  - Sin direccionamiento (matcheo por estructura del mensaje)
- Flujo de datos
  - Unidireccional
  - Bidireccional

Canales

- Conectan un proceso emisor con un proceso receptor
- Tienen un nombre
- Son tipados
- Sincrónicos o asincrónicos
- Unidireccionales

Productores y Consumidores

channel of Integer ch

Productor

```
task body Producer is
  I : Integer
begin
  loop
    Produce(I);
    ch <= I;
  end loop
end Producer
```

Consumidor

```
task body Producer is
  I : Integer
begin
  loop
    ch => I ;
    Consume(I);
  end loop
end Consumer
```

Selective Input

- Es una sintaxis permitida por los lenguajes que soportan canales
- Permite escuchar en varios canales de forma bloqueante y desbloquearse con el primero que recibe un mensaje

```
either
  ch1 => var1
or
  ch2 => var2
or
  ch3 => var3
```

Filósofos Comensales

channel of Boolean forks[5]

Filósofo i

```
task body Philosopher is
  dummy : Boolean
begin
  loop
    Think;
    forks[i] => dummy;
    forks[i+1] => dummy;
Eat;
forks[i] <= true;
forks[i+1] <= true;
  end loop;
end Philosopher
```

Fork i

```
task body Fork is
  dummy : Boolean
begin
  loop
    forks[i] <= true;
    forks[i] => dummy;
  end loop
end Fork
```

Remote Procedure Calls

Permiten al cliente ejecutar funciones en un servidor localizado en otro procesador.

- Se requiere la implementación de *stubs* en ambos extremos
- Los stubs conforman interfaces remotas utilizadas para compilar cliente y servidor
- Localización de servicios
- Parameter marshalling

.UBA fiuba FACULTAD DE INGENIERÍA

Índice

Canales en Unix

- Unix provee Pipes y FIFOs para conectar dos procesos independientes, orientados a **bytes**.
  - Los FIFOs poseen una representación en el file system.
- Unix también provee colas de mensajes (*Message queues*), orientados a **mensajes** como unidades independientes.

Índice

Pasaje de mensajes

"Do not communicate by sharing memory; instead, share memory by communicating."

Canales

- Un canal tiene dos extremos: un emisor y un receptor.
- Una parte del código invoca métodos sobre el transmisor, con los datos que se quiere enviar.
- Otra parte chequea el extremo de recepción por la existencia de mensajes.
- Múltiples productores, un consumidor.
- **Transfieren el ownership del elemento enviado.**
- Para crear múltiples productores, se clona el extremo de envío.

Canales

```rust
use std::sync::mpsc;
use std::thread;

fn main() {
  let (tx, rx) = mpsc::channel();

  thread::spawn(move || {
    let val = String::from("Hola");
    tx.send(val).unwrap();
  });
  let received = rx.recv().unwrap();
  println!("Recibido: {}", received);
}
```

Bibliografía

- **Principles of Concurrent and Distributed Programming**, M. Ben-Ari, Segunda edición (capítulos 6)
- Bibliografía de Rust.
- **The Rust Programming Language**, *https://doc.rust-lang.org/book/*
- **Programming Rust: Fast, Safe Systems Development**, 1st Edition, Jim Blandy, Jason Orendorff. 2017.