

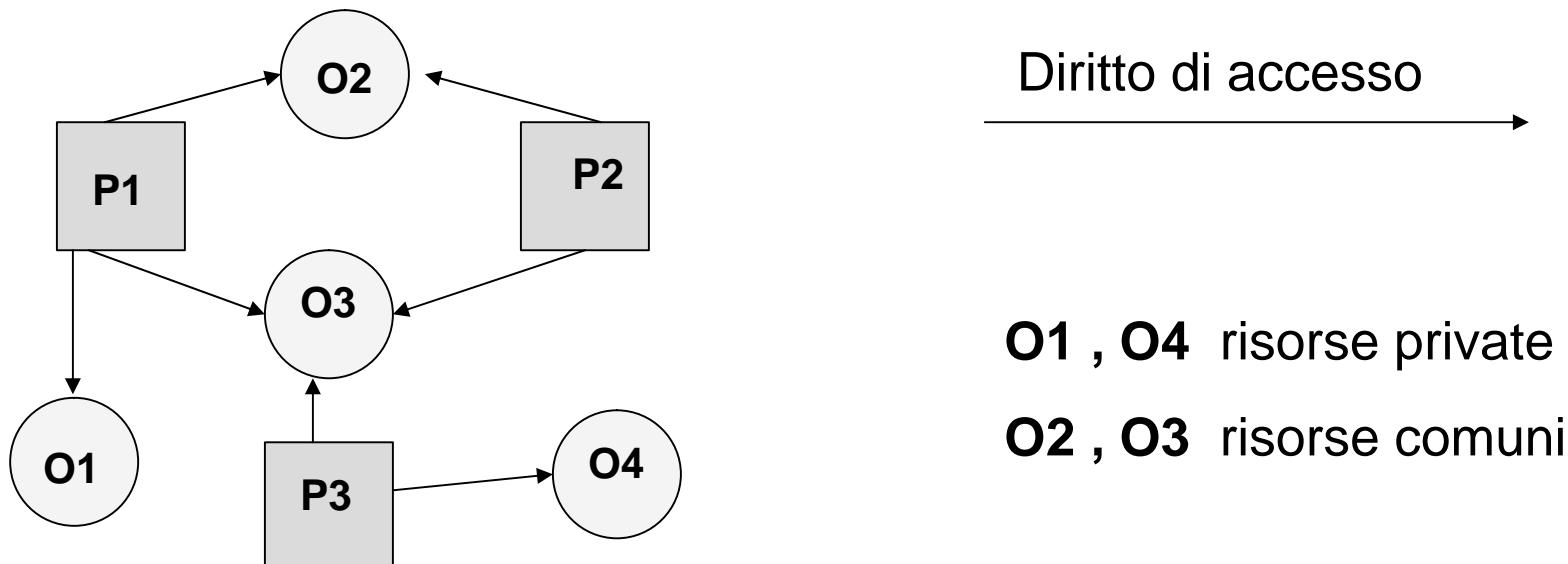
# Modelli di interazione tra processi

# Modelli di interazione

- Modello a ***memoria comune*** (ambiente globale)
- Modello a ***scambio di messaggi*** (ambiente locale, message passing)

# Modello a memoria comune

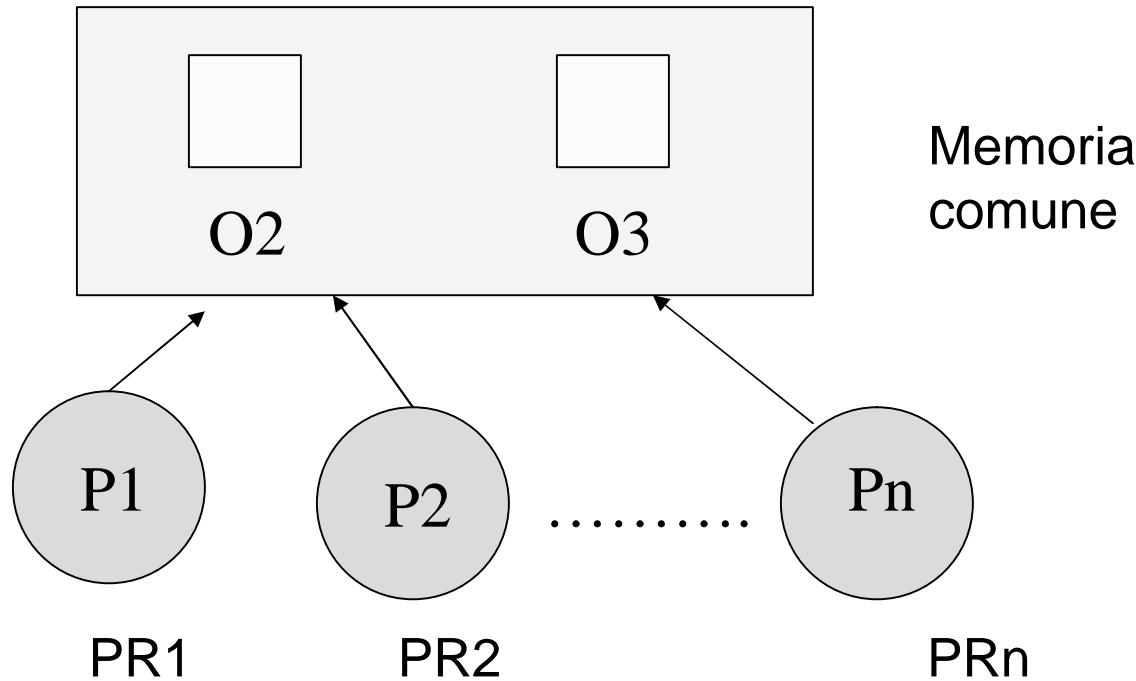
- Il sistema è visto come un insieme di
  - processi
  - oggetti (risorse)



*Tipi di interazioni tra processi:*

- **competizione**
- **cooperazione**

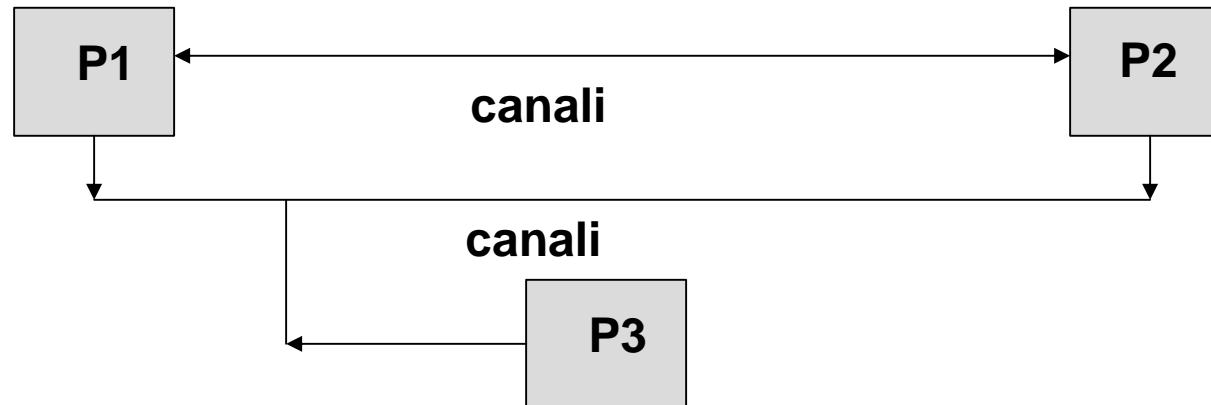
- Il modello a memoria comune rappresenta la naturale astrazione del funzionamento di **un sistema in multiprogrammazione** costituito da uno o più processori che hanno accesso ad una memoria comune



- Ad ogni processore può essere associata una memoria privata, ma *ogni interazione avviene tramite oggetti contenuti nella memoria comune*

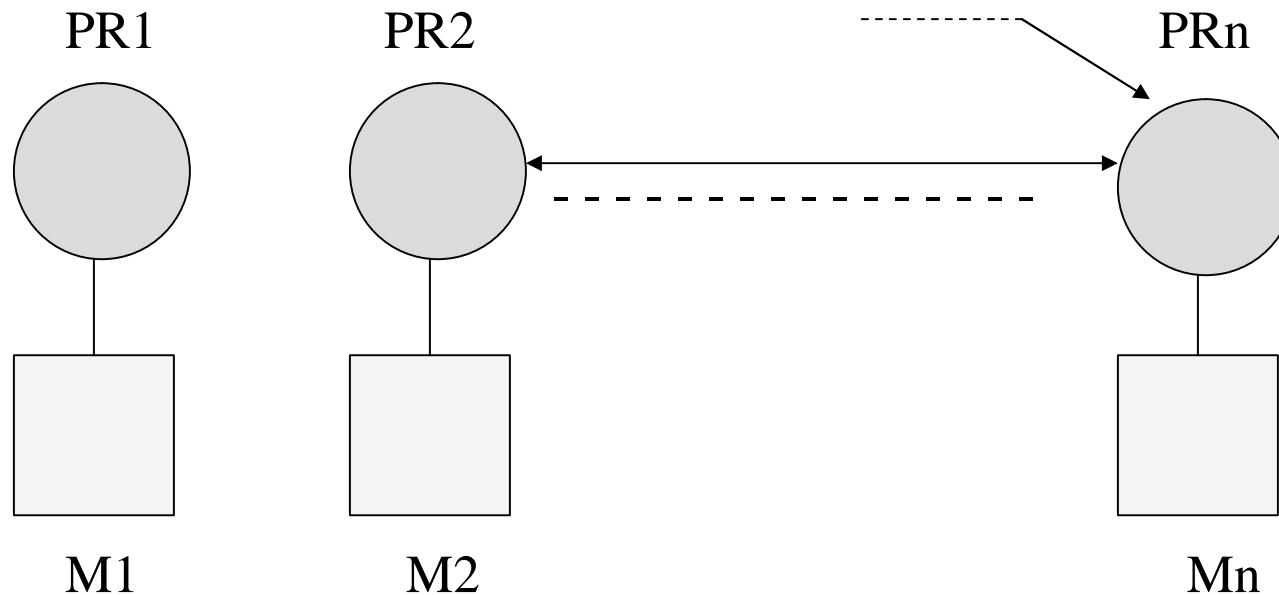
# Modello a scambio di messaggi

Il sistema è visto come un insieme di processi ciascuno operante in un *ambiente locale* che non è accessibile a nessun altro processo.



- Ogni forma di interazione tra processi avviene tramite scambio di messaggi.
- Non esiste più il concetto di **risorsa accessibile direttamente ai processi**; sono possibili due casi:
  - alla risorsa è associato un **processo servitore**
  - la risorsa viene passata da un processo all'altro **sotto forma di messaggi**

- Il modello a scambio di messaggi rappresenta la naturale astrazione di un sistema *privo di memoria comune*, in cui a ciascun processore è associata una memoria privata



- Il modello a scambio di messaggi può essere realizzato *anche in presenza di memoria comune*, che viene utilizzata per realizzare canali di comunicazione.

# **Modello a memoria comune**

# Aspetti caratterizzanti

- Ogni applicazione *viene strutturata* come un insieme di componenti, suddiviso in due sottoinsiemi disgiunti:
  - **processi** (componenti attivi)
  - **risorse** (componenti passivi).

**Risorsa:** Qualunque oggetto, fisico o logico, di cui un processo necessita per portare a termine il suo compito.

- Le risorse sono raggruppate in *classi*; ogni classe identifica l'insieme di *tutte e sole* le operazioni che un processo può eseguire per operare su risorse di quella classe.
- Il termine *risorsa* si identifica con quello di *struttura dati* allocata nella memoria comune.

**Risorsa privata** di un processo P (o **locale** a P): P è il *solo* processo che può eseguire operazioni sulla risorsa.

**Risorsa comune (o globale)**: è una risorsa su cui *più processi* possono operare.

→ In un modello a memoria comune i processi interagiscono *esclusivamente* operando su **risorse comuni** (competizione e cooperazione).

- **Meccanismo di controllo degli accessi**: è necessario definire quali processi ed in quali istanti possono *correttamente* accedere alla risorsa.
- **Allocatore o gestore di una risorsa R**: entità che ha il compito di definire in ogni istante t l'insieme  $SR(t)$  dei processi che possono accedere ad R in quell'istante.

# Allocazione delle risorse

## Risorsa allocata staticamente.

- Il gestore di R definisce l'insieme  $SR$  all'istante  $T_0$  (istante iniziale dell'elaborazione) senza modificarlo durante l'elaborazione.
- Il gestore della risorsa è il *programmatore* che in base alle *scope rules* del linguaggio stabilisce quale processo possa vedere e quindi elaborare la risorsa.
- Il compito di controllare gli accessi è svolto dal *compilatore* che, secondo le regole di visibilità, assicura che *soltanto* i processi ai quali la risorsa è stata allocata possano accedervi.

# Allocazione delle Risorse

## Risorsa allocata dinamicamente

- L'insieme SR è variabile nel tempo:
  - a) *Risorsa dedicata.*  $Sr(t)$  contiene *al più un processo*
  - b) *Risorsa condivisa.*  $Sr(t)$  contiene *più processi contemporaneamente*
- Il gestore della risorsa opera a *tempo di esecuzione*. Nel modello a memoria comune il gestore è una *risorsa* (nel modello a scambio di messaggi è un *processo*).
- $Sr(t)$  è inizialmente vuoto. Per operare sulla risorsa il processo *deve chiedere il permesso al gestore*.
- Il gestore può accettare, ritardare o rifiutare la richiesta .
- *Schema logico* seguito dal processo:
  - 1) *richiesta* della risorsa,
  - 2) *uso*
  - 3) *rilascio* del diritto di accedere.

	risorsa dedicata	risorsa condivisa
risorsa allocata staticamente	<b><i>privata</i></b>	<b><i>comune</i></b> ai processi cui la risorsa è allocata
risorsa allocata dinamicamente	<b><i>comune</i></b>	<b><i>comune</i></b>

Una risorsa allocata staticamente e dedicata ad un solo processo è una ***risorsa privata***.

In tutti gli altri casi le risorse sono ***comuni***, o perché condivise tra più processi o perché dedicate, ma *dinamicamente*, a processi diversi in tempi diversi.

# Meccanismi linguistici per la programmazione di interazioni

- Per presentare *alcuni meccanismi linguistici* usati nella programmazione di interazioni tra processi si seguirà il seguente schema:
  1. **Presentazione** del meccanismo
  2. **Esempi** di uso del meccanismo
  3. **Traduzione** del meccanismo linguistico in termini di un *meccanismo primitivo* di sincronizzazione fornito dal supporto a tempo di esecuzione (**nucleo**)
- Il meccanismo primitivo di riferimento è quello **semaforico** con le due operazioni **wait** e **signal** (Dijkstra).