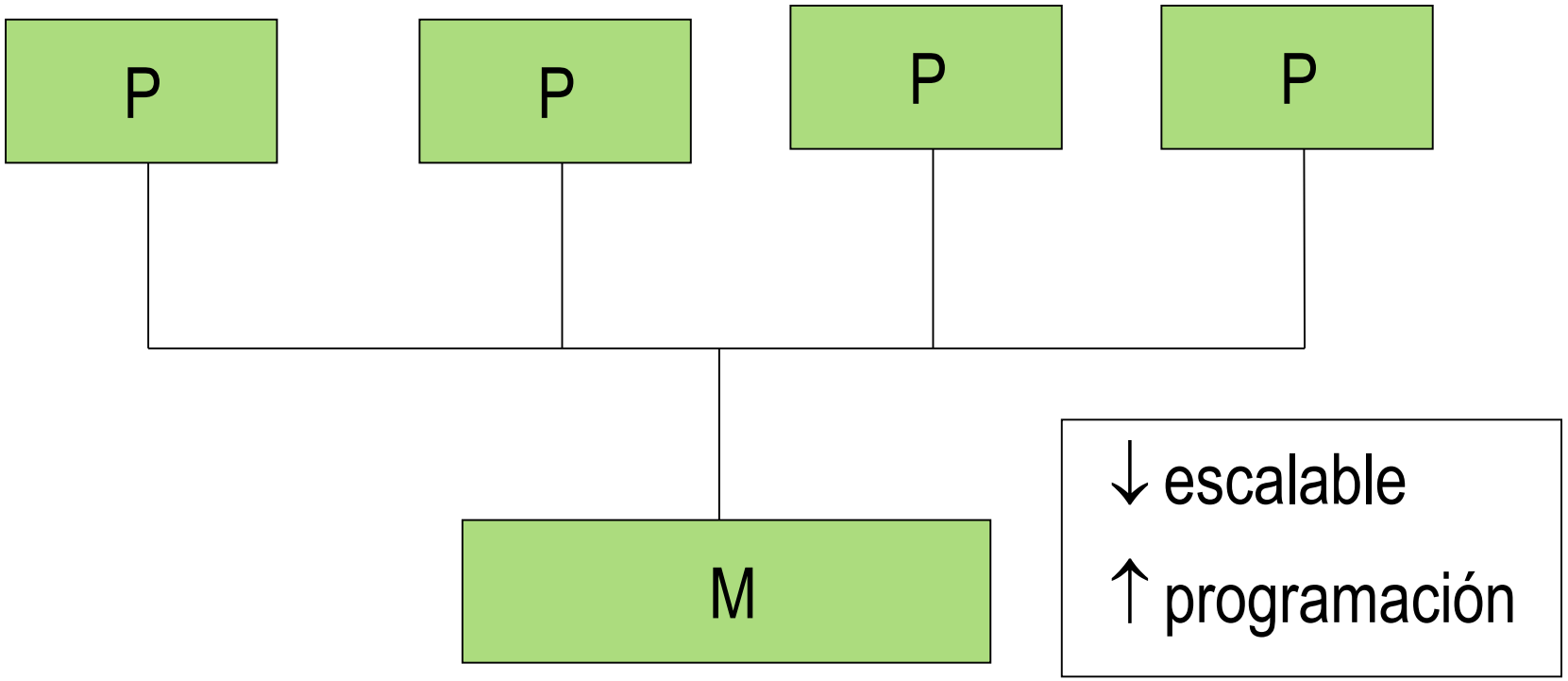
Sistemas Distribuidos

Memoria compartida distribuida

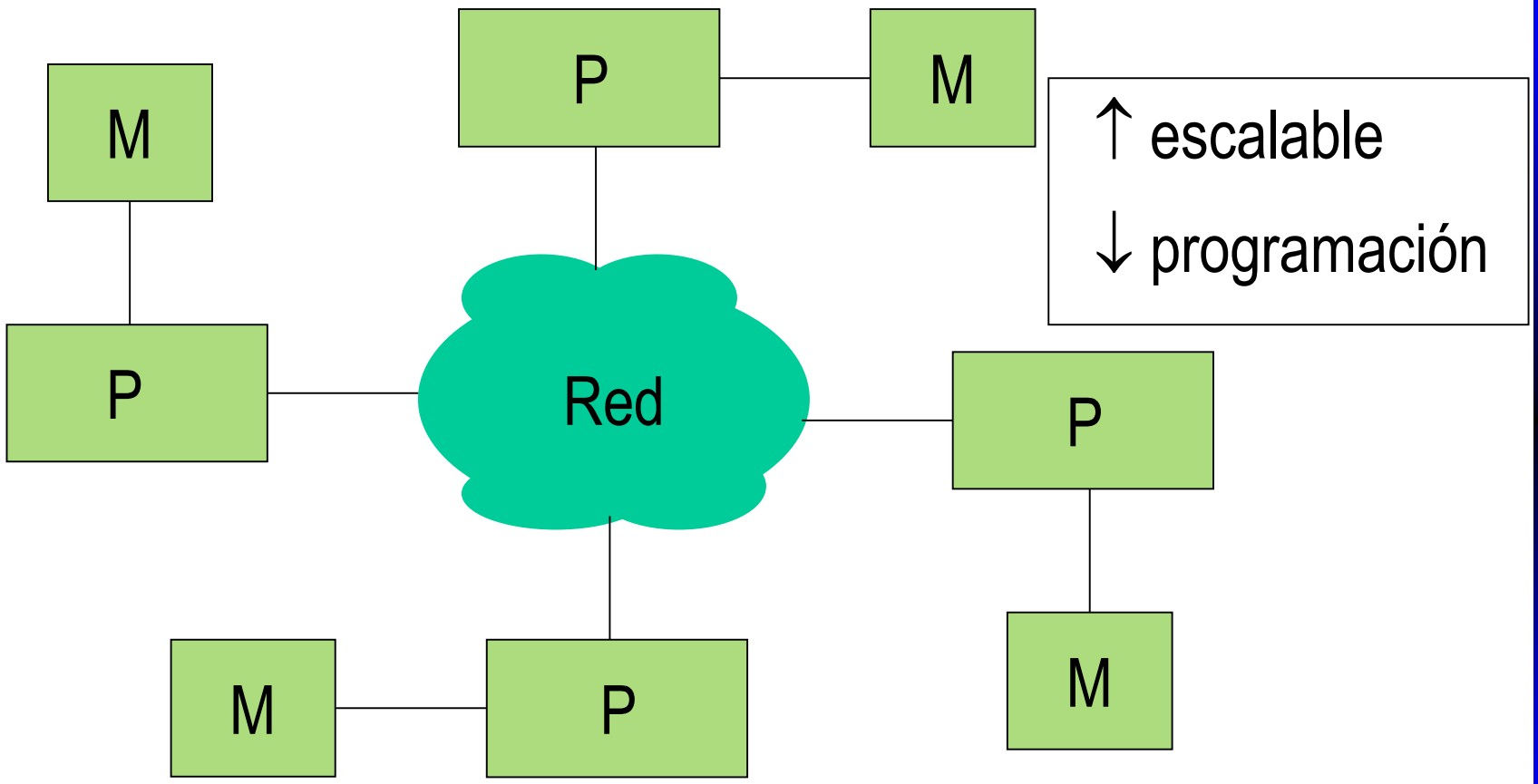
(DSM, *Distributed Shared Memory*)

Fernando Pérez Costoya

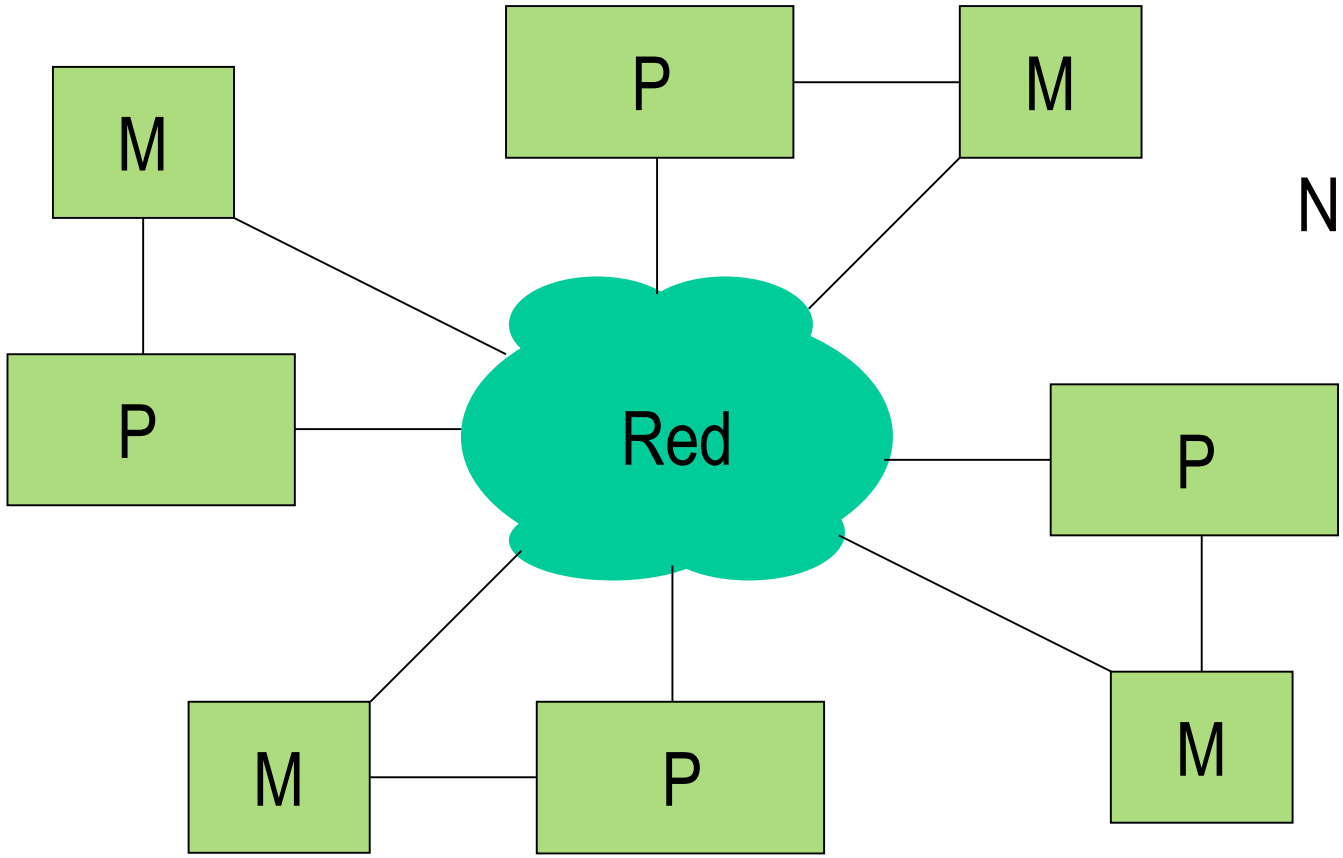
Memoria centralizada y compartida



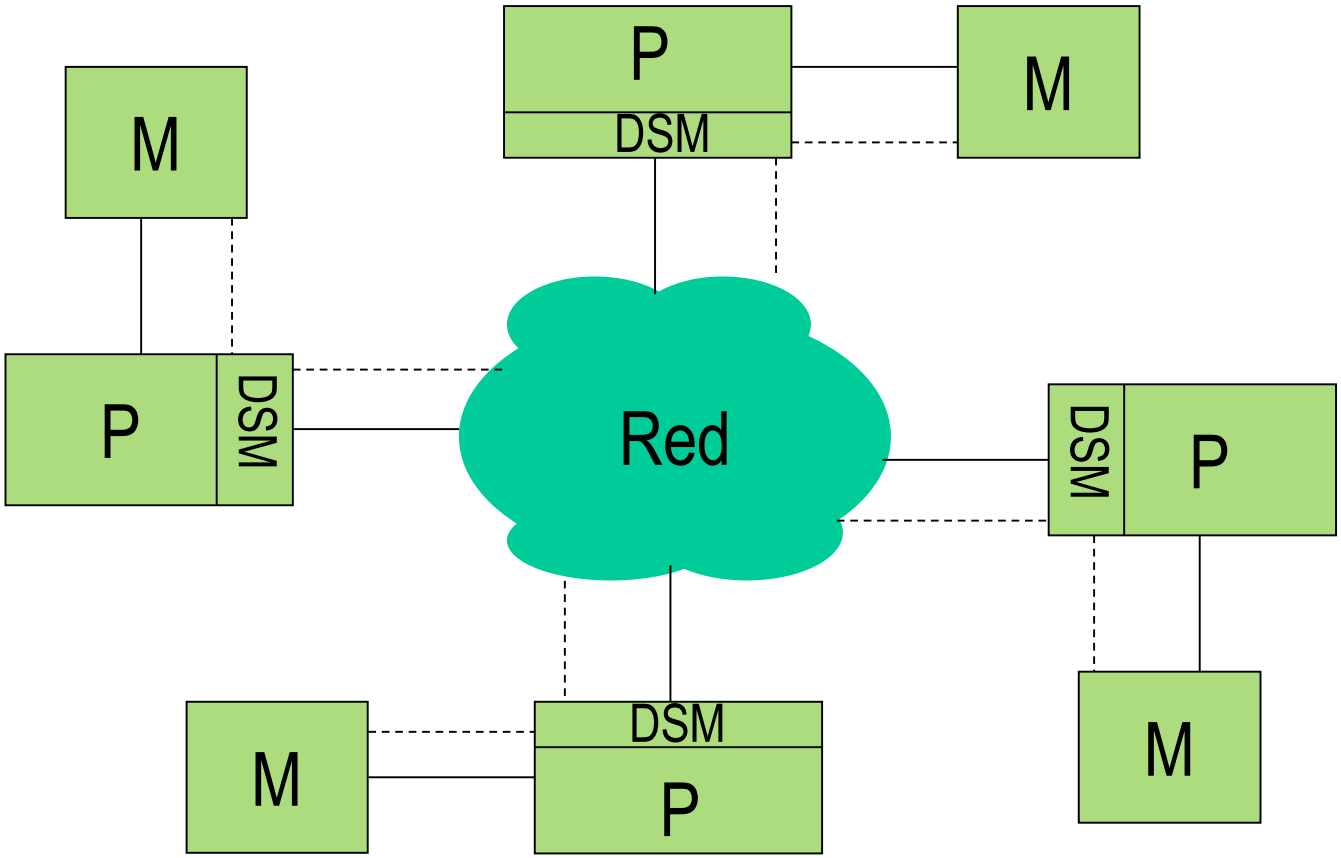
SD: Memoria distribuida y privada



DSM HW: Memoria distribuida y compartida



DSM SW: Memoria distribuida y compartida



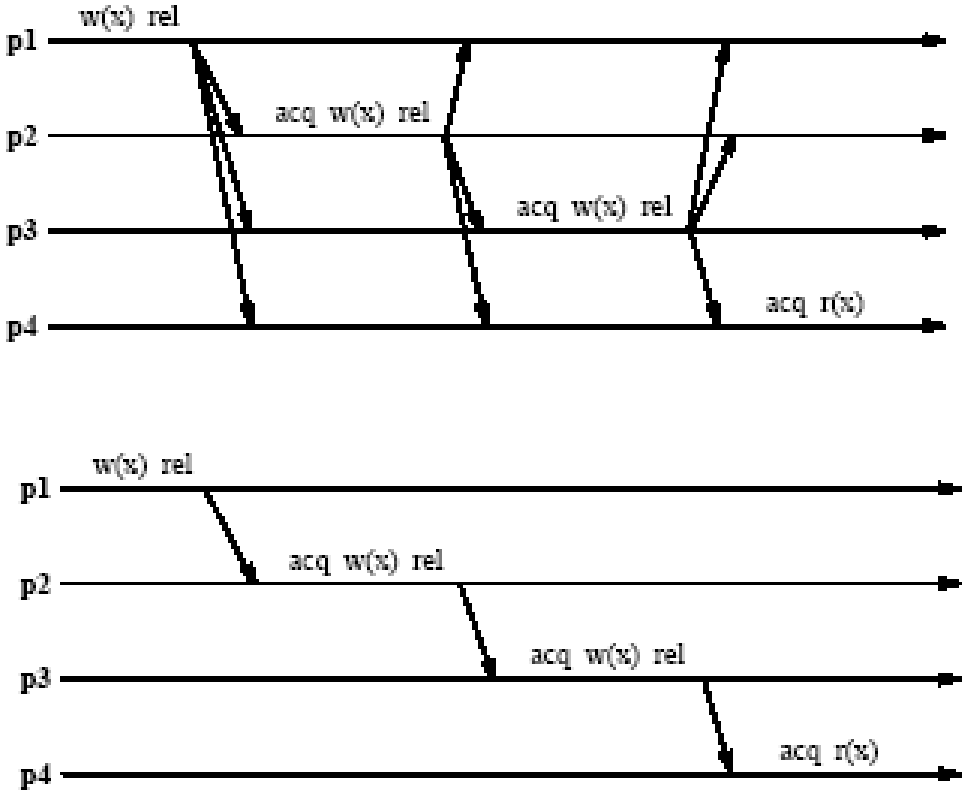
Introducción

- Multiprocesadores con mem. compartida vs. s. distribuidos:
  - HW más complejo y menos *escalable*; SW más sencillo y conocido
- Modelo de programación en sistemas con mem. compartida
  - Llamadas a procedimiento (o invocación de métodos)
  - Comunicación mediante datos compartidos
  - Sincronización mediante semáforos o mecanismos equivalentes
- Modelo de programación tradicional en s. distribuidos
  - Paso de mensajes para comunicación y sincronización
- Nuevo modelo de programación en SD: RPC (o RMI) + DSM
  - Ejecutar en SD aplicación paralela basada en m.comp.
- Dos implementaciones alternativas:
  - VM-DSM: basada m. virtual → fallo página se sirve de nodo remoto
  - RT-DSM: la gestiona el *runtime* del lenguaje

Modelos de coherencia

- Si cada escritura en memoria genera mensaje: DSM inviable
- ¿Cómo lograr implementación relativamente eficiente?
  - Programa bien construido → no condiciones de carrera
    - Accesos conflictivos a datos deben usar sección crítica (SC)
  - Sólo asegurar coherencia de los datos dentro de la SC
  - Articular la propagación de los cambios con entrar o salir de la SC
- 3 esquemas habituales:
  - *Eager Release Consistency* (ERC)
    - Al salir de SC se propagan a todos cambios en variables compartidas
  - *Lazy Release Consistency* (LRC)
    - Al entrar en SC se contacta con último proceso que estuvo en SC
      - Se le solicitan todos los cambios sobre las variables compartidas
  - *Entry Consistency* (EC): similar a LRC pero
    - Toda variable compartida está asociada a un cerrojo
    - Al entrar SC de un cerrojo, se contacta con último proceso que lo usó
      - Se le solicitan sólo cambios sobre v. compartidas asociadas a ese cerrojo

ERC vs. LRC



Lazy Release Consistency for Distributed Shared Memory.

Tesis de Peter Keleher, 1995

Coherencia de entrada (EC)

```
int main() {
    shared tipo1 v1; shared tipo2 v2; shared tipo3 v3;…
    cerrojo c1, c2;
    ……..
    asociar(c1, v1);
    asociar(c1, v2);
    asociar(c2, v4);
    ……..
    adquirir(c1);  // entrada en sección crítica: solicita últimos cambios en variables asociadas a c1
    ……..
    v1.xx...
    v2.yy...
    ……..
    liberar(c1);
    ……..}
```