



Taller de Programación



AGENDA

Evolución de Arquitecturas

Conceptos de Concurrencia

Ejemplos

Ambiente CMRE



NUESTRA VIDA –



Navegadores

**Cuentas
Bancarias**



**Sistemas
Operativos**



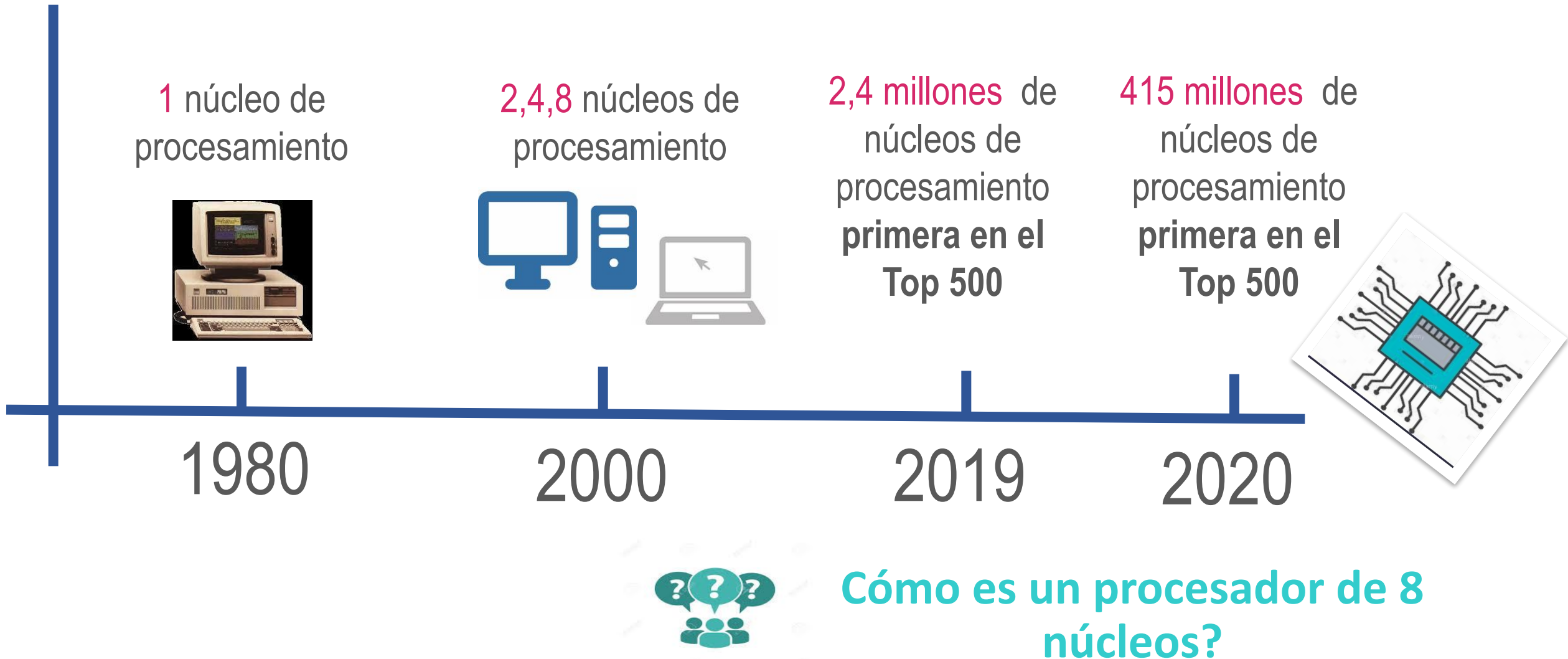
Smartphones



**Qué características
comunes hay en estos
ejemplos?**

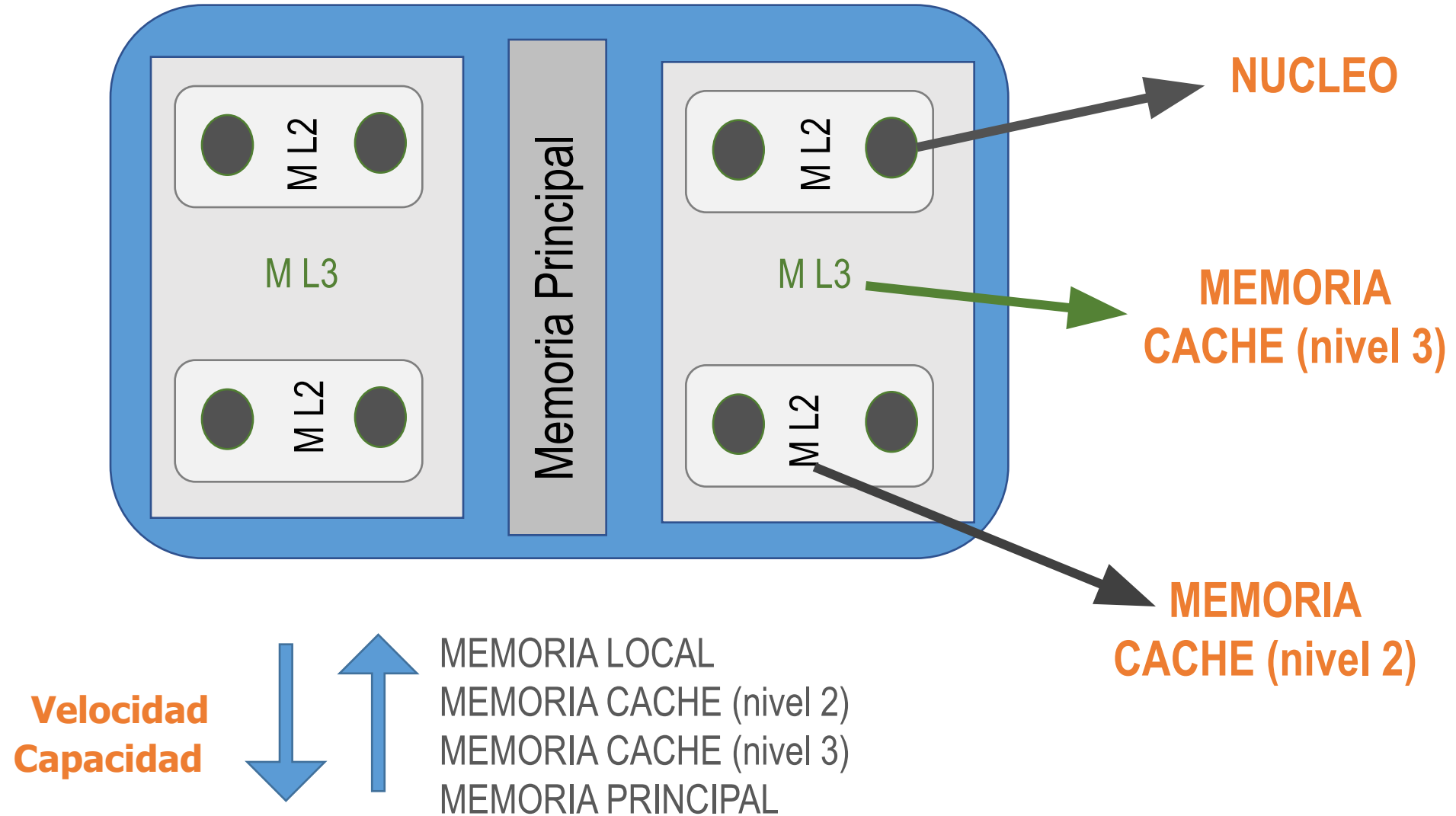


Evolución de las Arquitecturas



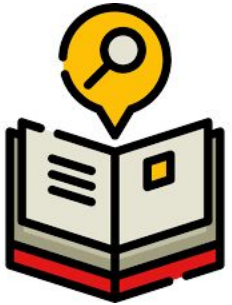


Evolución de las Arquitecturas



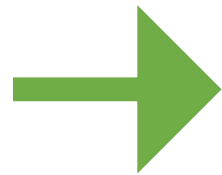


CONCURRENCIA



Un **programa concurrente** se divide en tareas (2 o más), las cuales se ejecutan al mismo tiempo y realizan acciones para cumplir un objetivo común. Para esto pueden: compartir recursos, coordinarse y cooperar.

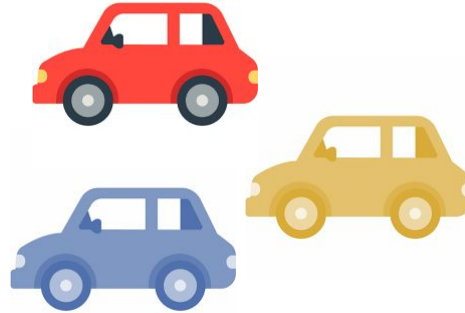
**COMUNICACIÓN
SINCRONIZACIÓN**



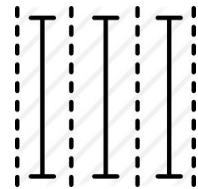
Concepto clave en la Ciencia de la Computación
Cambios en HARDWARE y SOFTWARE



CONCURRENCIA ...



Son los procesos que se ejecutan



Son los múltiples procesadores

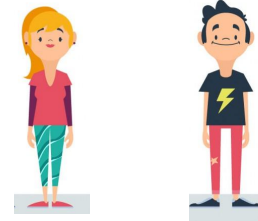
Se debe coordinar/sincronizar para que los autos no choquen

Objetivo: examinar los tipos de autos (procesos), trayecto a recorrer (programas), caminos (hardware), y reglas (comunicación y sincronización).



CONCURRENCIA - Ejemplos

Supongamos que nuestra pareja de Paula y Juan comparten una cuenta bancaria.



CUENTA BANCARIA

Supongamos ahora que ambos salen a sus trabajos y deciden detenerse en un cajero para extraer 1000 pesos



Si en la cuenta hay 50000 pesos es de esperar que después de las dos extracciones queden 48000.

¿Qué ocurre si ambos se conectan al mismo tiempo ?

CONCURRENCIA



CONCURRENCIA - Ejemplos

Variable Compartida

CUENTA BANCARIA: saldo



Integrante 1:

```
{  
  accede a la cuenta  
  saldo:= saldo - 1000;  
}
```



¿Cómo se protege
la variable saldo?

Integrante 2:

```
{  
  accede a la cuenta  
  saldo:= saldo - 1000;  
}
```



Cualquier lenguaje que brinde
conurrencia debe proveer
mecanismos para **comunicar** y
sincronizar procesos.



En este caso quiero **proteger** el
acceso a la variable compartida (dos
procesos no accedan al mismo
tiempo, sincronicen)

Semáforos (P y V)
Monitores
Pasaje de Mensajes



CONCURRENCIA - Ejemplos

CUENTA BANCARIA: saldo

Variable Compartida

Integrante 1:

```
{  
  P(saldo)  
  accede a la cuenta  
  saldo := saldo - 1000;  
  V(saldo)  
}
```



¿Cómo funciona?

Integrante 2:

```
{  
  P(saldo)  
  accede a la cuenta  
  saldo := saldo - 1000;  
  V(saldo)  
}
```



¿Este código puede ser más eficiente?



CONCURRENCIA - Ejemplos

CUENTA BANCARIA: saldo

Variable Compartida

Integrante 1:

```
{
  accede a la cuenta
  P(saldo)
  saldo := saldo - 1000;
  V(saldo)
}
```



¿Cómo funciona?

Integrante 2:

```
{
  accede a la cuenta
  P(saldo)
  saldo := saldo - 1000;
  V(saldo)
}
```



¿Alcanza si hago el
cambio en uno de los
dos integrantes?



CONCURRENCIA - Ejemplos



En un programa existen 3 procesos, un arreglo de longitud M y un valor N y se quiere calcular cuántas veces aparece el valor N en el arreglo.



cont

V



Proceso 1

Proceso 2

Proceso 3

Dado el siguiente código para cada proceso, ¿cómo se puede mejorar?

```
Proceso 1:  
{inf:=...; sup:= ...;  
  P(cont)  
  for i:= inf to sup do  
    if v[i] = N then  
      cont:= cont + 1;  
  V(cont)  
}
```

```
Proceso 2:  
{inf:=...; sup:= ...;  
  P(cont)  
  for i:= inf to sup do  
    if v[i] = N then  
      cont:= cont + 1;  
  V(cont)  
}
```

```
Proceso 3:  
{inf:=...; sup:= ...;  
  P(cont)  
  for i:= inf to sup do  
    if v[i] = N then  
      cont:= cont + 1;  
  V(cont)  
}
```



PROGRAMA CONCURRENTE - Características

Programa Secuencial

```
<!-- HTML: <title>
<meta name="description" content="HTML Tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target="_top">
<style type="text/css" media="screen">@import "/us.css";</style>
<link rel="stylesheet" type="text/css" href="/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
<link rel="search" type="application/opensearch+xml" title="HTML So
htmlsource-search.xml">
<script>
</script>
<script src="/scripts.js" type="text/javascript"></script>
<style type="text/css">
</style>
</-->
```



Programa Concurrente

```
<!-- HTML: <title>
<meta name="description" content="HTML Tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target="_top">
```

```
<style type="text/css" media="screen">@import "/us.css";</style>
<link rel="stylesheet" type="text/css" href="/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
<link rel="search" type="application/opensearch+xml" title="HTML So
htmlsource-search.xml">
<script>
</script>
```



Programa Paralelo

```
<!-- HTML: <title>
<meta name="description" content="HTML Tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target="_top">
```



```
<style type="text/css" media="screen">@import "/us.css";</style>
<link rel="stylesheet" type="text/css" href="/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
<link rel="search" type="application/opensearch+xml" title="HTML So
htmlsource-search.xml">
<script>
</script>
```



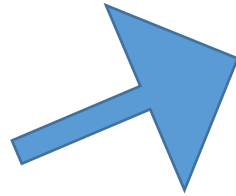


PROGRAMA CONCURRENTE - Características

Programa
Concurrente

```
<meta name="description" content="HTML tutorial">  
<meta name="author" content="Andrew">  
<meta name="copyright" content="2008-2011 and beyond...">  
<meta name="robots" content="all">  
<meta name="viewport" content="width=788">  
<base target="_top">
```

```
<style type="text/css" media="all">  
<link rel="stylesheet" type="text/css" href="/us.css">  
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">  
<link rel="search" type="application/opensearch+xml" href="/opensearch.xml" title="HTML So...>  
<script>  
</script>
```



COMUNICACIÓN



SINCRONIZACIÓN





PROGRAMA CONCURRENTE - Comunicación

Programa Concurrente

```
<html><head><title>
<meta name="description" content="HTML tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target=""></head>
```

```
<base target="_top">
<style type="text/css" media="e">@import "/us.css";</style>
<link rel="stylesheet" type="te" ss" href="/print.css" media="
<link rel="shortcut icon" type="e/ico" href="/favicon.ico">
<link rel="search" type="applic" opensearch" title="HTML So
htmlsource-search.xml">
<script>
</script>
```



proceso 1



proceso 2



ENVÍO DE
MENSAJES

MEMORIA
COMPARTIDA



PROGRAMA CONCURRENTE - Comunicación

Programa Concurrente

```
<!-- HTML: head, title -->
<meta name="description" content="HTML tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target="">
```

```
<!-- CSS: top -->
<style type="text/css" media="all">
<link rel="stylesheet" type="text/css" href="/us.css">
<link rel="stylesheet" type="text/css" href="/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
<link rel="search" type="application/opensearch+xml" title="HTML Search" href="/search.xml">
</script>
```



Forma de un mensaje

Origen
Destino
Contenido

ENVÍO DE MENSAJES

ENVIAR
RECIBIR

Es necesario establecer un canal (lógico o físico) para transmitir información entre procesos.

También el lenguaje debe proveer un protocolo adecuado.

Para que la comunicación sea efectiva los procesos deben “saber” cuándo tienen mensajes para leer y cuando deben transmitir mensajes.



PROGRAMA CONCURRENTE - Comunicación

Programa Concurrente

```
<meta name="description" content="HTML tutorial">
<meta name="author" content="Andrew">
<meta name="copyright" content="2000-2011 and beyond...">
<meta name="robots" content="all">
<meta name="viewport" content="width=780">
<base target="">
```

```
<script type="text/css" media="all">
<link rel="stylesheet" type="text/css" href="/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
<link rel="search" type="application/x-sparql+xml" href="/search.xml">
</script>
```



Recurso Compartido

LIBRE?

si

no

Bloqueo
Uso
Libero

MEMORIA COMPARTIDA

**BLOQUEAR
DESBOQUEAR**

Los procesos intercambian información sobre la memoria compartida o actúan coordinadamente sobre datos residentes en ella.

Lógicamente no pueden operar simultáneamente sobre la memoria compartida, lo que obliga a bloquear y liberar el acceso a la memoria.

La solución más elemental es una variable de control que habilite o no el acceso de un proceso a la memoria compartida.



PROGRAMA CONCURRENTE – AMBIENTE CMRE

The screenshot shows the CMRE environment interface. On the left, there is a sidebar with controls for 'Elemento' (Flores), 'Avenida', 'Calle', 'Cantidad' (0), and an 'Agregar' button. Below this is a 'ROBOTS' section with tabs for 'Robot', 'Flores', 'Papeles', and 'Color'. A 'Miniatura...' section is also present. The main area is a large grid representing the 'Ciudad'. On the right, there is a panel titled 'ROBOTS EN EJECUCION' which contains a list of robots and their status. A green box highlights the top toolbar with icons for file operations and execution, with an arrow pointing to the text 'Guardar, compilar, ejecutar los programas'. A yellow arrow points from the main grid area to the text 'Código del programa'. A red arrow points from the 'ROBOTS EN EJECUCION' panel to the text 'Información sobre los robots'. A blue arrow points from the 'Ciudad' label to the main grid area.

Guardar, compilar, ejecutar los programas

Código del programa

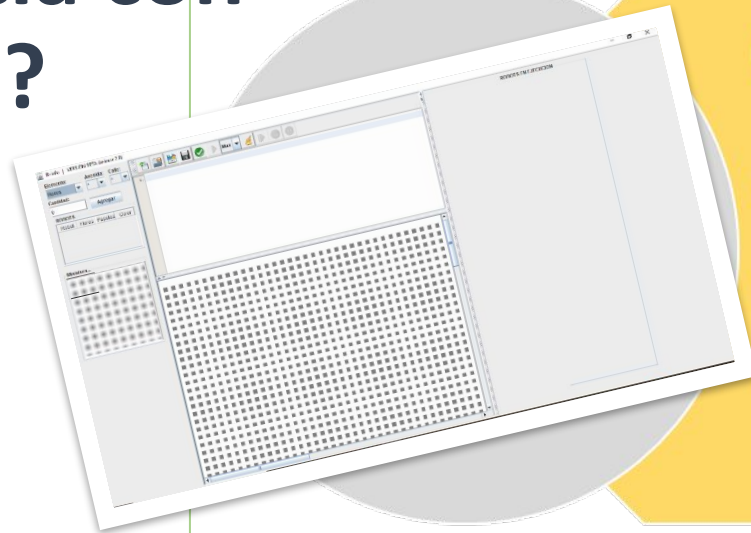
Información sobre los robots

Ciudad



PROGRAMA CONCURRENTE – AMBIENTE CMRE

**¿Cómo se relacionan
los conceptos de
conurrencia con
CMRE?**

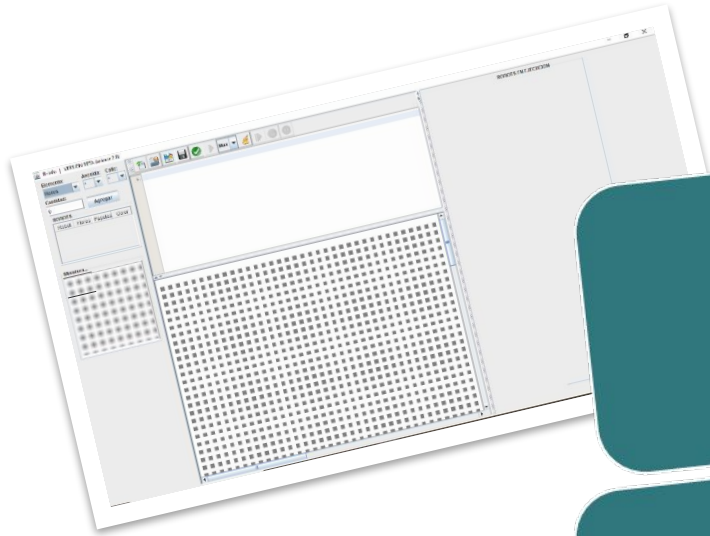


Conceptos

- Recursos Compartidos
- Sincronización
- Procesadores heterogéneos



PROGRAMA CONCURRENTE – AMBIENTE CMRE



Robots

- Se permite declarar más de un robot.

Areas

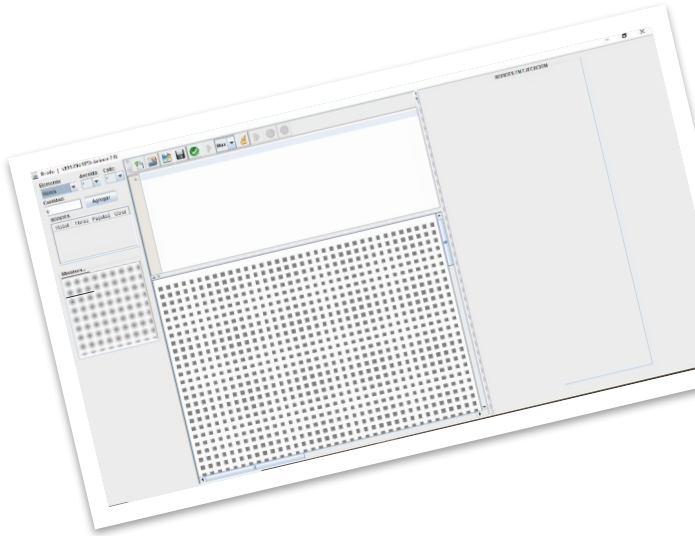
- Areas privadas, compartidas y parcialmente compartidas

Comunicación y
Sincronización

- Enviar y recibir mensajes
- Bloquear y desbloquear esquina



PROGRAMA CONCURRENTE – AMBIENTE CMRE



programa nombre

procesos

// Procesos utilizados por los robots

areas

// Áreas de la ciudad

robots

// Robots del programa

variables

// Variables robots

comenzar

// Asignación de áreas

// Inicialización de robots

fin

Estructura de un programa



PROGRAMA CONCURRENTE – AMBIENTE CMRE



```
programa nombre
```

```
procesos
```

```
// Procesos utilizados por los robots
```

```
areas
```

```
// Áreas de la ciudad
```

```
robots
```

```
// Robots del programa
```

```
variables
```

```
// Variables robots
```

```
comenzar
```

```
// Asignación de áreas
```

```
// Inicialización de robots
```

```
fin
```

```
proceso nombre (ES flores:numero;E valor:boolean)  
variables
```

```
    nombre : tipo
```

```
comenzar
```

```
    //código del proceso
```

```
fin
```





PROGRAMA CONCURRENTE – AMBIENTE CMRE

programa nombre

procesos

// Procesos utilizados por los robots

areas

// Áreas de la ciudad

robots

// Robots del programa

variables

// Variables robots

comenzar

// Asignación de áreas

// Inicialización de robots

fin

ciudad1: areaC(1,1,10,10) //área Compartida

ciudad2: areaP(15,15,20,20) //área Privada

ciudad3: areaPC(30,32,50,51) //área Parcialmente compartida

areaC
Compartida

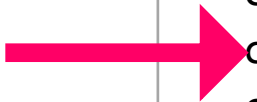
Cualquier robot pueden circular por la misma

areaP
Privada

Sólo puede haber en ella un único robot

areaPC
Parc. Comp.

Se debe seleccionar qué subconjunto de robots pueden circular por la misma





PROGRAMA CONCURRENTE – AMBIENTE CMRE

R-info | VERSIÓN BETA (release 2.9)

Elemento: Flores Avenida: Calle: Cantidad: 0 Agregar

ROBOTS

Robot	Flores	Papeles	Color
robot1	0	0	Red
robot2	0	0	Blue
robot3	0	0	Magenta

Miniatura...

```
1. programa areasEjemplo
2. areas
3.   areal: AreaC(1,1,10,10)
4.   area2: AreaP(12,1,15,10)
5.   area3: AreaPC(17,1,30,10)
6. robots
7.   robot florero
8.   variables
9.     avenida:numero
10.    calle:numero
11.  comenzar
12.    avenida2:=PosAv
13.    calle2:=PosCa
14.  fin
15. variables
16.   robot1:florero
```

Área compartida

Área privada

Área parcialmente compartida

ROBOTS EN EJECUCION

robot1 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

robot2 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

robot3 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

Windows taskbar: Escribe aquí para buscar, 8:31, 17/10/2019



PROGRAMA CONCURRENTE – AMBIENTE CMRE



```
programa nombre  
  
procesos  
    // Procesos utilizados por los robots  
  
areas  
    // Áreas de la ciudad  
  
robots  
    // Robots del programa  
  
variables  
    // Variables robots  
  
comenzar  
    // Asignación de áreas  
    // Inicialización de robots  
fin
```



```
robot tipo1  
variables  
    ...  
comenzar  
    // Código del robot 1  
fin
```



PROGRAMA CONCURRENTE – AMBIENTE CMRE



```
programa nombre
```

```
procesos
```

```
// Procesos utilizados por los robots
```

```
areas
```

```
// Áreas de la ciudad
```

```
robots
```

```
// Robots del programa
```

```
variables
```

```
// Variables robots
```

```
comenzar
```

```
// Asignación de áreas
```

```
// Inicialización de robots
```

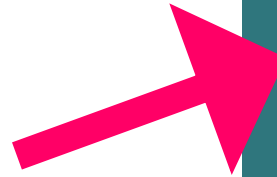
```
fin
```

```
r1: tipo1
```

```
r2: tipo1
```

```
r1: tipo1
```

```
r2: tipo2
```





PROGRAMA CONCURRENTE – AMBIENTE CMRE



R-info | VERSIÓN BETA (release 2.9)

Elemento: Flores Avenida: * Calle: *

Cantidad:

0

Agregar

ROBOTS

Robot	Flores	Papeles	Color
robot1	0	0	Red
robot2	0	0	Blue
robot3	0	0	Magenta

Miniatura...

```
7. robot florero
8. variables
9.   avenida:numero
10.  calle:numero
11. comenzar
12.   avenida2:=PosAv
13.   calle2:=PosCa
14. fin
15. variables
16.   robot1:florero
17.   robot2:florero
18.   robot3:florero
19. comenzar
20.   AsignarArea(robot1,areal)
21.   AsignarArea(robot2,area2)
22.   AsignarArea(robot3,area3)
```

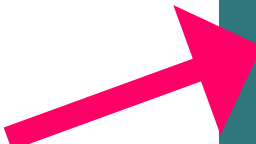


PROGRAMA CONCURRENTE – AMBIENTE CMRE



```
programa nombre
procesos
    // Procesos utilizados por los robots
areas
    // Áreas de la ciudad
robots
    // Robots del programa
variables
    // Variables robots
comenzar
    // Asignación de áreas
    // Inicialización de robots
fin
```

Un robot puede estar
asignado a 1 o más
de un áreas del
programa



```
//AsignarArea(variableRobot,nombreArea)
AsignarArea(r1,ciudad1)
iniciar(r1, 5, 5)
```



EJERCITACION – Clase teórica



Para poder realizar esta actividad en el horario de teoría el alumno tiene que haber instalado el entorno en su computadora

Analice la solución presentada en el **Ejercicio1-a**. Qué hace? Es correcta?.

Analice la solución presentada en el **Ejercicio1-b**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-c**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-d**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-e**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-f**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-g**. Qué hace? Es correcta?



EJERCITACION – Clase teórica



Estos ejercicios se dejan planteados para que el alumno los analice y se relacionan con lo que se verá la siguiente clase

Ejercicio 1-ha: Realice un programa donde un robot recorra el perímetro de un rectángulo de un tamaño 5 (alto) x 3 (ancho) juntando flores. Al finalizar informe las flores juntadas. Inicialmente el robot se encuentra en la esquina (2,2). **Debe modularizar el rectángulo. El rectángulo debe recibir alto y ancho y devolver las flores.**

Ejercicio 1-hb: Realice un programa donde dos robots recorren el perímetro de un rectángulo de un tamaño 5 (alto) x 3 (ancho) juntando flores. Al finalizar informe las flores juntadas por cada uno. Inicialmente los robots se encuentran en la esquina (2,2) y (6,2) respectivamente. **Debe modularizar el rectángulo. El rectángulo debe recibir alto y ancho y devolver las flores**



Ejercicio 1-hc: Qué tiene que cambiar en su código si el robot 1 debe realizar un rectángulo de 5 (alto) x 3 (ancho) juntando flores y el robot 2 un rectángulo de 8 (alto) x 2 (ancho) juntando flores. **El rectángulo debe recibir alto y ancho y devolver las flores**