



# Taller de Programación



# AGENDA

Evolución de Arquitecturas

Conceptos de Concurrencia

Ejemplos



# NUESTRA VIDA – Hoy...



NAVEGADORES



SAMARTPHONE

SISTEMAS  
OPERATIVOS



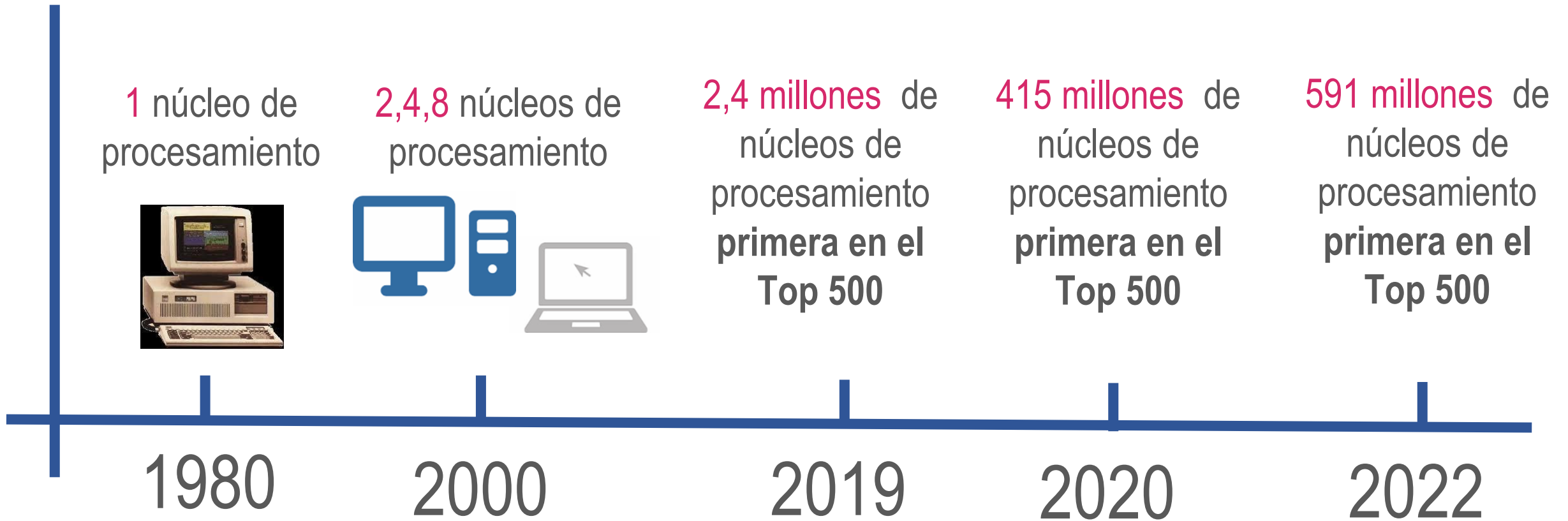
CUENTAS  
BANCARIAS



Qué  
características  
comunes hay en  
estos ejemplos?



# Evolución de las Arquitecturas

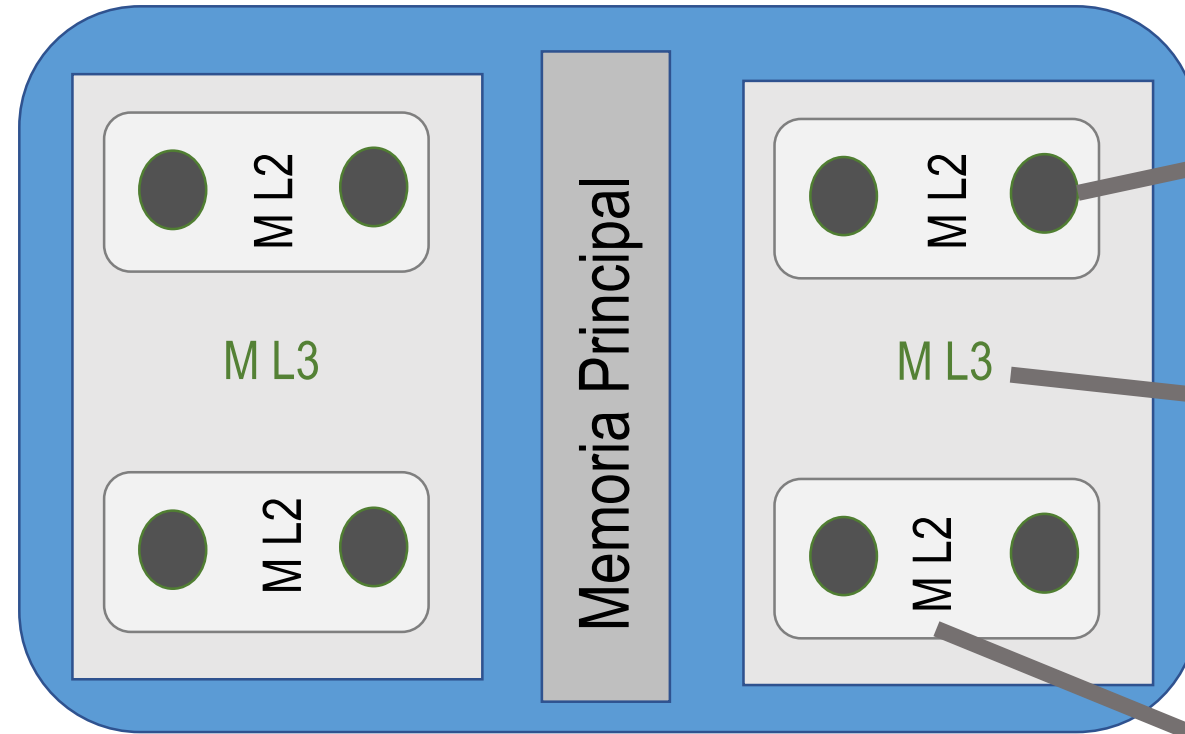


**Cómo es un procesador con más de un núcleo?**



# Evolución de las Arquitecturas

8  
NÚCLEOS



NÚCLEO

MEMORIA  
CACHE  
(nivel3)

MEMORIA  
CACHE  
(nivel2)

VELOCIDAD



MEMORIA LOCAL  
MEMORIA CACHE (nivel 2)  
MEMORIA CACHE (nivel 3)  
MEMORIA PRINCIPAL

CAPACIDAD





# 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.

## CARACTERISTICAS

Concepto clave en la Ciencia de la Computación

Cambios en HARDWARE y SOFTWARE

## CONCEPTOS

**COMUNICACIÓN**  
**SINCRONIZACION**



# CONCURRENCIA - Ejemplos

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



En algún momento 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.

**CUENTA  
BANCARIA**



**Podría ocurrir que  
ambos accedan a la  
cuenta en el mismo  
instante**

**CONCURRENCIA**





# CONCURRENCIA - Ejemplos

**CUENTA**  
**saldo**

Integrante 1:

```
{  
  ingresa la clave  
  saldo:= saldo - 1000;  
}
```



**BANCARIA:** VARIABLE COMPARTIDA



¿Cómo se  
protege la  
variable saldo?

Integrante 2:

```
{  
  ingresa la clave  
  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**  
**saldo**

**BANCARIA:** VARIABLE COMPARTIDA

Integrante 1:

```
{  
  P(saldo)  
  
  ingresa clave  
  
  saldo:= saldo - 1000;  
  
  V(saldo)  
}
```

¿Cómo  
funciona?

Integrante 2:

```
{  
  P(saldo)  
  
  ingresa clave  
  
  saldo:= saldo - 1000;  
  
  V(saldo)  
}
```

¿Este código puede  
ser más eficiente?




# CONCURRENCIA - Ejemplos

**CUENTA**  
**saldo**

**BANCARIA:** VARIABLE COMPARTIDA

Integrante 1:

```
{  
  ingresar clave  
  P(saldo)  
  saldo := saldo - 1000;  
  V(saldo)  
}
```



¿Cómo funciona?

Integrante 2:

```
{  
  ingresa clave  
  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

```
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)  
}
```

¿cómo se puede



# PROGRAMA CONCURRENTE - Características

## Programa Secuencial

```
<html><html:rootdir/></html:rootdir/>
<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="all"><@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>
</html>
```



## Programa Concurrente

```
<html><html:rootdir/></html:rootdir/>
<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">
```

```
<base target="_top">
<style type="text/css" media="all"><@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>
</html>
```



## Programa Paralelo

```
<html><html:rootdir/></html:rootdir/>
<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">
```

```
<base target="_top">
<style type="text/css" media="all"><@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>
</html>
```





# COMUNICACIÓN

# SINCRONIZACIÓN





# PROGRAMA CONCURRENTE - Comunicación

## Programa Concurrente

```
<html head/head>  
<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=788">  
<base target="">
```

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



proceso 1



proceso 2



COMUNICACION

PASAJE DE MENSAJES

MEMORIA COMPARTIDA



# 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=788">  
<base target="" href="">
```

```
<link target="_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>  
</script>
```



Forma de un mensaje

Origen  
Destino  
Contenido

## PASAJE 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

## MEMORIA COMPARTIDA

**BLOQUEAR  
DESBLOQUEAR**

```
<!-- HTML header for all pages -->
<!-- You can set the page title here -->
<!-- The page content goes here -->
<!-- HTML footer for all pages -->
<!-- You can set the page title here -->
<!-- The page content goes here -->
<!-- HTML footer for all pages -->
```

```
<!-- HTML header for all pages -->
<!-- You can set the page title here -->
<!-- The page content goes here -->
<!-- HTML footer for all pages -->
<!-- You can set the page title here -->
<!-- The page content goes here -->
<!-- HTML footer for all pages -->
```

Recurso  
Compartido

LIBRE?

si

no

Bloqueo  
Uso  
Libero

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.

**Cómo utilizamos**