

CC5303 – Sistemas Distribuidos

## **2.- Migración**

*Parte 1*

Sebastián Blasco V.

# Contenidos

- Procesos Servidores
- Esquemas de Ejecución
- Migración
- Agentes

# Procesos Servidores

- Se implementan para ofrecer servicios a un conjunto de clientes.
- Hay varios aspectos de diseño que deben considerarse al implementar servidores.



# Procesos Servidores: Tipos de Servidores

## Respecto a su **Concurrencia**:

- Iterativos
  - El propio servidor manipula la petición y, si es necesario, devuelve una respuesta a la petición del cliente.
  - Un solo hilo espera peticiones y las procesa.
- Concurrentes
  - No manipula por sí mismo la petición, pero la pasa a un hilo separado o a otro proceso, después de lo cual de inmediato queda en espera de la siguiente petición entrante.
  - Un hilo (o proceso) espera peticiones y ante cada petición, le pasa la petición a otro hilo (o proceso).

# Procesos Servidores: Tipos de Servidores

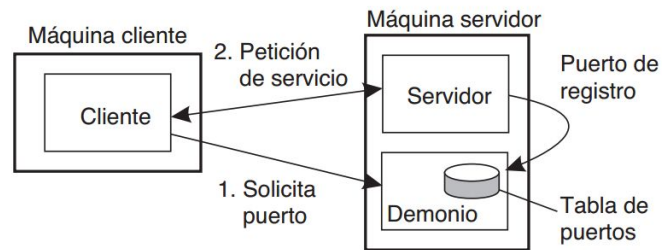
Respecto a su **Localización** (punto de entrada):

- Punto de entrada fijo y conocido
  - los clientes conocen de antemano la ubicación (como al puerto de conexión).
  - Ej. FTP, HTTP, etc.
- Punto de entrada desconocido
  - Resoluble mediante un servicio de nombres o un supervisor.

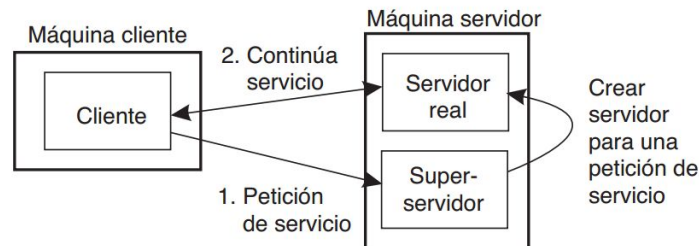
# Procesos Servidores: Tipos de Servidores

Respecto a su **Localización** (punto de entrada):

- Punto de entrada fijo y conocido (a)
- Punto de entrada desconocido (b)



(a)



(b)

# Procesos Servidores: Tipos de Servidores

Respecto al **Control** que se puede ejercer sobre ellos:

- Sin Control
  - Los clientes que quieren desconectar, simplemente abortan la conexión, por ejemplo terminando la propia aplicación cliente
- Con Control
  - Se establece un canal de comunicación adicional entre cliente y servidor, de forma que el cliente puede actuar sobre el servidor: para detenerlo, pararlo, reanudarlo, etc.

# Procesos Servidores: Tipos de Servidores

## Respecto a su **estado interno**:

- Sin estado
  - Los servidores no guardan información respecto al estado de los clientes ni respecto a su conversación
  - Ej. un servidor HTTP
- Con estado
  - El servidor guarda información de cada cliente. Ante caídas y recuperaciones del servidor, se debe recuperar la información de cada cliente.



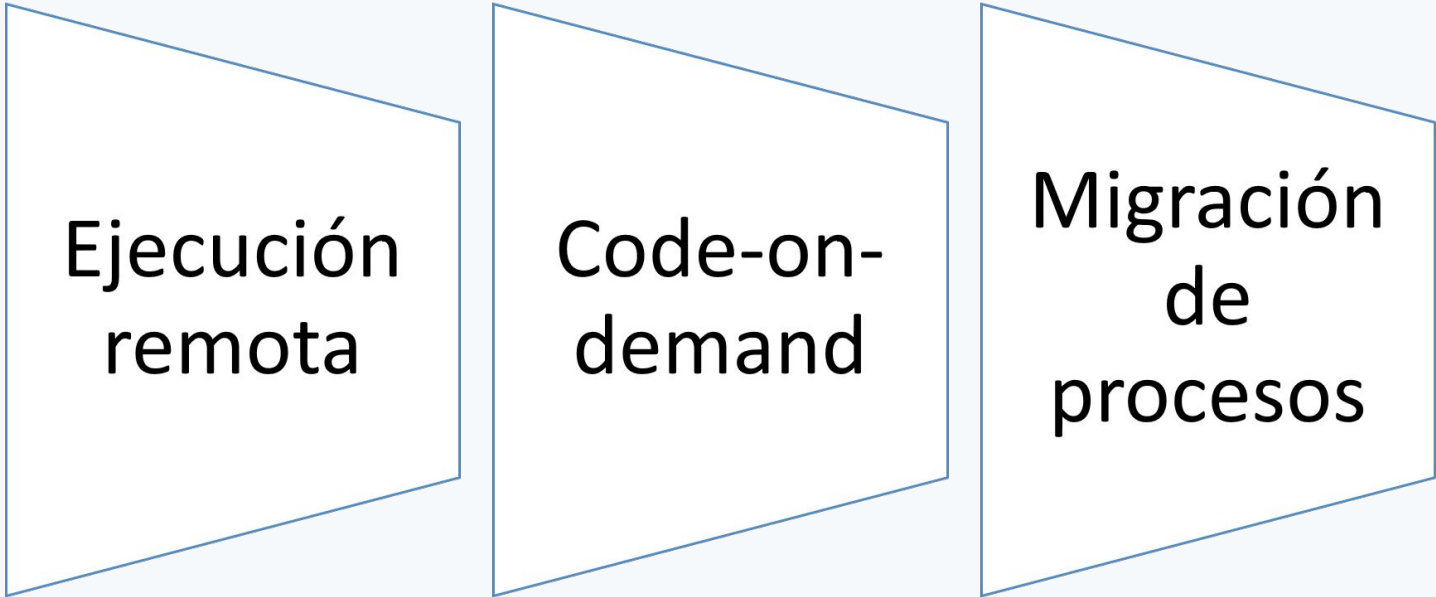
# Procesos Servidores



El esquema cliente servidor tradicional está pensando para el **paso de datos**

Hay veces en que es mejor realizar un **paso de programas**, que deben desarrollar distintas tareas según donde se ejecuten. Esto se conoce como *migración de código* y depende de los llamados *esquemas de ejecución*.

# Esquemas de Ejecución



Ejecución remota

Code-on-demand

Migración de procesos

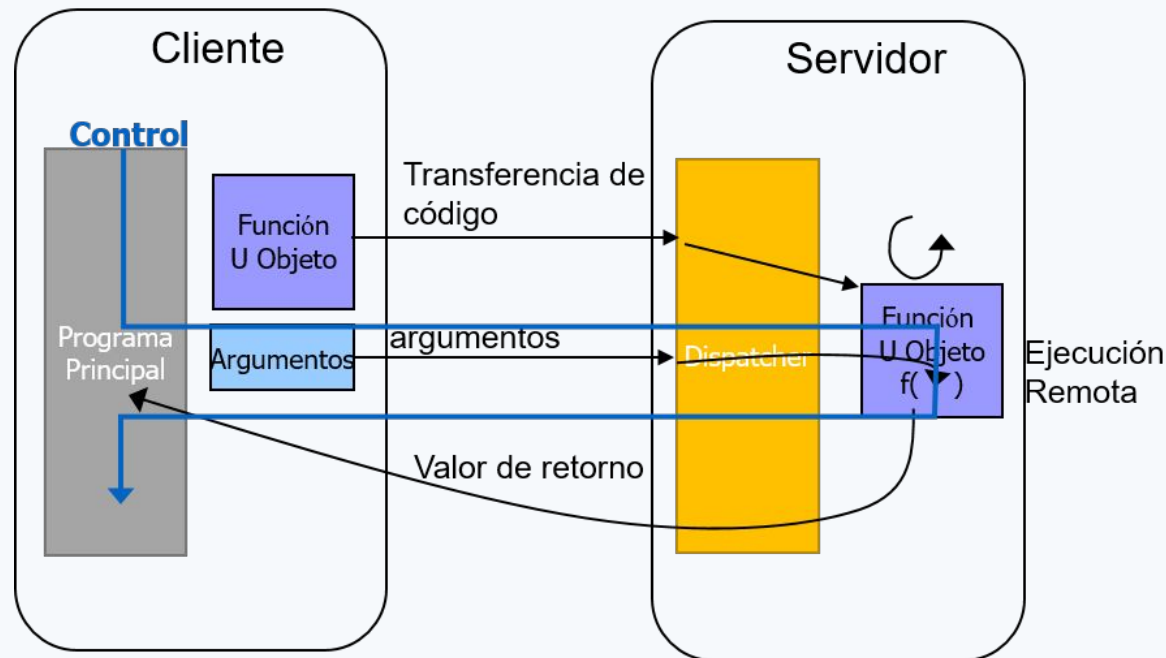
# Esquemas de Ejecución

Ejecución remota

Code-on-demand

Migración de procesos

- Ejecución Remota
  - El código es enviado junto con los argumentos
  - El servidor actúa como un típico servidor cíclico



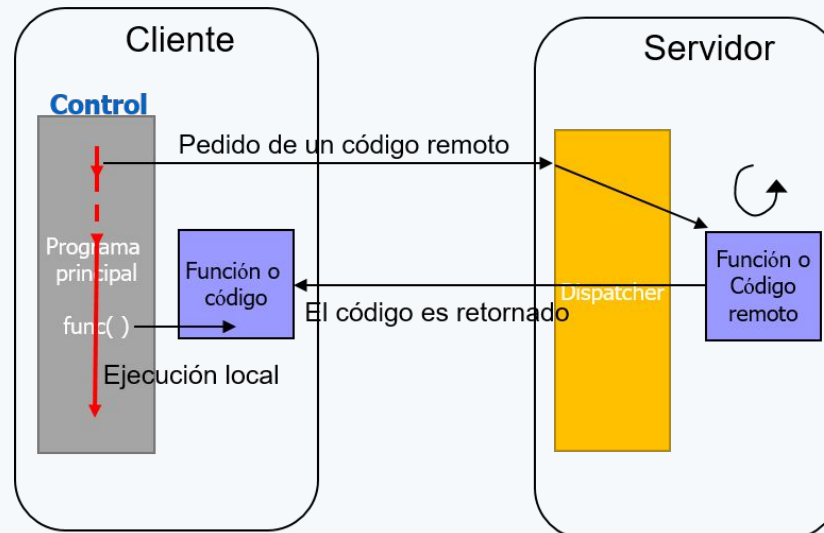
# Esquemas de Ejecución

Ejecución remota

Code-on-demand

Migración de procesos

- Code-on-Demand
  - El servidor actúa como un servidor de códigos remotos
  - El código remoto es llevado al cliente
  - El cliente lo ejecuta localmente
  - El control de ejecución permanece en el cliente mientras está suspendido por la descarga del código.



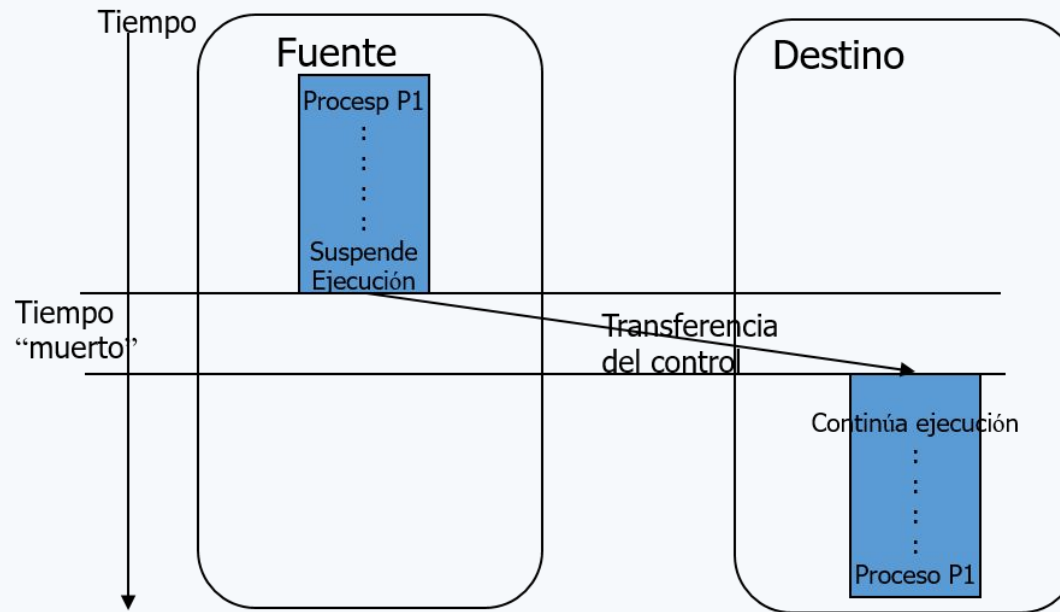
# Esquemas de Ejecución

Ejecución remota

Code-on-demand

Migración de procesos

- Migración de Procesos



# Esquemas de Ejecución

Ejecución  
remota

Code-on-  
demand

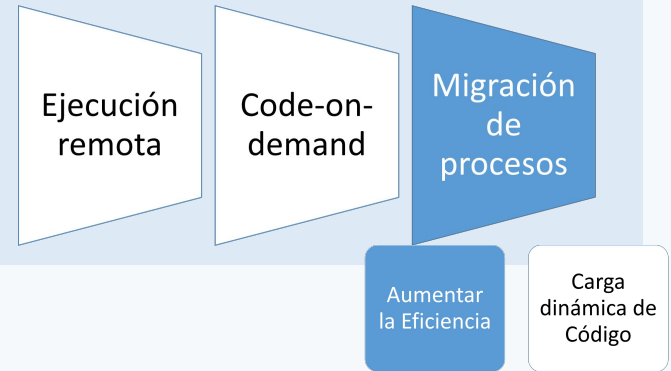
Migración  
de  
procesos

- Migración de Procesos
  - Motivos para migrar código?

Aumentar  
la Eficiencia

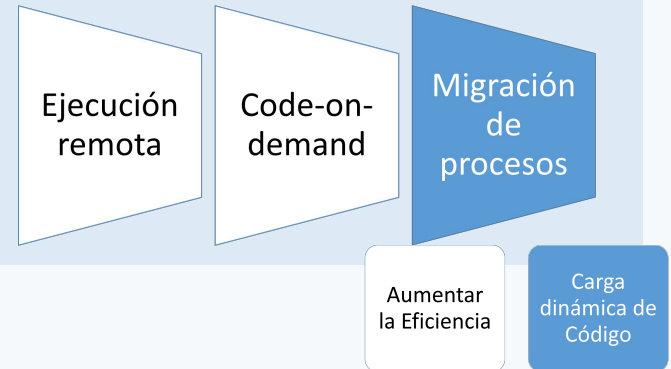
Carga  
dinámica de  
Código

# Esquemas de Ejecución



- Aumentar la eficiencia
  - Repartir carga computacional
    - Si una máquina tiene mucha carga, se observará una mejora global del rendimiento del sistema distribuido, si se migra su código a otra máquina con menos carga.
  - Disminuir carga de la red de comunicaciones
    - Si se observa mucho tráfico entre dos máquinas debida a la interacción entre dos “procesos”, puede ser conveniente migrar uno de ellos a la máquina del otro.
  - Aumentar la velocidad de procesamiento
    - Distribuyendo el procesamiento entre máquinas para aumentar el grado de paralelismo.

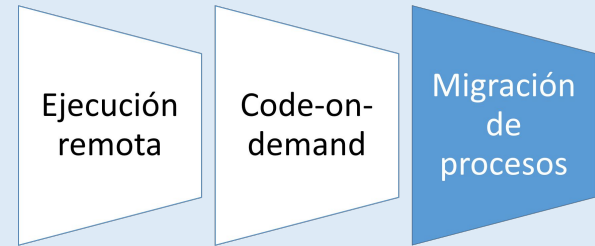
# Esquemas de Ejecución



- Permitir la carga dinámica de código:
  - Uso de partes del código no conocidas a priori
    - Por ejemplo podemos construir clientes que hagan uso de protocolos genéricos de comunicación con los servidores, pero que carguen del servidor el protocolo específico que se utiliza en cada momento.
  - Mejorar la distribución/instalación del código en sistemas grandes
    - Muchas veces no es posible instalar una aplicación en todo un sistema, de forma que la mejor opción consiste en que los clientes descarguen el código que deben ejecutar conforme lo usen.

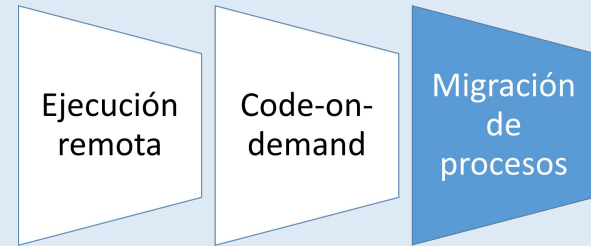


# Esquemas de Ejecución



- Vamos a migrar, pero... ¿Qué es un programa finalmente?
  - Segmento de Código
    - Con las instrucciones de código fuente
  - Segmento de Ejecución
    - Con el estado y contexto de valores empleados en la ejecución. Ej. variables, pila, contadores, etc.
  - Segmento de Recursos
    - Con las referencias a los recursos externos empleados por el programa. Ej. archivos, impresoras, etc.

# Esquemas de Ejecución



Una operación de migración implica tratar la relocación de los 3 segmentos.

Segmento de Código | Segmento de Ejecución | **Segmento de Recursos**

- Migrar el segmento de recursos muchas veces puede ser problemático. Por ejemplo migrar la referencia a un socket abierto TCP de una máquina a otra.
- La clave para entender las posibilidades de migración del segmento de recursos radica en el enlace que exista entre
  - el recurso y el proceso
  - el recurso y la máquina.