

## Tema 1: Introducción

Definiciones, objetivos, dificultades

Jose Luis Díaz  
Curso 2011-2012

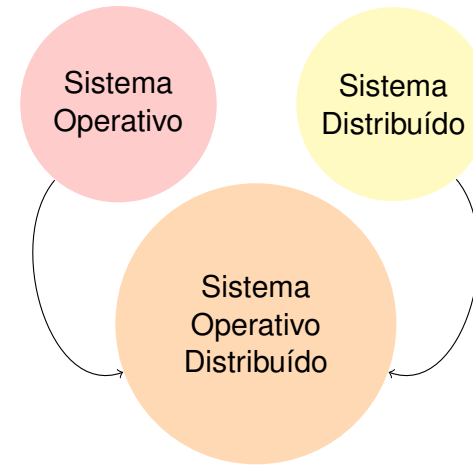
## Contenidos

- 1 Definición
  - Ejemplos de Sistema Distribuido
- 2 Características de los Sistemas Distribuídos
  - ¿Por qué hacerlos?
  - Objetivos (o retos)
- 3 Arquitectura de los Sistemas Distribuidos
  - Componentes
- 4 Conclusiones

## Esquema

- 1 Definición
  - Ejemplos de Sistema Distribuido
- 2 Características de los Sistemas Distribuidos
- 3 Arquitectura de los Sistemas Distribuidos
- 4 Conclusiones

## ¿Qué es un sistema operativo distribuido?



## Sistema Operativo

Es un *software* que

- Gestiona los recursos de un computador
- Proporciona al usuario (o programador) una visión de «alto nivel»

## Sistema Distribuido

### Definición de Tanenbaum

*«Conjunto de computadores independientes que ante los usuarios del sistema parecen un solo computador»*

### Definición de Colouris

*«Conjunto de computadores autónomos, unidos por una red, que ejecutan un software diseñado para dar una utilidad computacional integrada»*

### Definición de Lamport

*«Sabes que estás ante uno cuando la avería en un computador del que nunca habías oído hablar, te impide trabajar con el tuyo»*

## Sistema Distribuído

### Nuestra definición

- Varios ordenadores independientes
- Una red que los une
- Una *capa software* que permite la comunicación de datos entre procesos en diferentes máquinas
- Una aplicación o conjunto de aplicaciones que usa estos recursos de forma coordinada para resolver un problema o proporcionar una utilidad

## Conclusión

### Sistema Operativo Distribuído

Es un *software* que

- Gestiona los recursos de un *Sistema Distribuido*
- Proporciona al usuario (o programador) una visión de alto nivel del sistema distribuído

Idealmente

- Hace que el sistema aparezca ante el usuario como un solo computador

Como mínimo simplifica la creación de aplicaciones distribuídas

T1: Introducción

└ Definición

└ Ejemplos de Sistema Distribuido

## Ejemplos

Ya que la definición es muy amplia, existen muchas posibilidades, muy dispares entre sí.

- *Clusters*
- *Intranets*
- *Grid Computing*
- Redes *peer to peer* (p2p)
- Internet y el WWW

T1: Introducción

└ Definición

└ Ejemplos de Sistema Distribuido

## Clusters

Son un conjunto de computadores, generalmente homogéneo, dedicado una tarea muy específica.  
Coordinador central asigna el trabajo

### Ejemplo



- **Objetivo:** Almacenar una copia de la web y realizar búsquedas rápidas en ella.
- **Arquitectura:** 600 000 PCs de gama alta  
¡En 1999 eran solo unos pocos *Pentium II!*

T1: Introducción

└ Definición

└ Ejemplos de Sistema Distribuido

## Intranet



Red interna a una empresa que utiliza los mismos protocolos (y el mismo *software*) que el que se usa en *internet*.

Típicamente permite compartir información y otros recursos a los trabajadores de la empresa, como

- Documentos
- Bases de datos
- Impresoras
- Programas
- etc.

T1: Introducción

└ Definición

└ Ejemplos de Sistema Distribuido

## Grid Computing

Intento de utilizar los recursos computacionales poco usados de un conjunto de computadores  
No hay *coordinador central*

Ejemplo

**Folding@home** distributed computing

- **Objetivo:** Estudiar la forma en que se pliegan las proteínas (innumerables aplicaciones médicas).
- **Recursos:** CPUs o incluso GPUs de voluntarios, mientras no están haciendo otra cosa.

T1: Introducción






└ Definición

└ Ejemplos de Sistema Distribuido

## Redes *peer to peer* (P2P)

Concepto análogo al de *grid computing*, pero intentando además aprovechar recursos de almacenamiento.  
Especial hincapié en aspectos como robustez, anonimato, ...

### Ejemplos

- 
 Los sistemas P2P más populares son los de intercambio de archivos.
- 

 Concebidos para compartir información científica o distribuir rápidamente nuevas versiones de programas libres.
- 

 Usados en cambio para compartir material protegido por *copyright*.
- También existen P2P para “donar” CPU

T1: Introducción

└ Definición

└ Ejemplos de Sistema Distribuido

## Internet



## Esquema

- 1 Definición
- 2 Características de los Sistemas Distribuidos
  - ¿Por qué hacerlos?
  - Objetivos (o retos)
- 3 Arquitectura de los Sistemas Distribuidos
- 4 Conclusiones

## Ventajas

- En ocasiones, los propios recursos están distribuidos.
- O la naturaleza del trabajo exige compartir datos.
- Precio: Un conjunto de PCs suele ser más barato que un *mainframe* con una potencia equivalente.
- Escalabilidad. Si el sistema se queda pequeño, se puede ampliar agregando componentes.
- Más potencia: la potencia de un único computador está limitada.
- Tolerancia a fallos. Si se diseña convenientemente, un fallo en una de sus partes puede no afectar al conjunto.



T1: Introducción

└─ Características

└─ ¿Por qué hacerlos?

## Desventajas

- Complejidad del *software*
- La red puede ser un cuello de botella
- Seguridad: si la red es accesible, también pueden serlo los datos, incluso los que deberían ser privados o secretos

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Objetivos

(O retos, según se mire)

La construcción de un sistema distribuído plantea importantes retos al programador. No obstante son objetivos deseables para el sistema. Algunos de ellos son:

- Transparencia
- Heterogeneidad
- Fiabilidad
- Escalabilidad
- Flexibilidad
- Seguridad

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Transparencia

El objetivo máximo de la transparencia es lograr que el usuario no perciba que está trabajando con varias máquinas.

Hay varios niveles de transparencia:

- Localización.
- Replicación.
- Concurrencia.
- Paralelismo.
- Migración.
- Fallos.

T1: Introducción

└─ Características

└─ Objetivos (o retos)

### Ejemplo de transparencia

Al acceder a Google, no sabemos si es un solo supercomputador con toda su base de datos en un disco, o son 150000 PCs colaborando entre sí.

### Ejemplo de NO transparencia

Al compartir una carpeta desde Windows (montar una *unidad de red*), hay que saber el nombre de la máquina que la comparte. Una vez montada, la unidad parece un disco local (transparencia). Pero el usuario sabe que cada vez que accede a esa unidad, en realidad está accediendo a otra máquina. Si un archivo que estaba en el disco compartido se mueve a otra máquina, el usuario lo notará.

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Heterogeneidad

Los componentes de un sistema distribuido pueden ser muy diferentes unos de otros. Esta heterogeneidad afecta a:

- **Redes:** Cable, inalámbrica, fibra óptica, satélite, teléfono...
- **Hardware de computadores:** PC, Macintosh, *mainframes*, teléfonos móviles, PDAs...
- **Sistemas operativos:** Windows, Linux, Unix, Mac OS X, PalmOS...
- **Lenguajes de programación:** C, C++, C#, Java, Perl...
- **Diferentes programadores:** deben ponerse de acuerdo en un protocolo, para que sus programas puedan comunicarse.

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Fiabilidad

El desarrollador debe tener siempre presente que cualquier parte del sistema puede fallar. Esto plantea varias cuestiones:

- **Detección:** ¿puede saberse si se ha producido un fallo?
- **Corrección:** ¿qué hacer si se detecta un fallo?
- **Previsión:** ¿qué medidas pueden tomarse de antemano previendo que habrá fallos?
  - No centralizar.
  - Posibilidad de recuperación: que una máquina pueda volver al estado en que estaba cuando cayó.
  - Redundancia: varias copias de la información, o del hardware.
  - La redundancia plantea un nuevo problema: coherencia.

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Escalabilidad

La escalabilidad hace referencia a la capacidad de crecer del sistema.

Para que un sistema sea escalable debe prestar atención a:

- Prever desbordamientos de recursos software. Ejemplo, el número de bits destinado a almacenar una cantidad, que puede crecer.
- Evitar cuellos de botella debidos a recursos centralizados.
- Escalabilidad de los algoritmos. Cuando el número de datos aumenta al doble, el tiempo de una búsqueda aumentará más del doble.
- Escalabilidad del hardware. Sería deseable que al duplicar el hardware, se dupliquen las prestaciones.

T1: Introducción

└─ Características

└─ Objetivos (o retos)

## Flexibilidad

La **flexibilidad** es la capacidad de incorporar extensiones en la funcionalidad del sistema.

La clave es el uso de **sistemas abiertos**

### Ejemplo: WWW

- Función inicial: servir documentos
  - Extensiones:
    - Otros tipos de información
    - Aplicaciones más “interactivas”
    - Servicios Web
    - Lo que está por venir...
- Basados en estándares (contrapuesto a “tecnología propietaria”)
  - Interfaces públicos
  - Código fuente disponible
  - Regulado por organizaciones y no por fabricantes

T1: Introducción

└ Características

└ Objetivos (o retos)

## Seguridad

La red de comunicaciones puede estar expuesta a terceros (ej: Internet). Esto plantea problemas de seguridad:

- Obtención no autorizada de información
- Modificación no autorizada de información
- Suplantación de personalidad (falsificación)
- Ataques “vandálicos” a los servidores (*Denial Of Service*)
- Troyanos en el código móvil
- etc...

La solución pasa por el uso de *técnicas criptográficas*.

T1: Introducción

└ Arquitectura

## Esquema

- 1 Definición
- 2 Características de los Sistemas Distribuidos
- 3 Arquitectura de los Sistemas Distribuidos
  - Componentes
- 4 Conclusiones

## Arquitectura *software*

Para cumplir con todos o alguno de los objetivos anteriores, un sistema distribuido necesitará varios componentes (o servicios) que interactúan entre sí

- Servicios de comunicación
- Sistemas de ficheros
- Servicios de nombrado
- Servicios de sincronización y coordinación
- Servicios de seguridad
- Gestión de procesos
- Memoria compartida distribuída

## Servicios de comunicación

- Un proceso en una máquina necesitará comunicarse con otros procesos en otras máquinas.
- Hay diferentes tecnologías para lograrlo
  - Envío directo del mensaje por un *socket* (bajo nivel)
  - Llamadas a procedimientos remotos
  - Invocación remota de objetos
  - Servicios Web
- Cada proceso que se comunica puede jugar un rol
  - Cliente / Servidor
  - Redes entre iguales (P2P)

Dedicaremos gran parte del curso a este apartado

## Sistemas de archivos

- Un archivo (o fichero) es una abstracción conveniente para el programador o el usuario. Es un nombre asignado a una colección de datos.
- Incluso en sistemas no distribuidos, un archivo no existe “realmente”: es un conjunto de sectores dispersos por el disco
- En un sistema distribuido el archivo podría estar disperso por diferentes discos
- Un sistema de archivos distribuido sería una capa de *software* que permite al programador y al usuario “abrir ficheros” para leerlos o escribirlos, aunque los ficheros no estén en la máquina local
- La forma de usar los archivos remotos debería ser la misma que para los archivos locales (transparencia)

## Servicios de nombrado

- Para usar un recurso, el usuario debe conocer su nombre
- Es necesario inventar una forma de asignar nombres, que evite duplicidades
- A la vez, el nombre de un recurso no debería depender de la máquina que lo posee (transparencia).
- Es necesario un servicio que, a partir del nombre, encuentre el objeto (lo asocie con una máquina concreta): **servidor de nombres**
- El servicio servidor de nombres podría ser un cuello de botella

## Servicios de nombrado. Ejemplo

### Ejemplo: DNS

- El sistema de nombrado en Internet se denomina DNS
- El usuario identifica las máquinas por nombres como "www.uniovi.es"
- Pero el protocolo de internet los identifica por un número como 156.35.33.99 (dirección IP)
- El servicio DNS asocia nombres con IPs
- Es descentralizado para evitar cuellos de botella
- Es jerárquico para que sea más fácil de administrar

## Sincronización

Cuando tienes un reloj, siempre sabes la hora. Cuando tienes varios nunca estás seguro.

Los algoritmos a veces:

- Dependen de la hora en que se ejecutan.
- Requieren exclusión mutua.

En un sistema distribuido son problemas difíciles, al no existir reloj único, y al evitarse los recursos centralizados.



## Sincronización. Ejemplo

### Ejemplo: reserva de entradas *on-line*

Usando un sistema de compra *on-line*, dos usuarios intentan comprar la misma entrada “casi a la vez”.

Cuando se conectan, ambos ven la localidad libre. Ambos la eligen y poco después ambos pulsan el botón “comprar”.

- ¿Cómo evitar que la entrada sea comprada dos veces?
- ¿Cómo saber quién la compró antes?
- ¿Cómo gestionar este problema?

## Seguridad

Ya hemos visto los problemas de seguridad asociados a los sistemas distribuidos.

Un componente del sistema debería ocuparse de la seguridad:

- Cifrado y descifrado de la información
- Protocolo para ponerse de acuerdo en los métodos de cifrado y las claves
- Mecanismos de autenticación
- Gestión de credenciales
- etc.

## Gestión de procesos

- Un componente clave del sistema operativo es el *planificador*
- La planificación en un sistema distribuido involucra más de una CPU
  - Un planificador global podrá decidir qué ordenador ejecuta qué trabajos
  - Intentará aprovechar las máquinas inactivas
  - P.ej: todas las máquinas ejecutan el mismo código, pero cada una recibe un volumen de datos distinto
- Posibilidad de *migración* del código
  - Problema: lograr código móvil en arquitecturas heterogéneas
  - Hacer migrar el código una vez comenzó su ejecución puede ser complejo

## Memoria compartida distribuida

- Cuando hay un solo computador:
  - Todos los procesos están en la misma memoria física
  - Esto hace sencillo el compartir memoria. Y es una forma simple de comunicación entre procesos
  - Resulta muy cómodo tener variables compartidas a la hora de diseñar algoritmos paralelos
- En un sistema distribuido
  - Cada computador tiene su propia memoria
  - No existe memoria compartida
  - Pero sería tan cómodo tenerla...
- La memoria compartida distribuida es un *software* que simula la existencia de una memoria compartida, en un sistema distribuido.

## Esquema

- 1 Definición
- 2 Características de los Sistemas Distribuidos
- 3 Arquitectura de los Sistemas Distribuidos
- 4 Conclusiones

## Conclusiones

Un sistema distribuido es un conjunto de computadores **heterogéneos**, comunicados por una **red**, que **colaboran** con un objetivo común.

El diseño de *software* para sistemas distribuidos plantea muchas dificultades y retos

El diseño de un *sistema operativo distribuido* tiene toda la complejidad del diseño de sistemas operativos, más el de los sistemas distribuidos.