

Fundamentos de Computación distribuida

sockets **RPC**
RMI

Contenido

introducción

computación

arquitecturas

comunicación

conclusiones

introducción

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Programación SD

programación	T
1. Fundamentos de computación distribuida	6
2. Tecnologías Web y middleware	4
3. Servicios de nombres	1
4. Sistemas de establecimiento de tiempo	2
5. Seguridad	6
6. Coordinación distribuida	4
7. Sistemas de archivos distribuidos	4

introducción

resultados de aprendizaje

Contenido

introducción

computación
arquitecturas
comunicación
conclusiones

- ② Comprender los conceptos de heterogeneidad, extensibilidad, escalabilidad, seguridad, concurrencia, tolerancia a fallos y transparencia en el contexto de los sistemas distribuidos.
- ② Describir los principales paradigmas de computación distribuida, las características propias de cada modelo y sus aplicaciones.
- ② Justificar el uso de los diferentes mecanismos de comunicación en entornos distribuidos e internet en función de los requisitos y necesidades del problema.
- ② Diseñar la arquitectura de un sistema distribuido en función de los requisitos establecidos combinando paradigmas de computación distribuida y mecanismos de comunicación.
- ② Implementar protocolos de aplicación mediante la interfaz de sockets.
- ② Implementar aplicaciones distribuidas basadas en las tecnologías RMI y de Servicios Web utilizando frameworks y plataformas de terceros.

Contenido

introducción

computación
arquitecturas
comunicación
conclusiones

@ Fundamentos de computación distribuida

- Introducción a la computación distribuida
 - Evolución de los modelos de computación distribuida
 - Definiciones y propiedades
- Enfoques de sistemas distribuidos
 - SOR, SOD y Middleware
- Paradigmas de computación distribuida
 - C/S, P2P, MOM, SOA, agentes, colaborativos
- Mecanismos de comunicación distribuida
 - Recursos: IPC, sockets, RPC, RMI, ORB

Contenido

introducción

computación
arquitecturas
comunicación
conclusiones

- @ **Sistemas Distribuidos. Conceptos y Diseño**
G. Coulouris et al
Addison Wesley, 2001, 2012
Temas 4 y 5
- @ **Sistemas Distribuidos. Principios y paradigmas**
A.S. Tanenbaum
Prentice Hall , 2008
Temas 1 y 2
- @ **Computación Distribuida. Fundamentos y Aplicaciones**
M.L. Liu
Person Education , 2004
Temas 2,3,4,5,7 y 12
- @ **Service-Oriented Architecture: Concepts, technology and Design**
T. Erl
Prentice Hall, 2005
Temas 3,4,5 y 8

Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones

paradigmas de computación

Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones

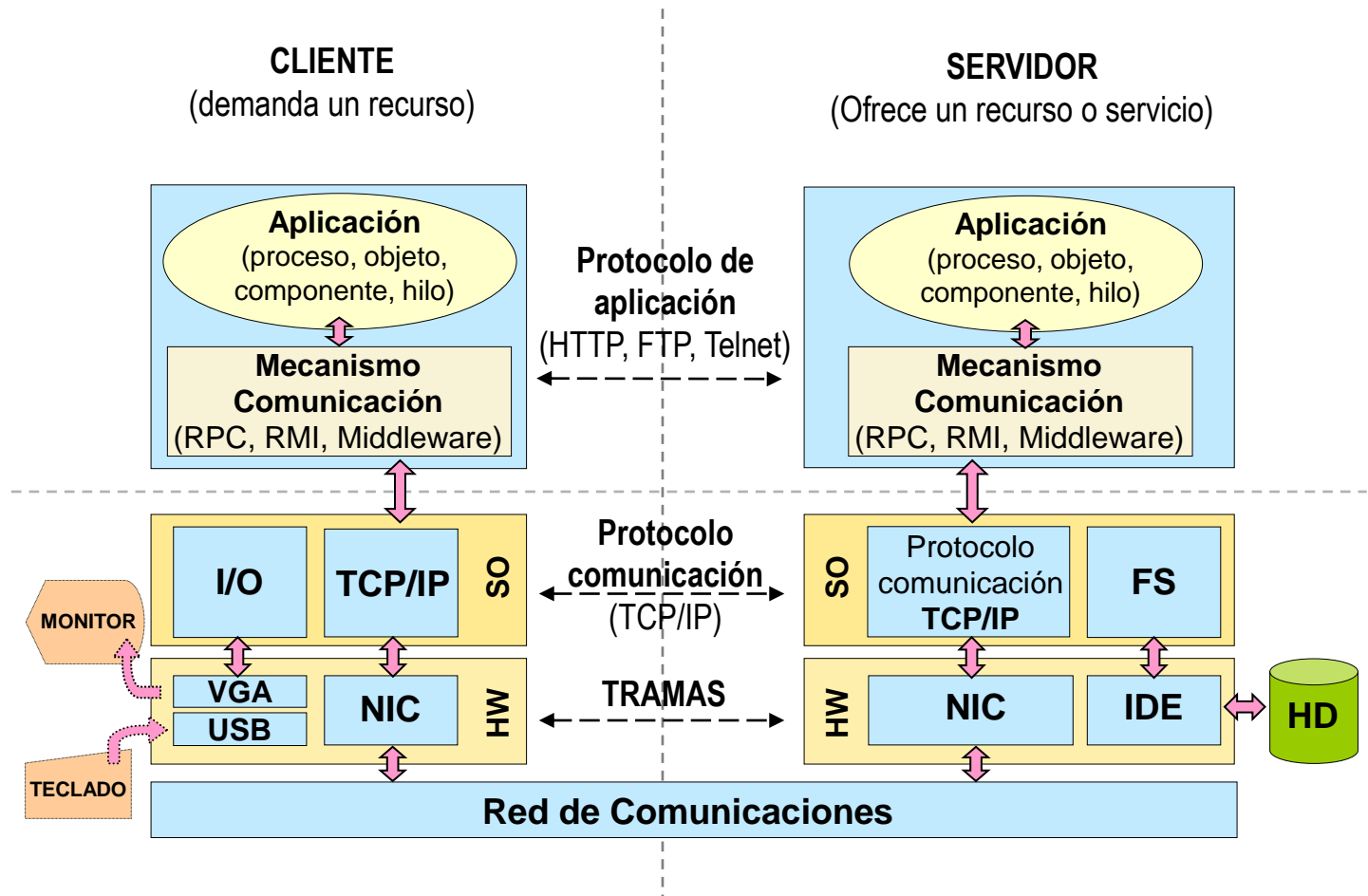
- Ⓜ **Sistema Distribuido** → Elementos de computación independientes, interconectados, que comunican y coordinan sus acciones a través de una red de comunicaciones
- Ⓜ Ejemplos de SD: Internet, intranets privadas, computación ubicua
- Ⓜ **Computación Distribuida** → La que se desarrolla en un SD: servicios y aplicaciones de red

introducción

elementos de un sistema distribuido

Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



Contenido

- introducción
- computación**
- arquitecturas
- comunicación
- conclusiones

@ Heterogeneidad

- Capacidad de los SD para estar compuestos por una **variedad** (de diferentes tipos) de componentes

- Estandarización
- Representación de datos
- Representación de código
- Representación de objetos
- Protocolos
- Integración
- Lenguajes intermedios
- UNIX, Windows

Hardware

- Representación datos

Red

- Ethernet, 802.11, ATM

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

- Capacidad de un SD de poder ser **extendido** pudiendo incorporar nuevos componentes:
 - Hardware
 - Redes
 - Computadores
 - Software
 - Aplicaciones
 - Servicios
 - Módulos

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

@ Escalabilidad

- Un SD es **escalable** si puede trabajar de forma correcta aunque se incrementen el número de:
 - Usuarios que lo utilizan
 - Recursos que se usan
 - Peticiones que se realizan a un servicio
 - Requerimientos de las aplicaciones
 - ...
- ¿Cómo se consigue?
 - Incorporación de forma dinámica de nuevos recursos HW/SW

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

@ Escalabilidad

@ Seguridad

- Entornos proclives a ataques externos
- Confidencialidad
- Integridad
- Disponibilidad
- Firewalls, SSL, HTTPS, Radius, Kerberos

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

@ Escalabilidad

@ Seguridad

@ Concurrencia y sincronización

- Posibilidad de que dos elementos del SD **accedan de forma simultánea** a un mismo recurso compartido
- Hay que garantizar el acceso concurrente para evitar inconsistencias
 - Acceso de forma controlada / exclusiva
 - Prioridad en los accesos a recursos
 - Secuenciación de las operaciones concurrentes

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

@ Escalabilidad

@ Seguridad

- Redundancia de componentes
- Sistemas de respaldo

@ Concurrencia y sincronización

@ Tolerancia a fallos

- Es necesario garantizar que el SD sea capaz de funcionar cuando uno de sus elemento falla – QoS (24x7)

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Heterogeneidad

@ Extensibilidad

- De acceso: a recursos remotos como si fueran locales
- De ubicación/localización: a recursos remotos sin conocer su ubicación
- De movilidad: recurso cambia de ubicación sin que el usuario sea consciente
- De escalabilidad: el sistema crece en recursos sin que el usuario sea consciente
- Frente a fallos: el usuario no es consciente de fallos en HW/SW

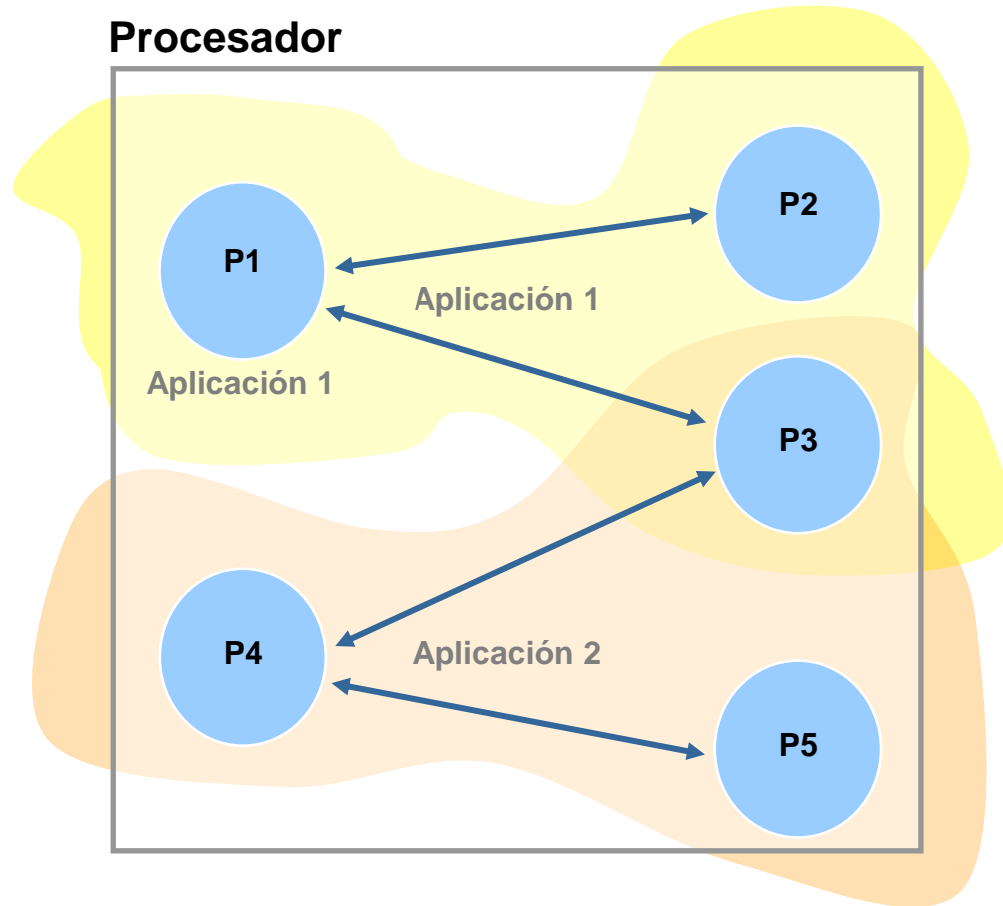
@ Concurrencia y sincronización

@ Tolerancia a fallos

@ Transparencia

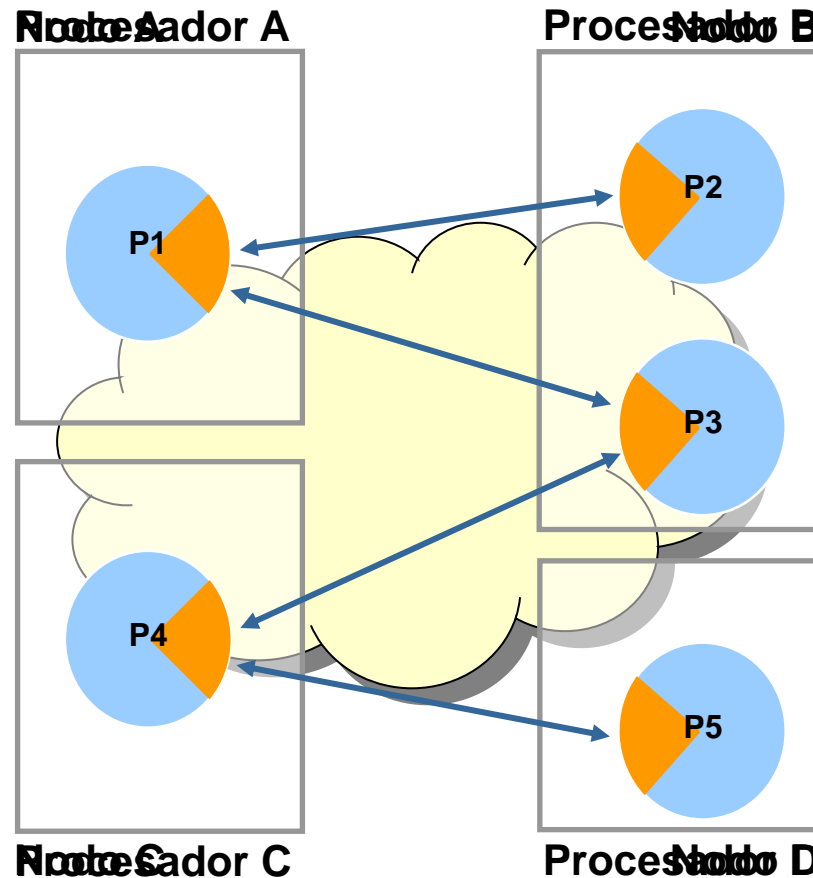
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



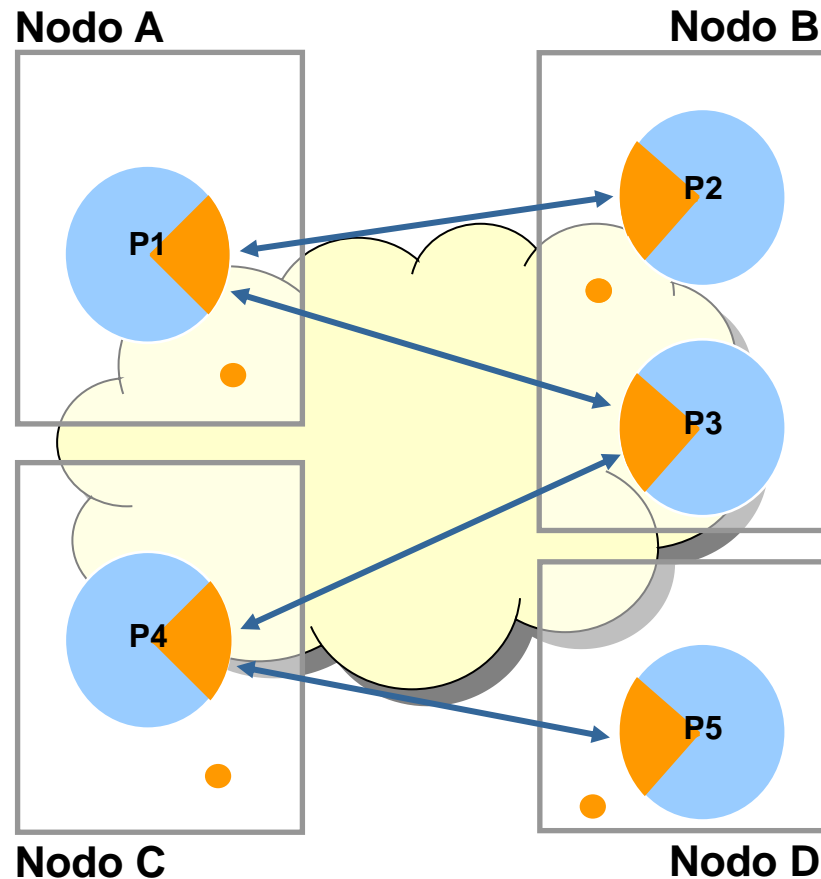
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



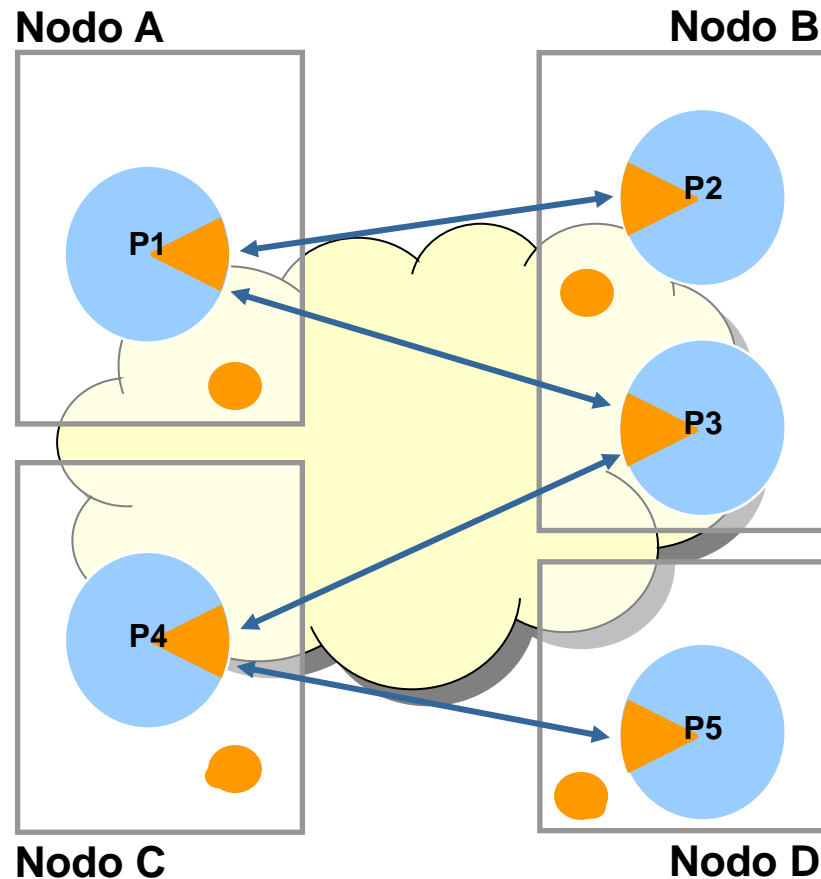
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



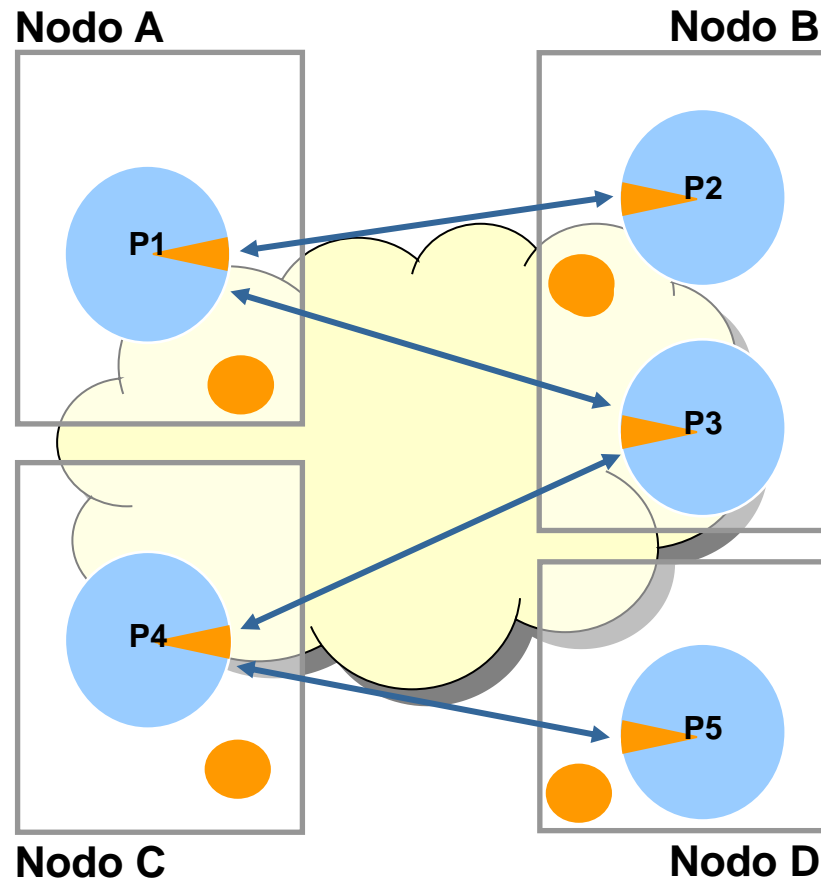
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



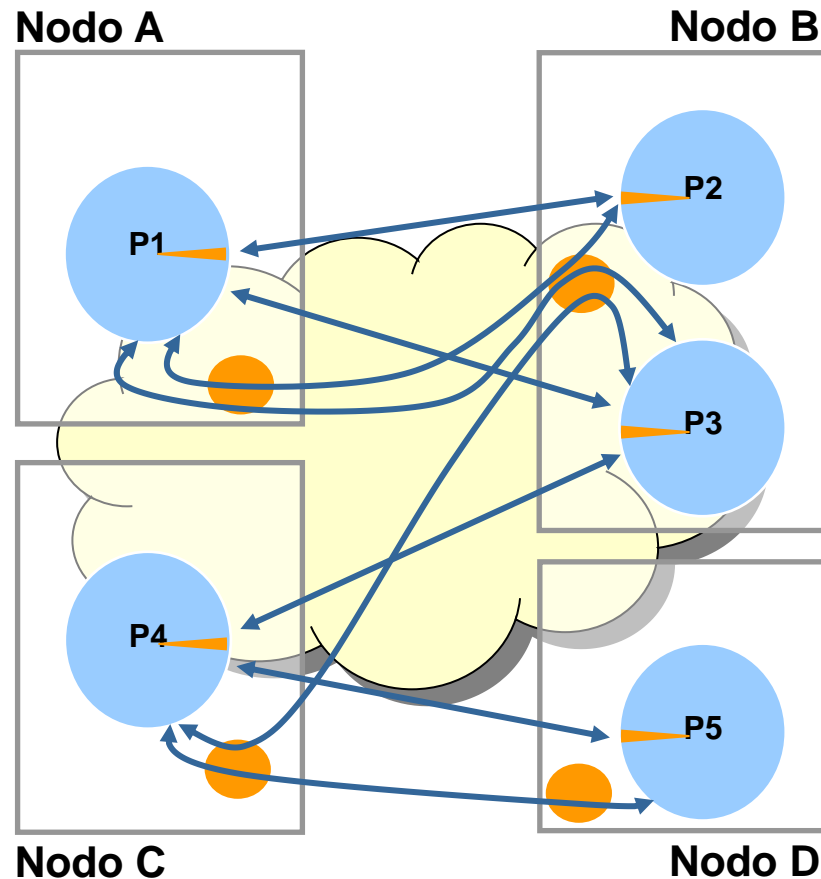
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



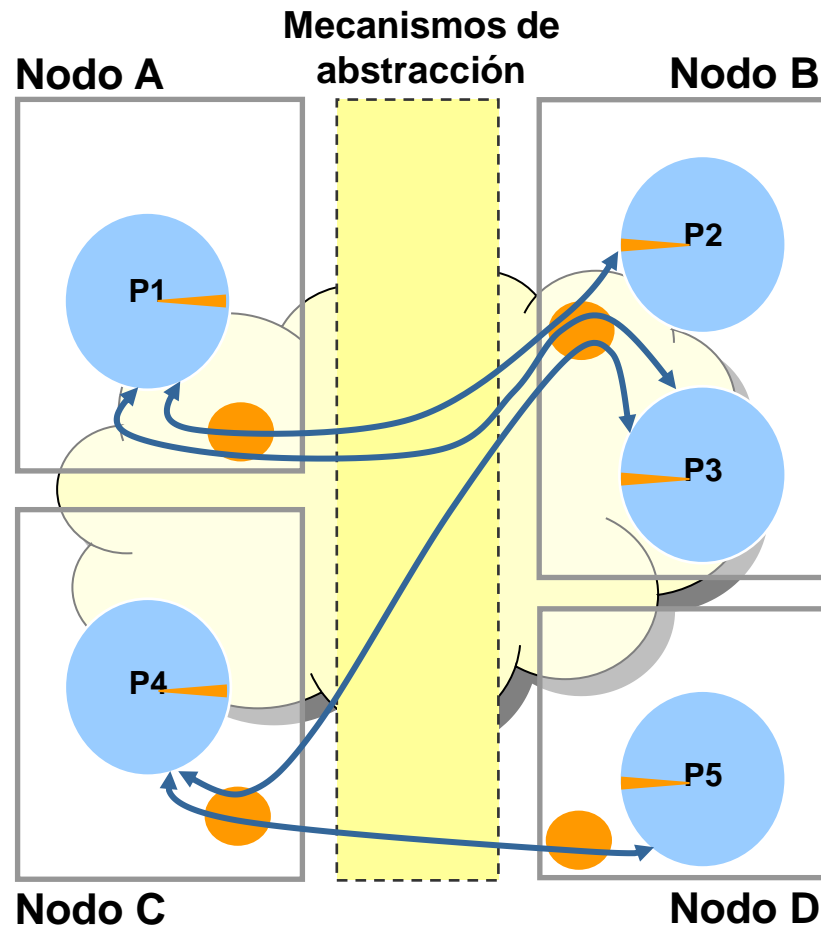
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



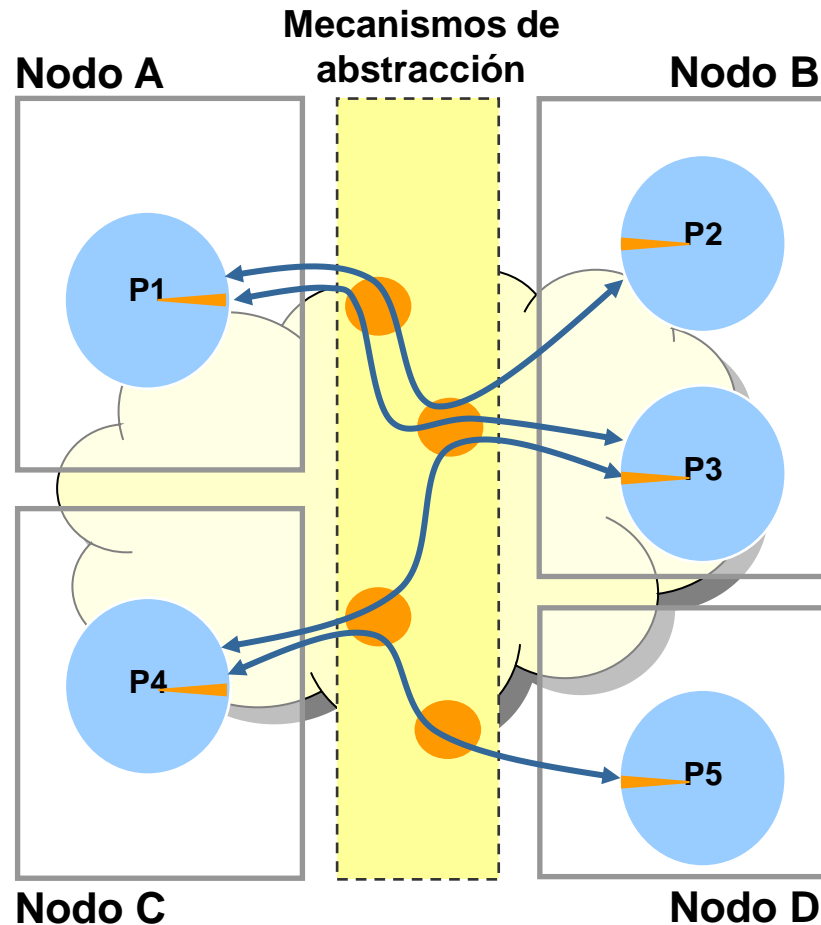
Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones



paradigmas de computación

evolución

Contenido

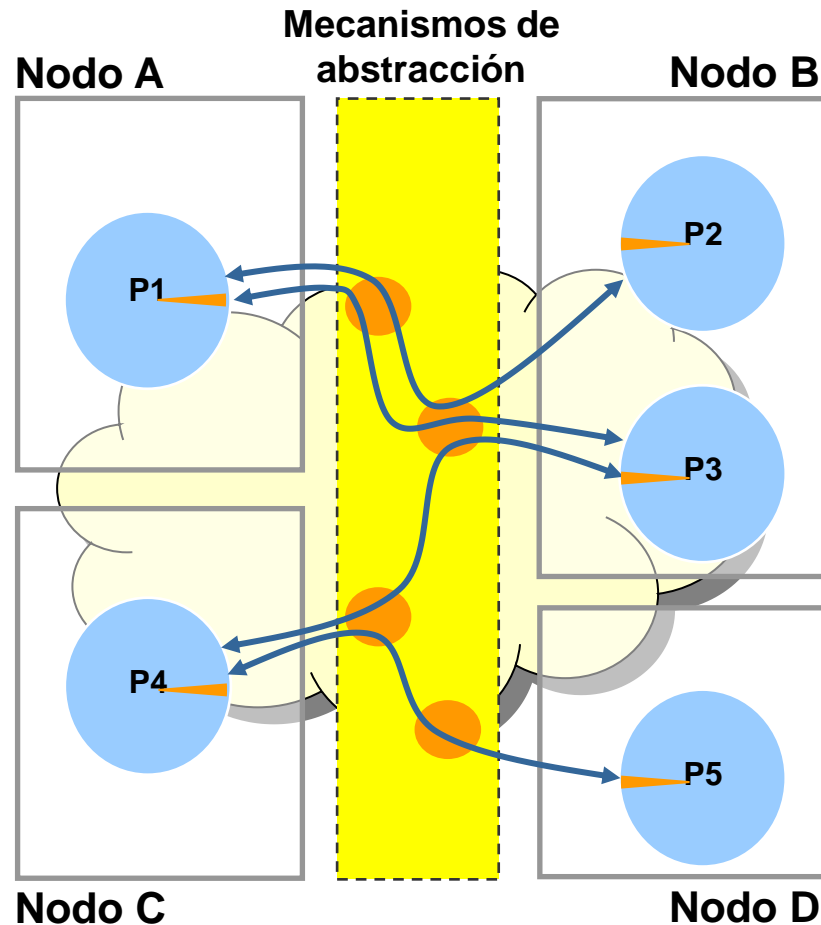
introducción

computación

arquitecturas

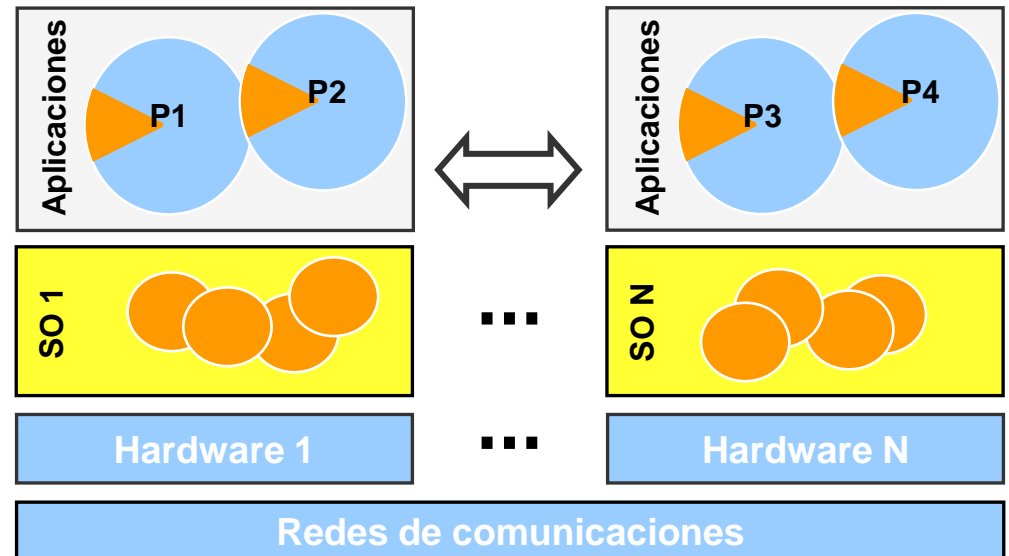
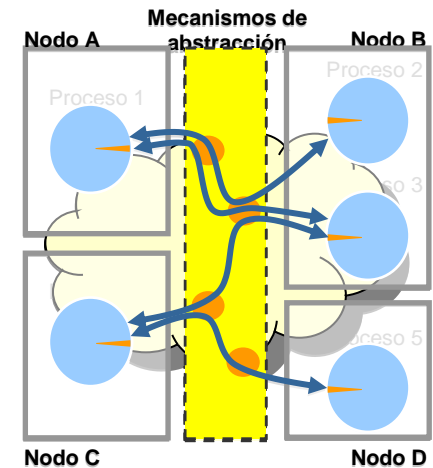
comunicación

conclusiones



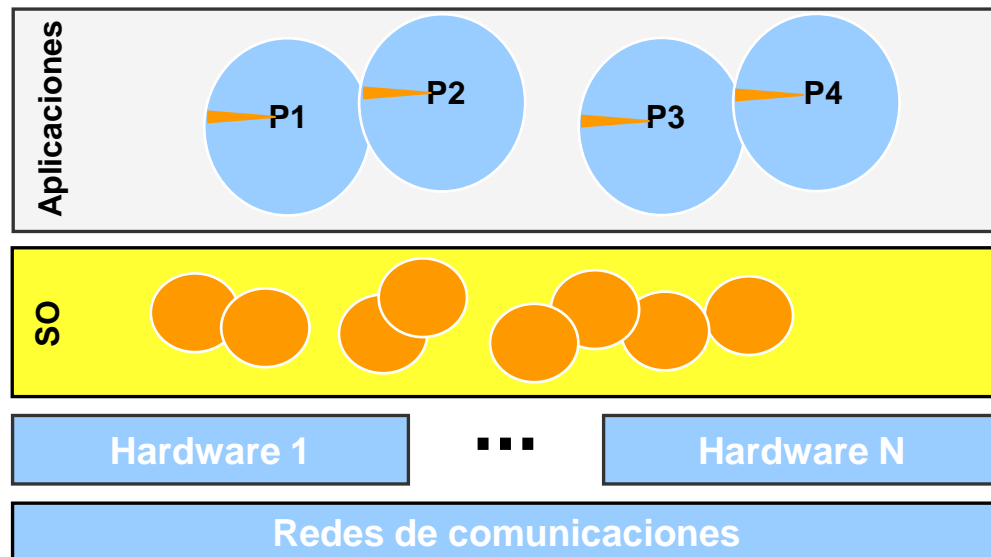
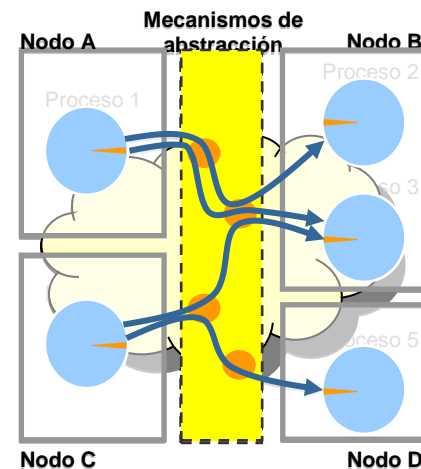
sistemas operativos en red

- Ubicación en el SO
- Heterogéneo → Específico del SO
- Ejemplos:
 - Linux, Windows, Novell NetWare
- Ventajas
 - Flexibilidad
 - SO → técnicas maduras
- Desventajas
 - Falta de transparencia
 - Mayor esfuerzo de integración



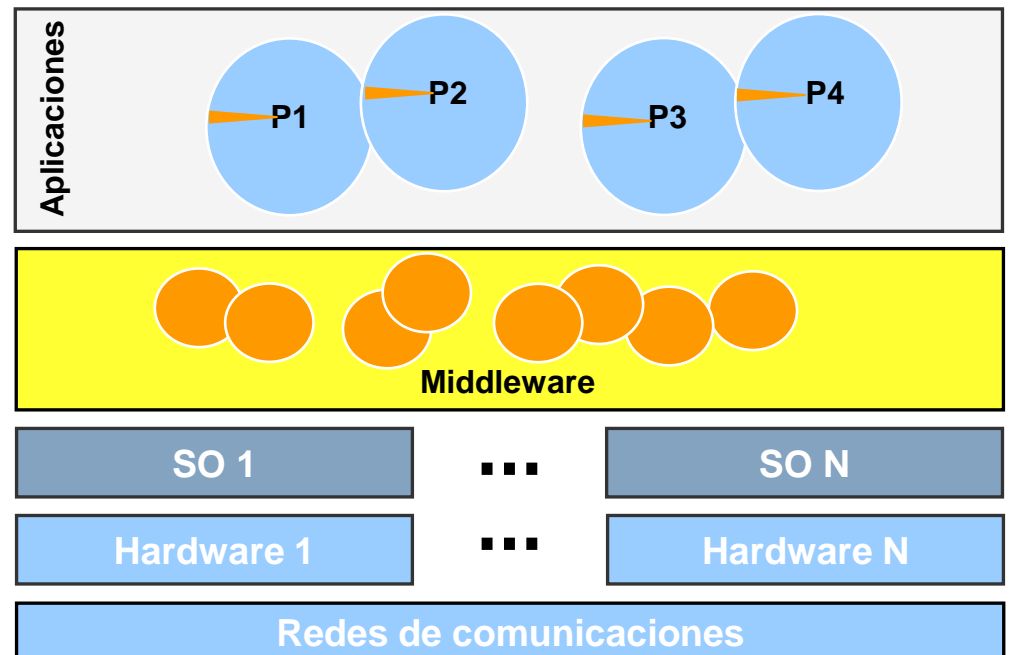
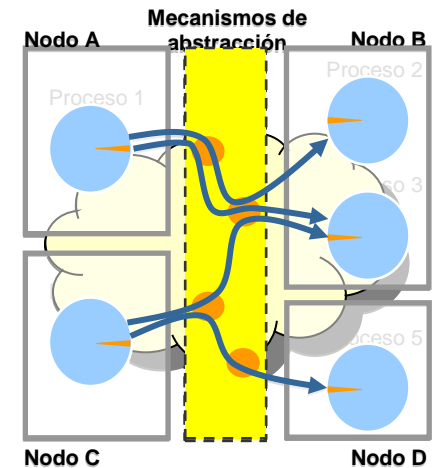
sistemas operativos distribuidos

- Ubicación en el SO
- Homogéneo → SO global
- Ejemplos:
 - Mach, Amoeba
- Ventajas
 - Transparencia
 - Escalabilidad
 - Facilidad de integración
- Desventajas
 - Técnicas complejas
 - Comunicaciones de alta velocidad
 - Competencia de mercado



middleware

- Enfoque mixto
 - Modelo conceptual → SOD
 - Infraestructuras → SOR
- Capa por encima del SO
- Homogéneo
- Ejemplos:
 - CORBA,
 - J2EE
 - .Net Framework
- Ventajas
 - Flexibilidad
 - Transparencia
 - Integración
 - Madurez
 - Escalabilidad
- Desventajas
 - Plataformas heterogéneas
 - Necesidad de estandarización



Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones

modelos arquitectónicos

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

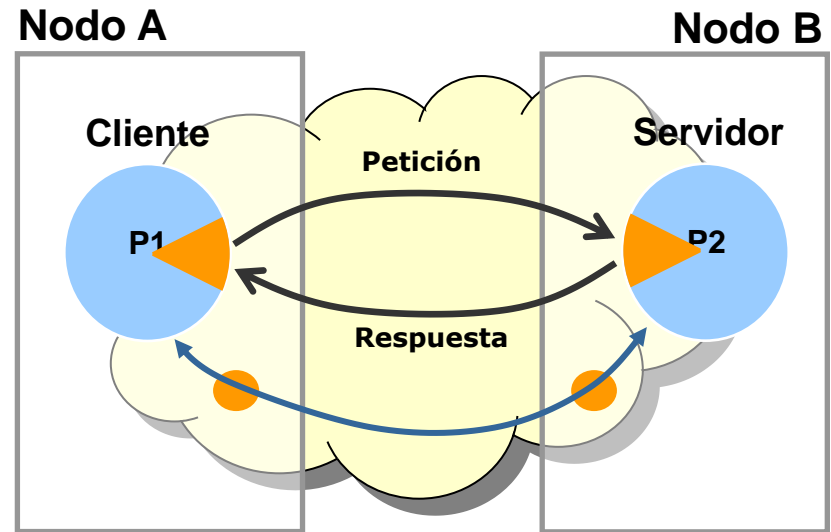
- @ Cliente/Servidor
- @ Peer-to-Peer
- @ Middleware orientado a mensajes
- @ Arquitectura orientada a servicios
- @ Cluster y grid

Contenido

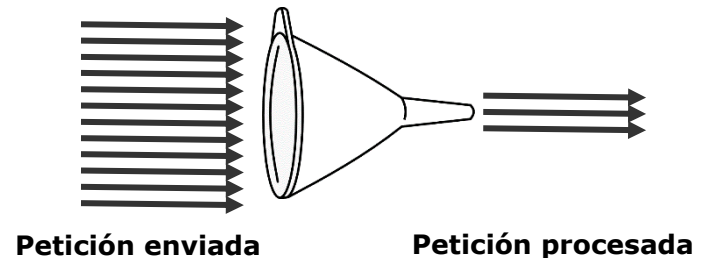
introducción
computación
arquitecturas
comunicación
conclusiones

- @ Cliente/Servidor
- @ Peer-to-Peer
- @ Middleware orientado a mensajes
- @ Arquitectura orientada a servicios
- @ Cluster y grid





- Sobre el paradigma de pasos de mensajes
- Abstracción del acceso de recursos remoto → servicios de red
- Gestión centralizada
 - Mayor control
- Procesos → roles
 - Servidor (Proveedor de servicio en espera pasiva)
 - Cliente (Solicita servicio)
- Aspectos
 - Mecanismos de concurrencia
 - Mantenimiento de sesión
 - Mecanismos de escalabilidad
- Ejemplo: HTTP, DNS, FTP, SMTP, ...

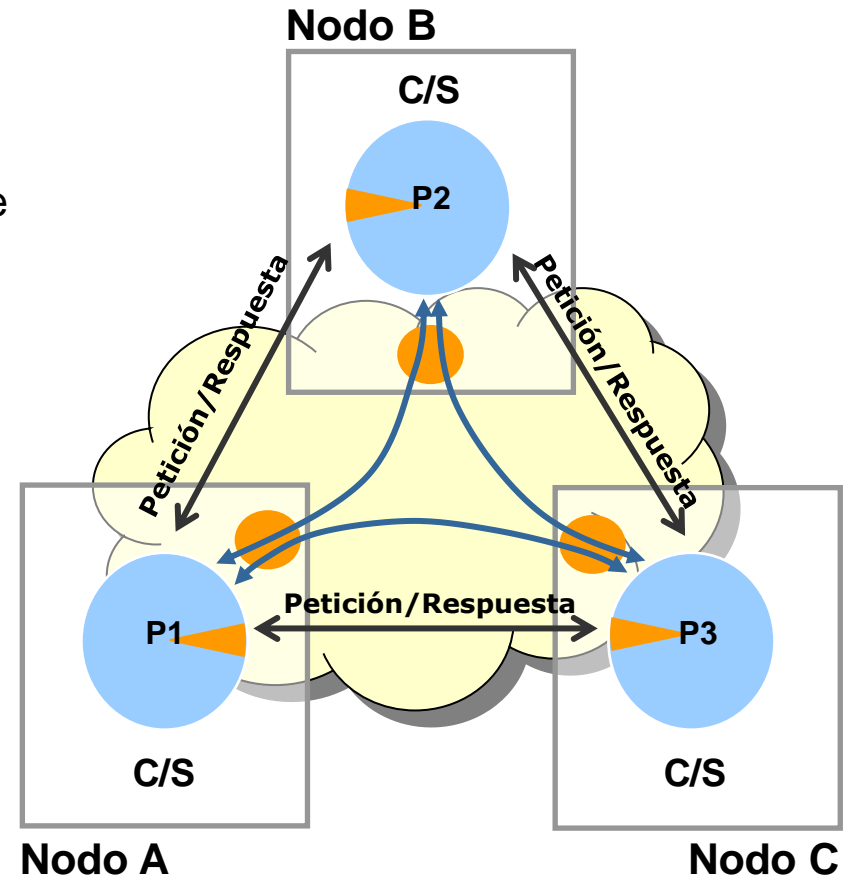


Cuello de botella



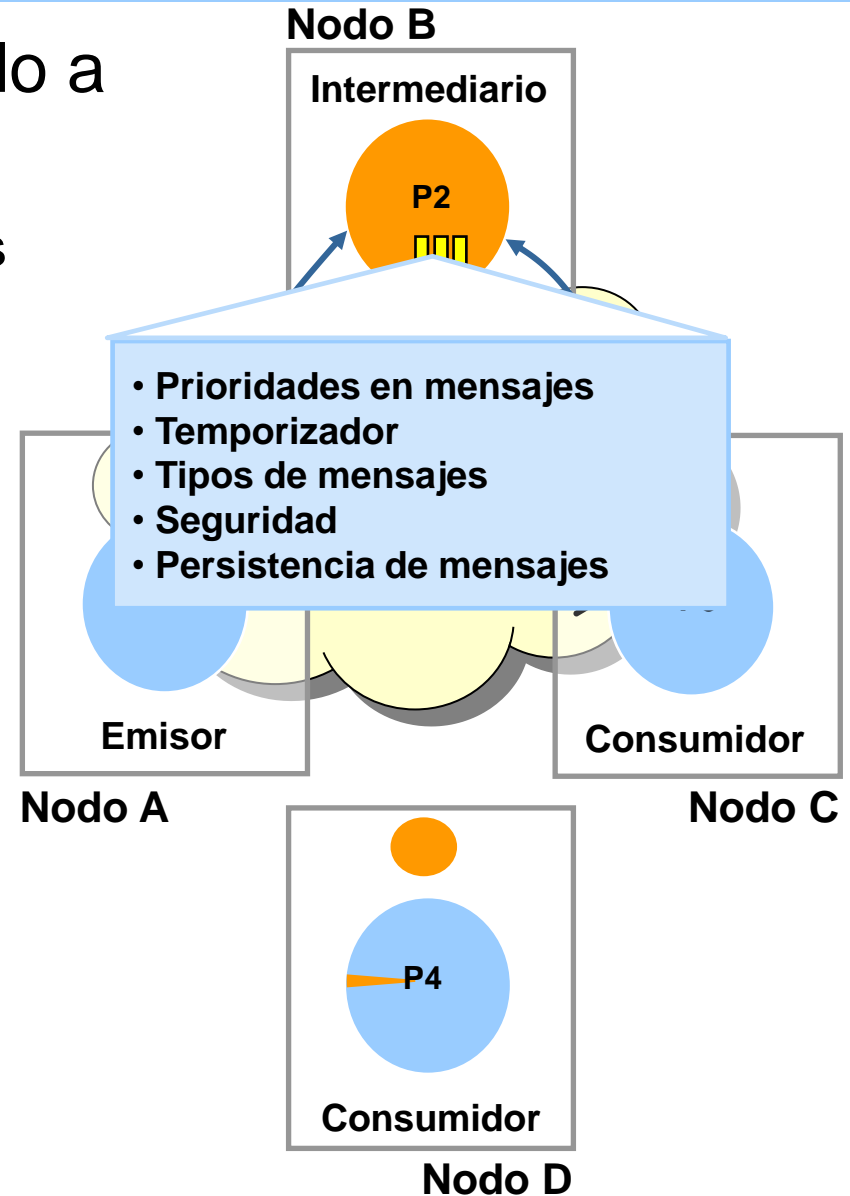
peer-to-peer

- Procesos → envían peticiones y prestan servicios
- Procesos de igual a igual
 - Cliente (envío de peticiones, recepción de respuesta)
 - Servidor (recepción de solicitudes, procesamiento de solicitudes, envío de respuesta, propagación de solicitudes)
- Gestión distribuida del recurso
 - Menor control
- Apropiado para aplicaciones tipo: mensajería instantánea, compartición de archivos, video conferencia y trabajo colaborativo
- Herramientas: JXTA 
- Ejemplos:
 - Napster (centralizada) 
 - Gnutella (descentralizada) 
 - BitTorrent (Híbrida) 

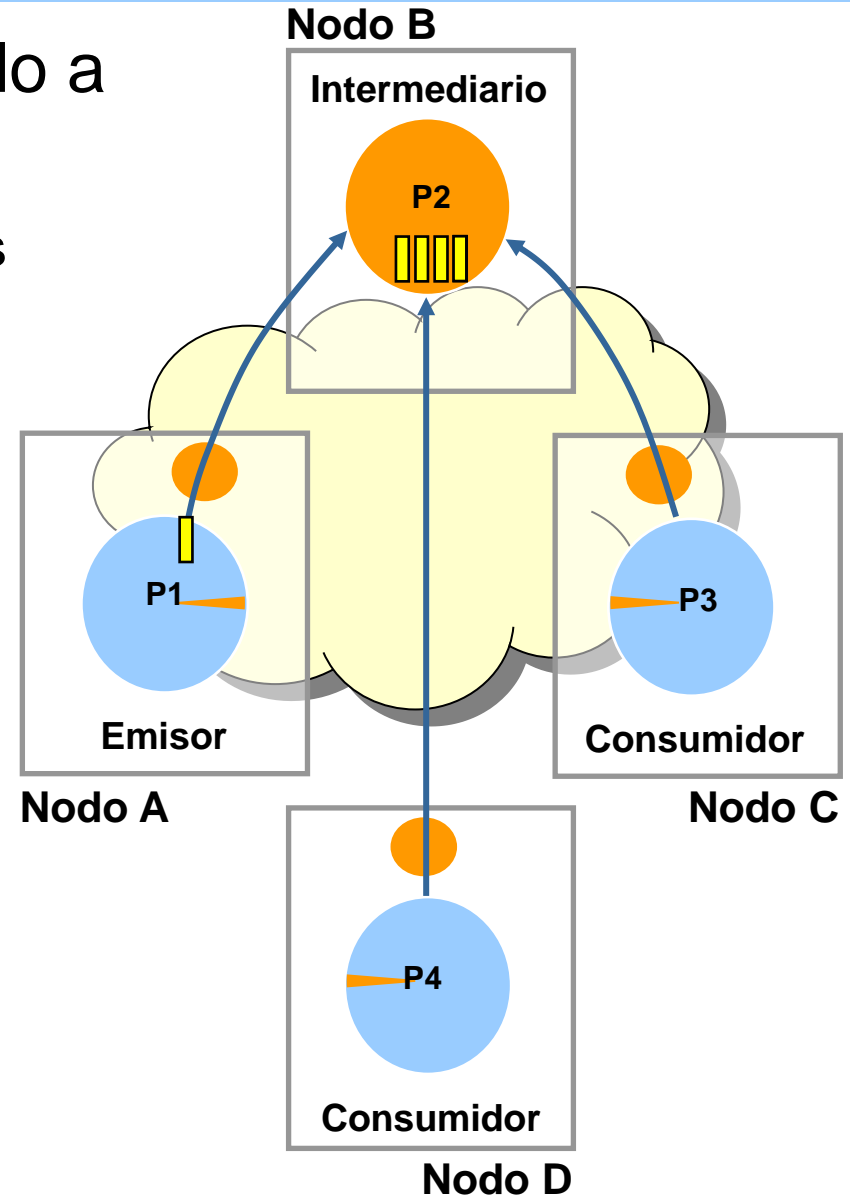


middleware orientado a mensajes

- MOM (Middleware Orientado a Mensajes)
 - Evolución del paso de mensajes
- Desacoplamiento
- Sistemas asíncronos
- Intermediario en el proceso de comunicación
- Herramientas:
 - JMS, MSMQ, MQSeries
- Dos tipos:
 - Punto a punto \rightarrow 1:1
 - Publicación/suscripción \rightarrow 1:M

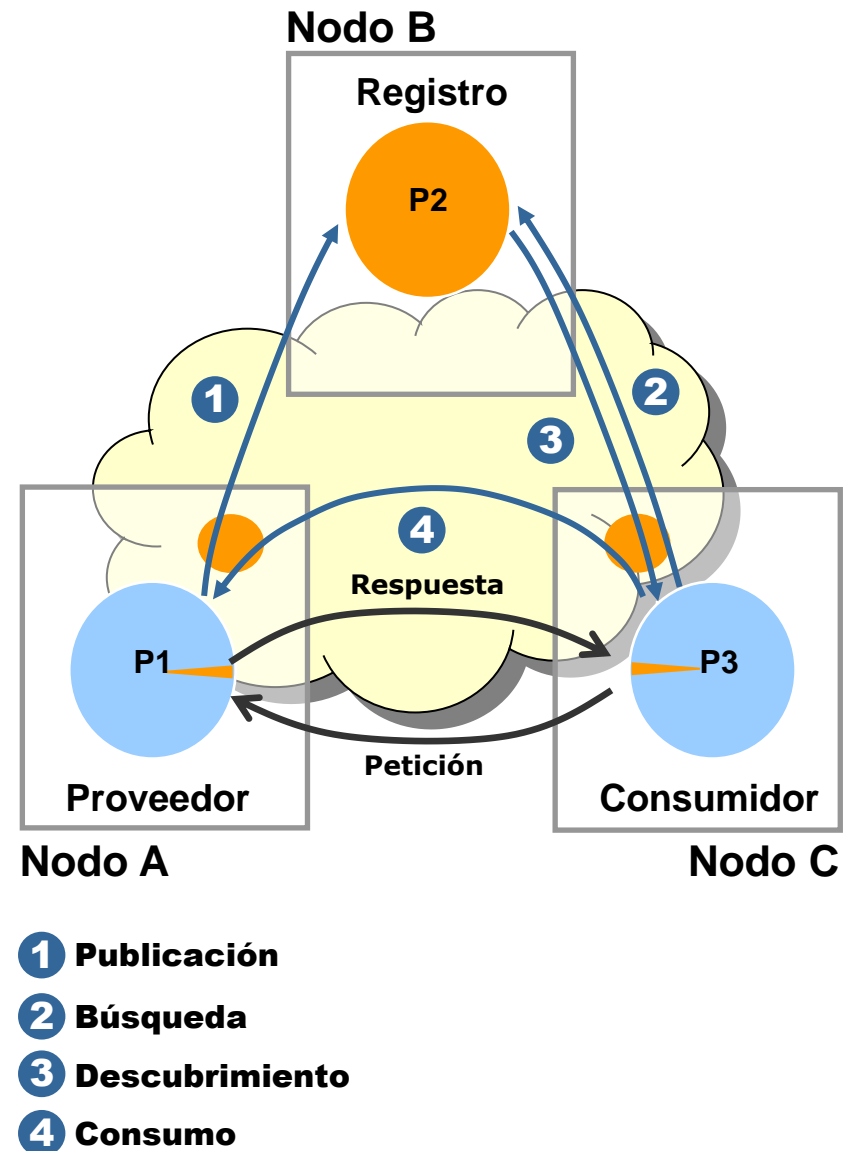


- MOM (Middleware Orientado a Mensajes)
 - Evolución del paso de mensajes
- Desacoplamiento
- Sistemas asíncronos
- Intermediario en el proceso de comunicación
- Herramientas:
 - JMS, MSMQ, MQSeries
- Dos tipos:
 - Punto a punto \rightarrow 1:1
 - Publicación/suscripción \rightarrow 1:M



arquitectura orientada a servicios

- Abstracción de acceso a actividades de negocio denominadas Servicios
→ Infraestructura de servicios
- Principios
 - Localización, descubrimiento y publicación
 - Interoperabilidad
 - Composición
 - Autonomía y autocontenidos
 - Reusabilidad
 - Desacoplamiento
 - Contrato bien definido
 - Sin estado
- Componentes
 - Proveedor de servicios
 - Consumidor de servicios
 - Servicio de registro
- Herramientas:
 - Servicios Web, JINI, UPNP



Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- ⌚ Infraestructuras hardware y software para ofrecer mayor capacidad de procesamiento y almacenamiento
- ⌚ Conjunto de computadores → un super computador
 - 1.000 computadores 1GHz → 1 computador 1THz
 - 1.000 computadores 1GB RAM → 1 computador 1TB RAM
 - 1.000 computadores 40GB HD → 1 computador 40TB HD

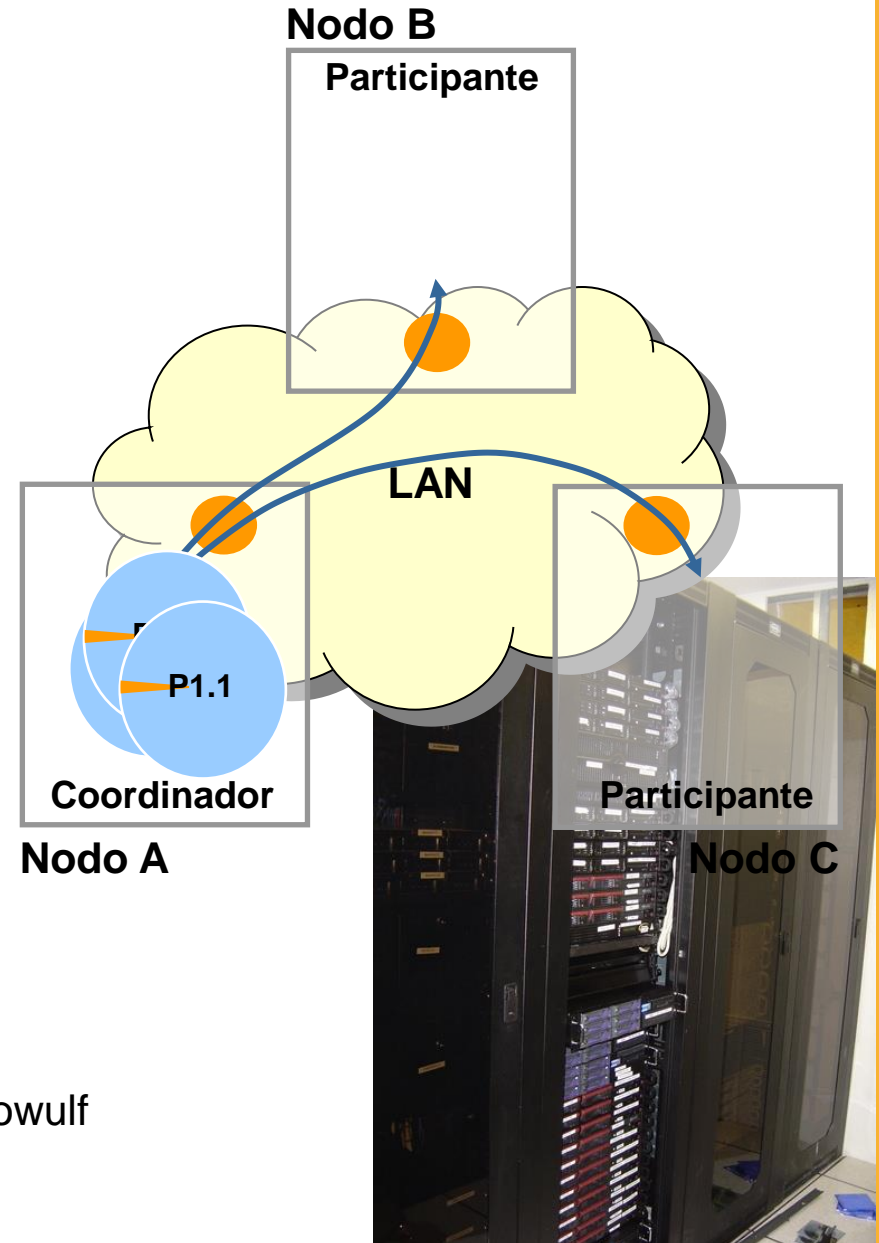
modelos arquitectónicos

Contenido

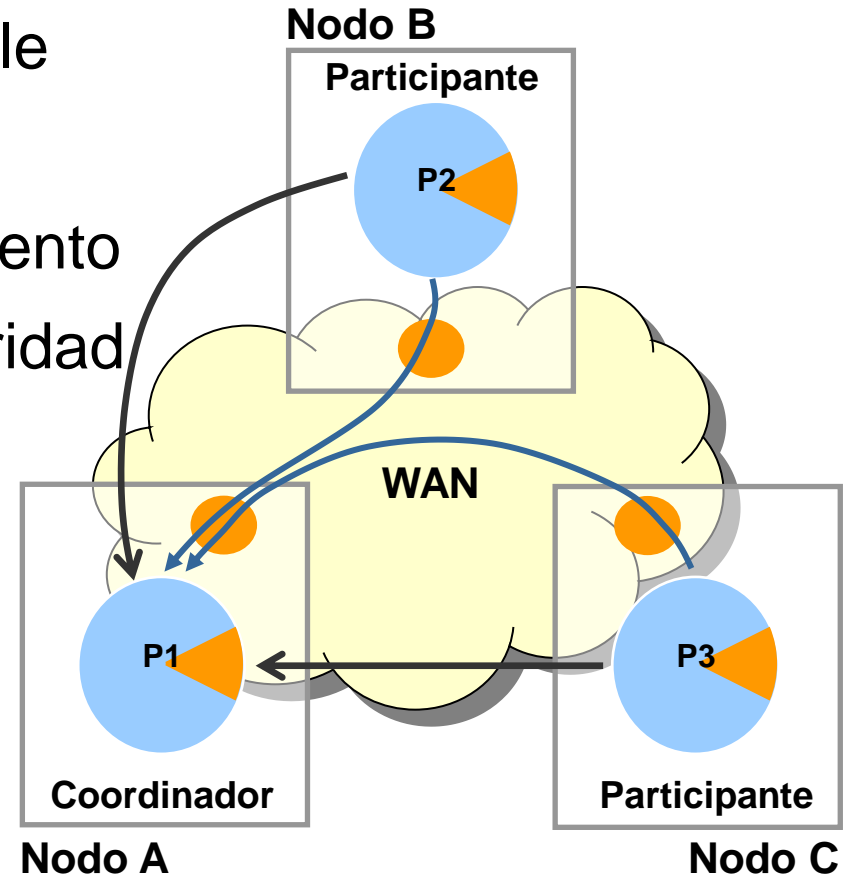
introducción
computación
arquitecturas
comunicación
conclusiones

cluster

- Homogeneidad
- Red local de alta velocidad
- Entorno dedicado → Pérdida de independencia
- Gestor de recursos centralizado
- Tipos
 - Alta disponibilidad
 - Balanceo de carga
 - Escalabilidad
 - Alto rendimiento
- Aplicaciones
 - Servidores Web y de aplicaciones
 - Sistemas de información
 - Supercómputo
 - MOSIX, OpenMosix, Heartbeat, Beowulf



- Heterogeneidad → más flexible
- Internet → desacoplamiento
- Procesamiento y almacenamiento
- Respeto de políticas de seguridad y aplicaciones internas
- No pierde la independencia
 - Tiempos muertos
- OGSA (Open Grid Service Architecture)
 - Grid sobre tecnología Web
- Herramientas:
 - Globus Toolkit 4.0 (código abierto)
- Aplicaciones
 - Proyecto BIRN, ESGII, ...



Contenido

- introducción
- computación
- arquitecturas
- comunicación
- conclusiones

mecanismos de comunicación

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

@ Fundamentos de comunicación

- Mecanismos de comunicación entre procesos (IPC)
- Transmisión de información
- Protocolos

@ Mecanismos de comunicación

- Paradigma de mensajes
- Llamada a procedimientos remotos
- Invocación de métodos remotos
- Intermediario de petición de objetos
- Servicios Web

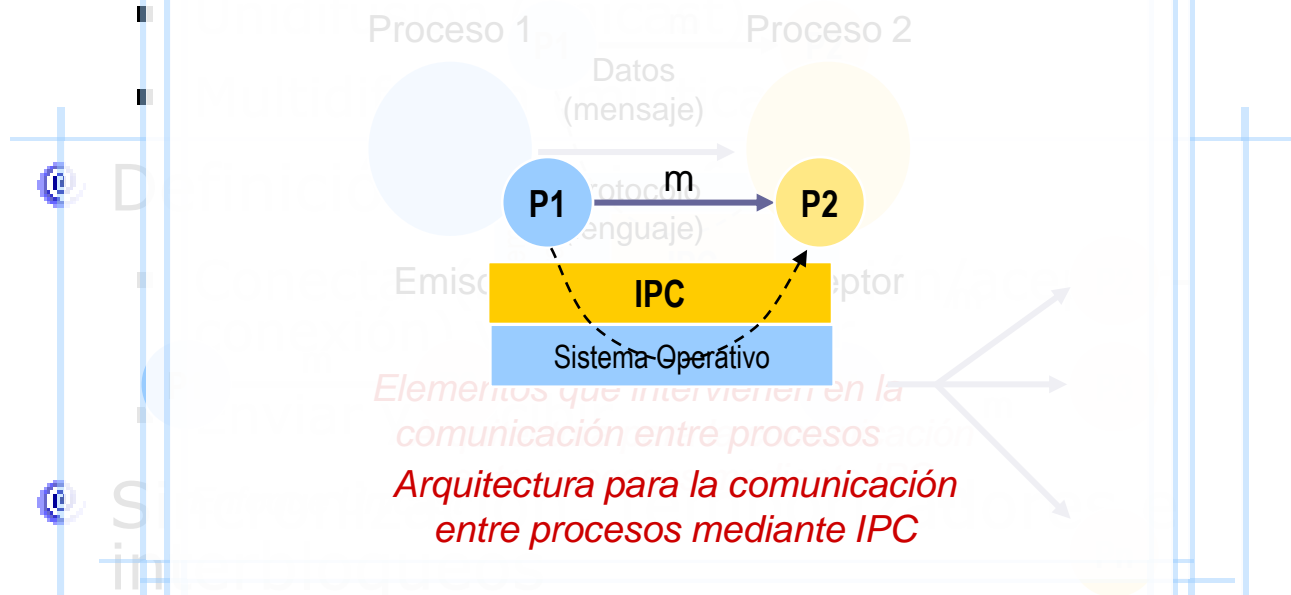
Contenido

- introducción
- computación
- arquitecturas
- comunicación**
- conclusiones

@ Mecanismo básico de los sistemas distribuidos

→ comunicación entre procesos distribuidos

@ Modelos básicos:



- Operaciones bloqueantes o síncronas
- Operaciones no bloqueantes o asíncronas

Enfoque Multicast

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

<agenda>

<contacto>

<nombre>Virgilio**</nombre>**

<apellidos>Gilart Iglesias**</apellidos>**

<localidad>Alicante**</localidad>**

<teléfono>555 77 9999**</teléfono>**

<email>vgilart@dtic.ua.es**</email>**

</contacto>

<contacto>

<nombre>Diego**</nombre>**

<apellidos>Marcos Jorque**</apellidos>**

<localidad>Elche**</localidad>**

<teléfono>555 66 8888**</teléfono>**

<email>dmarcos@dtic.ua.es**</email>**

</contacto>

</agenda>

*Archivo XML de ejemplo con la definición de una
agenda personal de contactos*

mecanismos de comunicación

protocolos

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- @ Basados en texto o binario
 - HTTP, SMTP, POP3
 - LDAP, DNS
- @ Tipo de patrón de mensaje
 - Petición-Respuestas
 - FTP, HTTP, SMTP
 - Solicitud-Respuestas
 - Unidireccional
 - Notificación
- @ Técnicas de comunicación
 - Poll
 - Push
- @ Orientados o no a la conexión
 - HTTP, FTP
 - DNS, DHCP
- @ Con o sin estado
 - FTP
 - HTTP

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- @ Paso de mensaje (Sockets)
- @ Llamadas a procedimientos remotos (RPC)
- @ Invocación de métodos remotos (RMI)
- @ Intermediario de petición de objetos (ORB)
- @ Servicios Web (WS)

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- ② Intercambio de información entre procesos → mensaje
- ② Interfaz mínima requerida
 - Enviar/Recibir
- ② Interfaz dependientes de conexión
 - Conectar/Desconectar

mecanismos de comunicación

sockets

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- Ⓜ Mecanismo IPC de intercambio de datos a través de un conector

- Punto final de un enlace de comunicación

- Ⓜ Berkeley → BSD de UNIX

- Ⓜ API de sockets → Biblioteca de programación

- Analogía con gestión de archivos UNIX
- UNIX, Linux, Windows, OS/2

- Ⓜ Comunicación local y remota

- Fácil de utilizar y más utilizada

- Ⓜ Tipos de comunicación

- Con conexión
 - Flujo de datos *Comunicación entre dos procesos remotos a través de sockets*
 - Confiable y ordenado

- Sin conexión

- Datagrama
- No confiable y sin orden




```
program SUMA_PROG {
  version SUMA_VERS {
```

```
    int SUMA(numeros) = 1;
```

Número de procedimiento

```
  } = 1;
```

Número de versión

```
  } = 0x20000001;
```

Número de programa

Declaración de procedimientos del proceso remoto

func(1,2)

func(arg1,arg2)

...

- 0x00000000 - 0x1FFFFFFF: Definidos por Sun

- 0x20000000 - 0x3FFFFFFF: **Definidos por el usuario**

- 0x40000000 - 0x5FFFFFFF: Temporales

- 0x60000000 - 0xFFFFFFFF: Reservados

Obtención del número de programa

*Llamada a una función
procedimiento local a un programa*

ubicado en un proceso diferente

▪ Evolución → XML-RPC

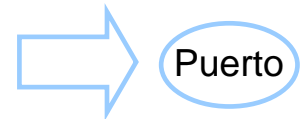
Comunicación
con procesos remotos

S
ción

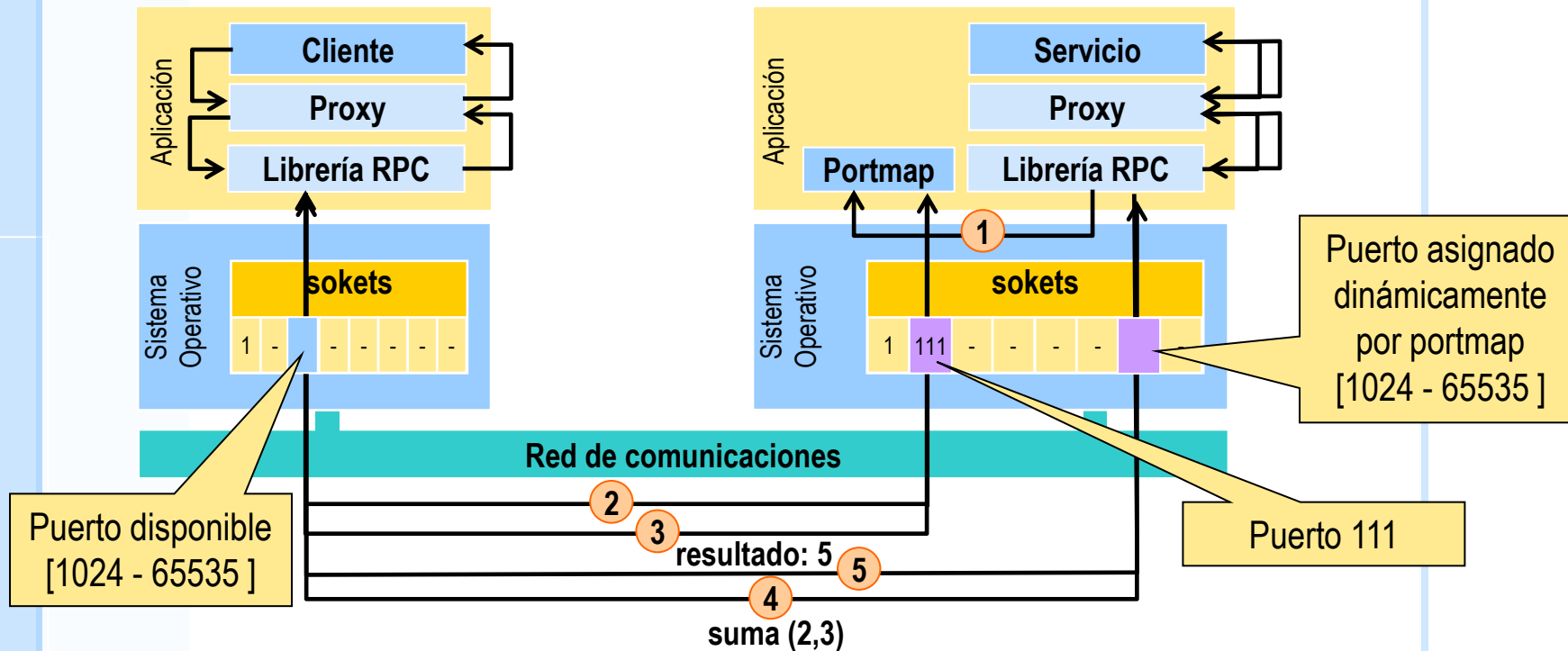
Proceso 2

Contenido de Funcionamiento de RPC

- Número de programa
- Número de versión
- Número de procedimiento
- Protocolo de transporte



- 1 Registro del servicio
- 2 Cliente realiza petición al servicio remoto
- 3 Portmap envía el puerto de conexión del servicio
- 4 El cliente instancia un método remoto a través de la librería RPC
- 5 El proceso servidor devuelve el resultado



modelos de comunicación

invocación de métodos remotos

Contenido

- introducción
- computación
- arquitecturas
- comunicación**
- conclusiones

@ RPC para lenguajes orientados a objetos

@ Detalles de comunicación

- Stub y skeletons

- Representación de datos

@ Propuesta de SUN

- RMI → JRMP

@ Propuesta de MS

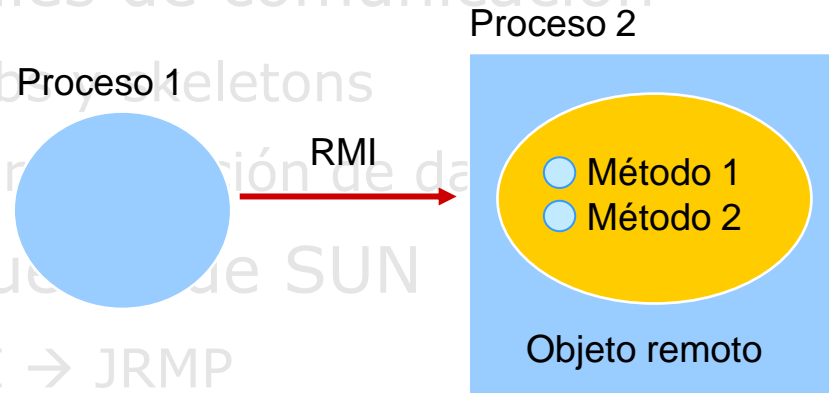
Invocación desde el proceso 1 a un método perteneciente a un objeto remoto ubicado en el proceso 2

- DCOM

- .NET Remoting (VS.NET)

 - Canales → TCP, HTTP

@ Objetos de la misma plataforma

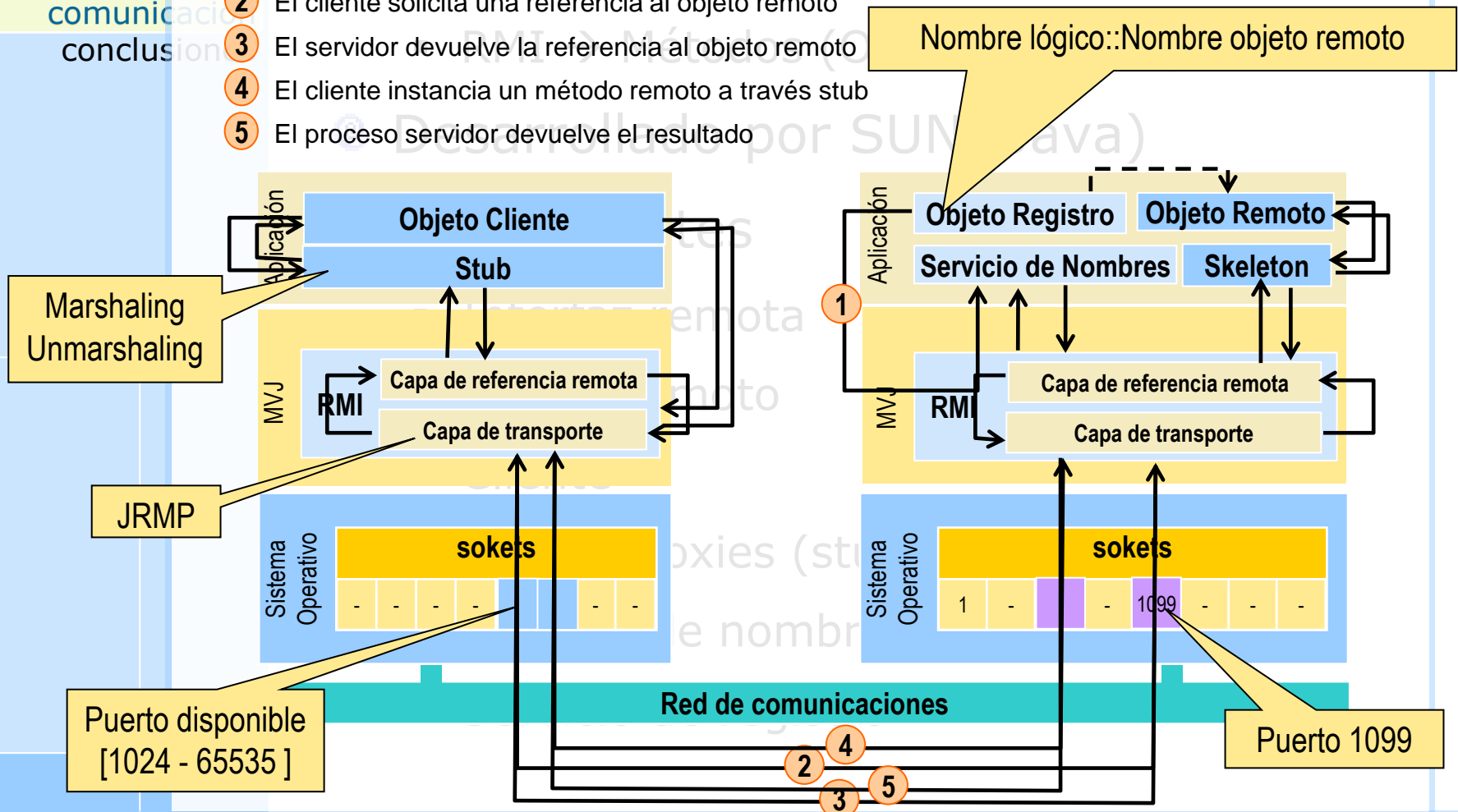


Contenido

Arquitectura RMI

introducción
computación
arquitectura
comunicación
conclusiones

- 1 Registro del objeto remoto
- 2 El cliente solicita una referencia al objeto remoto
- 3 El servidor devuelve la referencia al objeto remoto
- 4 El cliente instancia un método remoto a través stub
- 5 El proceso servidor devuelve el resultado

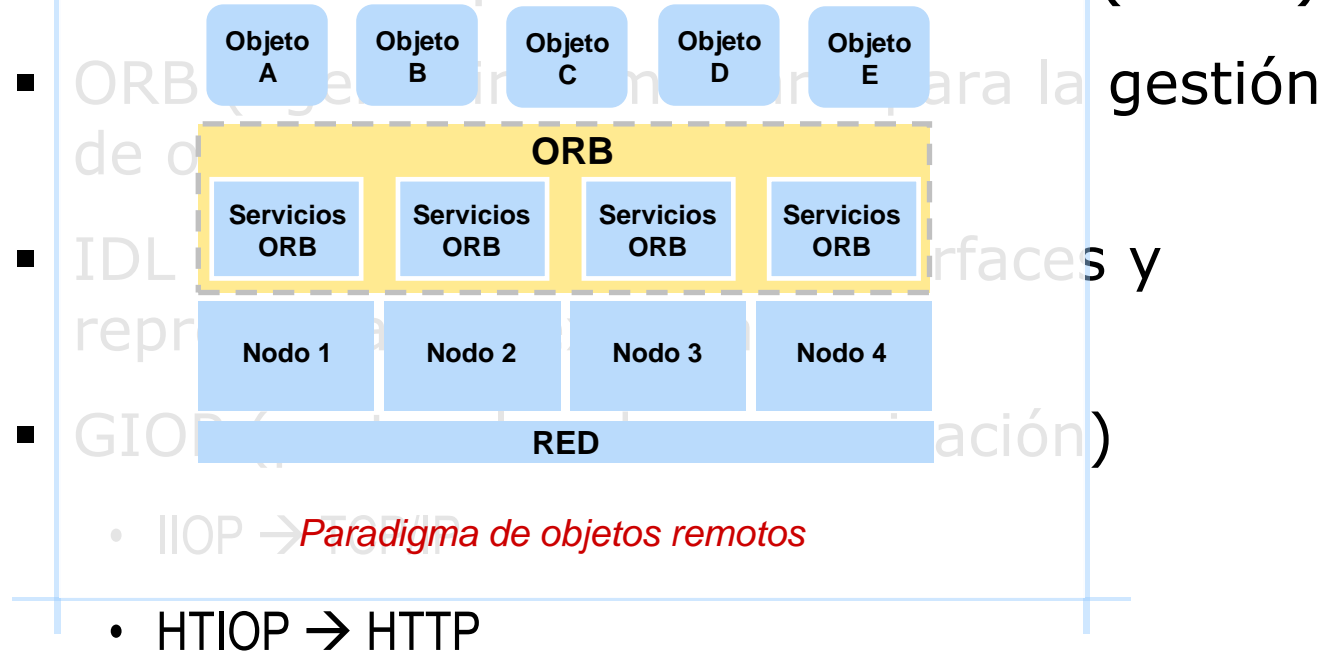


Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

Ⓢ Abstracción del acceso a objetos heterogéneos

Ⓢ Base de la arquitectura CORBA (OMG)



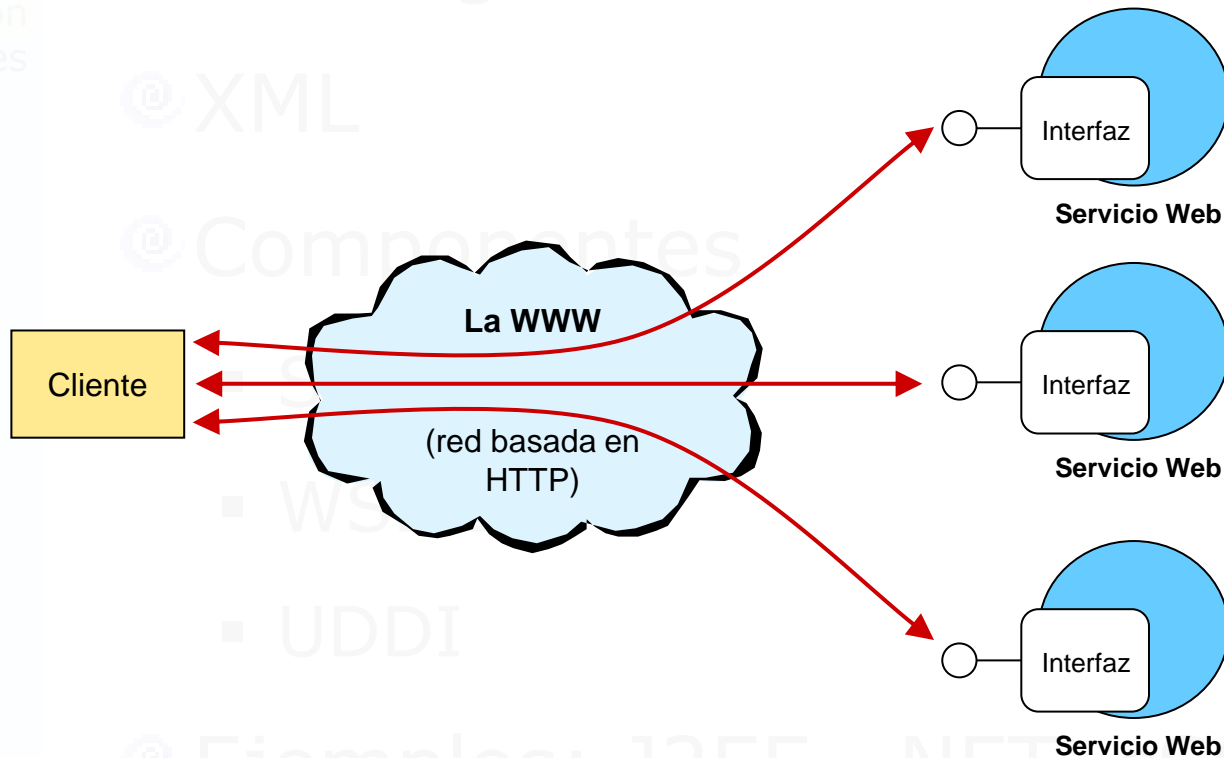
Ⓢ Ejemplos: J2EE, .NET



Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

Abstracción del acceso a objetos



Modelo conceptual de los Servicios Web

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- @ Abstracción frente a sobrecarga
- @ Escalabilidad
- @ Independencia de la plataforma
- @ Criterios adicionales:
 - Madurez, estabilidad de la tecnología y disponibilidad de herramientas de desarrollo
 - Tolerancia a fallos ofrecida por la herramienta
 - Mantenibilidad y Reutilización de código

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

conclusiones

Contenido

introducción
computación
arquitecturas
comunicación
conclusiones

- Conocer la evolución de los paradigmas de computación distribuida
- Dotar de transparencia a la integración de sistemas distribuidos y a su desarrollo
- Necesidad de conocer tecnologías y herramientas para seleccionar adecuadamente

Fundamentos de Computación distribuida

sockets **RPC**
RMI