TEMA 1 – Fundamentos de computación distribuida

Sistema Distribuido

Definición: Elementos de computación independientes, interconectados, que comunican y coordinan sus acciones a través de una red de comunicaciones.
Ejemplos: Internet, intranets privadas, computación ubica.

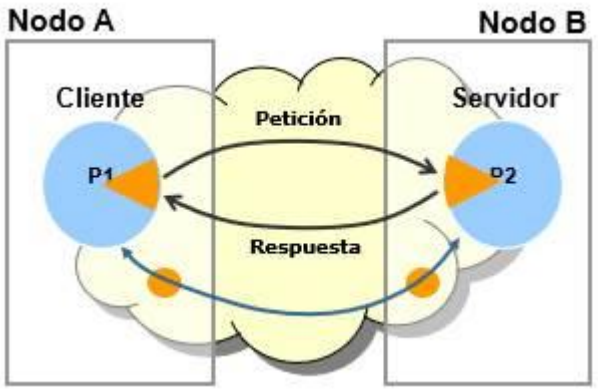
Computación Distribuida

Definición: es la que se desarrolla en un Sistema Distribuido.
Ejemplo: Servicios y aplicaciones de red,

CLIENTE/SERVIDOR

Consiste básicamente en un cliente que realiza peticiones a otro programa (el Servidor) que le da respuesta. La interacción cliente-servidor es el soporte de la mayor parte de la comunicación por redes. Ayuda a comprender las bases sobre las que están construidos los algoritmos distribuidos.

Imagen con el funcionamiento del modelo cliente/servidor:



Roles que componen el sistema:

- **Cliente:** Solicita un servicio al servidor y se queda esperando por una respuesta. Su tiempo de vida es finito una vez que son servidas sus solicitudes, termina el trabajo.
- **Servidor:** Proveedor de servicio en espera pasiva la llegada de peticiones de los procesos clientes y gestiona el acceso a dichos recursos en su nombre.

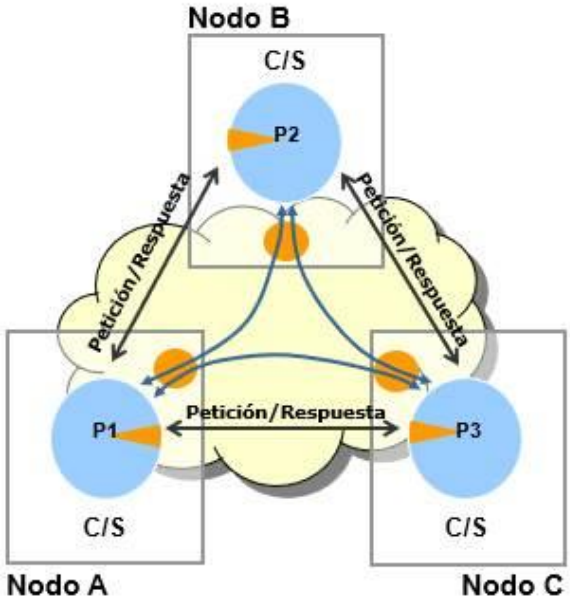
REQUISITOS

- Son necesarios mecanismos de concurrencia que permitan al servidor gestionar el acceso de muchos clientes al mismo tiempo.
- En algunas implementaciones puede ser necesario el mantenimiento de forma continua de los accesos que un determinado cliente realiza al servidor, mantenimiento de la sesión.

PEER TO PEER

La arquitectura P2P (peer-to-peer) está basada en la unión de ordenadores en una red y cada ordenador realiza la labor de cliente y servidor al mismo tiempo.

Imagen con el funcionamiento del modelo P2P:



Los procesos se realizan de igual a igual

- Cliente (envío de peticiones, recepción de solicitudes)
- Servidor (recepción de solicitudes, procesamiento de solicitudes, envío de respuesta, propagación de solicitudes)

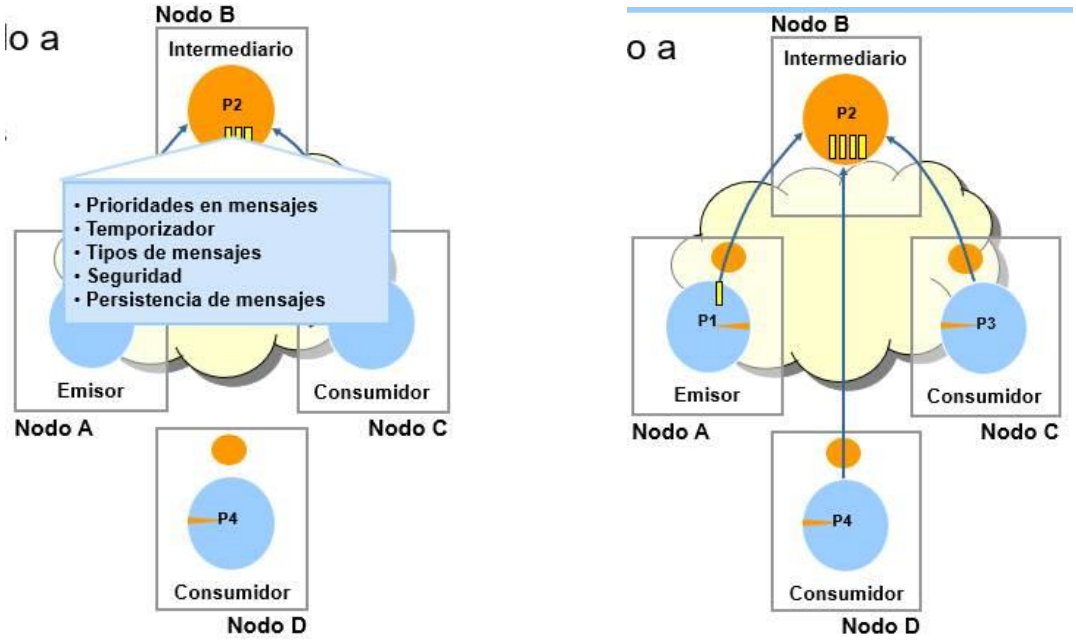
Este tipo de arquitecturas es apropiado para aplicaciones de tipo: mensajería instantánea, compartición de archivos, video conferencia y trabajo colaborativo.

Para ello hace uso de las herramientas como JXTA, por ejemplo: Napster (centralizada), Gnutella (descentralizada). Además, existen redes P2P hibridas que utilizan un servidor central para mantener la información sobre las redes o nodos, por ejemplo, BitTorrent.

MIDDLEWARE ORIENTADO A MENSAJES (MOM)

Permite a aplicaciones distribuidas comunicar e intercambiar información a través del envío y recepción de mensajes.

Imagen con el funcionamiento del modelo MOM:



La comunicación entre el emisor y el receptor del mensaje se produce de una forma completamente desacoplada. Este modelo es utilizado en intercambio de mensajes de forma asíncrona, donde el emisor no necesita una respuesta inmediata del receptor tras enviar el mensaje.

Los elementos que participan son los siguientes:

Proceso emisor: que es el responsable de la emisión del mensaje

Proceso intermedio: El que se encarga de almacenar los mensajes enviados por el emisor, el agente es un middleware que incluye un conjunto de servicios que permiten el tratamiento de mensajes.

El Middleware posee una gran variedad de características como: Evolución de paso de mensajes, gestión de seguridad, etc.

Tecnologías basadas en este modelo son: JMS, MSMQ, MQSeries

El proceso consumidor que se conecta al proceso intermediario para obtener los mensajes y procesarlos.

Se pueden distinguir dos tipos de MOM:

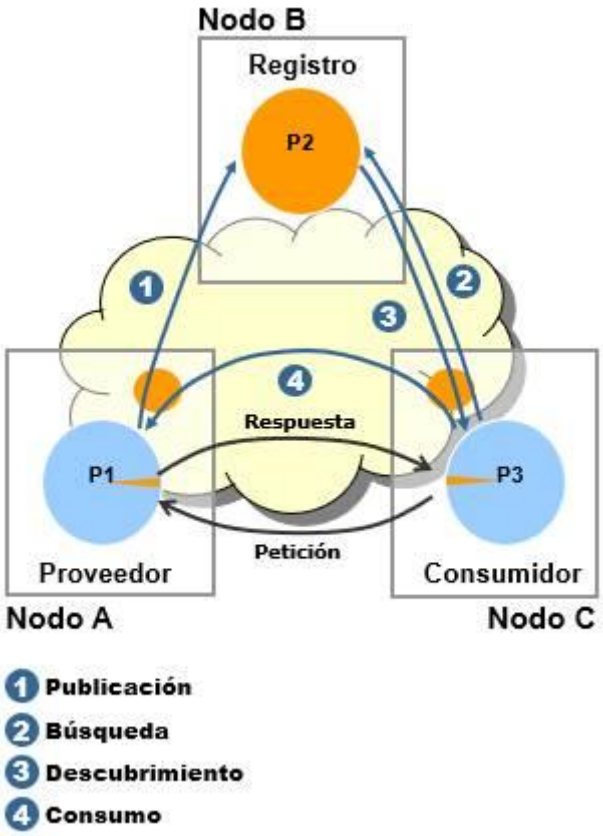
Punto a punto (1:1) → cada mensaje enviado por el emisor únicamente será procesador por un consumidor, cuando el mensaje lo tiene el consumidor al procesarlo ese mensaje es borrador del agente intermediario.

Publicación / Suscripción (1:M) → Un mensaje publicado por el emisor será procesado por todos los agentes consumidores que se hayan suscrito al proceso intermediario.

MODELO DE ARQUITECTURA ORIENTADA A SERVICIOS (SOA)

Presenta la gestión de recursos de un sistema distribuido como servicios de red que son publicados y descubiertos por los procesos clientes para su consumo.

Imagen con el funcionamiento del modelo SOA:



Un aspecto clave en este modelo es la aparición de un agente intermediario (servicio de directorio) que posibilita la localización de los servicios disponibles.

Los componen:

Proveedor de servicios: expone el acceso a un recurso como un servicio de red. Se encarga de publicar la información en el servicio para que permita localizar el servicio y acceder a él.

Consumidor de servicios: Accede a un recurso a través de un servicio de red. El consumidor se conecta al servicio para localizar el servicio que ofrezca la función que necesite el consumidor.

Servicio de registro: almacena la información necesaria sobre los servicios para que un consumidor pueda localizar y descubrir los servicios y consumirlos.

Dos características claves de este modelo son:

- **Interoperatividad.**
- **Reutilización**.
- Las tecnologías basadas en este modelo permiten sin gran esfuerzo diseñar aplicaciones complejas a partir de la composición de servicios heterogéneos.

Otras características de este modelo también son:

- Autonomía y autocontenidos.
- Reusabilidad.
- Desacoplamiento.
- Contrato bien definido, etc.

Herramientas de tecnologías basadas en el modelo SOA son:

- UPNP
- JINI
- Servicios Web

GRID Y CLUSTER

Aspectos comunes:

- Son infraestructuras hardware y software destinadas para ofrecer mayor capacidad de procesamiento y almacenamiento.
- Son un conjunto de computadores que conforman un super computador.

Cluster

Se caracteriza…

- …porque los elementos que lo componen se encuentran fuertemente acoplados a nivel de hardware o de software.
- Los nodos que forman el cluster son interconectados a través de redes de alta velocidad que permita una comunicación eficiente.
- El modelo Cluster generalmente suele aplicarse a un problema concreto y en un entorno dedicado.

En la mayoría de los casos, los clusters presentan las siguientes características:

- Alta disponibilidad.
- Balanceo de la carga.
- Escalabilidad.
- Alto rendimiento.

En la actualidad, las técnicas de cluster son aplicadas en:

- Servidores Web.
- Servidores de aplicaciones.
- Sistemas de información.
- Resolución de problemas con necesidades de supercómputo.

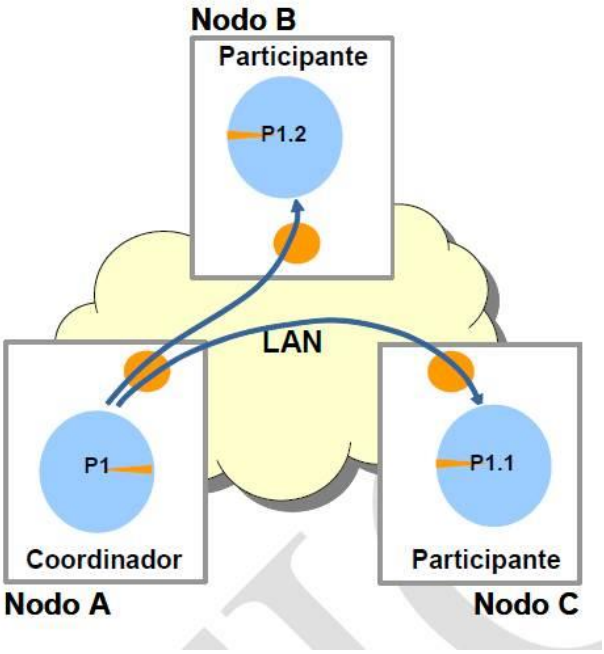
Una característica importante de los cluster, a diferencia de los Grids, es que existe una pérdida de independencia en cuanto al procesador. Esto significa que cuando un proceso es enviado a uno de los nodos del cluster consume procesador, y puede ralentizar la ejecución de otras aplicaciones en ese mismo nodo.

En u cluster exste un elemento que se encarga de la gestión de los recursos del sistema global y de repartir la carga entre los diferentes componentes. De este modo se muestra una visión de los recursos como uno único.

Ejemplos de cluster:

- MOSIX, OpenMoxix, Heartbeat, Beowulf.

Imagen del funcionamiento del modelo Cluster:



Grid

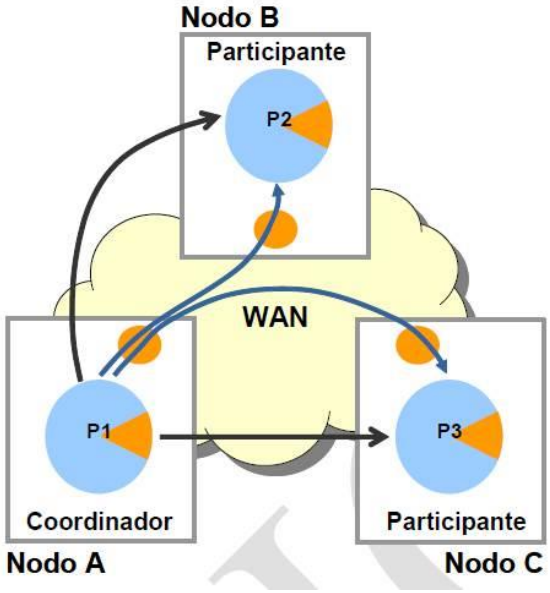
Se caracteriza…

- …porque los elementos que lo componen son totalmente heterogéneos conectados a través de redes de área amplia completamente distribuidas como Internet.
- En la mayoría de los casos, los recursos (procesamiento y almacenamiento) son cedidos por usuarios particulares para lograr el objetivo del sistema.
- Estas características (los dos puntos anteriores a este) hacen necesarios la existencia de mecanismos que garanticen la seguridad de acceso y transmisión de información entre otros.

Características importantes:

- Existe un completo desacoplamiento entre los nodos que conforman el Grid.
- A diferencia del cluster, un computador no suele perder la independencia de procesamiento, ya que el Grid consume recursos aprovechando los tiempos muertos del sistema.
- Cada nodo perteneciente al Grid es responsable de la gestión de sus recursos, obteniendo una gestión y planificación completamente distribuida.
- A diferencia del Cluster, no se busca ofrecer una visión única del sistema.

Imagen del funcionamiento del Grid:



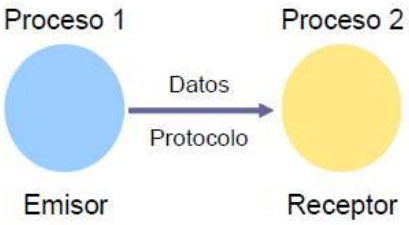
Información adicional (lo pongo por si las moscas porque habla de SOA pero no creo que sea necesario):

- Existen propuestas, como OGSA (Open Grid Service Architecture), para lograr una arquitectura estándar de la tecnología Grid que normalice los servicios propios de este modelo. Esta arquitectura se sustenta sobre el modelo SOA, visto anteriormente, usando tecnología de servicios Web como base para lograr una mayor interoperación en la comunicación de los componentes del Grid.

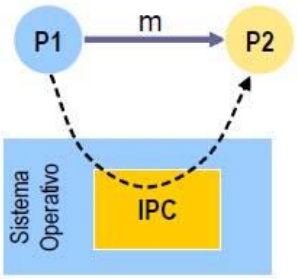
Mecanismos de comunicación entre procesos (IPC)

Consiste en el intercambio de información entre dos procesos, un emisor y un receptor, a través de la red de comunicaciones mediante un protocolo que establezca las reglas de comunicación.

Estos mecanismos pueden ubicarse dentro o por encima del sistema operativo como librerías.



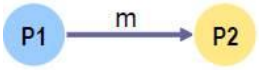
Elementos que intervienen en la comunicación entre procesos



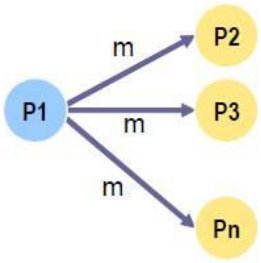
Arquitectura para la comunicación entre procesos mediante IPC

Modelos de comunicación:

- **Unidifusión**: comunicación desde un único proceso emisor a un único proceso receptor.
- **Multidifusión**: cuando el proceso de comunicación se establece desde un proceso emisor a varios procesos receptores.
- Imagen del funcionamiento de ambos modelos:



Enfoque Unicast



Enfoque Multicast

Aprender:

- Los mecanismos IPC **deben** definir interfaces que permitan establecer la comunicación entre procesos distribuidos. Las primitivas incluidas en los mecanismos IPC pueden variar el rango de complejidad en función de la abstraccion que ofrezcan.

Un **ejemplo de interfaz** de programacion que proporciona el mínimo nivel de abstracción para lograr la comunicación entre procesos puede contener las siguientes primitivas:

- Enviar.
- Recibir.
- Conectar (solicitar-conexión / aceptar-conexión).
- Desconectar.

Existen ciertos aspectos en la comunicación de procesos que se deben tener en cuenta en el diseño de interfaces de programación para la comunicación de procesos:

- **Sincronización**: para poder sincronizar los eventos que se producen en la comunicación entre procesos se usan operaciones bloqueantes. Cuando un proceso invoca una operación su procesamiento queda suspendido hasta que la operación invocada haya termiando.
- Las operaciones bloqueantes que facilitan la sincronización pueden llevar a un proceso a un estado de bloqueo permanente o temporalmente no aceptable.

Paradigma Paso de Mensajes

(No lo veo importante, pero lo pongo por si las moscas. Copiado y pegado del libro de Virgilio)

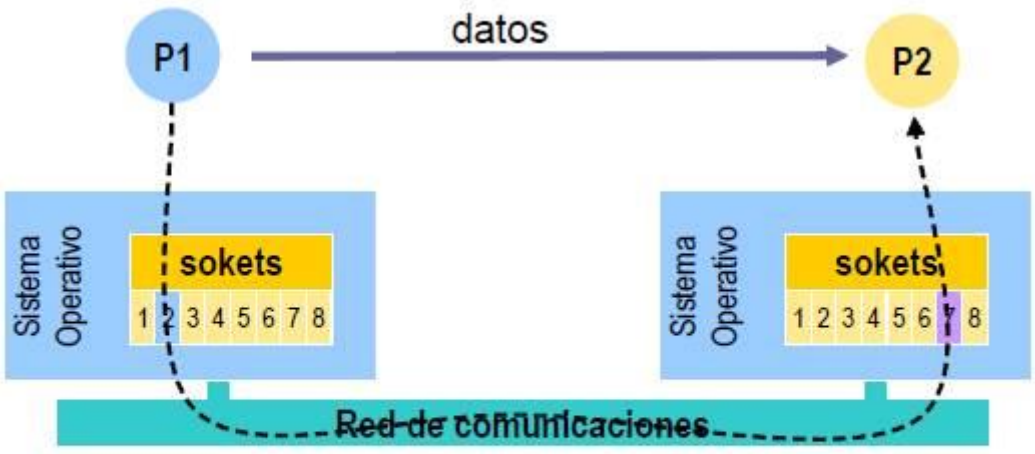
El paradigma de paso de mensajes es el modelo de comunicación entre procesos distribuidos más básico y sobre el cual se sustentan el resto de mecanismos. En el paradigma de paso de mensajes el proceso emisor envía información, que representa un mensaje, a un proceso receptor a través de un canal de comunicaciones.

La interfaz mínima requerida en este tipo de mecanismo debe ofrecer las operaciones enviar y recibir para poder transmitir los datos entre el emisor y el receptor. Además, el mecanismo puede estar orientado a la conexión, con lo cual la interfaz incluirá también las operaciones necesarias para tal fin: conectar y desconectar.

Sockets (Paso de mensaje)

Es un mecanismo IPC de intercambio de datos entre procesos que fueron originalmente desarrollados como una biblioteca de programación.

Imagen del funcionamiento de sockets:



Comunicación entre dos procesos remotos a través de sockets

Permite la comunicación entre:

- Procesos ubicados en una misma máquina.
- Procesos ubicados en diferentes máquinas.
- Familias de diferentes protocolos.
- Protocolos TCP/IP. (Familia INET. Es la **más usada**)

El API de socket ofrece primitivas que permiten orientar la comunicación a la conexión o no orientarla a la conexión:

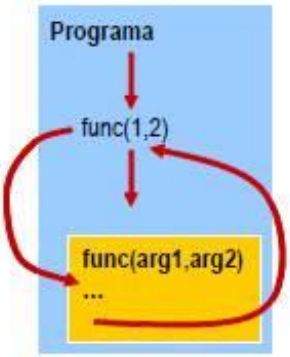
- **Orientada a conexión**:
  - Se garantiza la confiabilidad y el orden del mensaje.
  - En la familia INET, esta comunicación se realiza a través del protocolo TCP y en ella se establece una comunicación entre el emisor y el receptor de flujo de datos como si se tratase de una tubería.
- **No orientada a la conexión:**
  - La información se envía como paquetes discretos.
  - No existe garantía de que la información llegue al receptor.
  - Tampoco existe la garantía de que la información llegue en el orden adecuado.
  - INET hace uso del protocolo UDP para establecer este tipo de comunicación.

RPC (Llamadas a procedimientos remotos)

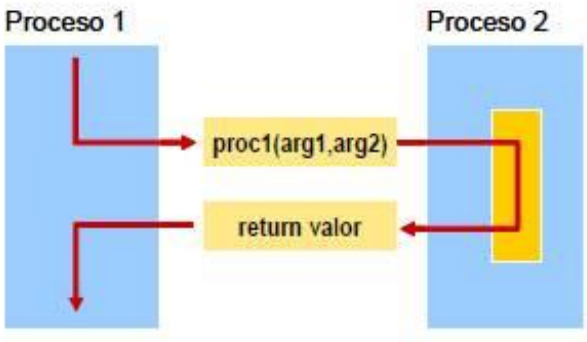
EL RPC…

- Busca poder invocar un procedimiento ubicado en una máquina remota.
- Se basa en el modelo de protocolo petición/respuesta.
- Cuando desde un proceso se realiza una petición para invocar un procedimiento remoto, el proceso emisor suspende su ejecución hasta que recibe una respuesta del proceso remoto.

Imagen del funcionamiento de RPC:



Llamada a una función o procedimiento local a un programa



Llamada a un procedimiento remoto ubicado en un proceso diferente

Características del RPC:

- Permite al desarrollador abstraerse de los detalles de la comunicación a través de la red.
- Esta abstracción se consigue utilizando las librerías llamadas **Stubs**. Los stubs son:
    - Aplicaciones proxies con las que realmente establece la comunicación la aplicación cliente.
    - Se encargan de codificar la información a un lenguaje de representación intermedio (XDR), ordenar la secuencia de datos, realizar el proceso de marshalling, establecer la comunicación, etc.

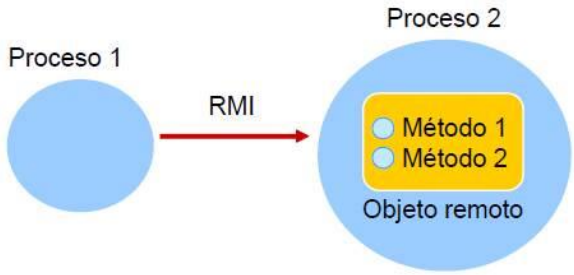
Una evolución de este modelo que está teniendo gran repercusión en la comunicación de servicio Web es XML-RPC. Se trata de una especificación que define el mecanismo de llamadas a procedimientos remotos usando XML como lenguaje de representación de datos.

RMI (Invocación de métodos remotos)

Importante:

- RMI es un mecanismo IPC (Mecanismos de Comunicación entre Procesos) similar al RPC (Llamadas a procedimientos remotos) pero orientado a la invocación de métodos de objetos en lugar de a procedimientos de aplicaciones.

Imagen del funcionamiento de RMI:



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

Características:

- Al igual que en RPC, se establecen librerías que permiten abstraer al desarrollador de los aspectos relacionados con la comunicación a través de la red.
- En la invocación a métodos remotos se establecen librerías similares (a RPC) llamadas Stubs (en el cliente) y Skeletons (en el servidor).
- RMI utiliza como protocolo de transporte binario JRMP.

Problema que saber sobre RMI:

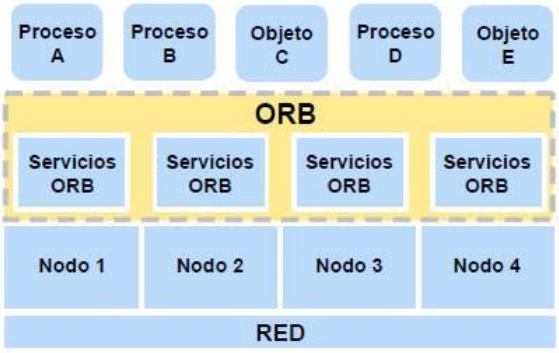
- La comunicación se realiza únicamente entre objetos escritos en el mismo lenguaje o plataforma.

ORB (Intermediario de petición de objetos)

Características:

- Es uno de los mecanismos de más alto nivel.
- Su principal objetivo es establecer la comunicación entre objetos escritos en diferentes lenguajes y diferentes plataformas.
- Es la base de la arquitectura CORBA (Common Object Request Brorker Architecture). Definida por el OMG.

Imagen del funcionamiento de ORG:



Paradigma de objetos remotos

Los elementos básicos que conforman esta arquitectura son: (añadido por si las moscas)

- ORB (agente intermediario para la gestión de objetos).
- IDL y CDR (definición de interfaces y representación externa).
- Giop (protocolos de comunicación)
    - IIOP → TCP/IP
    - HTIOP → http

En el libro pone que este modelo se presenta con más detalle en el tema 2 de la asignatura (Tecnologías Web y Middleware).

WS (Servicios Web)

Esta tecnología se ha establecido como una de las grandes promesas en la computación distribuida.

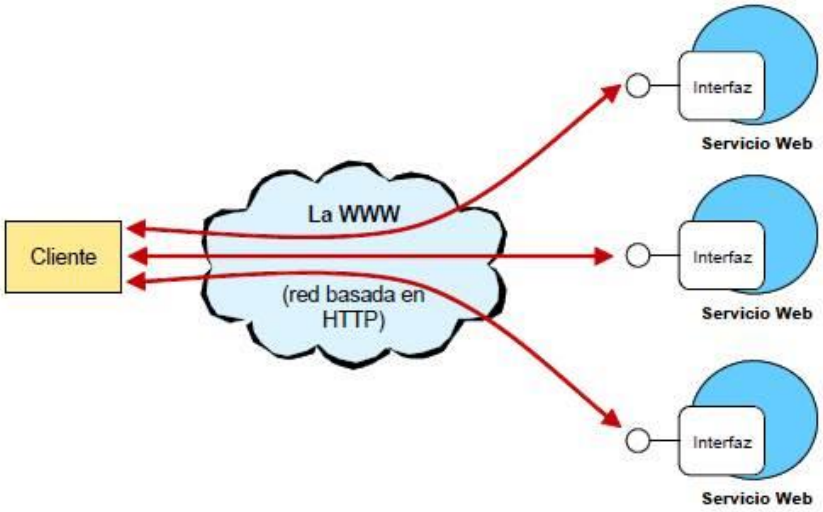
Características:

- Permiten (al igual que el modelo ORB) la interoperación de aplicaciones escritas en diferentes lenguajes y plataformas.
- Utiliza protocolos de un nivel de abstracción mucho mayor. Estos protocolos se fundamentan en el uso de XML como lenguaje de representación.
- Los servicios Web permiten un **completo desacoplamiento entre** las **aplicaciones**.
- Su uso es idóneo para la gestión de procesos de negocios (BPM) gracias a su completo desacoplamiento.

Los principales elementos o protocolos que conforman dicha tecnología son:

- **SOAP**: es el protocolo de transporte. Evolución del XML-RPC.
- **WSDL**: es el lenguaje de definición de interfaces.
- **UDDI**: es el servicio que almacena la información de los servicios y que permite la publicación, localización y consumo de servicios. Ofrece los servicios en términos de negocio.

Imagen del funcionamiento de Servicios Web:



Modelo conceptual de los Servicios Web