Maestría en Ingeniería de Software

Sistemas Distribuidos en Web I

MCC. Carlos Albeto Ochoa Rivera

Descripción general

Actualmente existe la tendencia de desarrollo de software que trabaje en un ambiente de red, con acceso a la información desde cualquier lugar a través de Internet, por ello en esta experiencia educativa se revisan los elementos básicos para poder desarrollar sistemas distribuidos bajo esta plataforma web siguiendo una metodología para su diseño e implementación.

Objetivo general

Al terminar el curso el alumno conocerá a detalle el funcionamiento de los sistemas distribuidos así como también las diferentes metodologías y lenguajes de script para desarrollar sistemas distribuidos en Web.

Contenidos temáticos

- Introducción a los sistemas distribuidos
  - Características
  - Ventajas y Desventajas
  - Campos de aplicación
  - El modelo cliente servidor
- Comunicación de aplicaciones
- El modelo de comunicación
  - El modelo RPC
  - Java RMI
  - CORBA, DCOM
  - Samba
- Metodologías para el desarrollo Web
  - Usabilidad
  - HDM
  - OOHDM Object-Oriented Hypermedia Design Model

Contenido temático...

- Herramientas para el Desarrollo Web
  - HTML
  - XHTML
  - CSS
  - DHTML
  - Elementos Multimedia
- Desarrollo Web dinámico
  - Javascript
  - Lenguajes de Script
  - Conectividad con base de datos
  - JAVA (Applets)

Evaluación

- Investigación y proyecto de programación con seguimiento mediante TSP (50%)
- Se aplicarán dos exámenes escritos que comprendan todo lo visto en clase antes de la aplicación del examen. (20%)
- Ejercicios prácticos. (30%)

Actividad

1. ¿Qué es un Sistema Distribuido?
2. ¿Cuál es la diferencia entre un Sistema Distribuido y una Red de Computadoras?
3. ¿Qué entiende por Arquitectura Cliente Servidor?
4. ¿Cuáles son los elementos de una Arquitectura Cliente Servidor?
5. ¿Qué características muestra el modelo Cliente Servidor?
6. ¿Cuáles son las ventajas y desventajas del modelo Cliente Servidor?
7. ¿Qué servicios ofrece el modelo Cliente Servidor?

Índice

- Definición y concepto
- Clasificación de sistemas distribuidos
- Aspectos referentes al proceso de ingeniería
- Caso particular: aplicaciones Web
- Aspectos relativos al coste

Definición y concepto

- Un sistema distribuido es aquel en el que dos o más máquinas colaboran para la obtención de un resultado. En todo sistema distribuido se establecen una o varias comunicaciones siguiendo un protocolo prefijado mediante un esquema cliente-servidor.

Definición y concepto

- En un esquema cliente-servidor, se denomina cliente la máquina que solicita un determinado servicio y se denomina servidor la máquina que lo proporciona. El servicio puede ser la ejecución de un determinado algortimo, el acceso a determinado banco de información o el acceso a un dispositivo hardware.

Definición y concepto

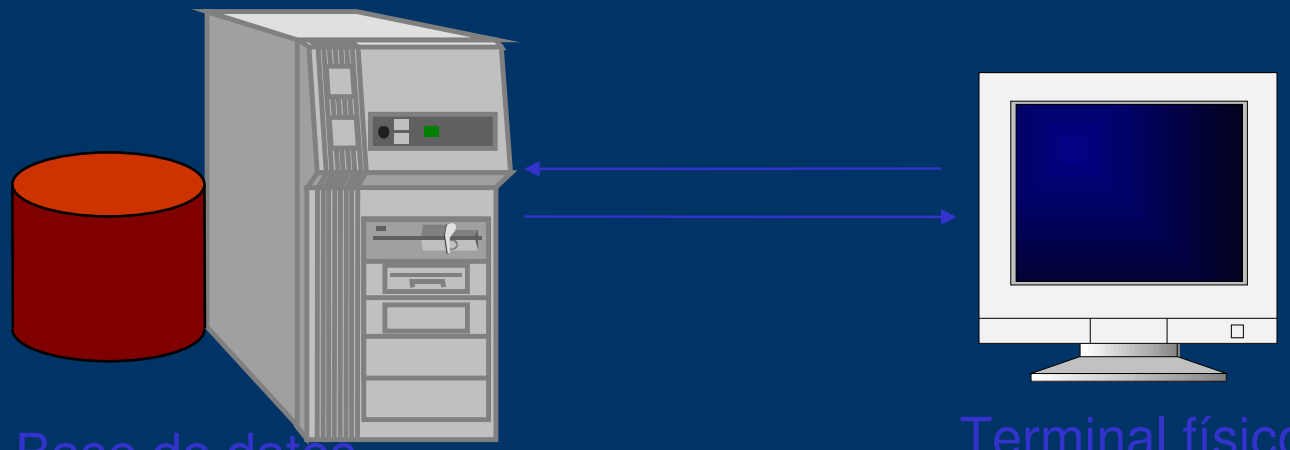
- Por extensión, se puede aplicar el esquema cliente-servidor dentro de una misma máquina, donde el proceso servidor y el proceso cliente son dos procesos independientes que corren dentro de la misma instancia de sistema operativo.
- Es por tanto un elemento primordial para que haya un sistema distribuido, la presencia de un medio físico de comunicación entre ambas máquinas, y será la naturaleza de este medio la que marque en muchos casos la viabilidad del sistema.

Clasificación

- Se clasifican los sistemas cliente servidor de acuerdo al nivel de abstracción del servicio que se ofrece. Se distinguen tres componentes básicos de software:
  - Interacción con el usuario
  - Lógica de Aplicación
  - Repositorio de datos

Clasificación

- 1. Representación distribuida. La interacción con el usuario se realiza básicamente en el servidor. El cliente hace de pasarela, de sistema de acceso a los elementos hardware pantalla y teclado.

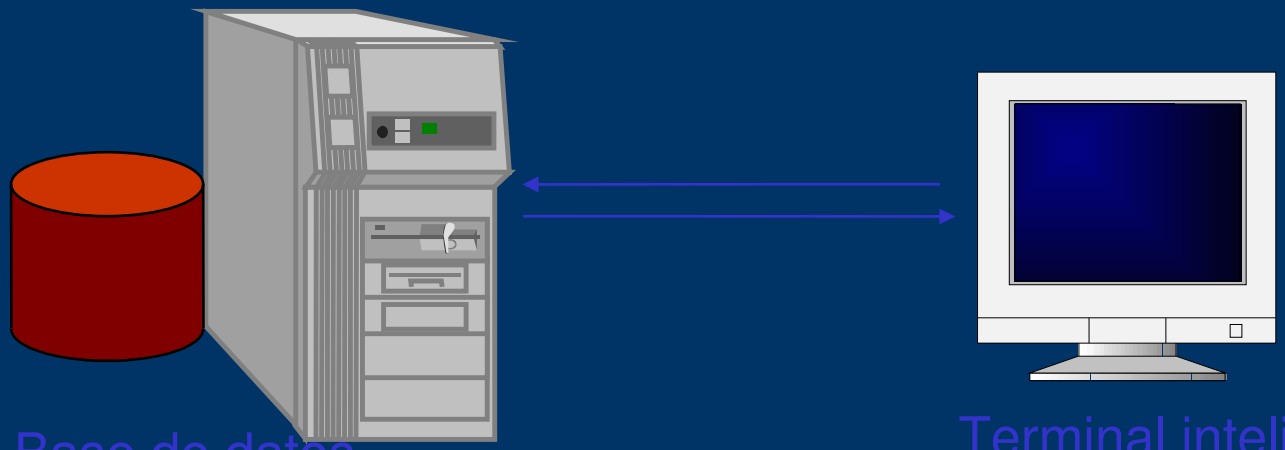


Base de datos
Lógica de aplicación
Interface de usuario

Terminal físico

Clasificación

- 2. Representación remota. Los datos se envían sin formatear, y es el cliente el responsable de formatear los datos y realizar las acciones de interacción con el usuario. En este caso, la aplicación y la base de datos se encuentran en el servidor

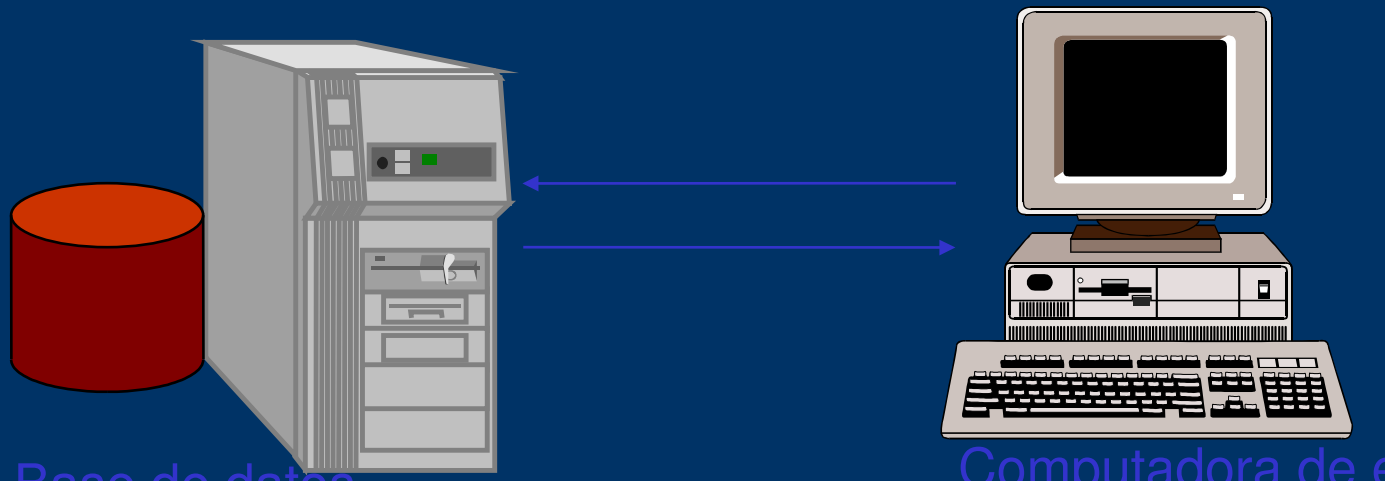


Base de datos
Lógica de aplicación
Interface avanzado de usuario

Terminal inteligente
Intarface básico de usuario

Clasificación

- 3. Lógica distribuida. En el cliente se llevan a cabo la interacción con el usuario y la parte más trivial de la lógica de la aplicación. En este caso, se llevan a cabo controles básicos de rango de campos, campos obligatorios, etc, mientras que el grueso de la lógica permanece en el servidor.

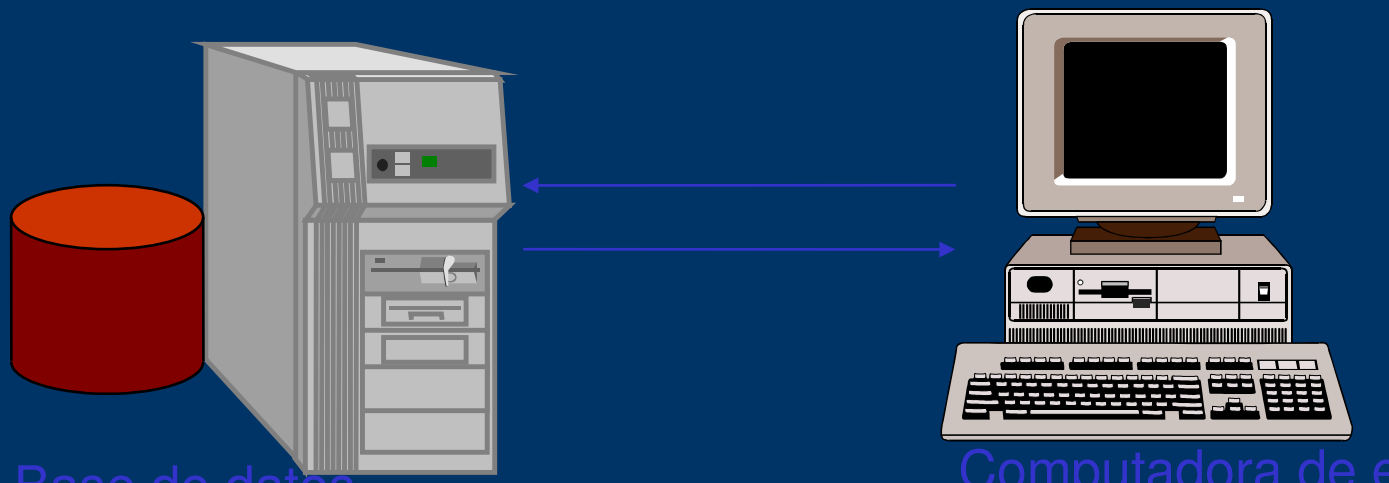


Base de datos
Lógica de aplicación

Computadora de escritorio
Lógica básica de aplicación
Interface de usuario

Clasificación

- 4. Gestión remota de datos. Tanto la interacción con el usuario como la aplicación residen en el cilente, siendo el servidor el depositario de los datos.

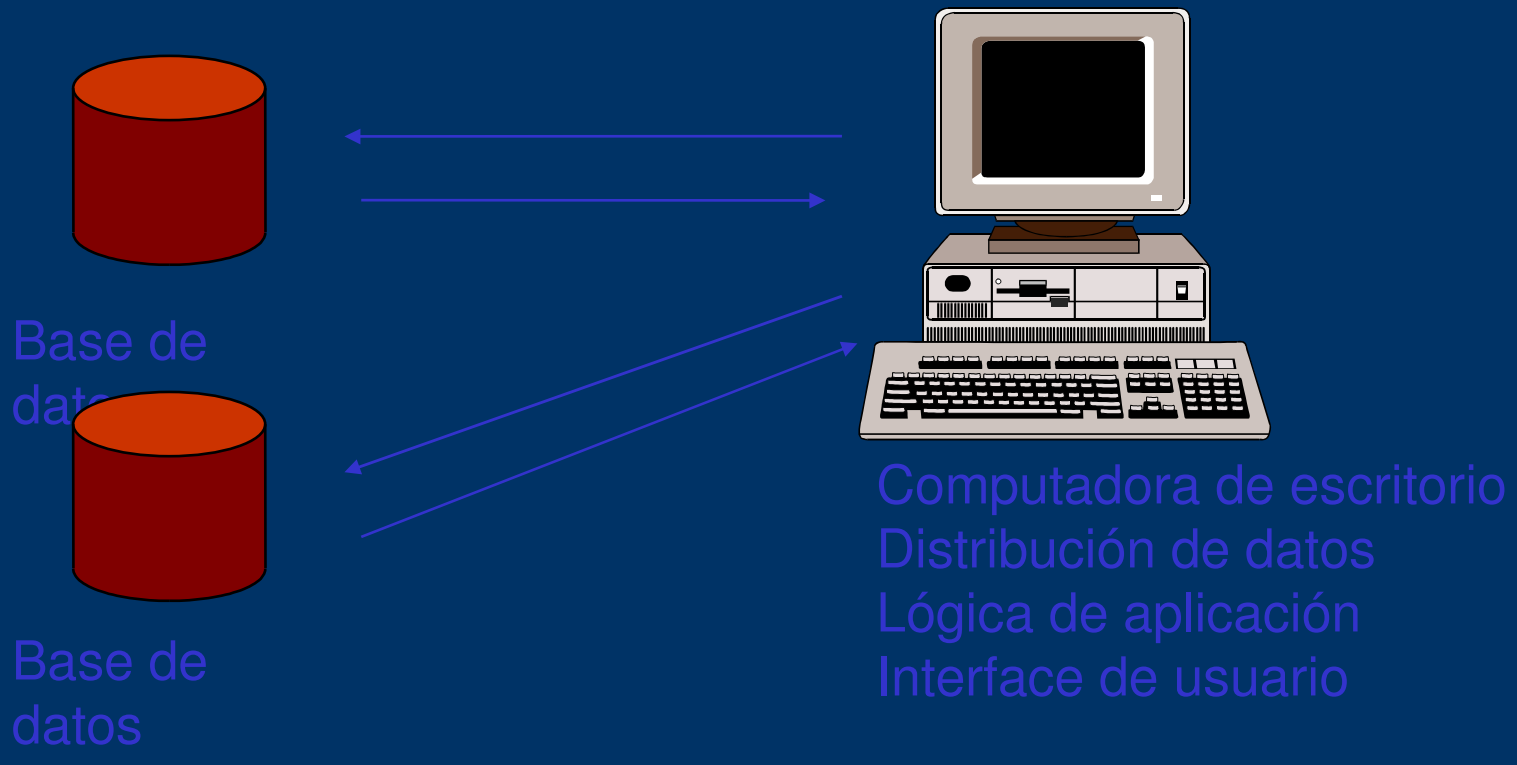


Base de datos

Computadora de escritorio
Lógica de aplicación
Interface de usuario

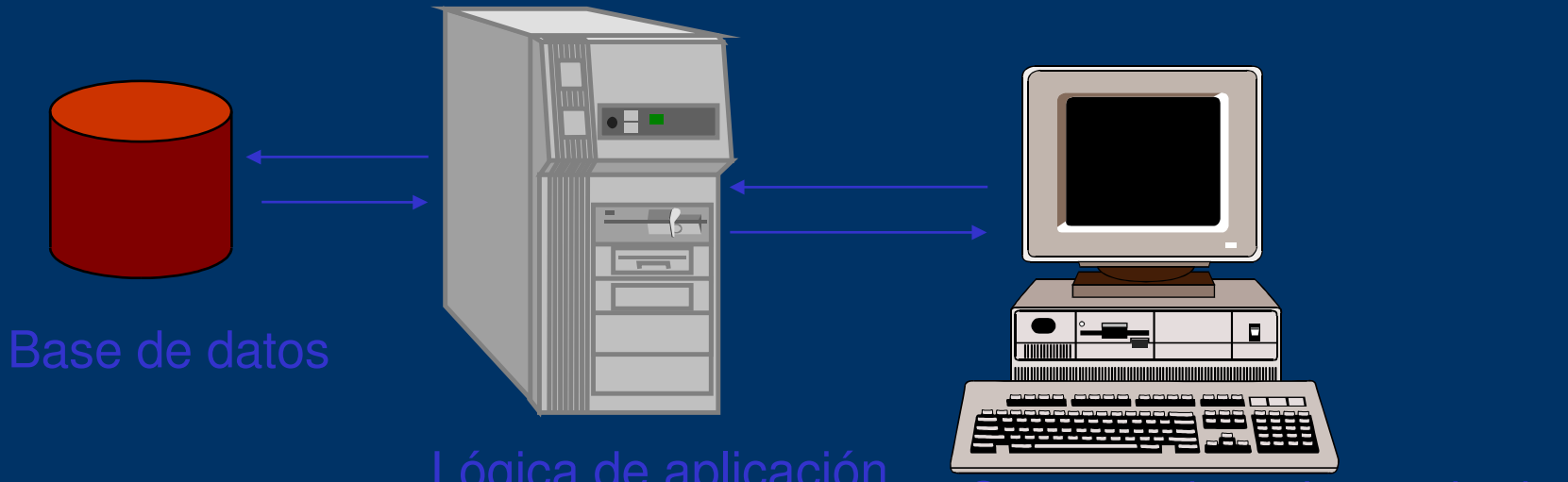
Clasificación

- 5. B.D. Distribuidas. El cliente debe conocer la topología de la red, así como la disposición y ubicación de los datos. En este caso, se delega parte de la gestión de base de datos a los clientes.



Clasificación

- Cliente servidor a tres niveles (three tier). La aplicación se distribuye en los tres niveles: aplicación, datos e interface de usuario



Aspectos a tener en cuenta en el proceso de ingeniería

- Protocolos de comunicaciones:
  - Son más importantes que la propia arquitectura distribuida o centralizada. Un buen protocolo permite que se pueda pasar, sin un coste adicional de rediseño o codificación, de una arquitectura centralizada a una distribuida, y viceversa:
    - Pipes
    - RPC
    - SQL Remoto
    - HTTP
    - X11
    - Otros

Aspectos a tener en cuenta

- Middleware. Es la herramienta o conjunto de herramientas que nos permitiran gestionar y coordinar los mecanismos de comunicación.
  - Independiza el servicio y su implementación, del S.O. y protocolos de comunicaciones
  - Permite la convivencia de distintos servicios en una misma máquina
  - Modelo OO: CORBA

Aspectos a tener en cuenta

- Fase de análisis:
  - Prácticamente no hay diferencias respecto a un S.I. tradicional
  - Se debe definir la política de empresa: fat client o fat server.
  - Se debe definir el coste en comunicaciones que puede asumir la organización.

Aspectos a tener en cuenta

- Fase de diseño
  - El diseño de entidades, en raras ocasiones se verán éstas afectadas
  - Aparecerán nuevos conjuntos de datos en los DFDs. No se trata de nuevas entidades, sino de información que debe viajar entre nodos
  - Respecto al diseño de tablas, se debe especificar su implementación:
    - Desde qué nodos debe ser accesible
    - Qué nivel de acceso se precisa desde cada uno de ellos
    - Cómo implementarlo

Aspectos a tener en cuenta

- Implementación BB.DD. Distribuidas
  - No hay entornos puramente distribuidos. Debe analizarse, tabla a tabla, qué distribuir, qué centralizar y cómo hacerlo:
  - Tabla única
  - Tablas con réplica simétrica on-line
  - Tablas con réplica simétrica off-line **
  - Tabla maestra más copias instantáneas
  - Tabla maestra más copias instantáneas actualizables **
  - Especial atención a las secuencias !!
  - Especial atencíón a los conflictos de réplica (**)

Aspectos a tener en cuenta

- Diseño de procesos
  - Se deberán tener en cuenta, no tan sólo los procesos de réplica y su periodicidad, sino el ancho de banda que consuman, máxime si implican tarificación por paquetes trasnmitidos:
    - Pipes y sockets -> Aproximación analítica
    - Middleware -> Información a transmitir + Sobrecoste en ancho de banda + Sobrecoste en tiempo de proceso
    - Protocolos propietarios (SQL) -> Recurrir a benchmarks o referencias del fabricante
  - Analizados los consumos de ancho de banda y tiempo estimado de proceso, se deberá replantear la idoneidad de ubicación de cada proceso
  - Extremar las pruebas cuando se requiera diseñar e implementar protocolos de comunicación

Aspectos a tener en cuenta

- Fase de pruebas. Debido a la complejidad del sistema, serán necesarias varias fases:
  - Pruebas de funcionalidad de la aplicación. Se puede llevar a cabo sobre máquinas de desarrollo y estaciones de trabajo de forma paralela
  - Pruebas de carga del servidor
  - Pruebas de integridad de datos. Son especialmente importantes en el caso de bases de datos distribuidas
  - Pruebas transaccionales
  - Pruebas de red

Desarrollos Web

- Caso particular de desarrollo cliente servidor con representación remota, en la cual disponemos de un protocolo standard: HTTP y un middleware denominado WebServer.
- Cada página puede desencadenar la solicitud de numerosos peticiones adicionales para finalizar el proceso de representación remota.
- Se dispone de un lenguaje standard de definición y formateo de páginas: HTML

Desarrollos Web

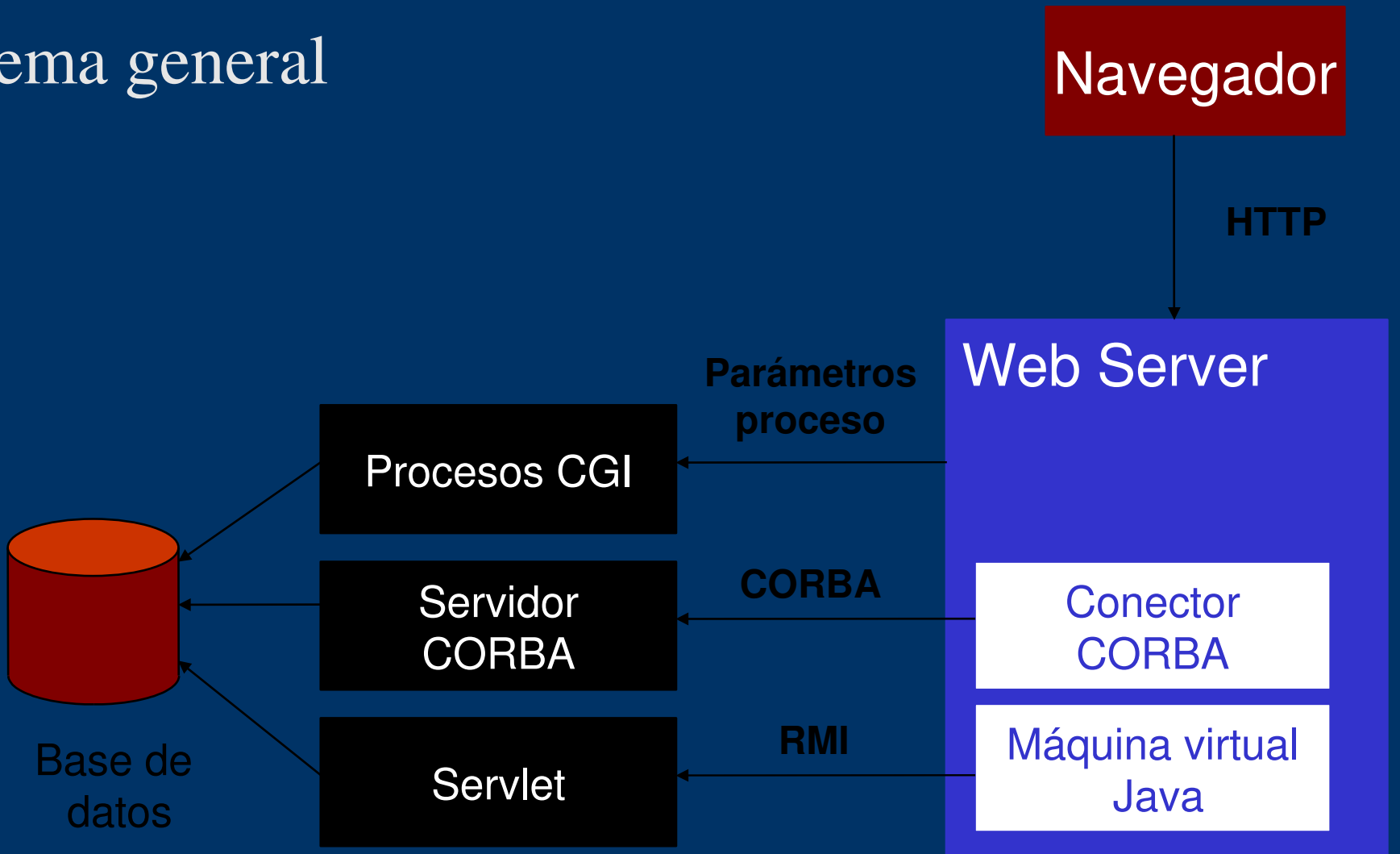
- Incrustación de la lógica de aplicación en el servidor Web:
  - CGI: Common Gateware Interface
    - Cada petición HTTP genera un nuevo proceso, el cual analiza la solicitud y genera un resultado. Cada proceso corresponde a una transacción.
    - Es flexible, ideal para pequeñas aplicaciones de uso reducido
    - No escala adecuadamente
  - Páginas ASP: Caso particular de CGI
    - Entorno propietario Microsoft
    - Aspectos de rendimiento bastante mejorados

Desarrollos Web

- Incrustación de la lógica de aplicación en el servidor Web
  - Servlets: Ejecución de aplicaciones Java en el servidor que procesan la petición y generan la página de respuesta
    - No generan un proceso adicional por cada petición
    - Utilizan un lenguaje de alto nivel (Java)
  - Objetos CORBA:
    - Pemite la integración de objetos CORBA con el servidor Web, creando una estructura cliente servidor multinivel
    - Es la solución más generalista y adaptable
    - Permite fácil, flexible y eficiente integración con BBDD

Desarrollos Web

- Esquema general



Nuevos tipos de dispositivos

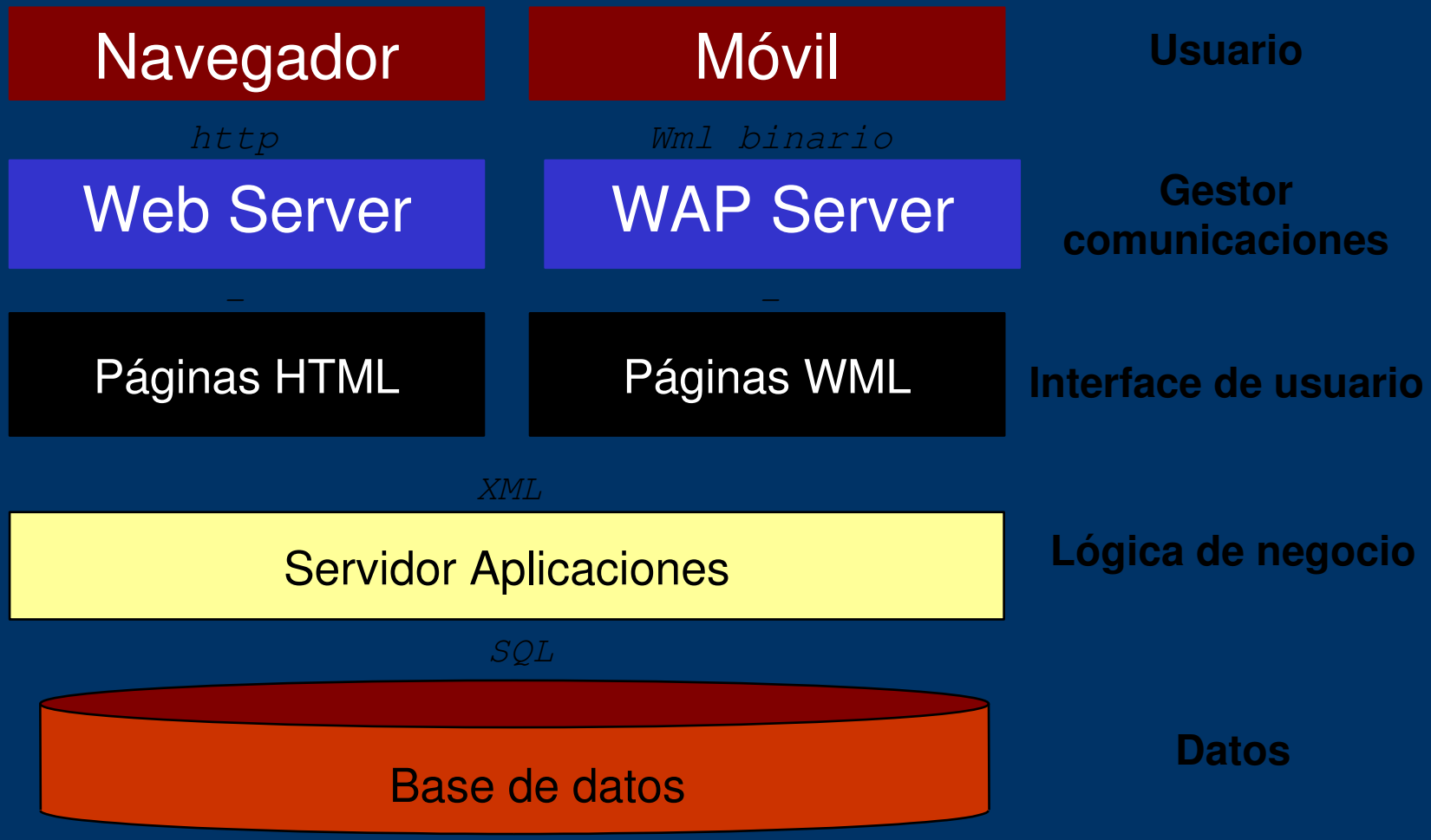
- Dispositivos que acceden hoy a internet:
  - Internet Explorer, Netscape, Set Top Box, Móviles WAP, PDAs Palm Pilot, Windows CE, ...
- Previsiones para los próximos años:
  - 2.002 el 50% de las transacciones habituales se podrán realizar desde dispositivos móviles
  - 2.003 el 80% de los usuarios realizarán algún tipo de transacción desde dispositivos móviles
  - 2.004 los se querrán realizar el 100% de las transacciones desde dispositivos móviles
  - 2.005 Se esperan más de 1.000 millones de usuarios móviles de internet

Nuevos tipos de dispositivos

- Problema a resolver:
  - Necesidad de adaptar el interface de usuario a cada tipo de dispositivo
- Medidas a tomar:
  - Separar la lógica de aplicación del interface de usuario
  - Utilizar métodos estándar de comunicación entre la lógica de aplicación y el interface de usuario
  - Uso de herramientas que permitan adaptar rápidamente las aplicaciones a los nuevos tipos de dispositivos que irán apareciendo

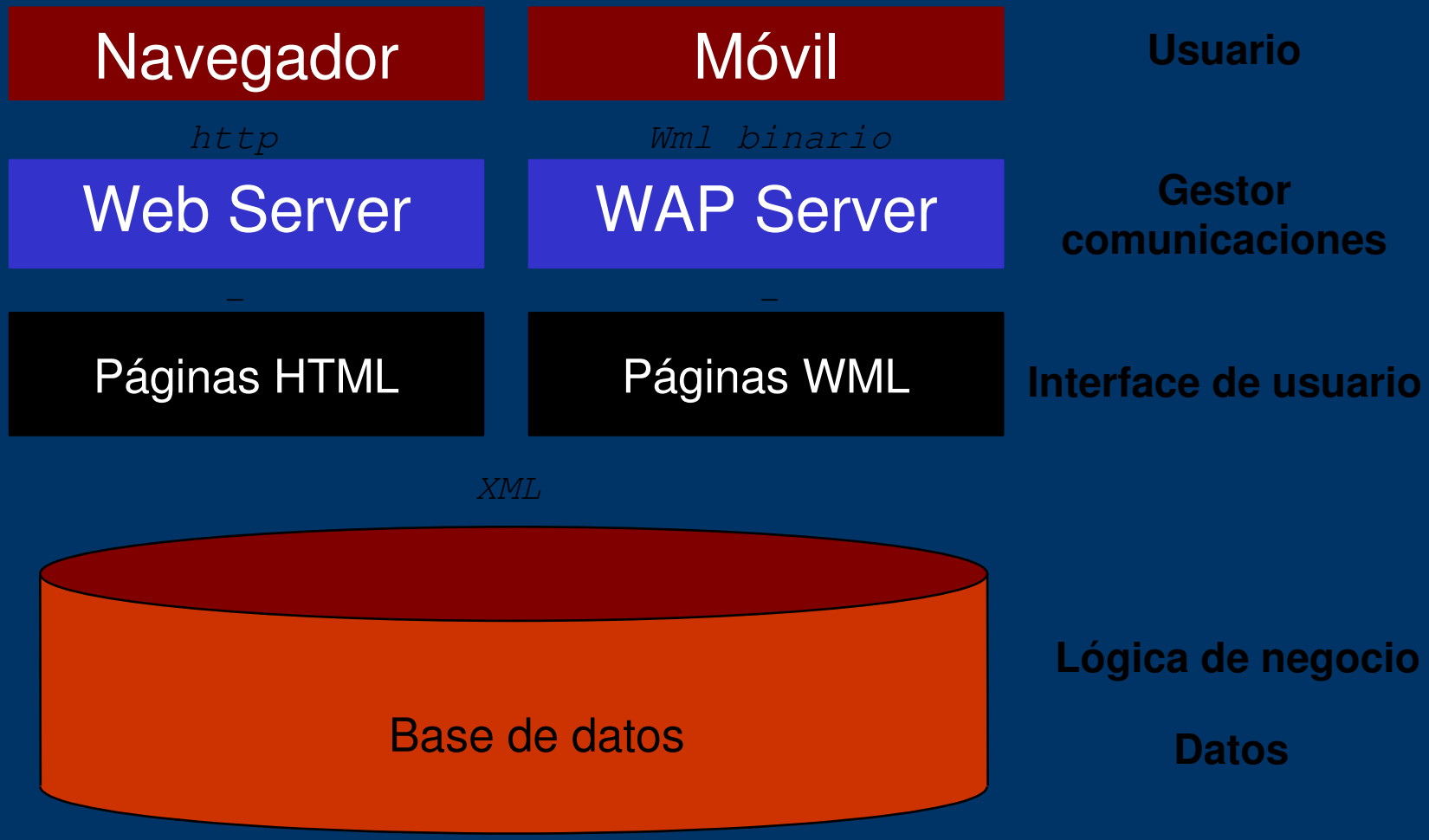
Nuevos tipos de dispositivos

- Tendencia actual



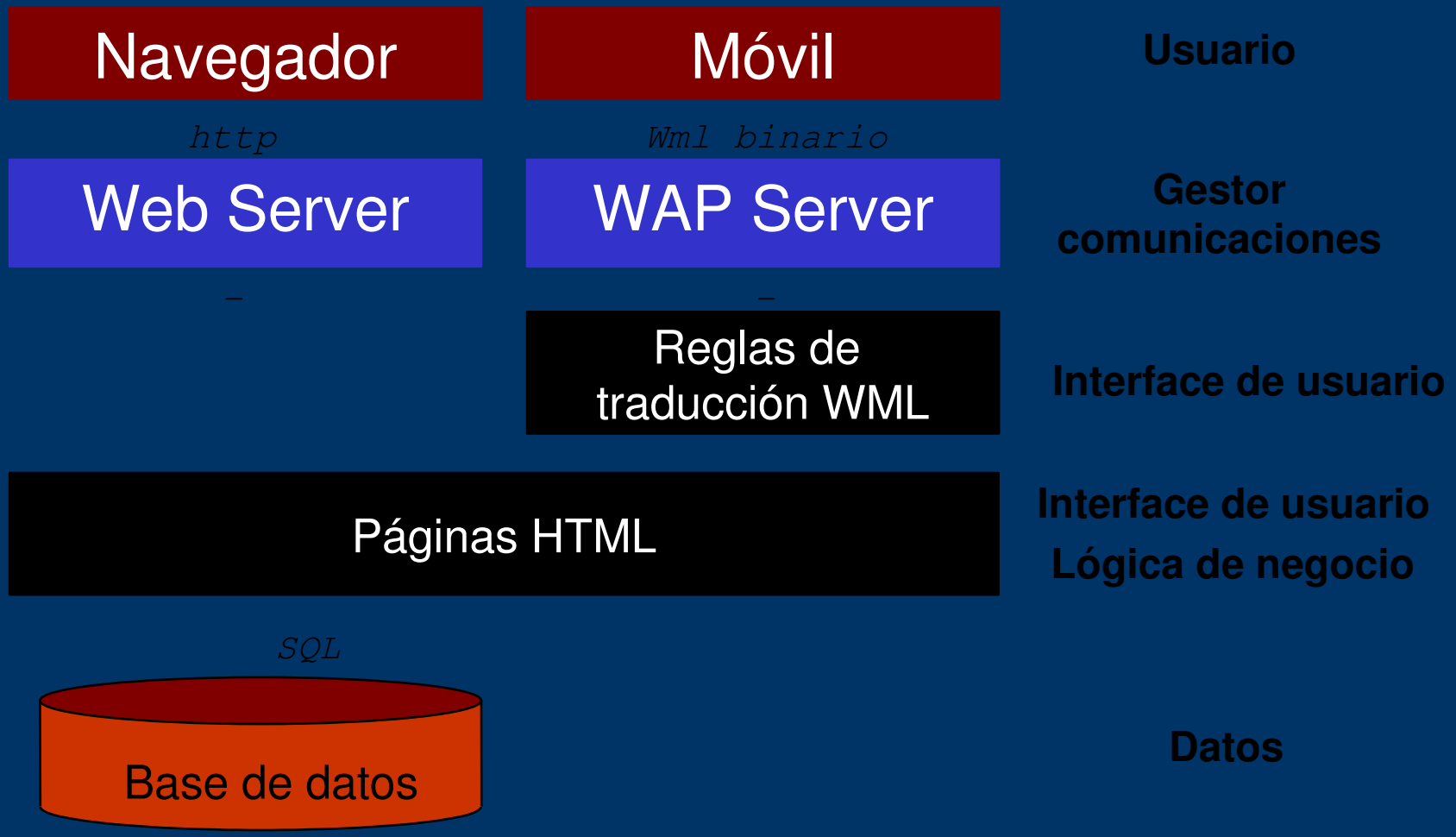
Nuevos tipos de dispositivos

- Variante de los fabricantes BBDD



Nuevos tipos de dispositivos

- Variante de los fabricantes pasarelas



Estrategia a seguir

- Valorar la durabilidad temporal de las tecnologías a aplicar
- Separar, en el diseño e implentación de la aplicación, las capas de lógica de aplicación e interface de usuario
- Prestar mucha atención a los nuevos tipos de dispositivos
- Examinar con lupa los “atajos” ofrecidos por los fabricantes

Costes sistema distribuido

- Elementos a valorar:
  - Coste de las comunicaciones: Valorar alternativas presentadas por los nuevos proveedores de telecomunicaciones. No descartar el tirar líneas propias
  - Evaluar el coste adicional en hardware, software y gestión que implica una arquitectura distribuida. Si las comunicaciones lo permiten, saldrá más rentable una arquitectura centralizada
  - El impacto de los protocolos de comunicaciones será vital en el desglose posterior de costes. Se deben dedicar todos los esfuerzos necesarios para evaluar cuál es el protocolo óptimo.