XUNTA DE GALICIA

ESCOLA GALEGA DE ADMINISTRACIÓN PÚBLICA

28. MODELO DE CAPAS: SERVIDORES DE APLICACIONES, SERVIDORES DE DATOS, GRANJAS DE SERVIDORES. INTEGRACIÓN DE CONTENIDO, SONIDO, IMAGEN Y ANIMACIÓN. SCRIPTS DEL CLIENTE.

Fecha de actualización 26 de julio de 2011

TEMA 28. MODELO DE CAPAS: SERVIDORES DE APLICACIONES, SERVIDORES DE DATOS, GRANJAS DE SERVIDORES. INTEGRACIÓN DE CONTENIDO, SONIDO, IMAGEN Y ANIMACIÓN. SCRIPTS DEL CLIENTE.

28.1. INTRODUCCIÓN Y CONCEPTOS

En una red formada por equipos informáticos los **nodos** suelen realizar tres **funciones** diferenciadas:

1. Facilitar la comunicación e interconexión de los nodos de la red.
2. Proporcionar servicios o información a otros nodos.
3. Realizar funciones de equipo de trabajo, haciendo uso de las comunicaciones, servicios e información disponible.

En este contexto los nodos o equipos que realizan las funciones de proporcionar servicio o información al resto se denominan **Servidores** y los que hacen uso de los servicios **Clientes**, formando en su conjunto lo que se da en denominar **Arquitectura Cliente-Servidor**. Se trata de una de las arquitecturas más extendidas en los entornos distribuidos, permitiendo la heterogeneidad en los clientes y un acceso transparente a la información. Los servidores permanecen a la escucha de la red en todo momento para

atender a las solicitudes o demandas de los clientes.

El esquema de funcionamiento **básico** seguiría el siguiente modelo:

1. El cliente solicita un servicio al servidor a través de la red.
2. El servidor a la escucha recibe la petición del servicio y la pone en la cola de demanda.
3. El servidor obtiene el resultado de la petición.
4. El servidor envía la respuesta de la petición al cliente a través de la red.
5. El cliente obtiene el resultado y lo procesa.

A partir de estos conceptos básicos podemos extraer las **características básicas** de la arquitectura Cliente-Servidor:

a) **Servicios**. Son la base de las peticiones entre los clientes y los servidores, se trata de cualquier entidad susceptible de ser demandada por uno o más clientes.
b) **Recursos compartidos.** Elementos y servicios de la red, tanto lógicos (software, datos e información), como físicos (hardware, impresoras, unidades en red, etc.).
c) **Comunicación asíncrona basada en el envío de mensajes.** Este tipo de arquitecturas emplean protocolos de comunicación asimétricos donde los clientes inician conversaciones y los servidores esperan que se establezca la comunicación escuchando la red. Toda la comunicación se realiza mediante el envío de mensajes y respuestas.
d) **Transparencia.** La localización, la organización lógica y física, así como la implementación de los servicios resulta transparente a los clientes. El uso de los mismos se limita a hacer una petición a la red y obtener la respuesta.
e) **Escalabilidad**. Horizontal en los clientes a la hora de permitir añadir

nuevos nodos sin más que añadirlos a la red y vertical en los servidores, de modo que administrando un único punto puede mejorarse la potencia, el rendimiento, el mantenimiento y la recuperación de errores.

Fruto de la escalabilidad de esta arquitectura surgen las **granjas de servidores**, consistentes en emplear varios servidores a la vez suministrando el mismo servicio y repartiéndose las peticiones o carga del sistema. La gestión de una granja de servidores será compleja debido a la necesidad de balancear la carga para obtener el mayor rendimiento posible.

Uno de los servicios más extendidos actualmente es el web o WWW, (siglas en inglés de *World Wide Web*), se trata de un sistema de publicación e intercambio de información distribuido que relaciona unos contenidos con otros a través de enlaces. En este servicio los clientes solicitan información a modo de páginas web, tratándose de documentos en lenguajes estándar como HTML o XML que incluyen diferentes tipos de información: texto, hiperenlaces, y elementos multimedia. Entre estos elementos multimedia encontramos:

a) **Texto.** Distinguiendo entre sin formato, con formato o enriquecido (tipo de letra, tamaño, color, color de fondo, etc.) e hipertexto texto con un vínculo o enlace a otro texto o documento.
b) **Sonido**. Digitalización del habla, la música u otros sonidos.
c) **Gráficos.** Representan esquemas, planos, dibujos vectoriales, etc. son documentos que se construyen a partir de una serie de primitivas: puntos, segmentos, elipses, etc. aplicándoles a continuación todo tipo de transformaciones o funciones: giro, cambio de atributos, escalado, efectos, etc.
d) **Imágenes**. Representaciones fieles de la realidad, como fotografías.

Son documentos formados exclusivamente por píxeles, punto a punto y por tanto no se estructuran o dividen en primitivas.

e) **Animación**. Representación de una secuencia de gráficos por unidad de tiempo, para ofrecer la sensación de movimiento. Asimismo ofrece posibilidades de interacción ante eventos.

f) **Vídeo**. Representación de una secuencia de imágenes por unidad de tiempo, para ofrecer la sensación de movimiento.

Documentos complejos agrupan diversos componentes multimedia en una misma página o documento. Los sistemas de publicación actuales permiten que la **multimedia digital on line** pueda transmitirse **en flujo** (en inglés *streaming*), que se encuentra disponible tanto on line en tiempo real como bajo demanda. En este modelo no es necesario descargar o acceder a la totalidad del documento para acceder a los contenidos sino que se proporciona acceso directo a cualquier parte del flujo y reproducción desde ese punto.

Otra característica de las páginas web o documentos HTML/XML es que pueden incluir **código de script para los clientes**. Este código representa un guión o secuencia de instrucciones a manera de un programa sencillo. Este programa puede ser interpretado por el navegador del equipo cliente con unos permisos limitados en el equipo y focalizados principalmente en la página o documento web en el que se encuentran incrustados o desde el que son llamados. Existen diferentes tecnologías para estos lenguajes de script siendo las más conocidas: Javascript, Visual Basic Script, Flash, y la evolución de Javascript: AJAX, aceptados con mayor o menor fortuna por los navegadores actuales, muchas de ellas denominadas tecnologías RIA en un acercamiento de las aplicaciones web a las aplicaciones de escritorio.

28.2 MODELO DE CAPAS: SERVIDORES DE APLICACIONES, SERVIDORES DE DATOS, GRANJAS DE SERVIDORES

La distribución de los sistemas de información fue evolucionando a lo largo del tiempo en función de las demandas y crecimiento de las redes y el aumento de la complejidad de las arquitecturas de red.

28.2.1. Arquitectura en una capa: Superordenador central.

La arquitectura más simple estaría formada por **un superordenador central** (en inglés *mainframe*) que centraliza toda la capacidad de procesamiento y almacenamiento de la red, también denominada monolítica. En este modelo el acceso a la información se hace directamente a través de la computadora principal o bien a través de clientes ligeros que se limitan a hacer las funciones de terminales. En esta arquitectura centraliza todo el coste de administración y mantenimiento se dedica al servidor central. Los terminales carecen de programas propios, y tienen recursos de memoria o disco mínimos, pudiendo incluso carecer de disco. Cualquier instalación o avance en el servidor repercute al momento en la red de modo que cualquier programa instalado estará disponible para todos los clientes. Por el contrario, si tenemos en cuenta la sostenibilidad del sistema en caso de caída o errores en el servidor central toda la red se ve afectada, al igual que si un cliente sobrecarga el sistema todos los demás se verán afectados en cuanto a rendimiento. A su vez los mainframes se organizan según arquitecturas paralelas tipo **SNA** (en inglés *Systems Network Architecture*) con un diseño de red con comunicación P2P a través de APPN (en inglés *Advanced Peer-to-Peer Networking*).

28.2.2. Arquitectura en dos capas: Modelo Cliente-Servidor.

En este modelo el sistema se estructura en dos capas, una capa a nivel de usuario que almacena y procesa parte de la información y otra capa remota a nivel de servicios que almacena y da funcionalidad a la totalidad de clientes de la red. De este modo se consigue descargar de parte de la carga de la red a los servidores centrales y mantiene la capa de servicios transparente a los usuarios con la posibilidad de escalar el sistema mejorando o aumentando el número de servidores sin que estos lleguen a notar el cambio en algo más que el rendimiento. Sin embargo, un modelo más distribuido en lo relativo a los clientes obliga a un mayor mantenimiento de los mismos por parte de los administradores. Otro de los puntos a tener en cuenta es la consistencia de los datos entre cliente y servidor, de manera que hace falta coordinar cada servicio por separado.
En esta línea el uso de protocolos de comunicación soporta el uso efectivo por parte de los clientes de los servicios de la red permitiendo la heterogeneidad de los clientes siempre y cuando los implementen.

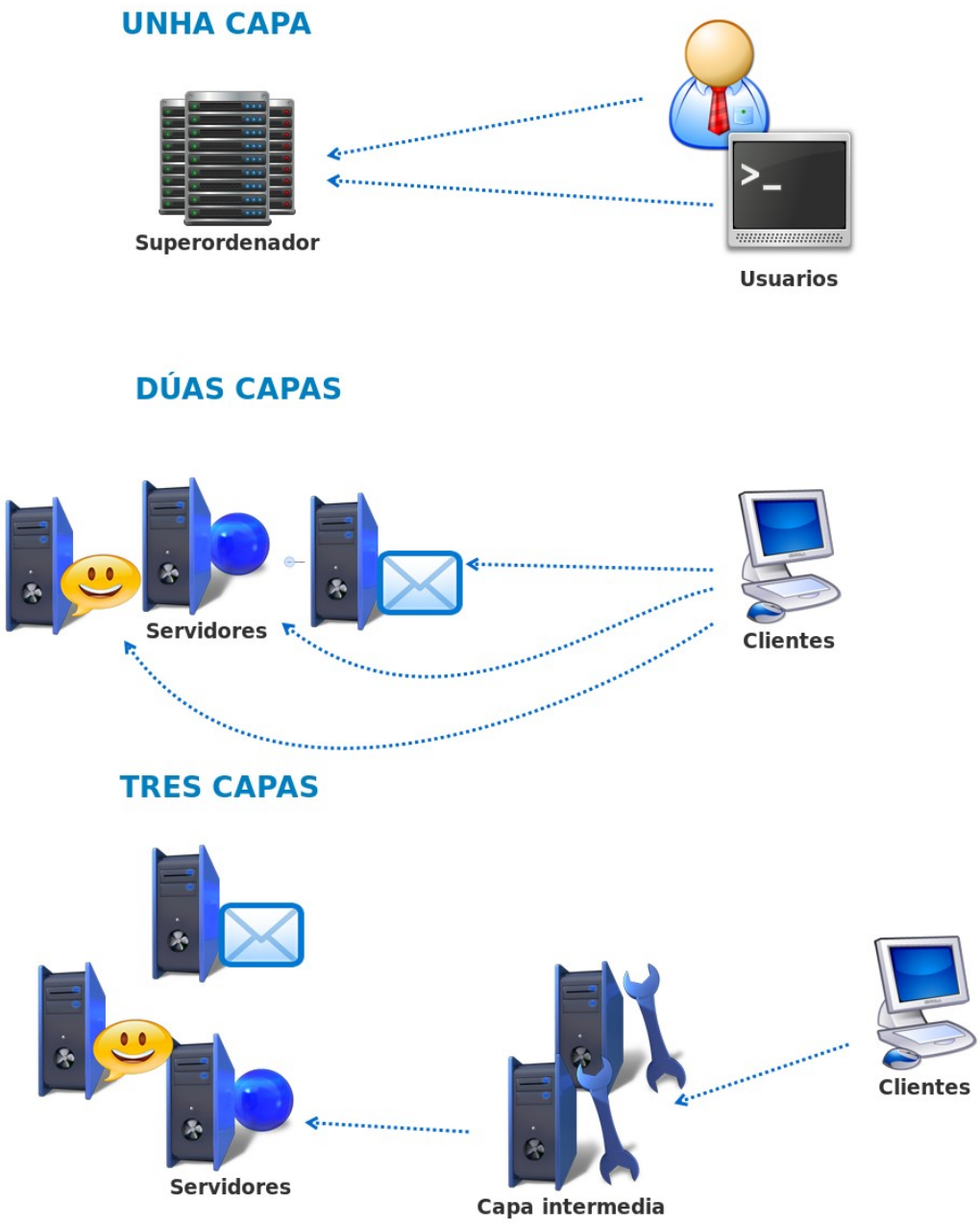


Figura 1: Arquitecturas en capas

28.2.3. Arquitectura en tres capas: Granjas de servidores.

Debido a los inconvenientes de los sistemas de una única capa, que obligan a mantener un servidor central de tamaño demasiado grande para un mantenimiento y rendimiento eficientes, y de dos capas, que obligan a

mantener cada servidor independiente del resto para un único servicio, se optó por el establecimiento de una capa más entre las de cliente y servidor. En esta capa se agrupan varios servidores en una DMZ soportando el mismo servicio dando lugar a una redundancia que tiene como ventajas una mayor tolerancia a fallos y un avance de rendimiento. A efectos de la red las granjas de servidores proporcionan un único servicio lógico o virtual integrado por cualquier número de servidores físicos. Cada servidor de la granja debe ser una réplica exacta del servidor lógico en cuanto a datos y software instalado. Para escalar el sistema se añade a la granja un nuevo servidor réplica del virtual y aumenta la disponibilidad de recursos. El ejemplo más habitual de granja de servidores es un servicio web, donde, si por ejemplo un servidor atiende a mil usuarios y tenemos previstos picos de diez mil usuarios simultáneos, pondremos una granja de diez servidores para atender el servicio y otros dos más en previsión de caída de alguno o picos puntuales todavía más altos. La escalabilidad de las granjas en función de los servicios ofertados define tipologías básicas como *Datacenters*, servidores de aplicaciones, de importación, de *front-end* existiendo elementos específicos para control de carga, Teredo o de dominio, entre otros.

28.2.3.1 Componentes intermedios: *Middleware*

Para dar el efecto de transparencia a los clientes, ese sistema requiere de una serie de componentes intermedios, es decir que se encuentran “por el medio” (en inglés *middleware*) de las capas principales. Estos componentes se encargan de recibir y repartir las peticiones de los clientes entre los servidores de la granja, cuidar el balanceo de carga, el mantenimiento de la sesión, etc. El ***middleware*** se describe como un conductor o intermediario entre sistemas, dirigiendo las peticiones de datos y servicios a otros nodos de la red. Entre sus principales características destacarían:

a) Simplificar el desarrollo de aplicaciones al capsular comunicaciones

entre sistemas.

b) Facilitar la interconexión de los sistemas de información con independencia de la red física.

c) Mejorar la escalabilidad del sistema, aumentando la capacidad sin pérdida de funcionalidad.

d) Mejorar la tolerancia a fallos del sistema, fiabilidad.

e) Aumentar la complejidad de administración y soporte.

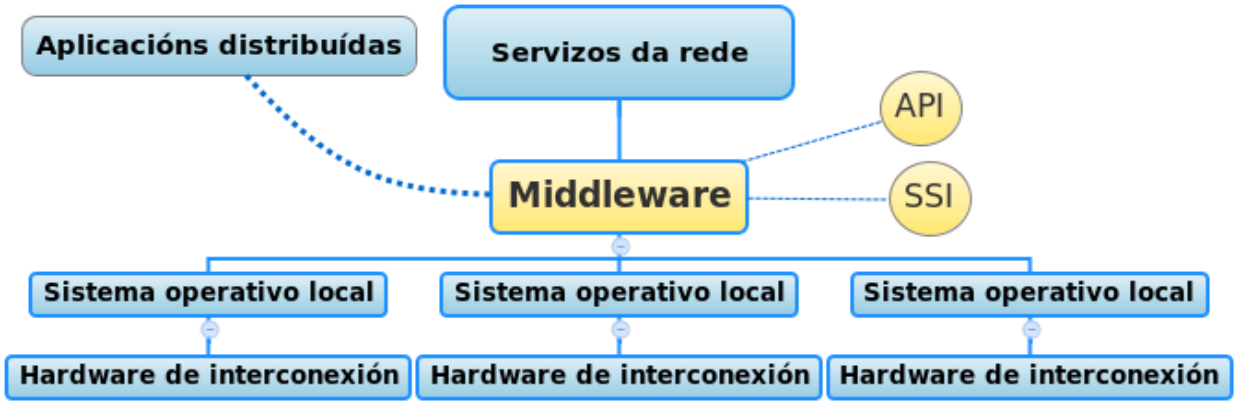


Figura 2: El middleware en un sistema distribuido.

En un sistema distribuido el *middleware* es el software de conectividad que permite disponer de un conjunto de servicios sobre plataformas distribuidas heterogéneas. Actúa como una capa de abstracción de las funciones del sistema distribuido haciendo transparente en la red los sistemas operativos y el hardware de interconexión de las redes de comunicaciones. Proporciona una Interfaz de Programación de Aplicación (**API**) para la comunicación y acceso a aplicaciones y servicios distribuidos. Por otra parte proporciona una interfaz única de acceso al sistema denominada **SSI** (del inglés *Single System Image*), la cual da al cliente la sensación de acceder a un único servidor, el virtual.

Para garantizar la heterogeneidad en la comunicación de los sistemas el *middleware* se estructura en tres **capas o niveles de comunicación** separados:

1) **Protocolo de transporte.** Protocolos de comunicaciones comunes a la capa de transporte de la red, como TCP o UDP. Establecen niveles de seguridad, control de sesiones, etc.
2) **Sistema Operativo en Red o NOS** (en inglés *Network Operating System*). Extensión del sistema operativo de los clientes que captura las peticiones y las dirige hacia el servidor acomodado para devolver a continuación la respuesta del mismo al cliente.
3) **Protocolo de servicio.** Protocolo específico del servicio o aplicación en el sistema Cliente-Servidor.

Los middleware acostumbran a clasificarse según el tipo de comunicación que realizan en el Sistema Operativo en Red y a los parámetros que comunican (infraestructura, acceso a datos, aplicaciones, etc.), los **tipos de middleware** más habituales serían:

1. **Llamadas a procedimientos remotos** (en inglés *Remote Procedure Call* o RPC). Los Clientes invocan directamente procedimientos o funciones de procesos que se ejecutan en servidores remotos, permitiendo distribuir la lógica de la aplicación remota a través de la red. Las llamadas pueden realizarse de manera asíncrona o síncrona. Mantiene al mínimo la información de la sesión y en caso de ruptura de la misma el cliente reinicia la comunicación de cero.
2. **Publicación/suscripción.** Este middleware realiza una monitorización del sistema detectando los servicios y procesos activos. Los componentes registran su interés en determinados eventos, cuando estos eventos son detectados por el monitor envía esa información a los suscriptores. La interacción es asíncrona recayendo por completo en el servicio de notificación/monitorización.
3. **Middleware orientado a mensajes** (en inglés *Message Oriented*

Middleware o MOM). La comunicación se basa en el envío de mensajes asíncronos por parte de los nodos (cliente, servidor, servicio o aplicación). Los mensajes se recogen en colas priorizadas en el nodo destino y se almacenan hasta que pueden responderse. El funcionamiento del sistema es análogo a cómo funciona un servicio de correo electrónico.

4. **Middleware basado en objetos** (en inglés *Object Request Broker* u ORB). Incorpora a RPC los paradigmas de orientación a objetos. Define una arquitectura cliente servidor donde los servicios devuelven objetos, siendo estos la unidad de comunicación. Los nodos piden los objetos por el nombre siendo estos entregados por un servicio de resolución de nombres. Ejemplos de implementaciones de este *middleware* serían: CORBA, RMI, COM, .NET Remoting, etc.
5. **Middleware de acceso a datos** (en inglés *Oriented Data Access Middleware*). Proporcionan la API transparente de acceso a datos agrupando las operaciones de manejo de la conexión con las bases de datos. Ejemplos de este tipo de API serían JDBC y ODBC. Por norma general realizan conexiones síncronas y operaciones transaccionales.
6. **Arquitecturas orientadas a servicios** (en inglés *Service Oriented Architecture* o SOA). Las funcionalidades o procedimientos se publican desde cualquier servidor a modo de servicios. Los servidores publican el servicio y permanecen a la escucha hasta que llega una petición, la procesan y devuelven una respuesta al cliente del servicio. Ejemplos de este middleware son los Servicios Web y los Servicios CORBA.

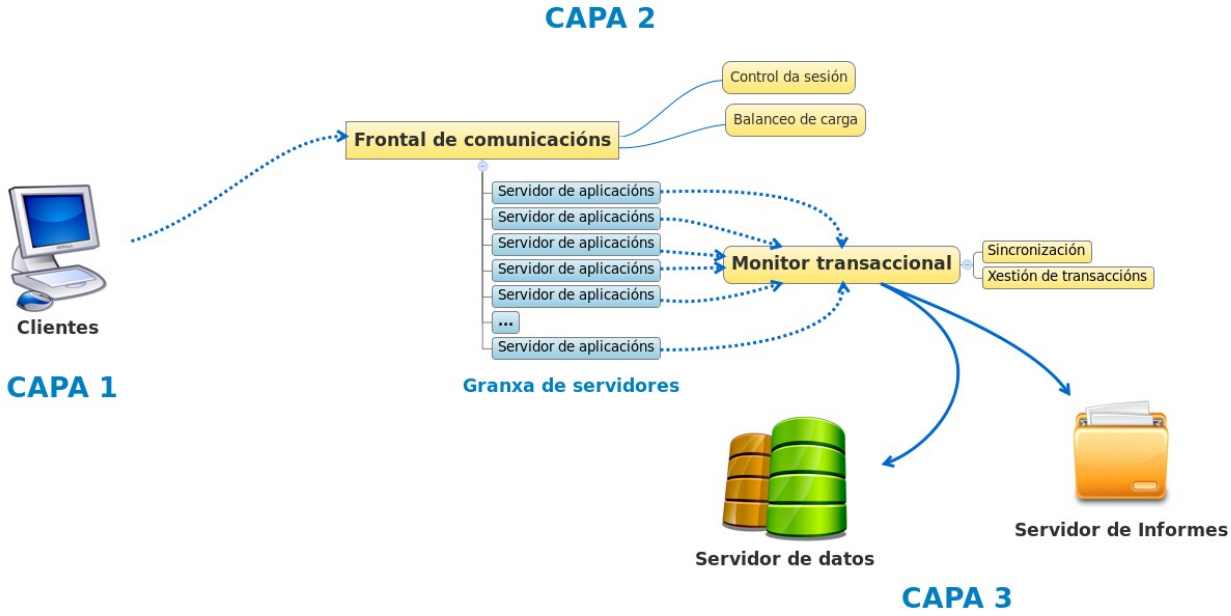


Figura 3: Granja de servidores.

Las granjas de servidores se completan con dos componentes fundamentales, funciones que de manera general realiza el *middleware*:

1. **El frontal de comunicaciones.** El frontal de comunicaciones (en inglés *front-end*) es el punto de acceso único a la granja de servidores, simulando un único servidor lógico. Aunque cada servidor de la granja pueda tener su propia dirección IP lo normal es que el frontal tenga uno propio y sea esta la forma de que los clientes accedan la él, o bien directamente o bien a través de un nombre de dominio (DNS).
   - ✓ **Balanceo de carga.** Consiste en dividir la carga de trabajo de los clientes entre los servidores de la granja. Puede implementarse vía hardware, software o una combinación de ambas. Para hacerlo vía hardware el frontal de comunicaciones debe disponer de un equipo específico, aunque hay enrutadores que permiten esta funcionalidad.

- ✓ **Control de la sesión**. Si a causa del balanceo de la carga diferentes peticiones de un mismo cliente van hacia servidores diferentes, hace falta coordinar a los servidores en el seguimiento de una sesión o que puedan compartirla.
- ✓ **Priorización**. En caso de tener peticiones simultáneas el frontal debe ser capaz de atender primero a los clientes críticos o de mayor prioridad, así como de asignar el procesamiento de sus tareas a aquellos servidores dotados de más recursos.

2. **Los monitores transaccionales.** Los monitores transaccionales son los encargados de mantener las redes de consistencia de los datos y los procesos que se realizan simultáneamente en los servidores de la granja. Tiene que garantizar que una modificación de datos fruto de una petición se realiza como una transacción, es decir, o se realiza completamente o no se realiza en absoluto. Cada transacción tiene que tener lugar independientemente de que tenga lugar otra simultánea, deben procesarse de manera aislada. Las principales funciones serán:
   - ✓ **La gestión de las transacciones.** Se encarga de controlar la atomicidad y secuencialidad de las transacciones, para garantizar la consistencia de datos y operaciones. La gestión de transacciones debe garantizar el correcto funcionamiento del sistema cuando la carga de trabajo o el número de usuarios son muy elevados. Debe contemplar la posibilidad de errores en las aplicaciones y caídas de elementos del sistema durante las transacciones, permitiendo operación de vuelta atrás. Vista en detalle la gestión de transacciones constará de:
     1. **Gestor de transacciones** (en inglés *Transaction Manager*). Controla el inicio de transacción, registra los recursos que precisa y gestiona las operaciones de confirmación de la

transacción (en inglés *commit*) y de vuelta atrás y recuperación del estado inicial de la transacción (en inglés *rollback*).

2. **Gestor de registro** (en inglés *Log Manager*). Guardar los estados de los recursos que están en uso por parte de las transacciones, elaborando un historial de versiones de los mismos. Esta información es compartida por los distintos gestores de transacciones siendo lo que permite garantizar la consistencia de los recursos empleados.
3. **Gestor de bloqueos** (en inglés *Lock Manager*). Gestiona el acceso simultáneo por parte de varios procesos a los recursos, permitiendo bloquearlos para evitar que dos o más accesos a la vez den lugar a inconsistencias. Asimismo lleva a cabo la detección de cuando se libera un recurso y envía una notificación al gestor de transacciones.

- ✓ **Sincronización**. La sincronización de las comunicaciones resulta compleja en este modelo, ya que un cliente pode acceder a un servicio empleando diferentes servidores de la capa intermedia, incluso simultáneamente. Según el comportamiento del servicio podemos encontrar soluciones síncronas, donde se espera siempre a la respuesta del servidor, simple pero con riesgo de bloqueo. Frente a las asíncronas donde se envía la petición y ya llegará la respuesta, con lo cual no hay bloqueos, pero la respuesta podría no llegar nunca sin más opción que detectarlo a través de tiempos de espera agotados.

28.2.6. Arquitecturas en n-Capas.

Las arquitecturas en tres capas pueden extenderse a n-capas cuando en la capa intermedia se incorporan otros elementos de interconexión como distribuidores o sistemas de cortafuego. También puede dividirse la capa de

servidores por servicios o diferentes capas de acceso a datos o presentación de información. La separación en capas es una organización lógica del sistema con la cual puede establecerse cualquier número de capas según las necesidades del mismo. Cualquier especialización de servidores que se quiera hacer en la red y provoque un nuevo agrupamiento podemos identificarla con una nueva capa.

28.2.7. Arquitecturas para Red entre iguales (P2P).

En los modelos distribuidos de igual a igual o P2P (en inglés *Peer-to-Peer*), todos los equipos (excepto los elementos de interconexión) tienen el mismo papel doble de cliente/servidor en la red. Hacen uso de servicios y los proporcionan. En esta arquitectura, por tanto, no se pueden agrupar los nodos y pierde sentido hablar de capas. Estas redes no resultan excelentes para todo tipo de servicios, en muchos casos, por ejemplo a la hora de funcionar cómo servidor web, requerirían un coste muy alto para el control de la consistencia del sitio web, mantenimiento y configuración del servidor, etc. Por el contrario, en situaciones donde los nodos caen a menudo, por ejemplo un servidor atacado continuamente, la redundancia de nodos garantiza que el sistema siga funcionando. Otros servicios como el intercambio de archivos, o el procesamiento compartido presentan más ventajas a la hora de emplear una arquitectura de este tipo como solución de implantación. Los modelos más habituales son centralizados, puros o descentralizados o híbridos, según el peso de cada nodo individual en la red o de la existencia de servidores con responsabilidad de control y gestión en el modelo. La adaptación de este modelo por parte de los ISP da lugar a P2P híbridas de servicio denominadas P4P (en inglés *Proactive network Provider Participation for P2P*). Otros modelos similares serían los P2M, que actúan en arquitecturas híbridas empleando el correo electrónico como soporte del envío de datos.

La gestión de este tipo de redes se realiza fundamentalmente vía software. En este caso el software deberá realizar las mismas funciones ya vistas para la arquitectura de tres o más capas: frontal de comunicaciones y gestión de transacciones. Por debajo acostumbran implementar servidores propios como Kademlia, eDonkey, Gnutella, FastTrack, BitTorrent u OpenNap entre otros.

28.2.8. SERVIDORES.

28.2.8.1. Servidor web.

Se trata de servidores que proporcionan el servicio WWW a través del protocolo HTTP. En esencia se trata de una aplicación ejecutándose en un servidor a la espera de peticiones HTTP por parte de un cliente respondiendo con los documentos solicitados generalmente páginas web y los objetos que enlazan: imágenes, archivos de script , animaciones, etc. En funciones más avanzadas estos servidores añaden seguridad a través de conexiones encriptadas con protocolos tipo HTTP Seguro o HTTPS. Por regla general, los servidores de aplicaciones se integran en arquitecturas de mínimo tres **capas**:

1) **Primera capa.** Capa de interacción con los usuarios, principalmente a través de navegadores web.
2) **Capa intermedia.** Capa de los servidores web, que pueden estar distribuidos en un modelo de granja de servidores. Cada servidor incorporaría los módulos necesarios para seguridad, lenguajes de servidor interpretados, correo, mensajería, acceso a datos y otras funcionalidades.
3) **Tercera capa.** Capa de servidores de acceso a datos, como servidores de archivos, base de datos o informes.

Entre los **servidores web de uso más extendido** actualmente se encontrarían:

- ✓ **Apache**. Uno de los más utilizados, por ser un servidor libre que ofrece prestaciones a nivel de otras soluciones propietarias además de una gran facilidad de uso y configuración.
- ✓ **Internet Information Server (IIS).** Servidor propietario con soporte para aplicaciones .NET o ASP entre otras.
- ✓ **Otros:** Java Web Server, AOLServer, Cherokee, Tomcat, lightHttpd, etc.

Los servidores web pueden disponer de módulos para la ejecución de programas de servidor interpretados, como son los de las tecnologías Python, PHP, ASP, JSP, Tcl,...

28.2.8.2. Servidor de aplicaciones.

Los servidores de aplicaciones son servidores web con capacidad de procesamiento ampliada, pudiendo ejecutar aplicaciones y componentes de lógica de negocio y recursos relacionados como el acceso a datos. Debido a esto permiten realizar el procesamiento de aplicaciones de cliente en el propio servidor. Proporcionan soporte como middleware o software de conectividad y para diferentes tecnologías de servidor. Por regla general, los servidores de aplicaciones se integran en arquitecturas de mínimo tres **capas**:

1) **Primera capa.** Capa de interacción con los usuarios, principalmente a través de navegadores web.
2) **Capa intermedia.** Capa de los servidores de aplicaciones, que pueden estar distribuidos en un modelo de granja de servidores. Un subconjunto de los servidores de aplicaciones darán servicio a los usuarios/clientes mientras otro grupo se encargará de soportar la operativa común del dominio, como librerías o aplicaciones y

servicios web de los que hagan uso las aplicaciones para usuarios/clientes.

3) **Tercera capa.** Capa de servidores de acceso a datos, como servidores de archivos, base de datos o informes.

El servidor de aplicaciones suele tener integrado un servidor web, para gestionar de manera independiente el servicio WWW a través del protocolo HTTP.

Además de este servicio presenta un amplio conjunto de herramientas:

- ✓ Servidor web integrado.
- ✓ Contenedor de programas de servidor (en inglés *servlets*).
- ✓ Contenedores de objetos de lógica de negocio (como por ejemplo EJBs).
- ✓ Sistemas de mensajería.
- ✓ Software de conectividad con bases de datos.
- ✓ Balanceo de carga.
- ✓ Gestión de límites y colas de conexiones (en inglés *Pool*) para bases de datos y objetos.
- ✓ Etc.

En esencia, un servidor de aplicaciones realiza las mismas funciones que un servidor web, pero cuando la demanda de uso es grande y estamos ante un sistema complejo la solución pasa por emplear un servidor de aplicaciones que ofrezca las siguientes **ventajas**:

- ✓ **Centralización**. Centraliza en los servidores la administración y configuración de la lógica de negocio de las aplicaciones, de manera que aspectos como el mantenimiento de los accesos a base de datos pueden realizarse de manera centralizada. Asimismo cambios derivados de actualizaciones, migraciones o recuperaciones ante errores tienen lugar desde uno único punto.

- ✓ **Seguridad.** Al existir un único punto de acceso a datos puede reforzarse la defensa y los sistemas de control de errores en ese punto, mejorando su gestión y protección.
- ✓ **Rendimiento.** Como punto intermedio permite gestionar las peticiones de los clientes a la Base de datos.
- ✓ **Escalabilidad.** Un mismo servidor de aplicaciones puede dar servicio a varios clientes, y por tanto aumentando el número de servidores se mejora el rendimiento del sistema.

Entre los **servidores de uso más extendido** actualmente se encontrarían:

- ✓ **Jboss, Glassfish.** Servidores de aplicaciones libres bajo licencia GPL.
- ✓ **BEA Weblogic, IBM Websphere, Oracle Application Server.** Alternativas propietarias integradas en paquetes de aplicaciones con funcionalidades de gestión y monitorización extendidas.
- ✓ **Tomcat, Internet Information Server (IIS), Jetty**. Proporcionan funciones parciales de servidores de aplicaciones, con lo cual en ocasiones se definen más bien como contenedores de programas de servidor.

28.2.8.3. Servidor de acceso a datos.

Los servidores de acceso a datos ocuparían la última capa de los sistemas de información encargándose del acceso directo a los datos, existiendo diferentes tipos según el sistema de información empleado para su almacenamiento o publicación:

- ✓ **Servidores de archivos.** En este tipo de servidores la información se almacena directamente en archivos, por tanto la función de estos equipos será la de permitir el acceso remoto a los mismos desde los clientes u otros servidores. Los protocolos más habituales ofrecen servicio solo desde redes locales pero en sistemas avanzados pueden

proporcionar servicios como FTP o WebDAV para conexión remota a través de Internet. Actualmente el término empleado para referirse a estos servidores es NAS (en inglés *Network-Attached Storage*), pero ésta tan sólo sería la tecnología más habitual frente a otras cómo DAS (en inglés *Direct Attached Storage*), basada en SCSI o SAN (en inglés *Storage Arena Network*) basada en fibra óptica. No requieren un software muy específico como otros tipos de servidores sino más bien soporte para diferentes protocolos y tecnologías. Por norma general acostumbran a estar dispuestos en RAID (en inglés *Redundant Arrays of Independent Disks*), equipos de almacenamiento redundante.

- ✓ **Servidores de bases de datos.** Albergan uno o más sistemas de gestión de bases de datos (en inglés *database management system*, o DBMS), software de gestión que se encarga de la comunicación entre las aplicaciones y las bases de datos. Permiten realizar operaciones de definición, manipulación y seguridad de los datos a través de una API de comunicación con las aplicaciones y un lenguaje estructurado de consulta como el SQL. Admiten accesos simultáneos a los datos, seguridad y gestión de transacciones.

Estos sistemas suelen presentar además programas o consolas de administración avanzadas para realizar las tareas generales de gestión de la base de datos.

- ✓ **Servidores de informes.** Pueden considerarse una capa intermedia entre los servidores de datos y los de aplicación, donde se establecen servidores o granjas de servidores que sirven los datos en documentos predefinidos multiformato: hojas de cálculo, PDF, XML, HTML, etc. El software de este tipo de servidores suele incorporar software de gestión para el servidor, y software de auto-edición de informes, para definir modelo de informes compuestos de cabeceras, imágenes, fórmulas, subinformes, etc. que generarán dinámicamente los informes a partir de consultas sobre los datos.

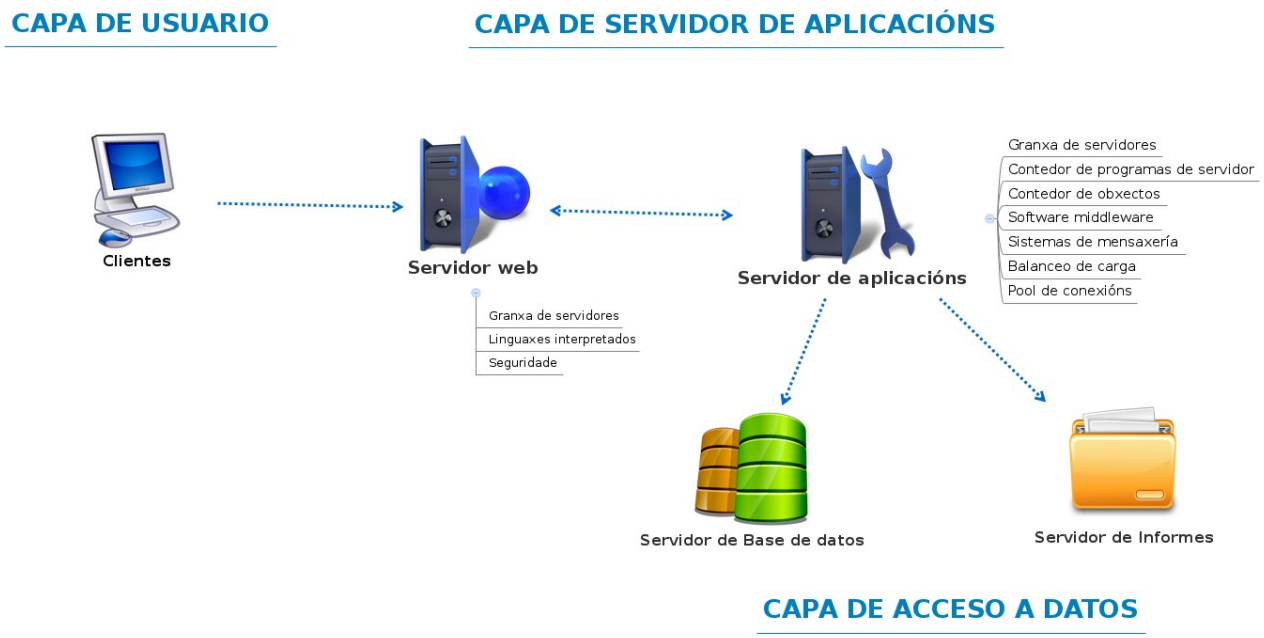


Figura 4: Arquitectura en 3 capas con servidor web, de aplicaciones y base de datos.

Entre los **servidores de uso más extendido** actualmente se encontrarían:

- ✓ **Servidores de archivos.** No requieren gestores especializados pero sí soporte software a los protocolos: CIFS, NFS, SMB, FTP, WebDAV, etc. Así como utilidades tipo Samba o FreeNAS.
- ✓ **Servidores de bases de datos.**
  - ✓ De licencia libre: PostgreSQL, MariaDB, Firebird, SQLite, Apache derby,...
  - ✓ Dual, dependiendo de su uso: MySQL.
  - ✓ Software propietario: SQLServer, Oracle, Access, Paradox, Informix, DBase, etc.
- ✓ **Servidores de informes.** Jasper Reports, Jreports, Crystal Reports, Oracle Reports etc.

28.3 INTEGRACIÓN DE CONTENIDO, SONIDO, IMAGEN Y ANIMACIÓN.

El modelo más habitual de integración de elementos multimedia a través de Internet es a través del servicio WWW, empleando la web. En este servicio los clientes solicitan información a modo de páginas web, tratándose de documentos en lenguajes estándar como HTML o XML, basados en etiquetas que se encargan de estructurar y referenciar los contenidos del documento. De este modo pueden incluirse diferentes tipos de información: texto, hiperenlaces, y elementos multimedia: sonido, imágenes, gráficos, vídeo y animaciones, además de otros formatos o tecnologías que a su vez integran estos elementos.

Para reproducir la mayoría de los formatos básicos de imagen, sonido y vídeo los navegadores suelen disponer de componentes idóneos mientras que para los formatos y tecnologías específicos acostumbran a precisar de *Plug-Ins* o complementos externos que precisan instalación y actualización independiente del navegador. Según esto se distinguirá por tanto dos formas de integración multimedia:

a) **Nativa.** En este caso el elemento multimedia se almacena en un archivo externo de un formato propio del tipo de elemento, si por ejemplo se trata de una imagen en GIF o JPG, y desde el documento HTML o XML se hace referencia al archivo a través del sistema de etiquetado. Casi todos los navegadores actuales, a excepción de los que son en modo texto, reconocen estos formatos básicos con lo cual serán capaces a partir del archivo de reproducir el contenido y transmitirlo de manera correcta al usuario.

b) **Dependiente.** En este otro caso los elementos multimedia emplean tecnologías externas que requieren complementos o *Plug-Ins* externos al navegador que deben instalarse aparte para poder representar la información correctamente. Estas tecnologías hacen las funciones de conectador y especificado su contenido a través del etiquetado HTML o XML son capaces de aparecer como un elemento multimedia básico. Dentro de estos conectadores destacarían los

vídeos y animaciones Flash y Silverlight, programas de cliente como los controles Active X y los applets de Java, y documentos en formatos enriquecidos como el PDF.

Los sistemas de publicación actuales permiten que la **multimedia digital on line** pueda transmitirse **en flujo** (en inglés *streaming*), tanto de sonido (en inglés *Podcast*) como de vídeo y videoconferencia. El mundo del vídeo on line es dentro de la multimedia uno de los que presenta mayores problemas a la hora de trabajar en entorno web, pues precisa más recursos de almacenamiento, ancho de banda para reproducción, problemas de conversión y mantenimiento de formatos de codificación (en inglés *codecs*).

Un último aspecto problemático es la sincronización de todos estos puntos en un proceso automático en un servidor, lo que da lugar a soluciones complejas y de poca sostenibilidad. Las principales tecnologías que dan soporte a este tipo de arquitecturas multimedia son: Windows Media, ASF (en inglés *Advanced Streaming Format*), Quicktime, Real Media, VideoLAN y Flash Video.

Elemento multimedia	Formatos de archivo	Observaciones
Sonido	AAC, MP3, RealAudio, WMA, OGG, MIDI, WAV, AIFF, etc.	*Los cuatro primeros producen pérdida de información en la conversión.*
Imagen	JPEG, GIF, BMP, TIFF, PNG, JPG, TGA, etc.	*El JPEG produce pérdida de información en la conversión.*
Vídeo	AVI, MPG, QuickTime (MOV y QT), WMV, Ogg, RMVB, DIVX, Matroska, etc.	*Tan sólo algunas tecnologías son adecuadas para flujos de vídeo on line.*
Gráficos	PNG, PSD, CDR, XCF, SVG, EPS, etc.	*Muchos pertenecen exclusivamente al programa de edición que los genera, excepto SVG y PNG.*

Animaciones	Flash (SWF), Silverlight (XAML), Javascript (JS), etc.	*Excepto Flash que comprime en un único archivo los demás emplean otras tecnologías de definición abiertas.*
Documentos enriquecidos	RTF, PDF, PostScript, etc.	*Llevan incrustados otros elementos multimedia.*

Tabla 1: Formatos de archivo multimedia.

El servidor de *streaming* permite que se pueda ver parte del video sin descargarlo por completo gracias al uso de un *buffer* que carga parte del archivo previamente. Los formatos de vídeo empleados deben permitir, por lo tanto, estas reproducciones parciales. El flujo busca conseguir lo máximo que permita el ancho de banda y protocolos del sistema, parando la reproducción y esperando a que continúe la carga si la información disponible no es suficiente.

La existencia de un servidor de flujo de vídeo o *streaming*, posibilita los siguientes **servicios**:

- ✓ **Vídeo bajo demanda** o VoD (en inglés *Video on Demand Media Streaming*). En este modelo el vídeo, incluyendo el sonido correspondiente y otros archivos complementarios como subtítulos o textos alternativos para tecnologías asistivas, se encuentra alojado en un servidor específico y los usuarios solicitan el envío de información según lo precisen, en cualquier punto del mismo, con lo que se produce una respuesta personalizada con un flujo parcial a partir de la posición solicitada. Los usuarios pueden realizar diferentes interacciones (simultáneas), yendo adelante, atrás o situarse en cualquier punto del vídeo. Con este sistema siempre se

envía la información almacenada y se hace una precarga de todo el vídeo a partir de la posición solicitada, no siendo necesario disponer del archivo o archivos completos para su visualización.

- ✓ **Vídeo en tiempo real.** (En inglés *Live Media Streaming*). El contenido se crea en el mismo momento de la difusión a modo de las videoconferencias. Se trata de un modelo orientado a la multidifusión de la información, produciéndose un envío del flujo de vídeo a los usuarios una vez que el archivo o parte del mismo pasa a estar creado en el servidor. No tiene porque ser el mismo flujo para todos los clientes, sino que permite flujos paralelos, con la posibilidad de pausas o retrocesos, pero no avances.

En una **arquitectura de streaming**, deberían considerarse cuando menos los siguientes elementos:

- ✓ **Sistemas de edición de vídeo.** Módulos de producción, compresión y conversión de vídeo en formatos aptos para *streaming*.
- ✓ **Sistemas de almacenamiento** que permita alta capacidad de entrega, elevado espacio, tolerancia a errores y sistemas de copias de seguridad. En sistemas con mucha demanda pueden requerir técnicas de tipo:
  - **Diseminación de archivos.** Emplea varios discos diseminando la información para permitir que el servidor acceda a ellos en paralelo.
  - **Almacenamiento terciario jerárquico.** Emplea diferentes soportes almacenando los archivos más demandados en los soportes de mayor rendimiento como discos, y los menos demandados en soportes como cintas, lo cual permite reducir costes.
  - **Técnicas en espejo.** Replica toda la información en diferentes discos separados en espejo, con lo cual se aumenta también la tolerancia a errores.

✓ **Servidor de streaming.** Disponiendo de líneas de alto ancho de banda, con soporte de **Calidad de servicio** o QoS (en inglés *Quality of Service*).

✓ **Clientes.** Con extensiones o complementos que soporten los formatos de la arquitectura. El complemento hará una función doble, gestionar la precarga del vídeo y permitir su reproducción parcial según la información disponible.

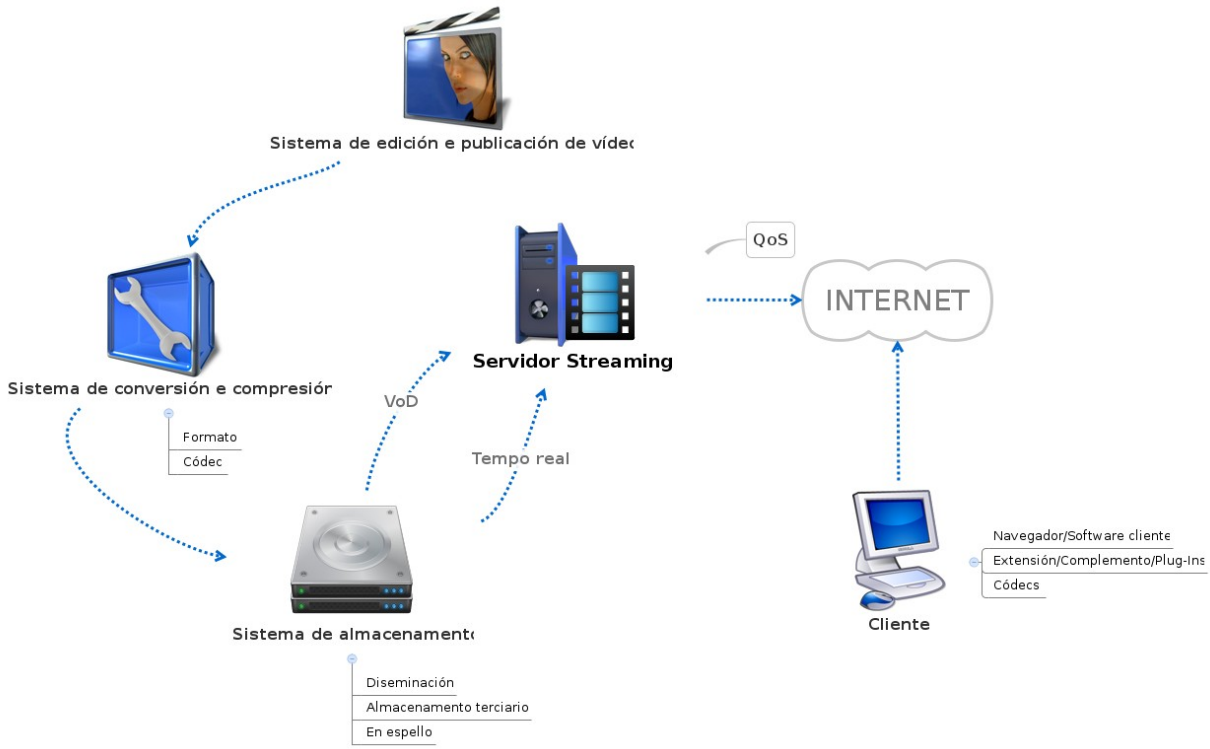


Figura 5: Arquitectura de un servidor Streaming.

Mención especial requieren, respecto del almacenamiento, las **bases de datos multimedia**, son sistemas gestores de bases de datos con orientación a objetos, identificando cada tipo de elemento multimedia con un objeto de la base de datos. Existen varios tipos principales:

1) **Bases de datos referenciales.** Hacen referencia detallada a objetos multimedia, incorporando información descriptiva tanto sobre

el elemento (título, autor, sinopsis, ...) como información técnica (formato, duración, *códec*, ...)

2) **Bases de datos descriptivas.** Incorporan información descriptiva o semántica del contenido del elemento, como puede ser la descripción de un vídeo paso a paso, o el texto alternativo de una imagen. El objetivo de estas soluciones es dar cabida a las búsquedas semánticas y soporte de accesibilidad para tecnologías asistivas.

3) **Elementos multimedia integrados.** Los objetos multimedia se almacenan como campos dentro de la base de datos y no como ficheros externos, por norma general el tamaño de los campos de las bases de datos relacionales se encuentra limitado en comparación con las necesidades de los elementos multimedia, pero el avance de rendimiento y eficacia del sistema pueden hacer necesarias este tipo de soluciones. El ejemplo más habitual serían los bancos de imágenes y algunos tipos de gestores documentales.

Además de la Web **en otros servicios** se puede realizar **integración multimedia** como son los de mensajería instantánea y el correo electrónico. Para el caso del correo, el protocolo **MIME** (en inglés *Multipurpose Internet Mail Exchange*) se desarrolló para permitir la integración de elementos multimedia en los mensajes de correo, con el objetivo de realizar una aproximación al HTML. El funcionamiento básico es asociar cada tipo de elemento multimedia a un tipo MIME (texto, imagen, documento HTML) esta información permite a los navegadores y clientes de correo electrónico determinar con qué tipo de contenido se está trabajando para representarlo correctamente con su complemento o *Plug-In* correspondiente.

28.4 SCRIPTS DEL CLIENTE.

Los *scripts* del cliente son programas interpretados diseñados para ejecutarse en los navegadores con el objetivo de dotar a las páginas de mayor interactividad con el usuario y dinamismo en una aproximación a las aplicaciones de escritorio. El **funcionamiento básico** de un *script* consiste en interpretar una serie de comandos a través de los cuales puede modificar y manipular objetos y reaccionar ante eventos de la interfaz cómo respuestas a periféricos (ratón, teclado, etc.), o cambios en los elementos del documento (botones, elementos de formularios, etc.). Sus usos básicos son validaciones, manipulación de formularios, procesamiento de funciones y carga asíncrona de datos.

Los *scripts* de cliente proporcionan las siguientes **ventajas**:

- ✓ Modificar el contenido de la página sin recargarla del servidor en función de las interacciones con el usuario.
- ✓ Modificar parámetros de configuración del navegador y otros elementos de la página web.
- ✓ Mejorar la interacción entre el usuario y el documento, en general la usabilidad.

Por el contrario, presentan una serie de inconvenientes o desventajas:

- ✓ Problemas de accesibilidad, pues se complica la posibilidad de presentar alternativas a usuarios que no soporten la tecnología de *script*.
- ✓ Problemas de seguridad, pues toda la lógica de interacción aparece sin protección descargada en el equipo del usuario, con lo cual disponen del código fuente del programa de *script*.

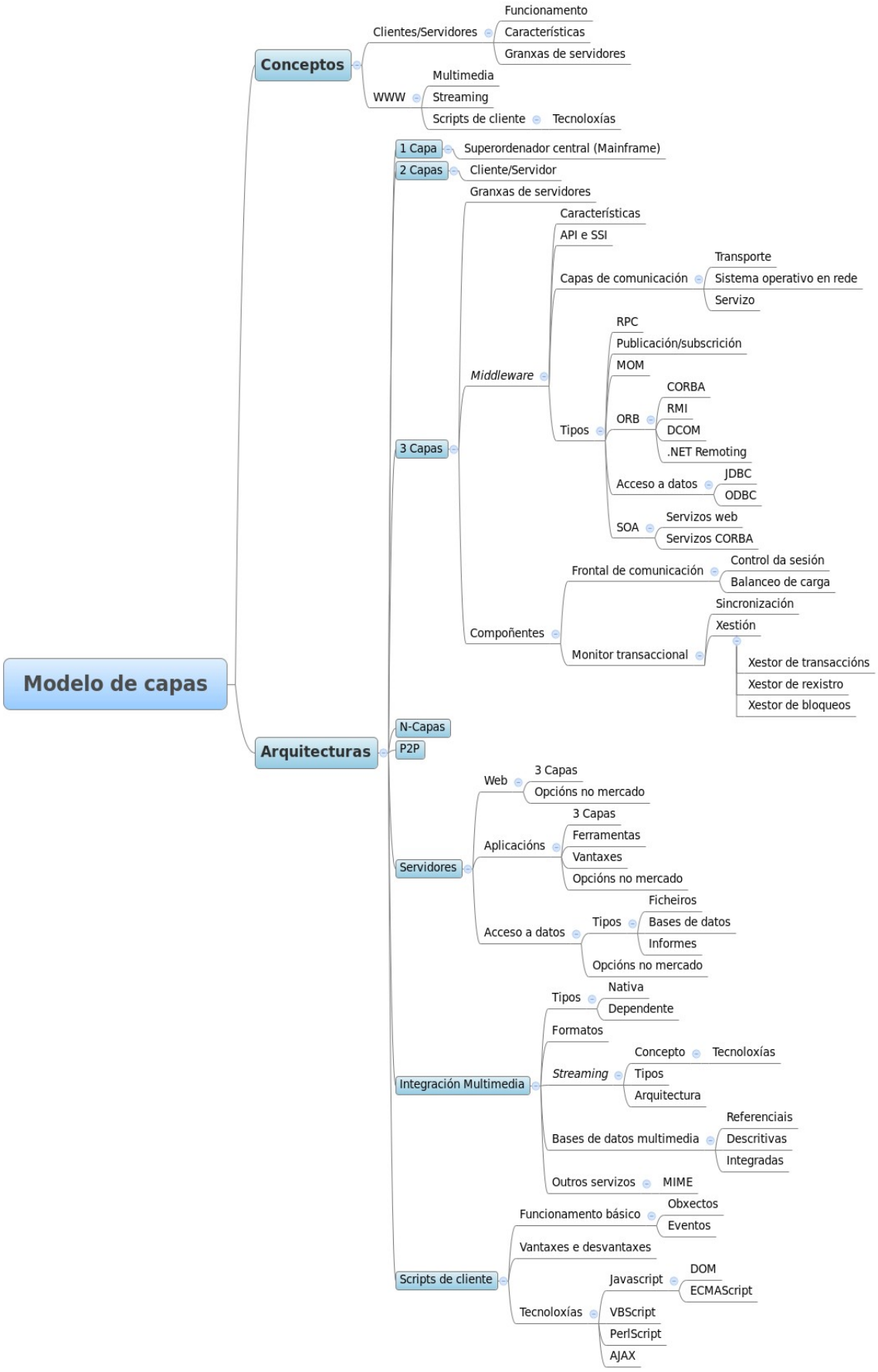
Las **tecnologías de *script*** más empleadas son Visual Basic Script, Javascript, con su evolución AJAX y PerlScript. El principal problema a la hora de seleccionar una tecnología cuando se diseña una página web es el

soporte que recibirá por parte de los navegadores, pues hay que recordar que los lenguajes de *script* serán en última instancia interpretados en el navegador.

a) **Javascript**. Basado en el lenguaje Java, es una de los lenguajes de script de uso más extendido. Hay que señalar, que tiene aplicación con otras tecnologías además de la web como en documentos PDF o aplicaciones de escritorio. Para permitir la interacción con los elementos de un documento web este lenguaje dispone de una API que implementa el DOM (en inglés *Document Object Model*) o Modelo de Objetos para la Representación del Documento, estandarizado por el W3C (en inglés *World Wide Web Consortium*). En un intento por estandarizar este lenguaje surge el ECMAScript una especificación del lenguaje aceptado cómo estándar ESO,
b) **Visual Basic Script.** Similar al Javascript en lo relativo a funcionamiento y estructura pero basado en Visual Basic. Tiene menor soporte dentro de los diferentes navegadores, excepto en el Internet Explorer.
c) **PerlScript**. Basado en lenguaje C su uso no está tan extendido como las tecnologías anteriores aunque tiene menos limitaciones. Debido a esto fue derivando hacia lenguaje de servidor.
d) **AJAX**. Acrónimo de Javascript Asíncrono y XML (en inglés *Asynchronous Javascript And XML*). Esta es una tecnología de *script* de cliente asíncrona, de suerte que puede realizar cargas de datos sin que afecten a la recarga de la página. AJAX es un conjunto de tecnologías que hace uso de:
    1) XHTML y hojas de estilo en cascada (CSS) para la estructura y diseño de los contenidos.
    2) El lenguaje Javascript cómo lenguaje de programación para funciones y definición del programa cliente.

3) El objeto *XMLHttpRequest* para intercambio de información asíncrona con el servidor. Por tanto, hace falta que el navegador soporte este objeto, siendo empleado en ocasiones el objeto *Iframe*.
4) XML y DOM como estándares asociados para intercambio de datos y manipulación del documento.

28.5. ESQUEMA


Modelo de capas
Conceptos
Clientes/Servidores
Funcionamento
Características
Granxas de servidores
WWW
Multimedia
Streaming
Scripts de cliente
Tecnoloxías
Arquitecturas
1 Capa
Superordenador central (Mainframe)
2 Capas
Cliente/Servidor
3 Capas
Granxas de servidores
Middleware
Características
API e SSI
Capas de comunicación
Transporte
Sistema operativo en rede
Servizo
Tipos
RPC
Publicación/subscrición
MOM
ORB
CORBA
RMI
DCOM
.NET Remoting
Acceso a datos
JDBC
ODBC
SOA
Servizos web
Servizos CORBA
Compoñentes
Frontal de comunicación
Control da sesión
Balanceo de carga
Monitor transaccional
Sincronización
Xestión
Xestor de transaccións
Xestor de rexistro
Xestor de bloqueos
N-Capas
P2P
Servidores
Web
3 Capas
Opcións no mercado
Aplicacións
3 Capas
Ferramentas
Vantaxes
Opcións no mercado
Acceso a datos
Tipos
Ficheiros
Bases de datos
Informes
Opcións no mercado
Integración Multimedia
Tipos
Nativa
Dependente
Formatos
Streaming
Concepto
Tecnoloxías
Tipos
Arquitectura
Bases de datos multimedia
Referenciais
Descritivas
Integradas
Outros servizos
MIME
Scripts de cliente
Funcionamento básico
Obxectos
Eventos
Vantaxes e desvantaxes
Tecnoloxías
Javascript
DOM
ECMAScript
VBScript
PerlScript
AJAX

28.6. REFERENCIAS

José Antonio Destrezas.
Mundo IP. Introducción a los secretos de Internet y las redes de datos. (2004).

Andrew S. Tanenbaum.
Redes de computadoras. (2003).

Sergio Luján Mora.
Programación de aplicaciones web: historia, principios básicos y clientes web. (2003).

TRADUCCIÓN DE FIGURAS:
Figura 1.- Arquitectura en capas

- Una capa
- Dos capas

Figura 2.- El Middleware en un sistema distribuido

- Aplicaciones distribuidas
- Servicios de la red

Figura 3.- Granja de Servidores

- Frontal de Comunicaciones

1. Servidor de aplicaciones
2. Gestión de transacciones

- Granja de Servidores

Figura 4.- Arquitectura en 3 capas con servidores web, de aplicaciones y base de datos

- Capa de servidor de aplicaciones

1. Granja de servidores
2. Contenedor de programas de servidor
3. Contenedor de objetos
4. Sistema de mensajería
5. Pool de conexiones

- Servidor Web
    1. Granja de servidores
    2. Lenguajes interpretados
    3. Seguridad

Figura 5.- Arquitectura de un servidor streaming

- Tiempo real
- Sistema de almacenamiento terciario
    1. Almacenamiento terciario
    2. En espejo

Autor: Juan Marcos Filgueira Gomis
Asesor Técnico Consellería de Educación e O. U.
Colegiado del CPEIG