UNIVERSIDAD DE GRANADA

DECSAI

Diseño y Desarrollo de Sistemas de Información

Grado en Ingeniería Informática

Tema 2 – Desarrollo de Sistemas de Información

©I. J. Blanco, F. M. Castro, C. Cruz, C de Mesa, M. J. Martín, J. Paños, D. Sánchez

Departamento de Ciencias de la Computación e Inteligencia Artificial

http://decsai.ugr.es

Contenidos

- El proceso de desarrollo de SI
- Componentes de un SI y su impacto en el desarrollo
  - Agentes externos
  - Datos
  - Software
  - Hardware
- Modelos de Arquitectura para SI
  - Arquitectura centralizada
  - Arquitectura cliente-servidor
  - Arquitectura por niveles
  - Arquitectura orientada a servicios

- Las tareas a realizar para el desarrollo de SI son las mismas que para cualquier otro sistema informático: Planificación, Análisis, Diseño, Implementación, Pruebas, Instalación/Despliegue, Mantenimiento.
- Para organizar y realizar estas tareas puede utilizarse cualquier metodología de desarrollo,

  https://en.wikipedia.org/wiki/Software_development_process
- Se utilizan asimismo las mismas herramientas UML que en cualquier otro proceso de desarrollo de Software (Diagramas de casos de uso, de secuencia, de componentes ...), a las que añadiremos otras.

- En DDSI nos vamos a centrar en particularidades del desarrollo de SI teniendo en cuenta sus componentes:
  - Agentes externos: personas y sistemas
  - Datos
  - Software
  - Hardware
- Particularmente, los datos toman un papel central en el Diseño y Desarrollo de SI

- Los agentes externos son elementos externos al sistema y que interactúan con el mismo. Pueden ser personas u otros sistemas informáticos, incluyendo otros SI.
- La interacción con el sistema de estos agentes consiste en enviar y/o recibir:
  - Datos
  - Eventos indicativos de acciones a realizar por el SI (envían) o que informan del estado del SI (reciben).
- La identificación de agentes externos y sus roles de interacción se debe realizar en la fase de Análisis.

- Los datos son el elemento central de un SI. Cuando hablamos de datos debemos tener en cuenta que estamos hablando de:
  - Aspectos estructurales de los datos (variables, dominios, etc.)
  - Restricciones, que nos indican qué configuraciones de datos almacenados son válidas.
- Ambos aspectos deben recogerse en la fase de Análisis. El Análisis conjunto guiado por las funciones nos permite recoger todos los datos necesarios para el sistema, y ninguno más, de manera sencilla e informal. Lo usamos en prácticas.
- En la fase de diseño se utilizan Modelos de Datos conceptuales y lógicos adecuados al SI para reflejar estructuras y restricciones (diagramas E/R o, equivalentemente, diagramas de clases UML).

- En la fase de Análisis, la descripción de los flujos de datos resulta útil para describir el sistema desde el punto de vista de la adquisición, almacenamiento, procesamiento y publicación de datos.
- Usaremos Diagramas de Flujo de Datos (DFD) o su equivalente Diagramas de Actividades UML (éste mucho más rico semánticamente, y que permite integrar flujos de datos y de control). Vemos y usamos DFDs en las prácticas.
- Permiten interactuar con el cliente y dentro del equipo de desarrollo, así como realizar labores de diseño del sistema centradas en los flujos de información entre componentes funcionales del sistema, base de datos y agentes externos.

- Hay múltiples aspectos a tener en cuenta relacionados con los datos y su ciclo de vida en el desarrollo de SI, que hay que analizar :
  - Adquisición de datos
    - Fuentes de datos: usuarios, sensores, otros SI
    - Métodos de adquisición: entrada directa de datos, procesos ETL, ...
  - Uso de los datos en el SI: flujo de datos, su transformación y almacenado a través del SI, e interacción con el exterior.
  - Archivado de datos: cuándo y cómo eliminar datos del sistema (ya no serán procesados) y su almacenamiento para futuras necesidades.
  - Purgado de datos: cuándo y cómo realizar el borrado de datos archivados.

- Proceso ETL: se usa para adquirir datos mediante integración de datos de distintas fuentes y/o formatos.
  - Extract: de otras BD, ficheros externos (XML, JSON, texto plano), web crawlers, screen scrapping …
  - Transform: limpieza de datos incorrectos o vacíos, transformación de codificaciones, ordenar, mezclar, agregar, etc.
  - Load: comprobación de restricciones semánticas, sobreescritura o almacenamiento histórico, etc.
- Es un proceso complejo, se suele utilizar software específico:

  https://www.softwaretestinghelp.com/best-etl-tools/
- Hoy en día, también ELT (típico en cloud-based data warehouses) o incluso TEL (p. e. transformando los datos para comunicación segura con Blockchain, y después E y L).

- Una transacción es una operación lógica que cambia el contenido de la BD.
- Pueden corresponder con una sola sentencia del DML (ej. una sola sentencia SQL) o varias.
- Hay que determinar si el SI requiere Atomicidad de las transacciones: o bien se ejecutan todas las sentencias correctamente, o no se ejecuta.
- La mayoría de las implementaciones de SQL permiten la definición y manejo de transacciones, ya que el modelo relacional está pensado para garantizar la consistencia e integridad de la BD a través de la Atomicidad.

- Otros aspectos importantes relacionados con los datos a tener en cuenta en el desarrollo:
  - Seguridad de los datos: es necesario garantizar el acceso restringido a los datos de acuerdo con las reglas de negocio de la organización.
  - Recuperación de datos: es necesario diseñar estrategias de recuperación de los datos ante fallos catastróficos, incluyendo copias de seguridad y protocolos de recuperación.
  - Aspectos éticos y legales. El sistema debe garantizar estos aspectos en la adquisición y publicación de datos, particularmente en interfaces de usuario.

- Fundamentalmente dos componentes:
  - Software del SGBD
  - Software de Procesamiento de la Información
- Los datos (estructura y restricciones) junto con los requisitos funcionales y no funcionales del SI, van a determinar el modelo de datos a utilizar (lo veremos en el Tema 4), restringiendo los SGBD que podemos usar.
- Es necesario utilizar un mecanismo de conexión entre el software de procesamiento y el SGBD, que debe estar disponible a través del lenguaje de implementación (lo veremos en el Seminario 1), lo cual repercute en la elección de éste.

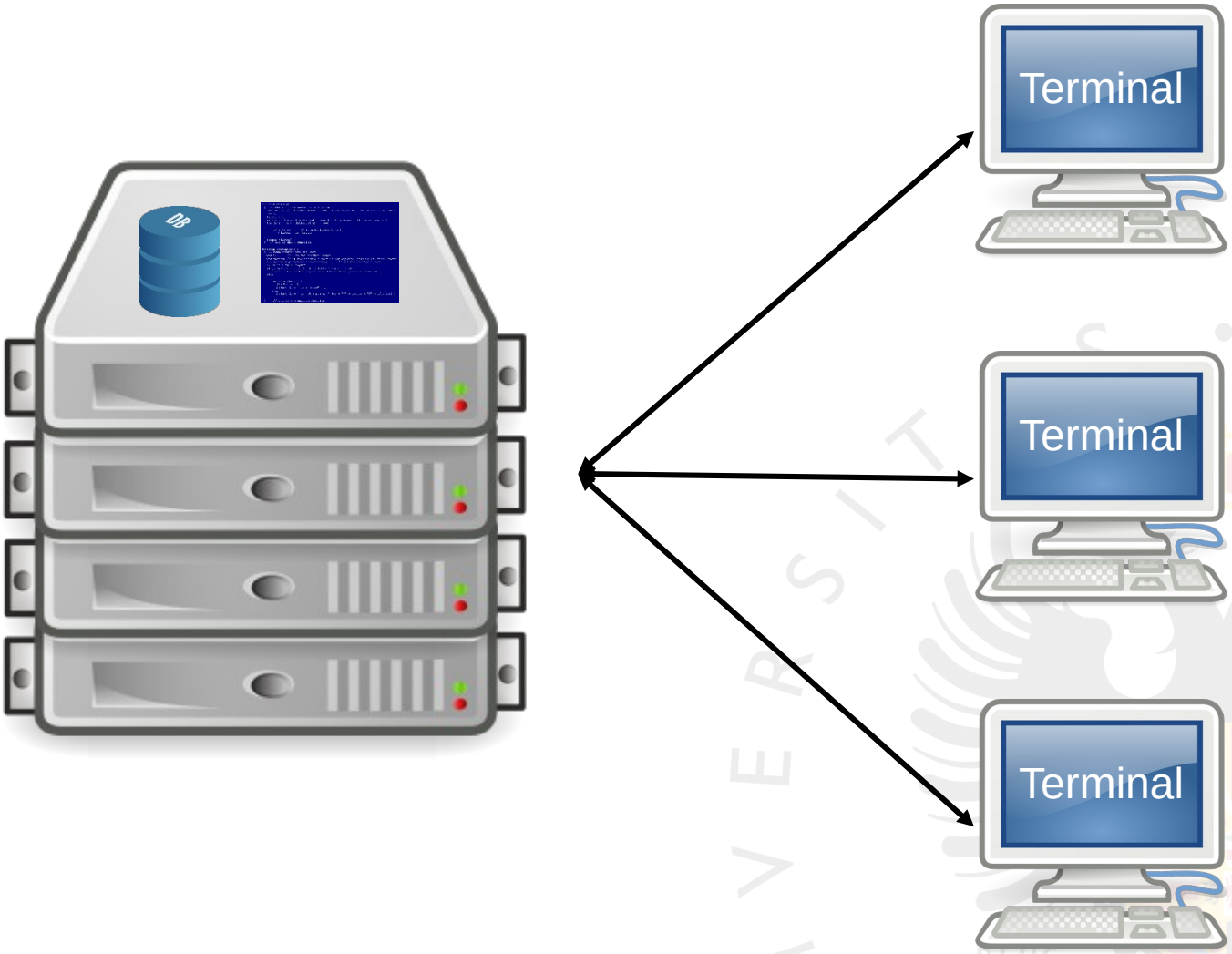
- La implementación del SI requiere la participación y comunicación entre el Administrador de la BD y el equipo de desarrollo.
- El ABD debe, entre otras cosas
  - Aprobar el diseño conceptual de los datos del sistema o proponer los cambios necesarios para su integración en la BD.
  - Colaborar en crear las vistas de usuarios con los permisos de acceso apropiados.
  - Adaptar el diseño físico a los requisitos del sistema.
  - Colaborar en la implementación de la funcionalidad del SI a nivel de BD.

- La mayoría de SGBD actuales permiten la inclusión de código como objetos adicionales en la BD. Para ello proporcionan al menos un lenguaje de programación (lo veremos en la Práctica 3).
- Suele ser código que se activa al realizar operaciones sobre datos, y que sirve para realizar operaciones o comprobaciones que garanticen el cumplimiento de restricciones de integridad.
- ¿Dónde implementar la funcionalidad de mi SI?
  - Si la funcionalidad es inherente a los datos e independiente de la aplicación concreta, puede implementarse en la BD.
  - En otro caso, es mejor en general implementarla externamente a la BD.

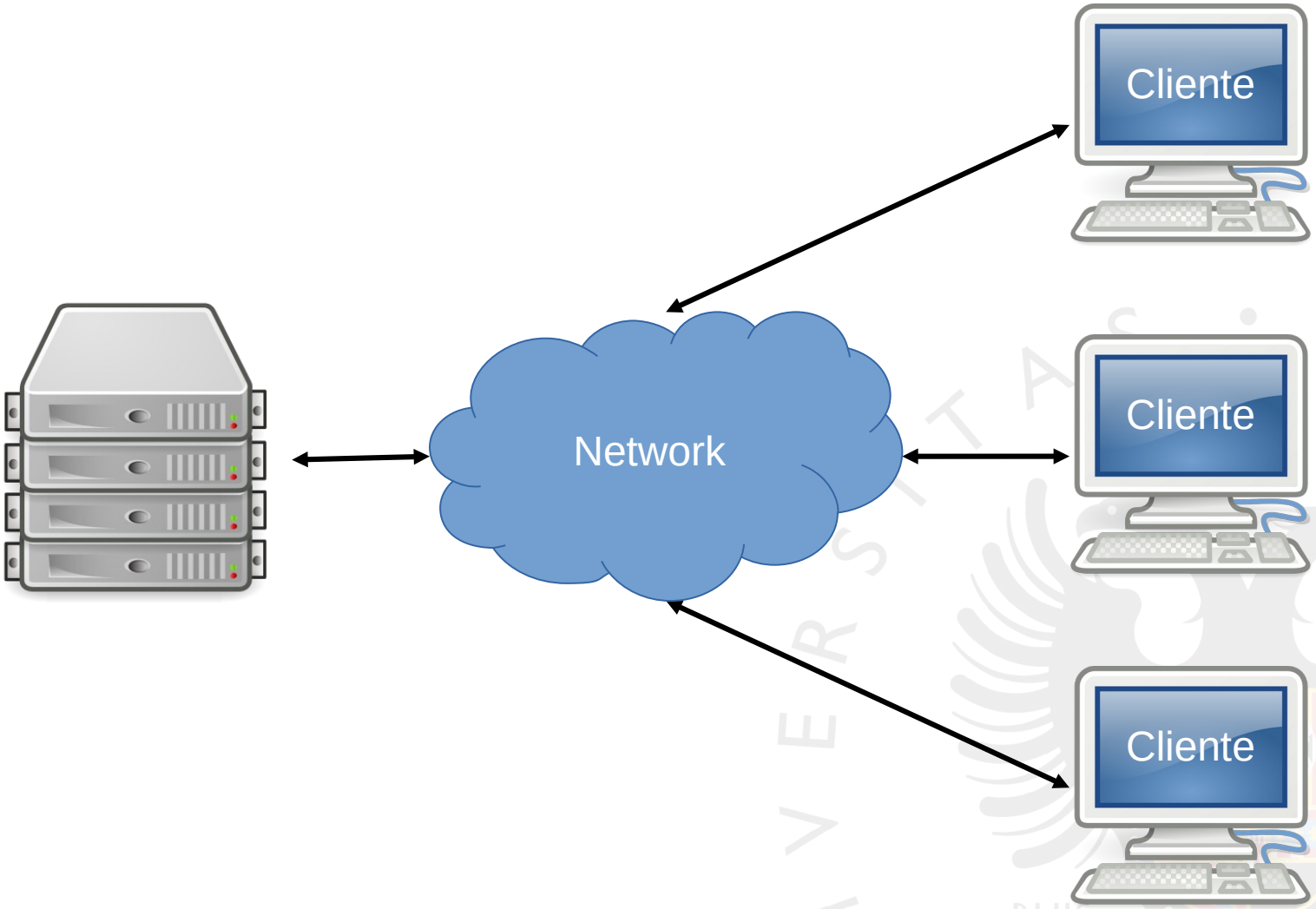
- Pueden usarse lenguajes comunes (C, C++, C#, Java, Python…)
- Existen multitud de herramientas para facilitar el desarrollo de SI (hablaremos de ello en el Seminario 2). Destacan entornos de desarrollo asistido por computador (CASE), IDEs específicos y el uso de lenguajes de cuarta generación.
- Algunas suites de SGBD incluyen estas y otras herramientas. Por ejemplo, ORACLE Developer Suite incluye, entre otros:
  - Oracle FORMS
  - Oracle Designer (herramienta CASE)
  - Oracle JDeveloper (un IDE basado en Java)
  - Oracle Application Developer Framework.
- Sugerencia: buscad información sobre éstas y otras herramientas en suites de SGBD.

- Los SI más simples pueden funcionar en cualquier computadora o LAN.
- Los SI más recientes, especialmente los que tratan mayor cantidad de datos, realizan procesamientos complejos y/o recurren al cloud computing, suelen requerir:
  - Clusters de ordenadores.
  - Redes complejas de comunicación (casi un “quinto componente” de SI), particularmente Internet y la Web mediante capas de abstracción específicas.
- El componente hardware es uno de los elementos más fácilmente distinguibles en las Arquitecturas de SI.

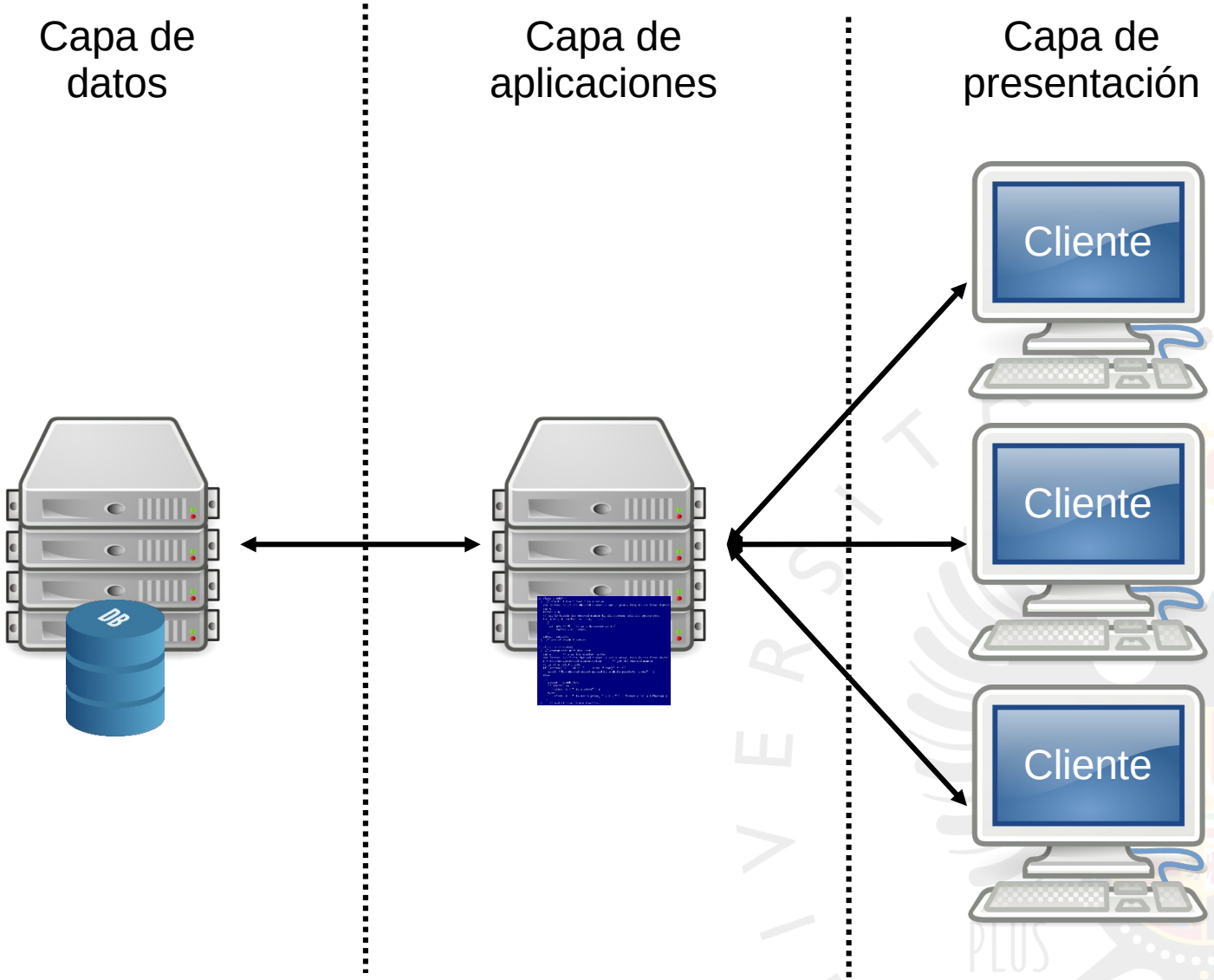
- La arquitectura de un SI es una definición formal de los procesos y reglas de negocio, estructura, infraestructura técnica y tecnologías empleadas en el sistema.
- Incluye un inventario detallado del hardware, software y tecnologías de red disponibles, así como planes detallados a medio y largo plazo para su actualización y reemplazo.
- La arquitectura de un SI suele corresponderse con uno de los cuatro modelos de arquitectura de un SI que veremos a continuación:


DB
Terminal
Terminal
Terminal

- La arquitectura centralizada es una de las arquitecturas más antiguas en los sistemas de información.
- Los datos y las aplicaciones se almacenan en un sistema central.
- No existen ordenadores que se conecten al servidor (al modo de los clientes) sino terminales sin capacidad de cómputo que actúan como periféricos del servidor para proveer de comunicación con el usuario.
- En desuso, aunque podemos encontrar sistemas antiguos funcionando con esta arquitectura.



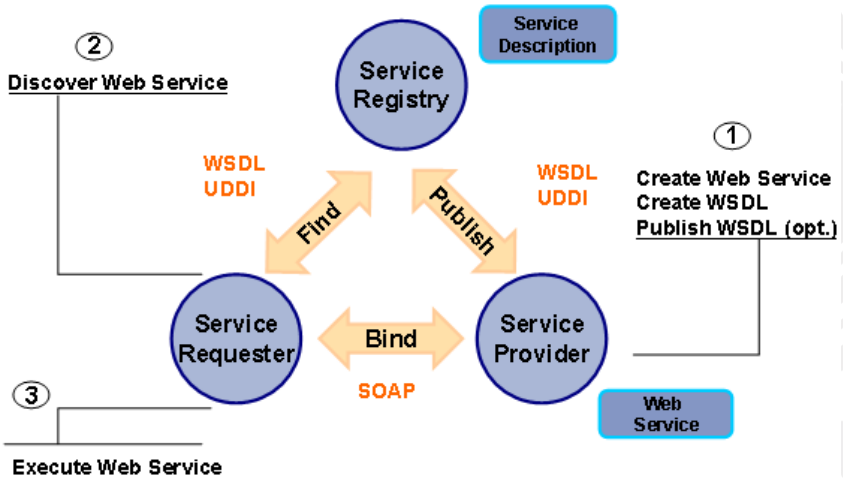
- En la arquitectura cliente-servidor tenemos dos tipos de procesos que suelen ejecutarse en distintas computadoras conectadas por una red de comunicación:
  - Procesos cliente, que realizan peticiones de acceso a la BD.
  - Procesos servidor, que esperan la llegada de peticiones y las satisfacen enviándoles los resultados
- El software del SI se divide entre procesos cliente y procesos servidor, que se reparten las tareas.
- Debe haber un mecanismo de intercambio de mensajes (peticiones y respuestas) entre los elementos de la arquitectura a través de la red.


Capa de datos
Capa de aplicaciones
Capa de presentación
DB
Cliente
Cliente
Cliente

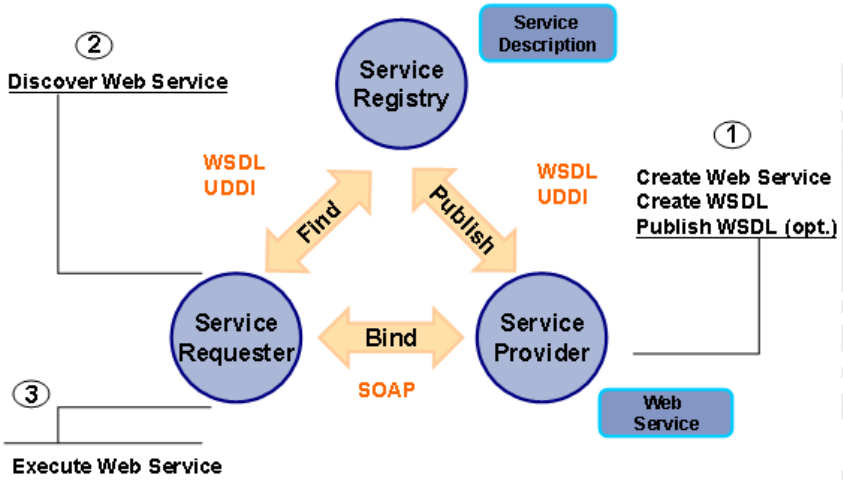
- La arquitectura por niveles es una extensión de la arquitectura cliente/servidor en la que se crea una capa intermedia para la ejecución de la aplicación de gestión de la información.
- Los PCs de la capa de presentación contienen procesos cliente simples cuya labor queda relegada a la de interfaz gráfico para la comunicación con el usuario (ej. navegador web).
- Los procesos de la capa de aplicaciones actúan como servidores frente a la capa de presentación y como clientes de la capa de datos. Puede haber varios PCs en esta capa y en la de datos.
- Los procesos de la capa de datos actúan como servidores de datos, conteniendo exclusivamente el código asociado fuertemente a los datos y común a todas las aplicaciones.

- La arquitectura orientada a servicios es una forma de diseñar e implantar arquitecturas por niveles basada en acoplamiento débil (loose coupling), permitiendo una mejor interoperabilidad entre servicios y clientes implementados en diferentes lenguajes y/o plataformas.
- Se utiliza un conjunto de protocolos y estándares abiertos en la Web para realizar peticiones de servicio e intercambiar datos entre procesos. Proporcionan una capa de abstracción sobre el lenguaje de programación, SGBD y plataforma donde se ejecuta cada servicio.
- Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web.

- Según el W3C, un servicio web es un sistema software diseñado para soportar la interacción máquina-a-máquina, a través de una red, de forma interoperable.
- En la arquitectura de servicios web existen tres partes:
  - Provider: proveedor de servicios Web
  - Requester: solicitante del servicio Web
  - Registry: publicador de servicios disponibles en la Web.



- Los protocolos utilizados en esta arquitectura utilizan como base XML y se transmiten sobre HTTP y otros estándares web:
  - WSDL: Web Services Description Language
  - SOAP: Simple Object Acces Protocol
  - UDDI: Universal Description, Discovery and Integration



La arquitectura centralizada era usada en la época de los mainframes por la falta de PCs o el bajo rendimiento y alto precio de los mismos, aparte de la falta de disponibilidad de redes de comunicación entre computadoras para intercomunicar adecuadamente los PCs.

- La caída del sistema central supone que el SI deja de funcionar para toda la organización.
- Los terminales no poseen capacidad de cálculo, y ofrecen interfaces muy rudimentarios.
- Actualmente en desuso.

La arquitectura cliente-servidor permite distribuir las tareas del SI entre distintas computadoras. En la arquitectura clásica, usualmente el SGBD y la funcionalidad asociada fuertemente a los datos se ubican en una computadora, y el resto de la funcionalidad del SI en otra.

- La arquitectura por niveles permite una mayor escalabilidad y tolerancia a fallos, pudiendo haber diversas computadoras en la capa de datos y de aplicaciones con funciones redundantes y/o distribución y replicación de datos.
- Interfaces más amigables.

La arquitectura basada en servicios permite una mayor interoperabilidad y menor acoplamiento entre los procesos. De esta forma, si se cambia el lenguaje de programación, SBGD o plataforma de uno de los procesos, no es necesario tocar el resto.

- Existen paquetes oficiales o de terceros para la programación de servicios web en los lenguajes más habituales de programación de SI (Java, Python-Django, PHP, .NET ...) y en suites de SGBD (Oracle SOA Suite, etc.).
- Puede ser más lenta que arquitecturas que tengan un mayor acoplamiento.
- Junto con la virtualización de hardware y la disposición y pago del acceso al mismo según las necesidades, forman la base de los SI en la nube, que reducen esfuerzos de administración y el gasto inicial en la implantación.

Resumiendo: ideas importantes

- El desarrollo de SI sigue las mismas fases y utiliza las mismas herramientas y procedimientos que el desarrollo de Software en general, a los que se añaden algunos aspectos particulares.
- Los datos juegan un papel central y requieren tareas específicas relacionadas con su adquisición, almacenamiento, procesamiento, aspectos legales, seguridad, restauración ante fallos, etc.
- Existen herramientas adicionales a utilizar en el desarrollo tales como DFDs y Diagramas E/R, que permiten análisis y diseño desde la perspectiva de los datos.
- Los SI pueden implementarse utilizando distintas arquitecturas, para las que existen gran cantidad de herramientas y lenguajes de desarrollo.