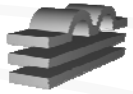
Sistemas de Información


Eduardo Mena

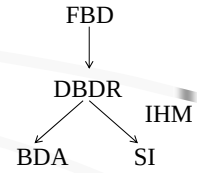
D.0.17, tutorías 13:00-15:00 (M, X, J)

emena@unizar.es

http://www.cps.unizar.es/~mena/

Bloque asignaturas BDs

- Asignaturas
  - Ficheros y BD (troncal)
  - Diseño de BD Relacionales (opt.)
  - BD Avanzadas (opt.)
  - Sistemas de Información (opt.)
  - Interacción Hombre-Máquina (opt.)



- Problemas detectados
  - DBDR debería ser troncal
    - el diseño de BDs es fundamental para un informático
    - proyección laboral
    - Se convertirá en obligatoria de primer ciclo
  - Interacción Hombre-Máquina poco relacionada con BDs

Sist. de Información: Objetivos

- Generalizar conocimientos de BDs
  - Conocer otras alternativas de gestión de datos
- Tipos de sistemas de información más populares
- Líneas de investigación en el área
- Desarrollo de un prototipo en entornos heterogéneos (y distribuidos)
  - Diseñar e implementar un sistema de información sencillo con acceso Web
  - Utilización de distintas tecnologías

Enfoque de la asignatura

- Punto de vista “empresarial”
  - Ejemplos de sist. de información existentes
  - Entornos comerciales (el fin último es hacer negocios)
  - Bibliografía abundante
- Punto de vista tecnológico
  - Generalización de las técnicas de manejo de información. Herramientas informáticas
  - Poca bibliografía. Dispersa y centrada en temas concretos

Temario

- Introducción
- Creación de un Sistema de Información
- Sistemas de Información Web
  - WWW, lenguajes de marcas, programación Web
- Data Warehouses
- Data Mining
- Sistemas de Información Geográfica (GIS)
- Sistemas Legados. Wrappers.
- Sistemas de Información basados en el conocimiento
  - Sistemas expertos, sistemas de agentes (móviles) inteligentes
- Sistemas de Integración de Información
  - Bibliotecas digitales, SBDF, sist. de información globales

Evaluación

- 50%: Examen teórico-práctico
  - No hay que empollar
- 50%: Práctica
  - Elección libre del tema
  - Heterogeneidad
  - Imaginación
  - Autosuficiencia
- Hay que aprobar ambas pruebas

Introducción

Introducción

- En las empresas se ha considerado muy importante gestionar
  - Dinero, materiales y personas
    - desde siempre, de forma progresiva
  - Información
    - últimamente

Introducción: papel de los Sist. de Información

- La gestión de la información es fundamental
  - Informatización cada vez mayor
  - Necesidad de datos actualizados
  - Mantener relaciones (complejas) entre datos
  - Integración de las distintas fuentes de información dentro de una organización
  - Toma de decisiones

¿Que es un Sistema de Información?

- Sistema que incluye todos los recursos dentro de una organización que participan en la recolección, administración, uso y diseminación de la información.
- Integración de hardware, datos, SGBDs, aplicaciones y personal (varios tipos de usuarios)

Los sistemas de información basados en funciones están diseñados para el apoyo exclusivo en un área de aplicación concreta (por ejemplo, contabilidad).

Lo normal es que las empresas tengan varios sistemas de estos, uno por departamento. Problema de redundancia de datos (varios departamentos tienen datos sobre los mismos clientes, pero no necesariamente los mismos datos): problemas de actualización.

Actualmente se tiende hacia sistemas de información integrados (o sistemas de información a secas) donde se realiza un tratamiento más centralizado de los datos sin perder la flexibilidad que necesita el acceso desde distintos departamentos.

Cuestiones éticas y sociales de los sist. de información

- Automatización ➔ menos trabajadores
- También mejora de condiciones laborales:
  - Teletrabajo, oficinas virtuales
  - Eliminación de algunos trabajos repetitivos
- Presión ante el hecho de ser una pieza más del sistema informático
  - Las técnicas de control son enormes
- Derecho a información. Propiedad intelectual. Privacidad de datos

Estamos empezando a ver cada vez más casos relacionados con:

• Utilización personal del sistema de e-mail de la empresa

• Utilización privada de nuestros datos personales

• Copias ilegales.

• Espionaje de las comunicaciones a nivel institucional

¿Donde empieza y acaba lo ilegal? La legislación va siempre muy por detrás de los rápidos avances tecnológicos de impacto social.

A nivel de individuo, muchas veces sufrimos una dependencia de los sistemas de información básicos (bancos, compañías de telefonía, administración).

Por otra parte, la implantación de nueva tecnología muchas veces favorece la comunicación entre individuos: Web, e-mail, chat, comunicaciones inalámbricas, foros virtuales (subastas, debates), cooperación (SETI@Home), etc. La sociedad va encontrando nuevas aplicaciones a la infraestructura tecnológica, algunas buenas, otras no tanto.

Creación de un Sistema de Información

Ciclo de vida de un SI

- Análisis de factibilidad
- Recolección y análisis de requerimientos
- Diseño
- Implementación
- Validación y prueba de aceptación
- Explotación (y mantenimiento)

•Análisis de factibilidad: estudiar áreas de aplicación, costes/beneficios y prioridades.

•Recolección y análisis de requerimientos: necesidades y problemas (interacción con los usuarios).

•Diseño: de los sistemas gestores de datos y de las aplicaciones que los usarán.

•Implementación: introducción de datos, programación de las aplicaciones y pruebas.

•Validación y prueba de aceptación: ¿satisface el sistema las necesidades de los usuarios y los criterios de rendimiento?

•Explotación (y mantenimiento): puede ir precedido de la adaptación de los usuarios al nuevo sistema. La fase operativa comienza cuando todas las funciones están disponibles y han sido validadas. Las tareas de supervisión y mantenimiento son muy importantes durante la fase de explotación.

Cuando surgen nuevos requerimientos o aplicaciones, se pasa por todas las fases anteriores hasta validar e incorporarlas al sistema.

Como cualquier otra nueva implantación (uso de una BD, teletrabajo) requiere una adaptación a la nueva filosofía de trabajo a todos los niveles.

Tecnologías que pueden ayudar

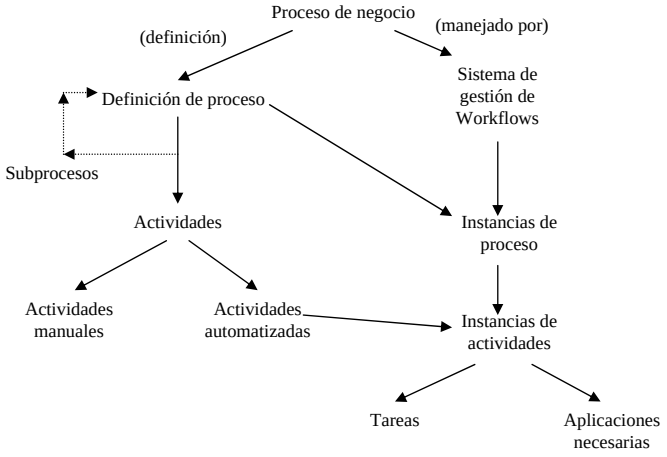
- Teoría de diseño de BDs
  - diseño de los datos
- Teoría de workflows
  - diseño del comportamiento dinámico del sistema
- Técnicas de IA
  - sistemas de ayuda a la toma de decisiones
- Técnicas de BD federadas
  - integración de distintas fuentes de datos
- Uso de la Web
  - punto de entrada a nuestro sistema de información

Diseño de los datos

- El volumen de los datos es enorme y se vuelve intratable (e inútil) con facilidad
- Hay que estructurar los datos
  - Detectar relaciones
  - Optimizar su almacenamiento/procesamiento
  - Documentar
  - Controlar su calidad (consistencia, actualización)
  - Verificar todo lo anterior con cierta periodicidad

Diseño dinámico: Workflows

- *Workflows*: automatización de un proceso de negocio
- Se incluye todo lo relacionado
  - Definición y modelado de procesos asi como de la información intercambiada
  - Sincronización de actividades realizadas por programas o personas
- Objetivo: definir procesos de negocio, razonar sobre ellos y generar implementaciones



Toma de decisiones

- A nivel estratégico, táctico, operativo y personal
- El SI mostrará un nivel de detalle adecuado para cada nivel de decisión
  - Filtrado de información: Dar la información correcta a la persona correcta de la forma correcta y en el momento correcto
- Decisiones programadas y no programadas

Estrategia: qué hacer cuando no hay nada que hacer.

Táctica: qué hacer cuando hay algo que hacer.

• El nivel estratégico se encarga del largo plazo (previsiones, anticipaciones en el mercado, análisis de tendencias).

• A nivel táctico se dan los pasos para conseguir los objetivos establecidos a nivel estratégico (informes de ventas por zonas, análisis de proyectos).

• A nivel operativo se tienen tareas que podrían durar días o meses (informes concretos más detallados, desarrollo de tareas concretas).

• A nivel personal se realizan tareas más o menos repetitivas (atención de pedidos, gestión de reservas).

En todos los niveles hace falta información para tomar decisiones (con más o menos detalles, más o menos globales, más o menos importantes para la productividad de la empresa).

Integración de fuentes de datos

- De una única fuente de datos, a un sistema de información global
- Motivado por la gran explosión de fuentes de datos
  - Los distintos departamentos manejan sus propios sistemas de información
  - Existen fuentes de datos externas interesantes
- Límite: no podemos pretender tenerlo todo en un único depósito de datos

La Web

- Actualmente es la “ventana” por la que nuestro sistema de información es visto por mayor número de usuarios
- Comercio electrónico
- Se pueden heredar ciertos problemas de la Web
  - Heterogeneidad, falta de seguridad
  - Desaparición de muchas “.com”

Objetivo: acceso a datos

- Tener una visión global de las fuentes de información

- Recuperación de datos: 100% preciso
  - Ej.: BDs
- Recuperación de información: se permite una cierta imprecisión

Fuentes de datos: sistema que almacena datos según una cierta organización más o menos compleja (BDs, sistemas de ficheros).

Depósitos de información: sistemas que almacenan información según una cierta organización. Compuestos de una o varias fuentes de datos. No dependen de otros depósitos de información, funcionan de forma independiente.

Algunos autores dan distinta semántica a los términos anteriores.

Información = datos + semántica (o significado)

Ejemplo:

Dato: entero 8

Contexto 1: los años de experiencia laboral del individuo

Contexto 2: sueldo anual en millones de pesetas

Acceso a información

- Entrada
  - Interfaz de usuario
  - API
  - Lenguaje de interrogación
- Salida
  - Método de visualización de resultados
  - Medida de la calidad de la respuesta, si se permite imprecisión
  - Enriquecimiento de respuestas

En muchos SI, las aplicaciones más frecuentes son las consultas. La actualización de datos se realiza desde las mismas fuentes de datos.

La entrada y salida marcan está íntimamente relacionada con la utilización que se dará al SI.

Además de la entrada y salida, también habrá que decidir lo referente al almacenamiento de datos y a su procesamiento, que pueden tener sus requisitos particulares.

Posible clasificación de los Sist. de Información

- Sistemas de procesamiento de datos
  - DP (*Data Processing*)
- Sistemas de información gerencial
  - MIS (*Management Information Systems*)
- Sistemas de ayuda a la toma de decisiones
  - DSS (*Decision Support Systems*)
- Sistemas de información ejecutiva
  - EIS (*Executive Information Systems*)
- Sistemas Expertos

• Sistemas de procesamiento de datos: manejo de transacciones sobre un área concreta, no cubren las necesidades de información no previstas. Manejo de pedidos, abastecimiento, contabilidad.

• Sistemas de información gerencial: utilizan una base de datos integrada y proporciona a los gerentes de los niveles estratégico, táctico y operativo un acceso sencillo a la información necesaria. Ayudan en problemas estructurados (por ejemplo, reabastecimiento de materias primas y cantidades a pedir). Generan informes y salidas de simulaciones con modelos matemáticos, detección de excepciones.

• Sistemas de ayuda a la toma de decisiones: a veces hay que tener en cuenta muchos factores y no sólo se puede confiar en la experiencia. Ayudan a seleccionar entre alternativas (algunos incluso obtienen las alternativas). Para problemas semiestructurados (por ejemplo, mejorar los tiempos de entrega) y no estructurados (por ejemplo, si conviene sustituir una pieza de plástico por otra de metal).

• Sistemas de información ejecutiva: como los anteriores pero exclusivamente para los niveles táctico y estratégico. De gran aceptación pero sin una definición comúnmente aceptada.

• Sistemas expertos: se comporta como un experto: responde preguntas, sugiere soluciones y explica el porqué de sus recomendaciones.

Bibliografía

- L. Long, “Introducción a las computadoras y al procesamiento de información”, cap. 12, Prentice Hall, 1995
- R. McLeod Jr., “Sistemas de Información Gerencial”, Pearson Educación/Prentice Hall, 2000
- K.C. Laudon & J.P. Laudon, “Administración de los Sistemas de Información. Organización y Tecnología”, Prentice Hall, 1996

Data Warehouses

Información en las empresas

- La información proviene de fuentes internas (sistema de producción) y externas (hasta un 20%)
- Problemas
  - Saturación de información
  - Difícil de acceder
  - No selectiva
- La información se necesita para:
  - Competir (comparación con otros productos)
  - Personalizar (simular que cada cliente es única)

Sobre la accesibilidad y selectividad de los datos, algunos estudios muestran que el 27% del tiempo de un directivo se invierte en buscar, acceder y dar formato a la información requerida.

Evolución de la economía:

• Postguerra. Orientada al producto, no hay problema para vender, producción en masa.

• 70's. Mejorar la calidad, aparecen normas y estándares.

• 80's. Entra en juego el factor tiempo, automatización de algunos procesos, plazos límite para cada fase de la producción.

• 90's. Mejora de los servicios al cliente: garantía, soporte, personalización (aumentar los servicios proporcionados, mantener la fidelidad de los clientes). De ahí tanto interés en nuestros datos personales.

Data Warehouse

- Definición: Colección de datos orientados al tema, integrados, no volátiles e historiados, organizados para el apoyo de un proceso de ayuda a la decisión
- Se guarda toda la información útil (proveniente de varias fuentes) en un único lugar

• Orientación al tema: disponer de toda la información sobre un tema (en vez de organizar los datos según los procesos funcionales). La información común a varios temas no debe duplicarse. Los Data Mart apoyan la orientación al tema (son divisiones del Data Warehouse).

• Datos integrados: los datos deben formatearse y unificarse para llegar a un estado coherente (por ejemplo, consolidar todas las informaciones respecto a un cliente).

• Datos historiados: los datos no se actualizan nunca (representan un valor en un momento concreto). Los datos se referencian temporalmente. Esto afecta al gestor de datos (optimizaciones, etc.).

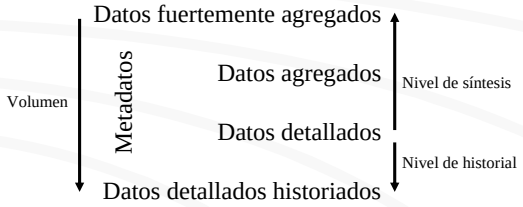
• Datos no volátiles: es una consecuencia de la historilización, la misma consulta formulada meses después sobre el mismo periodo de tiempo dará la misma información. Los sistemas de producción son volátiles.

Data Mart: BD orientada al tema puesta a disposición de los usuarios en un contexto de decisión descentralizado.

Infocentro: similar en su definición a un Data Warehouse pero centrado únicamente en el sistema de producción (sin datos externos ni historial)

Data Warehouse: Estructura

- Varias clases de datos



- Estructura multidimensional

Los datos detallados reflejan los eventos más recientes.

Los datos agregados constituyen el resultado de un análisis (un resumen de los datos detallados). Técnicas de Data Mining.

Los metadatos van orientados a distintos perfiles de usuarios (clientes, equipo de transformación de datos del sistema de producción, equipo de creación de datos agregados, etc.).

Cuantos menos datos mostramos, más usuarios. *Drill-down*: profundización en la estructura multidimensional yendo de lo global hacia el detalle (señal del éxito del Data Warehouse)

Modelos multidimensionales: matrices multidimensionales o cubos de datos (hipercubos si más de tres dimensiones). Ejemplo ejes: ventas, región, trimestre.

Construcción de un Data Warehouse

- Adquisición: Recopilar información de varias fuentes y unificarla
  - Extracción
  - Preparación (formateo, limpieza)
  - Carga
- Almacenamiento: basado en un SGBD
  - El historial influirá en la estructura física
- Acceso: distintos grupos de usuarios requerirán distintas consultas

Respecto al almacenamiento, actualmente sólo un 15% de la información manejada se almacena en formato electrónico (debido principalmente al problema de la conversión a formatos manejados por un ordenador).

OLTP y OLAP

- OLTP (*On-Line Transactionnel Processing*): entorno donde las respuestas se darán en un tiempo aceptable y serán consistentes
  - Transacciones predeterminadas
  - Utiliza pocas tablas
- OLAP (*On-Line Analytical Processing*): entorno de ayuda a la decisión (análisis de datos)
  - Transacciones muy variadas
  - Manejan volúmenes grandes de datos (+tiempo)
  - Se relacionan datos aparentemente sin relación

El SGBD (en realidad, el diseño de la BD) deberá tener en cuenta el tipo de transacciones que se realizarán. En un entorno Data Warehouse hay que potenciar las transacciones OLAP.

Factores de éxito del DW

- Integra datos de producción con datos externos y gestiona historiales
- Contiene la información útil
- Los datos son coherentes, actualizados y documentados (calidad)
- Ofrece acceso directo a los usuarios
- Aumenta el numero de accesos
- Da una flexibilidad que apoya el crecimiento
  - De usuarios, herramientas, y volumen

Errores a evitar

- Cargar datos solamente porque están disponibles (podrían ser no útiles)
- Crear el esquema de la BD de forma tradicional
- Crearlo pensando en la tecnología usada
- Concentrarse en los datos internos
- Creer que los problemas acaban una vez instalado el Data Warehouse

Bibliografía

- J.M. Franco, EDS-Institut Prométhéus, “El Data Warehouse. El Data Mining”, Eyrolles, 1997

Data Mining

Data Mining

- Búsqueda de información relevante (conocimiento) en grandes volúmenes de datos
- Descubrir de forma automática las reglas estadísticas y pautas de un conjunto de datos
- ¿Diferencia con *machine learning*? grandes volúmenes de datos grabados en disco
- Objetivo: obtener un cjto. de reglas

Ejemplo: “las mujeres jóvenes con ingresos anuales mayores de 8 millones de ptas. son las personas con mayores posibilidades de comprar coches deportivos de tamaño pequeño”

Tipos de reglas

- $\forall$X antecedente $\Rightarrow$ consecuente
  - X : lista de una o varias variables con rangos asociados
  - Ej.: $\forall$ transacción T, compra(T, pan) $\Rightarrow$ compra(T, leche)
- Rangos de las variables ➔ población
- Soporte: porcentaje de la población que cumple el antecedente o el consecuente
- Confianza: porcentaje con que el consecuente es cierto al serlo el antecedente

Ej.: $\forall$ transacción T, compra(T, pan) $\Rightarrow$ compra(T, leche)
Confianza 80%: el 80% de las transacciones que compran pan, también compran leche
Soporte 80%: el 80% de la población compra pan o leche

Data Mining Automático

- Descubrimiento automático de reglas
- Técnicas de *machine learning*, adaptadas para grandes volúmenes de datos
- Tres tipos de problemas:
  - Clasificación: reglas que dividan en grupos
  - Asociaciones: X ➔ Y
  - Correlaciones entre series

Ej. clasificación: agrupar los clientes de un banco en excelentes, buenos, medios y malos basándose en su historial de operaciones

Ej. asociación: “La mayoría de los que compran pan también compran leche”.

Ej. correlación entre series: “Siempre que el interés de los bonos del tesoro suben, el índice del mercado continuo baja antes de dos días”

Existen productos comerciales que usan distintas técnicas para extraer conocimiento. Utilizan acceso ODBC para acceder a los datos.

Clasificación

- Comienza con una muestra de datos de clasificación conocida
- Los datos se dividen según uno de sus atributos, sucesivamente
  - Atributos enumerados -> un cjto. por valor
  - Atributos con rango numérico -> intervalos
- Resultado: arbol de clasificación (taxonomía)
- Hasta usar todos los atributos o clasificar correctamente los datos

Taxonomía: del griego, ordenación de nombres

Ejemplos:

- Taxonomía de animales
- Taxonomía de productos de un supermercado

Asociaciones

- Se genera un mapa de bits para cada transacción (un bit para cada artículo a estudiar)
- Nos quedamos con los artículos más adquiridos
- Se generan todos los subcjtos. posibles de artículos y se cuenta el número de transacciones
- Los subcjtos. con un número alto de transacciones generan las reglas

Transacciones/productos	P1	P2	P3	P4	P5	P6	P7	P8	...
T1	X				X			X	
T2		X	X		X	X	X	X	
T3	X	X		X		X			
T4		X			X	X			
T5	X		X		X				
T6	X	X			X		X		
T7	X	X			X			X	
...									

Más adquiridos: P1, P2, P5

P1+P2: 3 (T3, T6, T7)

P1+P5: 4 (T1, T5, T6, T7) ¿ P1 → P5 ? ¿ P5 → P1 ?

P2+P5: 4 (T2, T4, T6, T7) ¿ P2 → P5 ? ¿ P5 → P2 ? ¿ P5 → P1 + P2 ?

Calcular el soporte y confianza de las reglas candidatas

Ejemplo: hacer ejercicio para alumnos/asignaturas optativas, y acabar calculando soportes

Data Mining guiado por el usuario

- El usuario plantea hipótesis
- El sistema comprueba si se verifica o no
- Las hipótesis se pueden ir refinando
- La visualización gráfica de datos ayuda al usuario a examinar grandes volúmenes de datos

Bibliografía

- J.M. Franco, EDS-Institut Prométhéus, “El Data Warehouse. El Data Mining”, Eyrolles, 1997
- R.A. Elmasri, S.B. Navathe, “Fundamentos de Sistemas de Bases de Datos, 3ª ed.”, Addison-Wesley, 2000

Sistemas de Información Geográfica

Sist. de Inf. Geográfica (GIS)

- Sistemas que relacionan, almacenan, manipulan y visualizan información referenciada geográficamente
- Sist. de Inf. que manejan datos espaciales
  - Algunos datos son referencias espaciales o coordenadas geográficas
  - Poseen operadores para manejar dichos datos espaciales

Información espacial

- Multidimensional (x,y,z,t,…)
- Voluminosa
- Naturaleza inexacta (no hay representaciones exactas de la Tierra)
- Las preguntas combinan topología, geografía, y otros atributos, con información aproximada
- Combina distintos contextos legales y económicos (varían de un país a otro)

Mapas

- Estructuración
  - Vectores (+versátil, -fácil de crear)
    - Creados con paneles digitalizadores
  - *Rasters* (-versátil, +fácil de crear)
    - Cada celda almacena el tipo de terreno
    - Creados mediante *scanning*
  - Se puede pasar de un formato a otro (con un cierto error)
    - Reconocimiento de formas en un raster → vectores
    - Pixelización de vectores → raster

Además no olvidemos que existirán otros muchos datos (de tipo textual, por ejemplo) asociados a cada elemento del mapa (renta per capita, población, etc.): son los atributos de los elementos geográficos.

Preguntas a los GIS

- ¿Qué hay en cierta posición?
- ¿Dónde hay cierto elemento?
- Muestra zonas que cuyos atributos cumplen ciertas condiciones
- Generación de nuevos gráficos
  - mapas de elevación, densidad de población, etc.

Utilidad de los GIS

- Generación de mapas
- Selección de lugares
- Creación de planes de emergencia
  - Ante terremotos u otras catastrofes
- Simulación de transformaciones medioambientales
  - Cambio en paisajes ante túneles, obras, urbanizaciones, etc.

Bibliografía

- Keith C. Clarke, “Getting Started with GIS”, Prentice-Hall, 1997, ISBN 0-13-294786-2
- http://www.usgs.gov/research/gis/title.html

Sistemas de Información Web

Sistemas de Información Web

- WWW
- Lenguajes de marcas
  - SGML
  - HTML
  - XML
- Programación Web
  - CGI
  - Applets, Javascript
  - Servlets
  - ASP, JSP

World Wide Web (WWW)

- Tim Berners-Lee
  - 1989: propuesta inicial
  - 1991: prototipo 1991
  - 1994: W3C
- HTTP (HyperText Transfer Protocol)
- Servidor Web
  - Servidores de ficheros
- URL = protocolo://nodo:puerto/path
- WWW ≠ Internet
- Gran éxito: Muy fácil de utilizar y crear,…y es “gratis”
- Previamente: FTP, WAIS, Gopher, Archie

Uso de la Web en pocas palabras:

• El usuario especifica una URL a su navegador, la cual identificará unívocamente un fichero dentro de toda la Web

• Tras analizar la URL suministrada, el navegador se conecta al servidor Web correspondiente (en el ordenador y puerto indicado por la URL), y le indica el fichero que desea obtener

• El servidor Web accede al fichero y se lo envía entero al navegador, indicando de qué tipo de fichero se trata

• El navegador interpreta el fichero recibido adecuadamente (depende del tipo de fichero), recurriendo a programas externos si fuera necesario

Las URLs también pueden referenciar ejecutables:

protocolo://nodo:puerto/path?param1=val1&param2=val2&...

La Web es actualmente la mayor fuente de información sobre prácticamente todos los temas. Ello es debido a la facilidad de crecimiento que tiene y a la incorporación de todo tipo de usuarios como creadores de páginas Web.

En algunos aspectos es un reflejo de nuestra sociedad (de lo bueno y de lo malo). Ha demostrado a donde puede llegar el poder de la información (cómo hacer una bomba atómica, listas de pederastas, libros on-line, banca electrónica, mejores relaciones sociales, peores relaciones sociales, etc.). Aún no estamos acostumbrados a poder acceder a tanta información.

Navegadores Web

- Netscape, Microsoft Internet Explorer
- Reciben URLs y muestran páginas (tras interpretarlas adecuadamente)
- MIME types
  - En el servidor y en el navegador
  - Plug-in’s y visualizadores externos
- Control de seguridad

MIME type	Extensión
application/postscript	ai eps ps
application/x-shockwave-flash	swf
application/x-stuffit	sit
audio/basic	au snd
audio/midi	mid midi kar
audio/mpeg	mpga mp2 mp3
image/jpeg	jpeg jpg jpe
image/tiff	tiff tif
text/html	html htm
text/plain	asc txt
text/xml	xml
video/mpeg	mpeg mpg mpe
video/quicktime	qt mov

Instalar un servidor Web

- Puerto de escucha (por defecto, 80)
- Definición del *DocumentRoot*
- *UserDir* (dónde poner las páginas)
- *ScriptAlias* (dónde poner los programas)
- Colocación de páginas y *scripts* (con los permisos adecuados)
- Muy importante la seguridad (permisos, logs)

Uno de los mejores servidores web: Apache (http://www.apache.org)

Problemas de la Web

- Falta de semántica (búsquedas poco eficientes)
- Cambios constantes (enlaces obsoletos)
- Falta de seguridad
- Gran velocidad de crecimiento

- Pero…está cambiando la sociedad (desde el punto de vista del acceso a información)

Lenguajes de marcas

- Texto con marcas o etiquetas de comienzo y final
- Lenguajes de marcas más conocidos:
  - SGML
  - HTML
  - XML

Para móviles, WML (Wireless Markup Language).

SGML

- *Standard Generalized Markup Language*
- Metalenguaje: reglas para definir un lenguaje de marcas
- Estructura
  - Estructura del documento
    - DTD (*Document Type Definition*). Sintaxis, no semántica
  - Contenido (con etiquetas)
- No se indica cómo visualizar el documento

Existen algunos estándares para especificar cómo visualizar documentos SGML, como DSSSL (Document Style Semantic Specification Language) y FOSI (Formatted Output Specification Instance).

SGML DTD

- Elementos estructurados en árbol
- Hijos obligatorios (-) y optativos (O)
  - Concatenación (,), OR (|), cero o una ocurrencia (?), cero o más ocurrencias (*), una o más ocurrencias (+)
- Contenido de una etiqueta
  - Otras etiquetas
  - Texto ASCII (PCDATA)
  - Datos binarios (NDATA)
  - EMPTY

```
<!-- SGML DTD para e-mail -->
<!ELEMENT e-mail        - - (cabecera, contenido) >
<!ELEMENT cabecera      - - (enviado-por, destino+, tema?) >
<!ELEMENT (enviado-por | destino | tema) - O (#PCDATA) >
<!ELEMENT contenido     - - (texto | imagen | sonido)* >
<!ELEMENT texto         - O (ref | #PCDATA)+ >
<!ELEMENT ref           - O EMPTY >
<!ELEMENT (imagen | sonido) - - (#NDATA) >

<!ATTLIST e-mail
            id                      ID          #REQUIRED
            fecha-envio DATE        #REQUIRED
            estado      (secreto | publico)   publico >

<!ATTLIST ref
            id                      IDREF       #REQUIRED >

<!ATTLIST (imagen |  sonido)
            id                      ID          #REQUIRED >
```


Marcas SGML

- <marca> … </marca>
  - Ej: <autor> Antonio Castro </autor>
- Pueden tener atributos:

  <marca atr=valor … > … </marca>
  - Ej: <autor nacido_en=Zaragoza>
- Comentarios: <!-- texto -->

```
<!-- Ejemplo de utilizacion del DTD anterior -->
<!DOCTYPE e-mail SYSTEM “e-mail.dtd”>
<e-mail id=18 fecha-envio=02102001>
  <cabecera>
    <enviado-por> Pablo Neruda </enviado-por>
    <destino> Federico Garcia Lorca </destino>
    <destino> Ernest Hemingway </destino>
    <tema> Fotos de mi casa en Isla Negra
  </cabecera>
  <contenido>
    <texto>
      Como prometi en mi carta anterior, os envio dos
      fotos digitalizadas para mostraros mi casa y la
      esplendida vista del oceano Pacifico desde mi
      dormitorio (foto <ref idref=F2).
    </texto>
    <imagen id=F1> “photo1.gif” </imagen>
    <imagen id=F2> “photo2.gif” </imagen>
    <texto>
      Un saludo desde el sur, Pablo.
  </contenido>
</e-mail>
```

HTML

- *HyperText Markup Language* (1992)
- Es una instancia de SGML (existe un HTML DTD), un subconjunto de SGML
- Definición de documentos multimedia y su visualización
- Los documentos pueden incluir "programas": DHTML (Dynamic HTML)
- Metainformación

Ejemplo:

```
<html>
<head>
<title> ejemplo HTML </title>
<meta name=xx content="un ejemplo">
</head>
<body>
<h1>Ejemplo       HTML</h1>
<p> <hr> <p> HTML tiene muchas
<i>etiquetas</i>, por ejemplo:
<ul>
<li> enlaces a <a href="http://www.cps.unizar.es/"> otras paginas </a>
<li> imágenes <img src="foto.gif" align=middle>
</ul> <p> <hr> <p> <img
src="at-work.gif"> Esta pagina esta en construccion
</body>
</html>
```


Principales etiquetas HTML

- Fuentes de texto: tamaño, estilo
- Separadores
- Listas: ordenadas y no ordenadas
- Imágenes
- Enlaces: internos y externos
- Tablas
- Acentos
- Otros: formularios, *frames*, *applets*, etc.

Fuentes de texto:
Tamaño: <H1>, <H2>, …
Estilo: itálica <I>, negrita <B>

Separadores: Nueva línea
, nuevo párrafo <P>, línea de separación <HR>

Listas: ordenadas <OL>, no ordenadas <UL> (items <LI>)

Imágenes: <IMG SRC="xyz.gif">

Enlaces:
Internos <A HREF="#etiqueta"> (referencia), <A NAME="etiqueta"> (referenciado)
Externos: <A HREF="URL"> (URL del mismo o distinto servidor Web)

Tablas: <TABLE>, <TH>, <TD>, <TR>

Acentos: Ramón Pérez (Ramón Pérez)

Limitaciones de HTML

- No permite a los usuarios crear sus propias etiquetas o atributos
- No soporta estructuras anidadas para representar datos estructurados
- No tiene mecanismos de verificación de los datos que contiene

XML

- *eXtensible Markup Language*
- Metalenguaje (subcjto. de SGML) que permite definir lenguajes de marcas
- Permite hacer validaciones que antes se hacia con scripts embebidos en HTML
- No tiene las restricciones de HTML pero impone una sintaxis más rígida
- No es obligatorio usar un DTD
  - Se registran las etiquetas según van apareciendo

XML es menos permisivo que HTML respecto a la sintáxis. Por ejemplo:

• Hay que cerrar todas las etiquetas. Se indica especialmente las etiquetas que no tienen contenido (ej:
)

•Distingue mayúsculas y minúsculas (al contrario que HTML)

•Los valores de los atributos van siempre entre comillas

Familia de estándares XML

- XML Namespaces (evitar colisión de nombres)
- XPath (XML Path Language)
- XQL (XML Query Language)
- XSLT (XML Stylesheet Language Transformation)
- XLink (XML Linking Language)
- XPointer (XML Pointer Language)
- XML Schema (más general que los DTDs)
- SAX (Simple API for XML)
- DOM (Document Object Model)

XML Namespaces

- Para combinar documentos (o fragmentos) de diferentes dominios
- Evita la colisión de nombres
- prefijo:etiqueta
- Atributo xmlns:prefijo=“cadena URI”. El atributo xmlns se puede incluir dentro de cualquier elemento.

Problema:

```
<libro>                                  <autor>
   <titulo> … </titulo>                     <titulo> … </titulo>
   <autor> … </autor>            +          <nombre> … </nombre>
   <editorial> … </editorial>               <nacionalidad> … <nacionalidad>
</libro>                                 </autor>
```

Colisionan el título del libro con el título (académico) del autor del libro.

Ejemplo con Namespaces:

```
<obra:libro xmlns:obra=“http://www.xyz.com”>
   <obra:titulo> … </obra:titulo>
   <persona:autor xmlns:persona=“http://www.abc.com”>
      <persona:titulo> … </persona:titulo>
      <persona:nombre> … </persona.nombre>
      <persona:nacionalidad> … </persona:nacionalidad>
   </persona:autor>
   <obra:editorial> … </obra:editorial>
</obra:libro>
```

XPath

- Para obtener valores de etiquetas imponiendo filtros o condiciones. Es un lenguaje de interrogación de XML
- Se navega por las etiquetas de un documento según la estructura jerárquica de XML
- Devuelve un cjto. de nodos referenciados

Ejemplo basado en el documento XML anterior:

Pregunta: ¿Cual es la nacionalidad de los autores que han escrito para la editorial “Prentice-Hall” ?

Expresada en XPath:

/obra:libro
[obra:editorial=“Prentice-Hall”]
/persona:autor
/persona:nacionalidad

XSLT

- Definir hojas de estilo que generen una presentación tomando como entrada un documento XML
- Una hoja de estilo es también un doc. XML xsl:template asocia una entrada con una salida (el contenido de la etiqueta)
- El atributo match discrimina nodos (usa XPath)
- xsl:apply-templates recorre todos los hijos

Ejemplo:

```
<?xml version=“1.0” ?>
<xsl:stylesheet version=“1.0” xmlns=“http://www.w3c.org/1999/XSL/Transform”>
   <xsl:template match=“/”>
      <HTML>
         <xsl:apply-templates>
      </HTML>
    <xsl:template match=“obra:libro”>
      <BODY>
         <xsl:apply-templates>
      </BODY>
    </xsl:template>
   <xsl:template match=“persona:autor”>
      <P> Nacionalidad:
         <xsl:value-of select=“persona:nacionalidad”>
      </P>
    </xsl:template>
</xsl:stylesheet>
```

XLink

- Para definir enlaces entre dos documentos (o recursos)
- Cualquier elemento puede ser enlazable (atributo xlink:type="…")
- Enlaces simples: como el <A> de HTML pero con más semántica
- Enlaces extendidos: como un índice de recursos (no hay que modificar los documentos enlazados)

Ejemplo de enlace simple:

```
<DISCO xmlns:xlink="http://www.w3c.org/1999/xlink">
   < MUSICO     xlink:type="simple"
                xlink:href="http://www.xyz.com/1234/mdavis/"
                xlink.title="Bibliografia de Miles David"
                xlink.actuate="onRequest"
                xlink.show="new"> Miles David </MUSICO>
</DISCO>
```


XPointer

- Para referenciar secciones concretas dentro de un documento
- Xpath + otras funciones
- Se puede apuntar a otro elemento por número, nombre, tipo o relación con otros elementos del documento
- No permite obtener parte de un documento remoto (vía HTTP)

XML Schema

- Problemas de las DTDs
  - Descritas en una sintaxis no XML
  - Tipado de datos limitado
  - Mecanismo de extensión y reutilización limitado
  - No soportan XML Namespaces
- Pero son bien conocidos, usados y soportados

XML Schema

- Modelo que describe cómo se organizan etiquetas y contenidos en un cierto tipo de documentos
- Establece restricciones sobre el orden de los elementos y sobre los tipos de datos
- Elemento schema con subelementos
  - element (un atributo)
  - simpletype (un atributo con restricciones)
  - complextype (un registro)

XML Schema

- Avances
  - Sintaxis XML
  - Tipos de datos predefinidos
  - Tipos de datos definibles por el usuario
  - Refinamiento de tipos de datos
  - Soporta XML Namespaces
- Pero no es soportado completamente por herramientas y *parsers*
  - Problema de implementar "*drafts*" o "*proposals*"

XML: ¿Cómo usarlo?

- Un parser XML y un motor XSLT
  - XP, Xerces (Apache), JAXP
  - XT, Saxon, Xalan (Apache)
- Normalmente, en Java
  - También hay para C++, Perl, …
- APIs
  - SAX, DOM
  - JDOM
  - Propietarias del parser

Algunos lenguajes ya aportan su propio API: Java Architecture for XML Binding (JAXB)

SAX

- Simple API for XML
- Utilización
  - Instanciar un parser SAX
  - Registrar el objeto que implementa la interfaz ContentHandler en el parser
  - Procesar el documento XML (a partir de una URI, etc.)
  - Según se lee el documento se dispararán métodos del objeto que implementa ContentHandler

DOM

- Document Object Model
- Utilización
  - Instanciar un parser DOM
  - El parser analiza el documento y devuelve un objeto de la clase org.w3c.dom.Document
  - Se almacena entero en memoria
  - Se puede manipular con los métodos DOM (navegando por nodos)

SAX vs. DOM

- SAX
  - Simple
  - Rápido
  - No estándar (por ahora)
  - Solamente lectura
  - Interpretar mensajes, objetos serializados, etc.
- DOM
  - Potente
  - No tan rápido
  - Estándar W3C
  - Creación y manipulación
  - Mantener documentos y estructuras complejas

XML: utilización

- Nuevo ASCII
  - ficheros configuración, logs
- Personalización de documentos
  - Separación descripción/presentación
- Soporte para aplicaciones *middleware*
  - Integración de aplicaciones
- Intercambio de documentos
  - Comercio electrónico

XML se está usando intensivamente para publicar contenidos en la Web. Cuando los clientes acceden, se aplican las hojas de estilo correspondientes. No hay que tener varios conjuntos de paginas HTML dependiendo de los distintos tipos de visualización que podemos hacer (con frames, sin frames, idiomas, etc).

Respecto a su uso en comercio electrónico, su ventaja respecto al EDI tradicional viene dada por su extensibilidad. Un formato podría ser ampliado fácilmente añadiendo un nuevo elemento y las aplicaciones que usaran la versión anterior no se verían afectadas (los XML parsers simplemente ignorarían el nuevo elemento puesto que no se tuvo en consideración su tratamiento).

XML: Orígenes

- Inicialmente, surge para aportar semántica a las páginas Web (HTML)
  - Para permitir búsquedas en la Web no sólo sintácticas, sino semánticas
- Con el tiempo se ha convertido en:
  - Un generador de HTML (personalizado)
  - Estándar de intercambio de información
- Conclusión: las páginas Web siguen sin ofrecer una descripción semántica de sus

Respecto a las búsquedas, veamos la diferencia entre HTML y XML con un ejemplo:

HTML

```
<P><B>Sr. Antonio Rojo</B>
<BR>
Maria de Luna 3
<BR>
Zaragoza 50015
</P>
```

XML

```
<DIRECCION>
   <NOMBRE>
      <TRATAMIENTO> Sr. </TRATAMIENTO>
      <NOMBRE> Antonio </NOMBRE>
      <APELLIDO> Rojo </APELLIDO>
   </NOMBRE>
   <CALLE>Maria de Luna</CALLE>
   <NUMERO> 3 </NUMERO>
   <PISO> </PISO>
   <CIUDAD> Madrid </CIUDAD>
   <COD_POSTAL> 50015 </CODPOST>
<DIRECCION>
```

Hacer un algoritmo que buscara el código postal sería mucho más fácil sobre XML. Además en XML ya tenemos herramientas para buscar valores de etiquetas.

XML: Errores "históricos"

- Reinventa muchas "ruedas"
  - DTD, XML Schema vs. Definición BDs
  - XPAth, XPointer, XQL vs. SQL
  - Namespaces vs. previos Namespaces
- Quería ser simple pero se ha convertido en un monstruo
  - "Saber" XML no es trivial
  - ¿Cómo será el XML del futuro?

Errores históricos:

- Organiza los datos en forma de árbol (DTD, XML Schema ≅ BD jerárquicas)
- Define su propio lenguaje de acceso a datos (XPath y XQL son navegacionales)
- Define su propio método de comunicación remota (no usa CORBA, ni ningún otro estándar)
- Nuevas referencias (URI ≅ URL, XML Namespace ≠ CORBA Namespaces o Java namespaces)

XSLT es más novedoso porque separa completamente la especificación del documento de su visualización (aunque ya había estándares similares para SGML).

Programación Web

- Creación de páginas dinámicas
- Llevar la potencia de la programación a la Web (e-commerce, interactividad, etc.)
- Mecanismos (cronologicamente):
  - CGI's (servidor)
  - Applets (cliente)
  - Javascript (cliente)
  - Servlets (servidor)
  - ASP, JSP, PHP (servidor)
  - Flash (cliente)
  - AJAX (cliente)

Desde el punto de vista del diseño de un sitio web, la utilización de técnicas de programación web supone un gran avance respecto a la modularidad, versatilidad, y adaptabilidad a cambios. En muy pocos contextos será suficiente con limitarnos a crear una serie de páginas HTML estáticas.

La combinación de BD + generación de HTML dinámico es una de las arquitecturas a las que hay que tender en prácticamente cualquier web.

CGI: Common Gateway Interface

- CGI: programa invocable desde un servidor Web
- La entrada puede parametrizarse desde el cliente (URL). Método GET y POST
- La salida estándar del programa es capturada por el servidor Web y devuelta al cliente
- EL CGI puede hacer cualquier cosa que permita el lenguaje de programación

Otra aproximación es crear extensiones (propietarias) al servidor Web propiamente dicho:

• NSAPI (API para Netscape)

• ISAPI (API para IE de Microsoft)

Dentro del CGI se recogen los parámetros y se actúa en consecuencia, volcando en la salida lo que queramos que sea la siguiente página Web que vea el usuario. También indicará el MIME type de la salida.

•Método GET:

- •Los parámetros van en la URL
- •El string de parametros se guarda en a variable de entorno QUERY_STRING (limitado a 240 caracteres en algunos SO)
- •Se puede anotar como bookmark
- •El usuario del navegador ve la parametrización

•Método POST

- •Los parámetros van ocultos dentro de la comunicación HTTP
- •El string de parametros se accede leyendo de la entrada estandar (longitud de caracteres en la variable de entorno CONTENT_LENGTH)
- •Sin límite de tamaño (pueden ser megas)
- •Invisible para el cliente
- •No pueden ser anotadas como bookmark y a veces ni recargadas

En ambos casos, la variable de entorno REQUEST_METHOD guarda GET/POST

Applets:

- *Applet*: programa Java listo para ser ejecutado en un navegador WWW
- Requerimientos:
  - El navegador debe soportar Java (versiones). Java Plug-in.
- Ventajas:
  - Ejecución de un programa dentro de una página
- Desventajas:
  - Restricciones de seguridad
  - No hay forma de “volver” a HTML

En sus orígenes, las restricciones de seguridad aplicadas a los applets eran muy exigentes (sin acceso a los recursos locales, comunicación solamente con el nodo origen) lo que hacía inviable el diseño de muchas aplicaciones interesantes.

Actualmente existen mecanismos muy potentes para definir la política de seguridad a seguir con los applets.

Javascript

- Javascript: añadido de Netscape para tener un programa embebido en HTML
- Sólo se parece a Java en el nombre y la sintáxis
- No es soportado por todos los navegadores (versiones)
- Útil en caso de detección de eventos de Netscape (ej. cambio de página HTML)

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
function writeYear() {
  document.write(document.lastModified)
}
</SCRIPT>

</TITLE> Ejemplo Javascript </TITLE>
<BODY>
Sobre esta página te puedo decir que fue modificada por ultima vez en
<SCRIPT LANGUAGE="Javascript">
writeYear()
document.write(" y reside en ",location.href)
</SCRIPT>
<P>
Nada mas.
</HTML>
```

Servlets: CGI's en Java

- Requerimientos:
  - El servidor Web debe soportar Java (versiones)
- Ventajas:
  - Camino de ida y vuelta a HTML. Cadenas de servlets.
  - Más fácil de desarrollar que un CGI en C
  - Es un nuevo thread y no un nuevo proceso
  - La misma instancia atiende sucesivas llamadas
- "Desventajas":
  - Necesita un servidor Web que lo soporte
  - Visualización de la página resultado mediante código

Ejemplo (HTML):

```
<HTML> <HEAD><TITLE>Ejemplo Servlet </TITLE>
<BODY>
<FORM METHOD=GET ACTION="/servlet/Hello">
Si no te importa que lo pregunte, dime tu nombre:
<INPUT TYPE=TEXT NAME="nombre"><P><INPUT TYPE=SUBMIT>
</FORM>
```

Ejemplo (servlet):

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello extends HttpServlet {
  public void doGet(httpServletRequest req, HttpServletResponse res)
                                        throws ServletException, IOException {
    res.setContentType("text/html");
    PrintWriter out=res.getWriter();
    String name=req.getParameter("nombre");
    out.println("<HTML><HEAD><TITLE>Hello, "+name+" </TITLE>");
    out.println("<BODY> Hello, "+name+" </HTML>");
  }
}
```


ASP: Active Server Pages

- Desarrollado por Microsoft
- Código embebido en HTML (VBScript o Jscript)
- Puede incluir objetos ActiveX
- Respecto a la página resultado, separa claramente la parte dinámica (código ejecutable) de la parte estática (HTML)

Ejemplo:

```
<html><head>
  <TITLE>hi.asp</TITLE>
  </head>
  <body bgcolor="#FFFFFF">
  Today is <%=now%> and all is well<br>
  <%if hour(now())>12 THEN%>
        Good Evening
  <%ELSE%>
        Good Morning!
  <%END IF%>
  </body>
</html>
```

El código ASP es ejecutado en el servidor y la página HTML resultante será enviada al cliente (quién nunca verá el código entre <% y %>)

JSP: Java Server Pages

- Java embebido en páginas HTML (scriptlet)
- Para separar parte dinámica de presentación
- Internamente se traduce a servlets
  - La primera vez tarda algo en ejecutarse
  - Se recompila autom. al cambiar el .jsp
- Puede incluir Java Beans
  - No todo el código en la página JSP

Ejemplo:

```
<HTML><HEAD><TITLE>JSP Example</TITLE>
<BODY>
<H1>
<%
  if (request.getParameter("name")==null) {
       out.println("Hello World");
  } else {
       out.println("Hello, "+request.getParameter("nombre"));
  }
%>
</H1>
</HTML>
```


PHP

- PHP: Hypertext Processor
  - Lenguaje *Open Source*
  - Sintaxis similar a C, Java, Perl
- Soporte para gran número de SGBDs
- Soporte para XML
- Diseñado para ser más seguro que CGI´s en C o Perl

Ejemplo:

```
<HTML>
<HEAD>
 <TITLE>PHP Test</TITLE>
 </HEAD>
<BODY>
<CENTER>
<H1>PHP Test</H1>
<?php echo "<P>Hello World</P>"; ?>
</CENTER>
 </BODY>
</HTML>
```

Flash

- Tecnología propietaria
  - Macromedia →Adobe
- Gran ventaja sobre applets en gráficos vectoriales
- Entorno de pago, Flash player gratuito
- Muy extendido en la Web (animaciones, juegos sencillos)

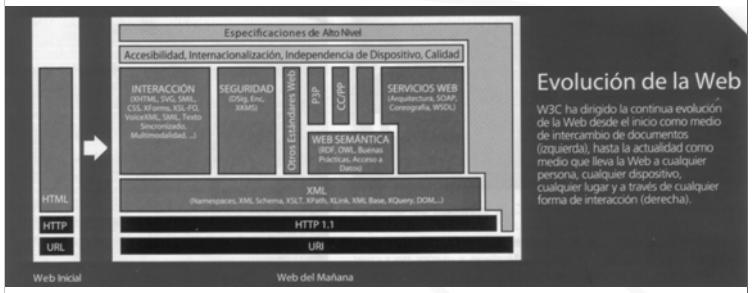
AJAX

- Asynchronous JavaScript And XML
- No es una nueva tecnología sino un cjto. de ellas:
  - XHTML (version XML de HTML) y CSS
  - DOM accedido desde JavaScript
  - Objeto XMLHttpRequest para recuperación de datos de forma asíncrona
  - Las páginas se actualizan concurrentemente con la interacción de usuario
- Debe soportarlo el navegador (lynx no)
- Lo usan todos los últimos productos Google

Servicios Web

- Objetivo:
  - Programas web → humanos y máquinas
- HTTP, XML
- Problemas actuales
  - Autodescripción (semántica)
  - Composición
  - Dos estándares
  - HTTP ?

Web Semántica



Comercio Electrónico

- Def. restringida: transacciones comerciales entre clientes y proveedores a través de Internet
- Def. más general: uso del ordenador para facilitar las operaciones de la compañía
  - EDI (*Electronic Data Interchange*)
  - La interacción con el cliente es un paso más dentro de la actividad comercial

Orígenes:

EDI (intercambio electrónico de datos): intentar que todo el proceso de negocio sea electrónico excepto la transferencia del producto físico.

Ventajas e Inconvenientes

- Mejor servicio a los clientes
  - Mayor disponibilidad
  - Más rapidez
  - Personalización
- Mejores relaciones con proveedores
- Mayor rendimiento de las inversiones
- Costos elevados
- Aceptar la nueva filosofía de trabajo
- Problemas de seguridad
- Software inmaduro o inexistente

Claves para e-commerce (1)

- Publicar datos de la empresa en la Web de forma correcta, legal y éticamente
  - Permitir a los clientes y suministradores interactuar con nosotros de forma eficiente
  - Cuidado con:
    - Información confidencial
    - Datos personales
    - Derechos de autor
  - Y también con los datos no actualizados
    - Precios

Amazon.com tuvo problemas legales al permitir a cualquier visitante ver el comportamiento de compra de sus clientes.

Buy.com anunció un precio incorrecto para un cierto tipo de monitor. Cuando recibió muchas órdenes de compra decidió no atenderlas pues perdería unas 50.000 ptas. por pedido. El asunto acabó en los tribunales por publicación de información incorrecta.

Claves para e-commerce (y 2)

- Analizar las estadísticas sobre el comportamiento de los visitantes
  - Determinar el mejor sitio para colocar publicidad (propia o de terceros)
- Integración de datos de Web (baja calidad) y de la empresa (alta calidad)
  - Control entrada datos por formularios Web
  - Validación datos según lo que tenemos almacenado anteriormente

En general, es una mala idea solicitar demasiados datos a los usuarios que quizá solamente quieran visitar nuestras páginas web. Además dicen las estadísticas que el 25% de los usuarios aportan datos falsos.

La verificación de los datos que recibimos a través de la Web es otro problema.: duplicación de usuarios debido a pequeños cambios sintácticos en los datos aportados.

También hay que tener mucho cuidado de que cada usuario solamente vea lo que se supone que puede ver. Y tener cuidado a la vez de no desconfiar de usuarios que no han intentado un acceso ilegal sino que simplemente se han equivocado, y son amenazados por nuestro sistema injustamente.

Por todo ello, conviene verificar que los datos que se van solicitando de la Web se corresponden con datos introducidos anteriormente, de cara a descubrir si la identidad de quien nos “habla” desde el navegador se corresponde con quien creemos (análisis probabilístico).

Bibliografía

- C. Liu et al., “Managing Internet Information Services”, O’Reilly, 1994
- http://www.xml.org/
- J. Hunter & Willian Crawford, “Java Servlet Programming”, O’Reilly, 1998
- E. Ladd & J. O’Donnell, “Using HTML 3.2, Java 1.1 and CGI”, Que, 1996
- R. Orfali & D. Harkey, “Client/server Programming with Java and CORBA”, Wiley, 1998
- http://home.netscape.com/eng/mozilla/3.0/handbook/javascript/

Sistemas legados

Sistemas legados

- Sistemas desarrollados con tecnología antigua pero que aun son útiles
- Objetivo: encapsulación para ofrecer conectividad con sistemas actuales
- Sistemas cerrados → sistemas abiertos
- Todo el software acaba siendo legado !!

Un caso muy frecuente actualmente es la posibilidad de crear acceso web a aplicaciones creadas según la tecnología de hace varias decenas de años (sistemas Cobol, por ejemplo).

Sustituir el viejo (pero efectivo) sistema Cobol por una nueva aplicación supondría un esfuerzo muy grande. La solución mejor pasa por “lavarle la cara” a la aplicación Cobol.

Hace poco Microsoft anunció que consideraba Windows XP como legado y que dejaba de mantenerlo.

El tiempo necesario para convertirte en un sistema legado es cada vez menor dada la velocidad con que se incorpora nueva tecnología al mercado. Solución: no olvidar el pasado al desarrollar nuevo software (ej. aplicaciones MSDOS en entornos Windows).

Distintas estrategias

- Encapsular completamente el sistema legado
- Sustituir el interfaz por uno nuevo
- Rehacer incrementalmente, usando ambos sistemas en paralelo
- Rehacer increm., migrando datos al comienzo
- Rehacer increm., migrando datos al final
- Rehacerlo todo desde cero

Las distintas estrategias estan ordenadas por orden creciente de riesgo (y también de la calidad del producto que obtenemos).

Rehacerlo todo desde cero

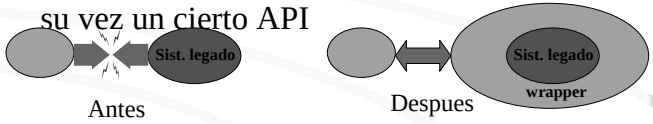
- Problemas de desarrollar desde cero
  - Falta de documentación sobre la aplicación antigua (millones de líneas de código)
  - Tiempo/dinero invertido en desarrollo, implantación, pruebas (para conseguir casi lo mismo)
  - Introducir nuevas técnicas (OO, GUIs, BDs distribuidas) aumentará aun más el riesgo
  - "Si algo funciona bien, no lo arregles"
- Alto riesgo de que el proyecto fracase

Muchas aplicaciones tardan años en ser depuradas correctamente y no siempre se documentan todos los cambios adecuadamente. Además las personas expertas en dichas aplicaciones han podido cambiar de trabajo, jubilarse, etc.

Si partimos de los requisitos iniciales y desarrollamos desde cero usando nueva tecnología probablemente tardaremos un cierto tiempo en volver a alcanzar el grado de calidad de la aplicación antigua.

Encapsulación

- Encapsulación = utilización de *wrappers*
- Wrapper: módulo especializado en acceder un sistema concreto ofreciendo a su vez un cierto API



- La OO es perfecta para este tipo de tareas de encapsulación

Tipos de wrappers

- Máquina-máquina
  - Ofertar un nuevo API (traducción de llamadas y resultados)
- Humano-máquina
  - Ofrecer un GUI donde sólo había un API
  - Ofrecer un API donde sólo había un GUI

Un wrapper puede ofrecer realizar un cambio de API para adaptarnos a las necesidades de nuevas aplicaciones. En realidad el wrapper sustituye las llamadas a las funciones nuevas por llamadas a las funciones antiguas, traduciendo los resultados en el formato anterior a resultados en el nuevo formato.

Algunos wrappers pueden estar especializados en ofrecer a un usuario humano un GUI para interactuar con una aplicación que fue diseñada para ser accedida solamente desde otro programa.

Mucho más frecuente es crear un wrapper que permite a un programa utilizar una aplicación que fue creada para ser accedida únicamente por humanos (el único punto de entrada es un GUI).

Allí donde hablamos de GUIs también podriamos estar hablando de simples interfaces de texto.

Wrappers

- Podemos tener distintos wrappers sobre una misma aplicación
  - Distintas formas de interactuar con la aplicación
- Pueden existir redes de wrappers
  - wrappers que se conectan con otros wrappers

El riesgo del uso de wrappers es muy bajo. Sin embargo no hay que olvidar que el sistema legado sigue allí y que antes o después habrá que aplicar una estrategia más agresiva.

Migración

- Distintos niveles
  - Interfaz
  - Lógica del programa
  - Estructuras de datos
- Migración parcial
  - Por ejemplo, cambiar de interfaz
- Migración Total
  - Menos riesgo que rehacerlo todo desde cero
  - Muy importante elegir el orden de migración

La migración o sustitución de una o todas las partes de un sistema legado no es una tarea fácil, pero todos los sistemas legados acaban siendo reemplazados antes o después.

La migración parcial del interfaz, sin tocar la lógica del programa ni las estructuras de datos, a veces no es posible si la lógica de presentación, programa y datos están entremezcladas, lo cual es muy frecuente en sistemas legados.

Migración en paralelo

- Primero, crear una nueva BD y migrar los datos
- Ir creando las nuevas aplicaciones, una a una
- Muy importante coordinar los datos nuevos y antiguos (bidireccionalmente)
- Tras un tiempo de empleo de ambas, pasar a usar solamente el nuevo sistema

Al convivir durante un tiempo ambas aplicaciones, cualquier cambio usando las aplicaciones legadas debe reflejarse en el nuevo sistema y viceversa.

Migración Database-First

- Construcción de un traductor de accesos “legados” a datos a SQL
- Creación de una BD relacional y migración de los datos
- Migración incremental de las aplicaciones
- Finalmente se desecha el traductor a SQL

El mayor problema de esta estrategia es que interceptar los accesos legados a datos para usar el traductor puede ser a veces imposible o no merecer la pena desde el punto de vista del coste.

Otro problema es que se migran los datos al comienzo, cuando el conocimiento sobre el sistema legado es menor.

Migración Database-Last

- Construcción de un traductor de SQL a accesos legados a datos
- Migración incremental de las aplicaciones
  - Usan SQL pero con el traductor usan las estructuras de datos antiguas
- Creación de una BD relacional y migración de los datos
- Finalmente se desecha el traductor a accesos legados a datos

A diferencia de la estrategia Database-First, la migración de los datos se hace al final del proyecto, cuando nuestro conocimiento es mayor porque ya hemos migrado las aplicaciones.

Sistemas de Información basados en el conocimiento

Sist. de Inf. basados en el conocimiento

- Se maneja un tipo particular de información: el conocimiento
- Se intenta simular comportamientos humanos (aprender, deducir, etc.)
- Entre ellos:
  - Sistemas expertos
  - Sistemas de agentes

Conocimiento: todo lo que se ha aprendido y organizado de acuerdo a aquellos conceptos, imágenes o relaciones que ha podido dominar, el conocimiento es una abstracción mental; supone cierto razonamiento y enjuiciamiento que organiza la información mediante su comparación y clasificación.

La información es la representación del conocimiento, comunica la estructura del conocimiento a través de datos, la información es la forma tangible y comunicable del conocimiento. Las escuelas, las universidades, tienen como actividad central propiciar el conocimiento a partir de la información.

Inteligencia Artificial: dotar a las máquinas de la capacidad para exhibir conductas que, si se observaran en seres humanos, se considerarían inteligentes. Principales áreas:

• Sistemas perceptivos: entendimiento de señales visuales, sonoras, etc.

• Procesamiento de Lenguaje Natural: entendimiento del lenguaje escrito.

• Robótica: imitar comportamientos motores de los humanos.

• Aprendizaje: adquisición de conocimiento adicional respecto al inicialmente introducido.

• Redes neuronales: aprendizaje, generalización, abstracción.

• Sistemas expertos: emulación de un experto.

• Sistemas de gestión de conocimiento: almacenamiento, recuperación, deducción.

• Sistemas de agentes: emulación del trabajo en equipo.

• Tutores inteligentes: enseñanza por ordenador.

• Teoría de juegos: estrategia y táctica.

Sistemas expertos

- Intentan simular el comportamiento de un experto
- Formado por una base de conocimientos, un motor de inferencia, un subsistema de explicación y un interfaz de usuario
- Su uso es limitado en ciertos campos
  - Medicina, inversión
- Es una herramienta muy valiosa para los expertos humanos

Experto: persona que domina un campo y que destaca claramente de aquellos que "simplemente" aplican los procedimientos lógicos. Muchas veces no sabe ni él mismo explicar algunas de sus decisiones, debido a su conocimiento basado en la experiencia y la intuición, que le hace reaccionar adecuadamente (al menos, mejor que los demás) ante situaciones no previstas o inéditas.

El subsistema de explicación es muy importante para ganarse la confianza del usuario del sistema experto.

Sist. Expertos: Base de conocimiento

- Conjunto de reglas y hechos, extraidos de expertos humanos
- También se maneja información incierta
- Debe ser creada por un ingeniero del conocimiento cooperando con los expertos
  - Es una tarea difícil, que puede provocar situaciones incontroladas del sistema

Motor de inferencia

- A partir del conocimiento propio y del adquirido durante una sesión (el problema a resolver), genera nuevos hechos
- Interactivamente solicitará la información sobre el problema que él crea relevante y no haya sido suministrada

Subsist. de explicación e interfaz de usuario

- El sist. experto explicará el motivo tanto de sus preguntas como de sus respuestas
- Muy importante para que confien en él

- El interfaz (al igual que el motor de inferencia y el subsist. de explicación) debe ser independiente del conocimiento manejado

Ventajas e inconvenientes

- Permite no perder años de experiencia de los expertos de una empresa
- Reduce la dependencia del personal crítico
- Puede adquirir el conocimiento de varios expertos
- Aumenta la disponibilidad de consejos expertos
- Excelente para entrenar nuevo personal

- Hay que confiar en ellos
- Los sistemas de reglas pueden ser inmanejables
- Se crearon tantas espectativas que fueron un fracaso
- Hay que actualizar el conocimiento base

Uno de los motivos para el fracaso de los sistemas expertos fue la pretensión de que sustituyeran a los expertos, cosa que no gustaba a los usuarios (confiar en una máquina) ni por supuesto a los mismos expertos.

Actualmente la filosofía a seguir es complementar a los expertos humanos con este tipo de sistemas, o como mucho tomarlo como una segunda opinión. Es decir, como un experto más pero nunca como el mejor experto.

Sistemas de agentes

- Def. Agente: modelado de un experto que puede cooperar con otros para realizar una tarea
  - Programas que actúan de parte del usuario
- Basados en modelos de cooperación
- Tipos:
  - Agentes software, agentes inteligentes, agentes móviles, sociedades de agentes, agentes funcionario, etc.

La idea de los agentes surge inspirada en contextos como el empleado de una agencia al que el “usuario” pide cosas como “Quiero alojamiento y billetes para viajar a cierto sitio”. El empleado de la agencia puede contactar con otros “agentes” de compañías aéreas y hoteles para estudiar las alternativas y elegir la que mejor convenga al usuario.

Los agentes son un paradigma de diseño que está aquí para quedarse, como ocurrió con la OO.

Características de los agentes

- Agenda de objetivos
- Cooperación con otros agentes
- Autonomía
  - Persistencia
- Inteligencia
  - Aprendizaje
  - Manejo de conocimiento
- Reactivos al entorno
- Movilidad

Dependiendo del contexto, algunos parámetros pueden no tener sentido, pero dan una idea de si lo que estamos construyendo es un agente o "simplemente" un programa.

Desde nuestro punto de vista, si desarrollamos un programa suficientemente independiente como para aprender y realizar tareas no triviales por si mismo e interactuar con otros programas similares, entonces hay muchas posibilidades de que lo que hemos hecho sea realmente un agente.

Agentes móviles

- Agentes que se mueven a través de la red, es decir, ellos deciden viajar
  - No es lo mismo que código móvil
- Los sistemas de agentes móviles (MAS) son la infraestructura que permite a los agentes viajar
- Los viajes se realizan de contexto a contexto
  - Contexto o place: entorno de ejecución de un agente

Que un agente movil "viaje" significa que se debe interrumpir su ejecución, transportar por la red tanto su código como su estado (sus datos), y continuar su ejecución en el punto siguiente a la orden de viaje.

Ventajas de los agentes móviles

- Mejor que C/S en redes inestables o de capacidad limitada, reducen tráfico de red
  - Entornos inalámbricos
  - Modelo indirecto frente a CS
- Reducen el software necesario en ordenadores cliente
  - La aplicación viaja donde esta el cliente
- Muy apropiados para ordenadores clientes de capacidad limitada

Modelo indirecto: cliente - intermediario - servidor

Normalmente la comunicación entre el cliente y el intermediario es inalámbrica (lenta, inestable, cara), y la del intermediario con el servidor es cableada (rápida, fiable, barata).

MAS (Mobile Agent Systems)

- Aglets (http://www.trl.ibm.com/aglets/)
- Voyager (www.objectspace.com)
  - Más que un MAS :-)
- Grasshopper (www.grasshopper.de)
  - Sigue el estandar MASIF
- Jumping Beans (www.JumpingBeans.com)

- Incompatibles entre ellos (de momento)

Actualmente se está trabajando en varios (esto ya es malo…) estándares, aunque el más “serio” parece ser FIPA (Foundation for Intelligent Physical Agents).

Bibliografía

- E. Pitoura and G. Samaras, "Data Management for Mobile Computing", Kluwer Academic Publishers, 1998
- A. Franklin and A. Graesser, “Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents”, http://www.msci.memphis.edu/~franklin/AgentProg.html
- Lista de MAS: http://www.agentbuilder.com/AgentTools/index.html
- FIPA: http://www.fipa.org

Sistemas de integración de información

Sistemas de integración de información

- Integración vs. Centralización
- Metainformación
- LAV vs. GAV
- Bases de datos federadas
- Bibliotecas digitales
- Sistemas de información globales

Integración vs. Centralización

- El número de fuentes de datos accesibles crece
- Centralización
  - mantener un único depósito de datos donde acceder desde distintos nodos
- Integración
  - enlazar virtualmente los distintos depósitos de datos (heterogéneos) para ofrecer una visión similar a un único depósito centralizado

La centralización solamente es posible en contextos concretos y de crecimiento controlado.

Cuando existe un gran número (creciente) de fuentes de datos, la integración es la mejor solución. Tecnológicamente es más complicado pero apuesta por la modularidad.

Se ha trabajado bastante sobre integración estática (a priori, semiautomática, semánticamente muy expresiva) pero actualmente se tiende hacia una integración dinámica (tipo *plug & play*, automática y por tanto menos expresiva).

Metainformación

- Información semántica sobre los datos
- Objetivo
  - Describir contenidos
  - Separar datos de significado
  - Permitir búsquedas “inteligentes”
- Problema:
  - Su generación es difícil de automatizar

Almacenar datos es interesante y útil pero es muy importante saber qué es lo que estamos guardando, hacer explícito cómo interpretarlos. Actualmente podemos acceder a muchos datos (la Web) pero sigue recayendo en los usuarios la interpretación de los mismos.

La metainformación actúa de catálogo de los depósitos de datos: podemos consultar que hay almacenado sin preguntar por los datos concretos. Ejemplo: ¿hay diccionarios de ingles-italiano en la biblioteca del CPS?

El espacio ocupado por la metainformación es mucho menor que el espacio ocupado por los datos.

Sin embargo, dado unos datos concretos es prácticamente imposible que un sistema automático deduzca qué representan, excepto en contextos conocidos. Aún no sabemos cómo hacer una generación automática de metainformación de alto valor semántico.

Bases de datos federadas

- Integración de distintas bases de datos autónomas, distribuidas y heterogéneas
- Generación de un esquema global
  - Fases de traducción e integración
  - Obtención de la información de enlace (LAV o GAV)
  - La preguntas se formulan sobre el esquema global
- Principal objetivo
  - ocultar la heterogeneidad a los usuarios

Para crear una base de datos federada se dan dos pasos:

1. Traducción: los esquemas de datos de los depósitos (que son heterogéneos) se traducen a un modelo de datos canónico, aprovechando para enriquecer semánticamente dichos esquemas.
2. Integración: se definen relaciones semánticas entre los distintos esquemas (homogéneos sintácticamente pero homogéneos semánticamente) de cara a obtener un esquema global.

Durante dichos pasos se generará la información de enlace necesaria para responder a las preguntas que se formularán sobre el esquema global.

LAV vs. GAV

- GAV (Global As View)
  - Enlaces desde el esquema global a los esquemas locales
  - Fácil reescritura de preguntas
  - Sensible a incorporación/eliminación de depósitos de datos
- LAV (Local As View)
  - Cada depósito de datos tiene su descripción
  - Fácil incorporación de nuevos depósitos
  - Procesamiento de preguntas complicado

Procesamiento de preguntas según GAV:

1. La pregunta se formula sobre el esquema global
2. Utilizando la información de enlace (entre términos del esquema global y los elementos de datos) se reescribe la pregunta en términos de los depósitos de datos
3. Separación de la pregunta multidepósito a varias preguntas monodepósito
4. Ejecución de las preguntas y obtención de las respuestas de cada depósito
5. Combinación de las distintas respuestas según el plan de ejecución

Procesamiento de preguntas según LAV:

1. La pregunta se formula sobre el esquema global
2. Hay que combinar las distintas vistas de los depósitos de datos disponibles para obtener una expresión equivalente a la pregunta formulada (costoso computacionalmente)
3. Una vez encontrada la combinación (o plan), se accede a los depósitos concretos
4. La distintas respuestas se combinan según el plan.

Bibliotecas digitales

- Biblioteca electrónica/virtual/digital
- Def. 1: Automatización total de servicios bibliotecarios
- Def 2: Gestión de colecciones de objetos (documentos) digitales
- De gran impacto actualmente en distintos contextos

Las técnicas de bibliotecas digitales no pretenden limitarse a su aplicación a entornos bibliotecarios sino extender la idea de almacén de libros a la idea de almacén de información.

La web de una biblioteca no constituye una biblioteca digital. El contexto de biblioteca digital supone la posibilidad de realizar de manera electrónica cualquiera de las operaciones que podríamos realizar en una biblioteca clásica, lo cual pasa necesariamente por la digitalización total de los documentos u objetos digitales.

En esta aproximación hay lugar para comunidades inicialmente muy distintas:

•Tecnología: BDs, recuperación de información, interacción hombre-máquina, digitalización, ingeniería del software

• Contenidos: bibliotecarios, documentalistas, catalogación

Sistemas de Información Globales

- Muchos depósitos de datos (miles, millones)
- Gran heterogeneidad a todos los niveles
- Altamente dinámico y cambiante
- Un ejemplo: La Web

- Adaptación de las técnicas conocidas a dicho contexto
- Aún es objeto de investigación