POLITÉCNICA
"Ingeniamos el futuro"

CAMPUS
DE EXCELENCIA
INTERNACIONAL

MiW

Visión General de la Ingeniería Web.
1. Web 1.0

Luis Fernández Muñoz
setillofm@gmail.com

INDICE

1.1. Previo a la Web

1. Hipertextos / Hipermedia
2. Formatos Multimedia
3. Sistemas Operativos
4. Lenguajes de Programación
5. Sistemas Gestores de Bases de Datos
6. Interfaces Gráficas de Usuario
7. Comunicaciones
8. Aplicaciones Distribuidas
9. Software Libre

1.1.1. Hipertextos / Hipermedia

- Existe la necesidad de gestionar eficientemente la eclosión de conocimiento científico tras la II Guerra Mundial
- **Vannevar Bush** considera que "la mente salta instantáneamente al dato siguiente, que le es sugerido por asociación de ideas, siguiendo alguna intrincada trama de caminos"
- Propone *Memex en 1945*: un dispositivo que almacena todo tipo de documentos y consta de un teclado y palancas que permiten la consulta de datos almacenados en microfilms proyectados en pantallas translúcidas. Además, incluye la posibilidad de que el usuario pueda escribir notas convirtiéndose en autor. Nunca fue materializado.

1.1.1. Hipertextos / Hipermedia

- **Nelson, T.**, filósofo y sociólogo, propone:
  - Hipertexto como la presentación de información en una red de nodos enlazados a través de los cuales los lectores pueden navegar libremente en forma no lineal.
  - Permite la coexistencia de varios autores, desliga las funciones de autor y lector, permite la ampliación de la información en forma casi ilimitada y crea múltiples rutas de lectura.
  - Además, sugiere que la interfaz "debe ser tan simple que un principiante en una emergencia pueda entenderlo en un plazo de 10 segundos".
  - El concepto de hipermedia es la generalización del concepto a de hipertexto aplicado distintos medios de información.

1.1.1. Hipertextos / Hipermedia

- **Nelson, T.**, filósofo y sociólogo, propone:
  - En 1960, lidera el proyecto *Xanadu* y junto con **Engelbart, D**. (inventor del ratón en 1968) desarrollaron la primera implementación de los conceptos de hipertexto e hipermedia.
  - Posteriormente, a principios de los 80's:
    - *IBM* desarrolló *Hypertext Editing System*, un sistema de hipertexto para ordenadores personales optimizado para documentos extensos
    - *Macintosh* desarrolló *Intermedia* e *HyperCard* precursor de las herramientas de presentación actuales (*PowerPoint*, …) y de la *Wiki*.

1.1.2. Formatos multimedia

- La historia de los formatos para información multimedia están salpicados de luchas de patentes y estándares, algoritmos de compresión con o sin pérdida de información, … casi todos ellos previos al nacimiento de la Web
- Texto digital:
  - Caracteres: *ASCII* por *ANSI* en 1963, *UNICODE* por *Unicode Consortium* en 1991, …
  - Publicación: *Tex* por Knuth en 1980, *Postcript* por *Adobe Systems* en 1985, *RTF* por *Microsoft* en 1987, y *PDF* por *Adobe Systems* en 1993, …
  - Estructuración: *GML* por *IBM* en 1960 y *SGML* por *ISO* en 1986 (precursores del XML)

1.1.2. Formatos multimedia

- Imágen digital:
  - *BMP* por *Microsoft* en 1986, *GIF* por *Compuserve* en 1987, *JPEG* (Joint Photographic Experts Group) por *ISO* en 1991, *PNG* recomendación de *W3C* en 1996, …
- Sonido digital:
  - *WAV* por *Microsoft/IBM*, *MP3* (Moving Pictures Experts Group) por *ISO* en 1986, …
- Video digital:
  - *MOV* por *Apple* en 1991, *AVI* por *Microsoft* en 1992, *MPG* (Moving Pictures Experts Group) por *ISO* en 1992, *RealVideo* por *RealNetworks* en 1995, …
  - En particular, la historia de los formatos de video digital están salpicados por *codecs* (para la transformación de datos en señal y viceversa) , *streaming* (que evita la descarga total previa a la reproducción), alta definición (con relación 16:9), …

1.1.3. Sistemas Operativos

- Existen multitud de sistemas operativos previos a la Web:
  - *Unix* de *AT&T* en 1969 para grandes sistemas y con interfaz gráfica *X Windows* de *MIT/Digital/IBM* en 1983.
  - *Mac OS* de *Apple* en 1978 con sistemas gráficos revolucionarios en usabilidad para el gran público *Lisa* y *Macintosh* en 1983 y 1984 respectivamente.
  - *MS/DOS* y *Windows* de *Microsoft* en 1981 y 1985 respectivamente para *IBM PC*
  - *GNU/Linux* de software libre en 1991.

1.1.4. Lenguajes de Programación

- Existen multitud de lenguaje previos al surgimiento de la Web. Caben destacar (I):
  - *Fortran* por **Backus, J.W.** de *IBM* en 1953 para desarrollar una alternativa más práctica al lenguaje ensamblador
  - *COBOL* por *CODASYL* en 1959 para crear un lenguaje de programación universal que pudiera ser usado en cualquier ordenador
  - *C* diseñado por **Ritchie, D**. de *Laboratorios Bell* en 1969 para programación de sistemas
  - *Smalltalk* por **Kay, A., Ingalls, D., Kaehler, T., Goldberg, A.** de Xerox PARC en 1970-80 para permitir expandir la creatividad de sus usuarios (ventanas, iconos, ratón, … OO, MVC)
  - *C++* diseñado por **Stroustrup, B**. de *Laboratorios Bell* en 1983 para incorporar el paradigma orientado a objetos en el lenguaje C
  - *Objective-C* diseñado por **Cox, B**. y **Love, T**. de *Steptone* en 1980 para la reusabilidad en el diseño software

1.1.4. Lenguajes de Programación

- Caben destacar (II):
  - *Python* diseñado por **van Rossum, G**. en 1980 de *CWI* para aumentar la legibilidad del código
  - *Perl* diseñado por **Wall, L**. en 1987 para facilitar el procesamiento de informes
  - *VisualBasic* diseñado por **Cooper, A**. para Microsoft en 1991 para rapido desarrollo de aplicaciones con interfaz gráfico de usuario y acceso a bases de datos
  - *Ruby* diseñado por ***Yukihiro Matsumoto*** en 1993 para equilibrar los paradigmas funcional e imperativo
  - *Java* diseñado por **Gosling, J**. de *Sun MicroSystem* en 1995 para la televisión interactiva
  - *ColdFusion* diseñado por los **hermanos Allaire** de *Allaire Corporation* en 1995 para el desarrollo rápido de aplicaciones
  - *COOL* diseñado por **Hejlsberg, A**. en 1999 para la docencia de compiladores. Renombrado como *C#* por *Microsoft* en 2000 para *.NET*

1.1.4. Lenguajes de Programación

- Al margen del objetivo de cada lenguajes, sus influencias y diferencias radican en:
  - La **sintaxis** (muchos comparten la propuesta de C) con el compromiso entre la legibilidad y la brevedad
  - El **paradigma** de programación: funcional, imperativo, orientado a objetos, orientado a eventos, … con el compromiso en el módulo constituyente de programas
  - La **ejecución** compilada vs interpretada con el compromiso entre tiempo de desarrollo frente al tiempo de ejecución
  - Los **sistema de tipos** estático, dinámico o híbrido con el compromiso en los costes de pruebas del software
  - La **portabilidad** sobre un sistema operativo o sobre cualquier sistema operativo gracias a las máquinas virtuales
  - La **eficiencia**, muchas veces, como consecuencia de la elección en el diseño en varios aspectos anteriores

1.1.5. Sistemas Gestores de Bases de Datos

- A partir de *SQL* publicado por *IBM* en 1981, surgen múltiples *SGBD* con API's (*ODBC/JDBC*) para su acceso:
  - *Adabas* de *Software AG* en 1969. Pionera
  - *Oracle* en de *Oracle* 1977
  - *Informix* de *RDS* en 1985
  - *Micorsoft SQL Server* en de *Microsoft* 1989
  - *MySQL* de *MySQLAB* en 1995 cedida al software libre
  - *Derby* de *Apache* en 1997 de software libre

1.1.6. Interfaces Gráficas de Usuario

- A través de Entornos de Desarrollo Integrado (IDE):
  - *Smalltalk* de *Xerox* en 1980. Pionero de la orientación a objetos y la arquitectura Modelo/Vista/Controlador (MVC)
  - *X Windows* del *MIT* para *UNIX* en 1983.
  - *Turbo Pascal* de *Borland* en 1983. Precursor de *Delphi* en 1995
  - *Turbo C* de *Borland* en 1987. Precursor de *Borland BuilderC/C++* en 1997
  - *Quick C* de *Microsoft* en 1987. Precursor de *Visual C++* en 1993 con arquitectura Documento / Vista y *Visual Studio* en 1998.
  - *Visual Basic* de *Microsoft* en 1991.
  - *AWT* de Sun Microsystemen 1995. Precursor de Swingen 1997 con MVC

1.1.7. Comunicaciones

- Terminales para puerto serie entre el equipo de datos y el equipo de circuito de datos (RS232)
- Redes de área local para interconexión de equipos de datos en área reducida (1km)
  - *NetWare* de *Novell* en 1983
  - *Windows NT* de *Microsoft* en 1993
- Redes de área global
  - *X-25* sobre la red pública de datos
  - *Internet* en 1980 sucesor de la red militar *ARPAnet* sobre *TCP/IP* de **Khanen, C**. en 1974

1.1.7. Comunicaciones

- Protocolos:
  - *Telnet* de *IETF* en 1969 para conexión remota
  - *FTP* por **Buhshanen** 1971 para transferencia de archivos
  - *ICMP* por **Postel** para control y notificación de errores
  - *SMTP* de *ARPANET* en 1982,
  - *POP* en *Berkeley* en 1983 y
  - *IMAP* por **Crispinen** 1986 para obtención de correos electrónicos
  - *NNTP* en *Berkeley* en 1986 para noticias (*news*)
  - …
- Sockets (TCP/IP): mecanismo para la entrega de paquetes de datos provenientes de la tarjeta de red a los procesos apropiados, fundamento de los protocolos y aplicaciones a medida

1.1.8. Aplicaciones Distribuidas

- Arquitecturas por capas:
  - Arquitectura 2 capas (solución de los 90'):
    - Cliente: lógica de presentación (vista) y lógica (controlador). Con *VisualBasic, Delphi, …* sobre *Windows, …*
    - Servidor: capa de persistencia (modelo) con *Oracle, Informix, …*
    - Conexión: *ODBC (open data base connectivity)*
- Arquitecturas por componentes (ante la heterogeneidad):
  - *RPC* (remote procedure call) del paradigma tradicional desde 1981 en *Xerox* (*Courier*)
  - *CORBA (common object request bróker arquitecture)* de *OMG* en 1991 independiente de plataforma, lengujes, … para el paradigma orientado a objetos precursor de *RMI (remote methods invocation)* de *Sun MicroSystem* en 1997 dependiente de Java
  - *DCOM* (distributed component object model) de *Microsoft* en 1995
- Requería el despliegue de ambos componentes!!!

1.1.9. Software Libre

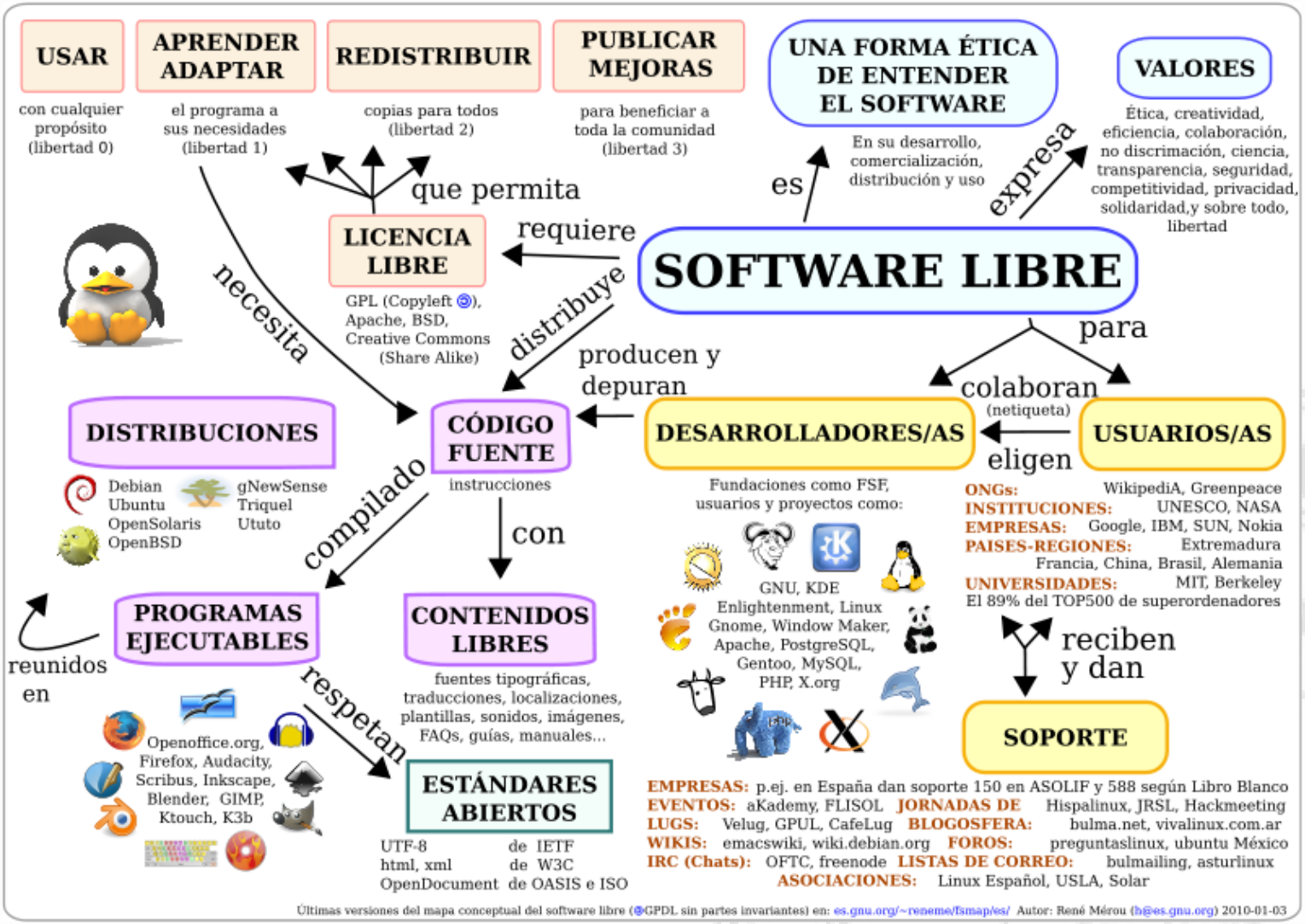
- En 1960/70 el software no era un producto comercial, lo era el hardware
  - En ámbitos universitarios y empresariales, creaban y compartían elsoftware sin ningún tipo de restricciones (*hackers*)
  - En 1980 comienzan sistemas operativos propietarios
- *Proyecto GNU* (**Stallman**, 1983):
  - un sistema operativo de software libre
  - *Linux* por **Torvalds** en 1992 es el núcleo del sistema operativo
  - Decenas de herramientas (analizadores, intérpretes, compiladores, …)

1.1.9. Software Libre

- **Free Software Foundation**(1985):
  - la libertad de usar el programa, con cualquier propósito
  - la libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
  - la libertad de distribuir copias del programa, con lo cual puedes ayudar a tu prójimo.
  - la libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

1.1.9. Software Libre

mapa conceptual de GNU



1.2. Fundamentos de la Web

1. Origen
2. Estándares
3. Funcionamiento

1.2.1. Origen

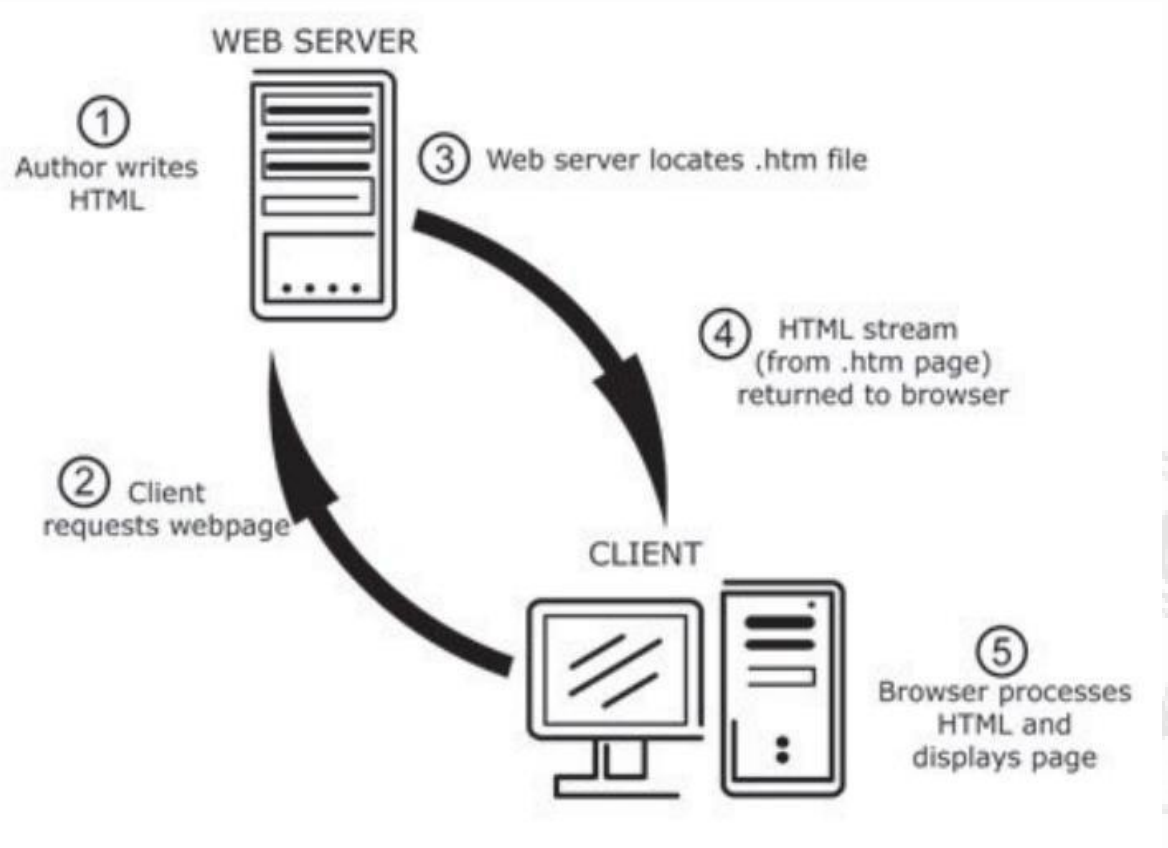
- *World Wide Web* (la *Web* o *WWW*) por **Tim Berners-Lee** en 1991 en el *CERN (Organización Europea para la Investigación Nuclear).*
  - Inspirado en el dispositivo *Memex* de **Vannevar Bush** y el proyecto *Xanadu* de **Ted Nelson**, la Web es servicio de hipertexto/hipermedia para compartir documentos en *Internet* (sobre *TCP/IP*).
  - Hoy en día es considerado como el acceso más sencillo y comprensible al universo de la información disponible en Internet
  - *World Wide Web Consortium* (*W3C*) se funda en 1994 para velar por los estándares de la Web
    - *Web Accessibility Initiative (WAI)* es una rama fundada en 1997 para velar por la accesibilidad con niveles A, AA y AAA

1.2.3. Estándares

- Se fundamenta en 3 estándares:
  - Localizador de Recursos Universal: *URL*. Secuencia de caracteres, de acuerdo a una sintaxis, que se usa para nombrar recursos de Internet para su localización o identificación
  - Lenguaje de Marcas de Hipertexto: *HTML*. Para describir la estructura mediante etiquetas y el contenido en forma de texto, así como para complementarlo con imágenes, …
  - Protocolo de Transerencia de Hipertextos: *HTTP*. Define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web. Orientado a transacciones, sigue el esquema petición-respuesta entre un cliente y un servidor sin estado, sin guardar ninguna información sobre conexiones previas
- Consecuencia: cualquiera que sepa poner etiquetas a su información y la aloje en un servidor web será accesible por todo aquel que tenga un cliente web (navegador presente en todos los sistemas operativos que no quieren estar fuera)

1.2.3. Funcionamiento

1. Autor escribe en el servidor sus contenidos en *HTML*
2. Usuario solicita en el cliente *Web* un documento mediante una *URL* en una trama *HTTP*
3. El Servidor *Web* localiza y responde con el documento referenciado con otra trama *HTTP*
4. El Cliente presenta el documento *HTML* al usuario



1.3. Dinamismo en el servidor

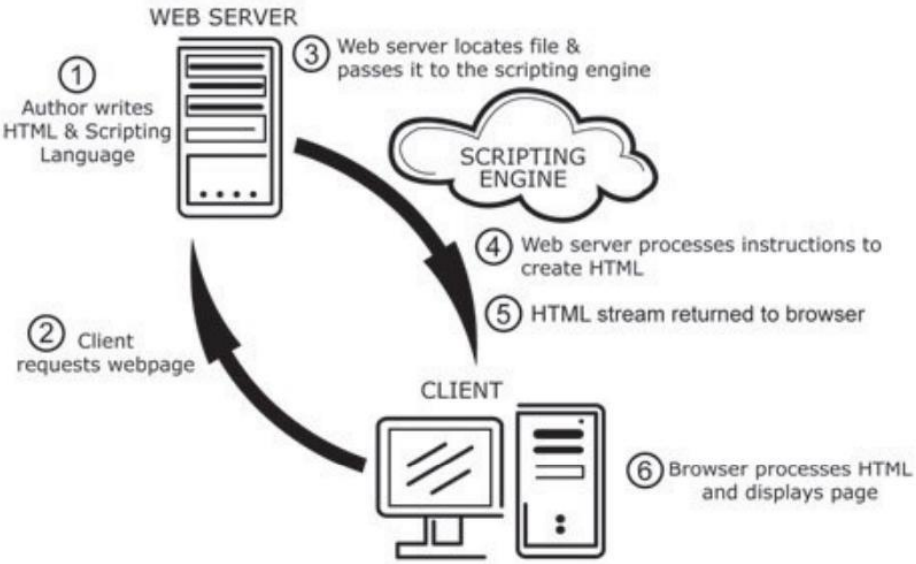
1. Justificación
2. Funcionamiento
3. Tecnologías

1.3.1. Justificación

- Inicialmente la Web era estática: solo se navegaba a través de documentos previamente escritos.
- Se requería el carácter dinámico en la Web para presentar documentos cuya información fluctuara en el tiempo (**CRUD**-*create/read/update/delete*).
  - Y así competir con la arquitectura dos capas pero sin problemas de distribución y mantenimiento sin desplazamientos (tiempo y dinero), portabilidad (interfaz independiente del sistema operativo), … todo el desarrollo radica en el servidor web
  - Por tanto, se incorporaron los formulario a HTML para introducir datos para las altas, bajas, modificaciones y consultas

1.3.2. Funcionamiento

1. Programador escribe en el servidor su *script* que genera HTML consultado datos
2. Usuario solicita en el cliente *Web* solicita un recurso mediante una URL en una trama HTTP
3. El Servidor *Web* localiza, **ejecuta** y responde con el documento generado en otra trama HTTP
4. El Cliente presenta el documento HTML al usuario



1.3.3. Tecnologías

- *CGI (Common Gateway Interface)*: protocolo que define cómo un servidor Web delega la generación de páginas Web a una aplicación o un fichero ejecutable.
  - primera implantación por **Wall** entrega en 1993 promovida por un grupo de noticias de *Usenet*
    - *Ejemplo de CGI*

```
#include <stdio.h>
#include <stdlib.h>
int main(void) {
char *data;
long m,n;
printf("%s%c%c\n", "Content-Type:text/html;charset=iso-8859-1",13,10);
printf("<TITLE>Multiplication results</TITLE>\n");
printf("<H3>Multiplication results</H3>\n");
data = getenv("QUERY_STRING");
if(data == NULL)
  printf("<P>Error! Error in passing data from form to script.");
else if(sscanf(data,"m=%ld&n=%ld",&m,&n)!=2)
 printf("<P>Error! Invalid data. Data must be numeric.");
else
 printf("<P>The product of %ld and %ld is %ld.",m,n,m*n);
return 0;
}
```

1.3.3. Tecnologías

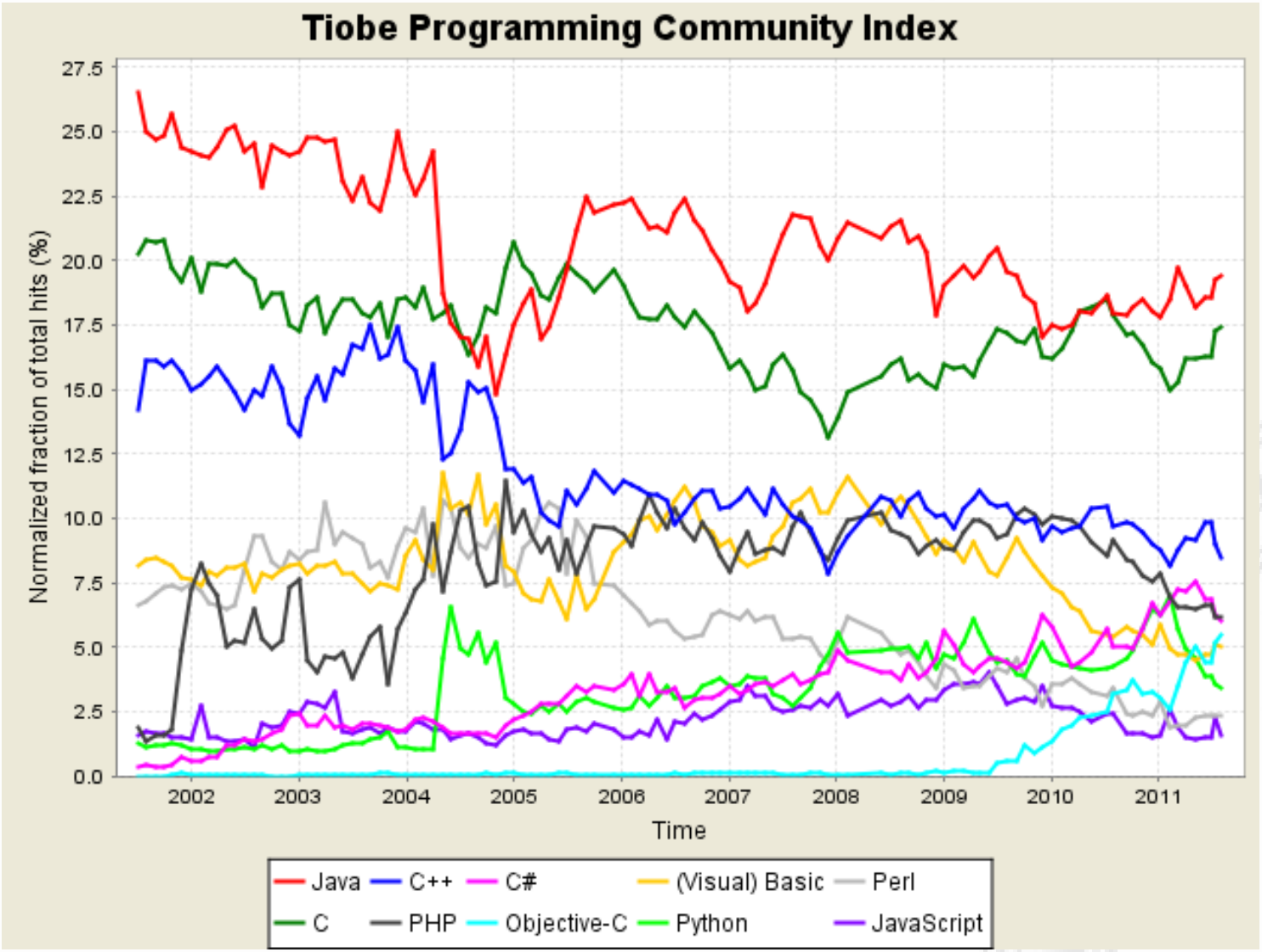
- Se crearon lenguajes de script específicos para generación páginas Web dinámicas que entremezclan código HTML con sentencia de programación: el código HTML se genera directamente en la salida del script; el código de programación se ejecuta con normalidad
  - *ASP (Active Server Page)* diseñado por *Microsoft* en 1998
  - *PHP (PHP Hypertext Pre-processor)* diseñado por **Lerdof, R.**, **Gutmans, A. y Suraski, Z**. en 1998
  - *JSP (Java Server Page)* diseñado por *Sun Microsystem* en 1999
  - *CFML (ColdFusion)* diseñado por *Macromedia* en 2001 como evolución de *Cold Fusion* de *Allaire* adquirida por ésta.

1.3.3. Tecnologías

- *Ejemplo de ASP*

```
<%@ Import Namespace="System.Data.OleDb" %>
<script runat="server">
  sub Page_Load
    dim dbconn,sql,dbcomm,dbread
    dbconn=New OleDbConnection ("Provider=
      Microsoft.Jet.OLEDB.4.0; data source=" &
      server.mappath("northwind.mdb"))
    dbconn.Open()
    sql="SELECT * FROM customers"
    dbcomm=New OleDbCommand(sql,dbconn)
    dbread=dbcomm.ExecuteReader()
    customers.DataSource=dbread
    customers.DataBind()
    dbread.Close()
    dbconn.Close()
  end sub
</script>
<html>
  <body>
    <form runat="server">
      <asp:Repeater id="customers" runat="server">
        <HeaderTemplate>
        <table border="1" width="100%">
          <tr>
            <th>Companyname</th>
            <th>Contactname</th>
            <th>Address</th>
            <th>City</th>
          </tr>
        </HeaderTemplate>
         <ItemTemplate>
          <tr>
            <td><%#Container.DataItem("companyname")%></td>
            <td><%#Container.DataItem("contactname")%></td>
            <td><%#Container.DataItem("address")%></td>
            <td><%#Container.DataItem("city")%></td>
          </tr>
         </ItemTemplate>
         <FooterTemplate>
        </table>
      </FooterTemplate>
      </asp:Repeater>
      </form>
       </body>
    </html>
```

1.3.3. Tecnologías



http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

1.4. Tecnologias XML

1. Justificación
2. Servicios Web
3. Redifusión

1.4.1. Justificación

- *IBM* ofrecía a sus cliente sistemas de gestión de información pero achacaba la imposibilidad de manipular su propia documentación interna (especificación de requisitos, informes de errores, manuales, gestión de proyectos, …) de forma automática porque se almacenaban en los formatos propietarios de los editores de texto que sustituyen la estructura de la información por la estructura de la presentación.
- *IBM* definió *GML (General Mark Language*) en 1960 y la *ISO* lo estandarizó bajo el nombre *SGML (Standard General Mark Language*) en 1986 con hojas de estilo (previos *CSS –cascading style sheet*).

1.4.1. Justificación

- Ambos lenguajes de marcado se basan en marcas o etiquetas de metadatos que encierran los datos con una etiqueta de apertura (“<metadato>”) y otra de cierre (“</metadato>”) de forma anidada
- Mediante los metadatos, no hay que “deducir” el significado de un dato por su posición en una tabla o dentro de una oración para su correcto procesamiento. Ejemplo:

```
<pedido>
    <codigoArticulo>458.34</codigoArticulo>
    <precioUnidad>89.345<precioUnidad>
    <cantidad>823</cantidad>
</pedido>
```

1.4.1. Justificación

- S/GML permite estructurar la información con meta-información para posibilitar el procesamiento de los datos frente a HTML que permite estructurar la presentación de los datos dificultando el procesamiento de los datos
  - Acorde al patrón MVC (desacoplando la vista y el modelo), otros lenguajes procesarán los datos en la estructura de la información (modelo) para generar los datos deseados con la estructura de presentación (vista) requerida

1.4.1. Justificación

- Ejemplo: un posible procesamiento del ejemplo anterior puede generar una presentación en HTML con la siguiente estructura:

```
<html>
 <body>
  <table>
   <tr>
    <td>Articulo</td>
    <td>Cantidad</td>
    <td>Articulo</td>
   </tr>
   <tr>
    <td>458.34</td>
    <td>89.345</td>
    <td>823</td>
   </tr>
  </table>
 </body>
</html>
```

1.4.1. Justificación

- Ejemplo: otro posible procesamiento del ejemplo anterior puede generar una presentación en HTML con la siguiente estructura:

```
<html>
  <body>
    <p> Se solicitan 823 unidades del artículo con código 458.34
                    al precio de 89.345 euros</p>
  </body>
</html>
```

1.4.1. Justificación

- *XML (eXtensibleMark Language)* es una simplificación (10%) de SGML estandarizado por W3C en 1998
  - XHTML (eXtensible Hipertext Mark Language) es una redefinición de HTML cumpliendo las normas de XML (estructura anidada de etiquetas abiertas y cerradas)
  - Actualmente exiten multitud de nuevos lenguajes extendidos de XML: MathML, javaML, VoiceXML, …
  - Conjuntamente a XML se han definido estándares para:
    - Automatizar los procesos de validación de documentos XML
    - Automatizar los procesos de presentación de documentos XML
    - Automatizar los procesos de manipulación de documentos XML

1.4.1. Justificación

- Para validar la estructura de la información de un *XML*:
  - Se define el lenguaje estándar *DTD (Definition Type Document)* en 1998 por la *W3C* que define la estructura de información de los documentos *XML*
  - Se define el lenguaje estándar *XML Schema* (vocabulario de XML) en 2001 por la *W3C* con la misma funcionalidad que el lenguaje DTD pero con mucha más potencia
  - Complementariamente, existen multitud de herramientas en muy diversas tecnologías (bibliotecas de lenguajes, *pluggins* en clientes Web, ...) que a partir de un documento *XML* y un documento *DTD* o *XML Schema*, determinan la corrección de la estructura de la información del documento *XML* acorde a la definición dada por documento *DTD* o *XML Schema*

1.4.1. Justificación

- Para generar presentaciones de un documento *XML*:
  - Se compatibiliza con el lenguaje estándar *CSS (Cascade Style Sheet)* por la *W3C* que define el formato de presentación de los datos del documento *XML*
  - Se define el lenguaje estándar *XSL (XML Stylesheets Transformation Language)* en 1999 por la *W3C* que define el filtrado, orden y formato de presentación de los datos del documento *XML*
  - Complementariamente, existen multitud de herramientas en muy diversas tecnologías (bibliotecas de lenguajes, *pluggins* en clientes Web, ...) que a partir de un documento *XML* y un documento *CSS* o *XSL*, generan la estructura de la presentación definido por el documento *CSS* o *XSL* con los datos del documento *XML*

1.4.1. Justificación

- Para procesar los datos de un documento XML:
  - Se define el estándar del modelo *DOM (Document Object Model)* en 1997 por la *W3C* que define la estructura jerárquica en memoria de los datos del documento *XML* para su recuperación, generación y manipulación y acceso.
  - Interfaz estándar *SAX* (*Simple Acces XML*) define el filtrado, orden y formato de presentación de los datos del documento *XML*
  - Complementariamente, existen multitud de bibliotecas en diversos lenguajes, *pluggins* en clientes Web, ...) que a partir de un documento *XML* y una biblioteca DOM o SAX permite manipular los datos del documento XML

1.4.1. Justificación

- Ejemplo:
  - Un programa basado en *SAX* o *DOM* puede generar un documento XML a partir de una fuente de datos (base de datos, sistema de ficheros, …) en el servidor
  - Un documento *DTD* o *XML Schema* puede validar la corrección de la estructura de información del documento *XML* generado en el cliente o en el servidor
  - Un documento *XSL* puede establecer la estructura de presentación de los datos del documento anterior generando otro nuevo documento *HTML*, *XHTML*, *texto*, *PDF*, … en el cliente o en el servidor

1.4.2. Servicios Web

- Servicios Web son un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en Internet, no en Web
  - Pila de protocolos:
    - Servicio de Descubrimiento: *UDDI* (para *XML-RPC* y *SOAP*)
    - Servicio de Descripción: *WSDL* (para *XML-RPC* y *SOAP*)
    - Servicio de Mensajería: *XML-RPC*, ***SOAP***, ***REST***
    - Servicio de Transporte: *HTTP*

1.4.2. Servicios Web

- *UDDI* (*Universal Description, Discovery and Integration*): de *IBM / Microsoft / SAP* en 2000 cuyo objetivo es ser accedido por los mensajes *SOAP* y dar paso a documentos *WSDL*, en los que se describen los requisitos del protocolo y los formatos del mensaje solicitado para interactuar con los servicios Web del catálogo de registros. Cada registro tiene tres partes:
  - Páginas blancas -dirección, contacto y otros identificadores conocidos.
  - Páginas amarillas -categorización industrial basada en taxonomías.
  - Páginas verdes -información técnica sobre los servicios que aportan las propias empresas.
- WSDL (*Web Services Description Language*): de *IBM / Microsoft / Aribaen* 2000 para determinar qué funciones están disponibles en un servidor. El cliente puede usar *SOAP* para hacer la llamada a una de las funciones listadas en el WSDL

1.4.2. Servicios Web

- *XML-RPC*: por **Wineren** en 1998 es un protocolo de llamada a procedimiento remoto que usa XML para codificar los datos y HTTP como protocolo de transmisión de mensajes. Los Servicios Web basados en RPC presentan una interfaz de llamada a procedimientos y funciones distribuidas, lo cual es familiar a muchos desarrolladores. Típicamente, la unidad básica de este tipo de servicios es la operación WSDL (WSDL es un descriptor del Servicio Web, es decir, el homologo del IDL para COM).

1.4.2. Servicios Web

- *SOA*: por *Microsoft/IBM* en 2003 es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML dando completitud a XML-RPC, donde la unidad básica de comunicación es el mensaje, más que la operación. Esto es típicamente referenciado como servicios orientados a mensajes

SOA: "La Web es el transporte universal de mensajes"

vs

Rest: "La Web es el universo de la información accesible globalmente" [Berners Lee, T.]

1.4.2. Servicios Web

- *REST*: por **Fielding** en 2000. Mediante la restricción de establecer la interfaz a un conjunto conocido de operaciones (*CRUD* => *POST/GET/PATCH/DELETE* de *HTTP*) para describir sobre recursos (*URI's*) cualquier interfaz que transmite datos (XML/JSON) específicos de un domino sobre *HTTP* (sin estado) sin capa adicional como hace SOAP.
  - En la actualidad se usa en el sentido más amplio para describir cualquier interfaz web simple que utiliza *JSON/XML* y *HTTP*.
  - Los recursos no son una biyección de las entidades de la BBDD pero sí pueden coincidir.
    - Por ejemplo: un ticket puede tener un creador, una lista de receptores, un supervisor, un contenido, fechas de creación, cierre, … que en la BBDD puede implicar a entidades de varias tablas; un mensaje con fecha y contenido sí podría implicar a una sola entidad de una tabla de la BBDD

1.4.2. Servicios Web

- Ejemplo de URI's para recursos:
  - *GET /tickets- Devuelve una lista de tickets*
  - *GET /tickets/12- Devuelve un ticket específico*
  - *POST /tickets- Crea un nuevo ticket*
  - *PUT /tickets/12- Actualiza el ticket #12*
  - *PATCH /tickets/12- Actualiza parcialmente el ticket #12*
  - *DELETE /tickets/12- Elimina el ticket #12*
  - *GET /tickets/12/messages - Devuelve una lista de mensajes para el ticket #12*
  - *GET /tickets/12/messages/5 - Devuelve el mensaje #5 para el ticket #12*
  - *POST /tickets/12/messages - Crea un nuevo mensaje en el ticket #12*
  - *PUT /tickets/12/messages/5 – Actualiza el mensaje #5 para el ticket #12*
  - *PATCH /tickets/12/messages/5 - Actualiza parcialmente el mensaje #5 para el ticket #12*
  - *DELETE /tickets/12/messages/5 -Borra el mensaje #5 para el ticket #12*
  - *…*

1.4.2. Servicios Web

- Ejemplo comparativo entre XML y JSON

```
<menu id="file" value="File">
   <popup>
      <menuitem value="New" onclick="CreateNewDoc()" />
      <menuitem value="Open" onclick="OpenDoc()" />
      <menuitem value="Close" onclick="CloseDoc()" />
    </popup>
</menu>
```

```
{"menu": {
    "id": "file",
    "value": "File",
    "popup": {
       "menuitem": [
          {"value": "New", "onclick": "CreateNewDoc()"},
          {"value": "Open", "onclick": "OpenDoc()"},
          {"value": "Close", "onclick": "CloseDoc()"}
       ]
    }
}}
```

1.4.2. Servicios Web

- Consecuencias:
  - Aplicaciones de una sola página web
  - El servidor suministra datos *XML/JSON* sin formato en *HTML* aliviando la carga de la pieza crítica en la arquitectura cliente/servidor de *HTTP*
  - Los clientes, potentes PC's disfrutando de buenas comunicaciones, asumen la modificación de los elementos del *DOM* en la presentación del *HTML* para formatear los datos recibidos *XML/JSON*
  - Se acomoda a la arquitectura MVC:
    - Modelos son las entidades de la *BBDD*
    - Vistas son la representación del recurso en *XML/JSON*
    - Controladores son los que implementan las operaciones sobre los recursos, a partir de los modelos y devolviendo las vistas

1.4.3. Redifusión

- Objetivo: difundir información actualizada frecuentemente a usuarios que se han suscrito a la fuente web de contenidos
- Elementos:
  - Fuente web (*web feed*) es un medio de redifusión de contenido web. Los interesados pueden usar un programa agregador para acceder a sus fuentes suscritas desde un mismo lugar.
  - Un agregador un tipo de software para suscribirse a fuentes de noticias y reúne los contenidos publicados en los sitios con redifusión web elegidos y muestra las modificaciones que se han producido en esas fuentes web (*Google Reader, Bloglines, My Yahoo! Netvibes, Menéame, …*)
- Formatos basados en XML (RDF):
  - *RSS* (*Really Simple Syndication*) de *Netscape* en 1999
  - *Atom* de *IETF* en 2005

1.5. Dinamismo en el Cliente

1. Clientes Web Ligeros
2. Clientes Web Pesados
3. Clientes Móviles

1.5.1. Clientes Ligeros

- *CSS (cascading style sheet)* por *W3C* en 1995. Hojas de estilo en cascada para separar "en parte" la estructura de la presentación del diseño
- *JavaScript* (*LifeScript*) de *Netscape* en 1995 orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico que se ejecuta en el cliente:
  - Interactúa con el usuario por eventos, con el contenido HTML mediante *DOM (document object model)*, con el navegador mediante *BOM (browser object model)* y con sesiones pasadas mediante *Cookies*: fragmento de información guardada en disco del cliente.
- *VBScript* de *Microsoft* en 1996 es la contrapartida sin éxito en la web. Actualmente. *ECMAScript* es el estándar de *W3C* en 1996 pero en Microsoft es *JScript*, como implantación propietaria.

1.5.1. Clientes Ligeros

- *AJAX (Asynchronous JavaScript And XML)* en 2005 permite la carga asíncrona de contenidos en una página existente sin requerir recarga completa con una combinación de tecnologías preexistentes:
  - *XHTML/HTML* y *CSS*
  - *DOM* con *ECMAScript* en *JavaScript* y *Jscript*
  - Peticiones asíncronas con *XMLHttpRequest*
  - Intercambio de información con *XML*, *JSON* o texto plano :
    - un formato ligero para el intercambio de datos
    - un subconjunto de la notación literal de objetos de JavaScript
- *JQuery* por **Resig, J**. en 2006 para simplificar la manera de interactuar con los documentos *HTML*, manipular el árbol *DOM*, manejar eventos, desarrollar animaciones y agregar interacción con la técnica *AJAX* a páginas web

1.5.2. Clientes Pesados

- Componentes (applets de PC Magazine en 1990) se ejecutan dentro del navegador pero no lo controlan. Proveen características de interacción que no puede realizar HTML.
  - Tecnologías:
    - *Applets* de *Java* en 1995. Precursor de *Java Web Star* para descargar aplicaciones de escritorio desde la *Web*.
    - *Active X* de *Microsoft* en 1996. Sucesor de la tecnología *OLE (object Linking and Embedding)*

1.5.2. Clientes Pesados

- *RIA (Rich Internet Applications)*: Son aplicaciones web que tienen la mayoría de las características de las aplicaciones de escritorio tradicionales y, así, mejorar la experiencia del usuario
  - Utilizan un navegador web estandarizado para ejecutarse y por medio de complementos o mediante una máquina virtual se agregan las características adicionales.
  - *Flash* de *Macromedia* (*Adobe System*) en 1996 para creación y manipulación de gráficos vectorial es con posibilidades de manejo de código mediante el lenguaje ActionScript en forma de estudio de animación que trabaja sobre "fotogramas" y está destinado a la producción y entrega de contenido interactivo
  - *Flex* de *Adobe System* en 2004 minimiza elegantemente el problema de adaptar la metáfora de la animación proveyendo un flujo de trabajo y un modelo de programación que es familiar a los desarrolladores de aplicaciones
  - *SylverLight* de *Microsoft* en 2007
  - *Java FX* de *JavaONE* en 2007

1.5.3. Clientes Móviles

- Desarrollo nativo:
  - *BlackBerry* en 1999 de *BlackBerry* con *Java*
  - *IOS* en 2004 de *Apple* con *Objetive-C*
  - *Android* en 2007 de *Android inc.* y despúes de *Google* con *Java*
  - *Windows Phone* en 2010 de *Microsoft* con *VB.NET* y *C#.NET*, *C++* y *XAML*
- Desarrollo independiente de la plataforma con estándares Web:
  - *Bootstrap* por **Otto, M**. y **Thornton, J**. de Twiter en 2011 is the most popular *HTML*, *CSS*, and *JavaScript* framework for developing responsive, mobile-first web sites.
  - *JQuery Mobile* evolución *responsive* de *Jquery*
  - *Cordova/PhoneGap* de **Nitobi** en *Apache/Adobe* con acceso a servicios de móvil (cámar, …)

1.6. Arquitecturas Web

1. Arquitectura MVC Física
2. Arquitectura MVC Lógica en el Servidor
3. Arquitectura MVC Lógica en el Cliente

1.6.1. Arquitectura MVC Física

- Arquitectura 3 capas:
  - Componentes:
    - Vista: Navegador Web / Capa cliente
    - Controlador: Servidor Web / Capa de lógica de negocio
    - Modelo: Servidor de BBDD / Capa de datos-persistencia
  - Plataformas:
    - *LAMP*: *Linux/Apache/MySQL/PHP-Perl-Python*
    - *Java 2*: */*/JDBC/JSP*
    - *Microsoft: Window/IIES/SQL Server/ASP*
  - Software (*PHP, JSP, ASP*) no es MVC porque cada script de generación de páginas web entrevera tanto la vista (*HTML*), el controlador de la lógica de negocio (*Java, Basic, PHP*) y el acceso a datos (*SQL*).

1.6.2. Arquitectura MVC Lógica en el Servidor

- MVC con servidor de aplicaciones:
  - *.NET* de *Microsoft* en 2000 mantiene los ASP e incorpora COM para los controladores en lenguajes VBasico C# con la *Base Class Library* con multitud de servicios para GUI, acceso a datos (*ADO.NET*), … todo sobre *Common Language Runtime(CLR)*
  - *JavaEE* de *Sun Microsystemen* 2001 mantiene los *JSP*, *servlets* y *applet* e incorpora *EJB (Enterprise Java Bean)* en un servidor de aplicaciones para los controladores (*SessionBeans*) y para los modelos (*EntityBean*) junto con multitud de servicios de seguridad, transacciones, correo, … todo sobre la máquina virtual (MV)
    - Servidores de aplicaciones:
      - *WebSphere* de *IBM* en 1998
      - *WebLogic* de *BEA WebLogic* (*Oracle*) en 1997
      - *Jbosss* de *Red Haten* software libre
      - *Glassfish* de *Sun Microsystem (Oracle)* en 2005

1.6.2. Arquitectura MVC Lógica en el Servidor

- MVC sin servidor de aplicaciones:
  - *Struts* de **McClanahan** de *Apache* en 2000 con *Java*
  - *Spring* de **Johnson, R.** con licencia de Apache en 2002 independiente de la plataforma (*Java2* o *.NET*)
  - *Java Server Face (JSF)* de cógigo abierto en 2004 con *Java*
  - *Ruby on Rails* de **Hansson** de código abierto en 2004 con *Ruby*
  - *Cake* de la comunidad de código abierto en 2005 con *PHP*
  - *Symphony* de la comunidad de código abierto en 2005 con *PHP*
  - *node js* de **Dahl, R**. de *Joyent* en 2009 con *JavaScript*

1.6.4. Arquitectura MVC Lógica en el Cliente

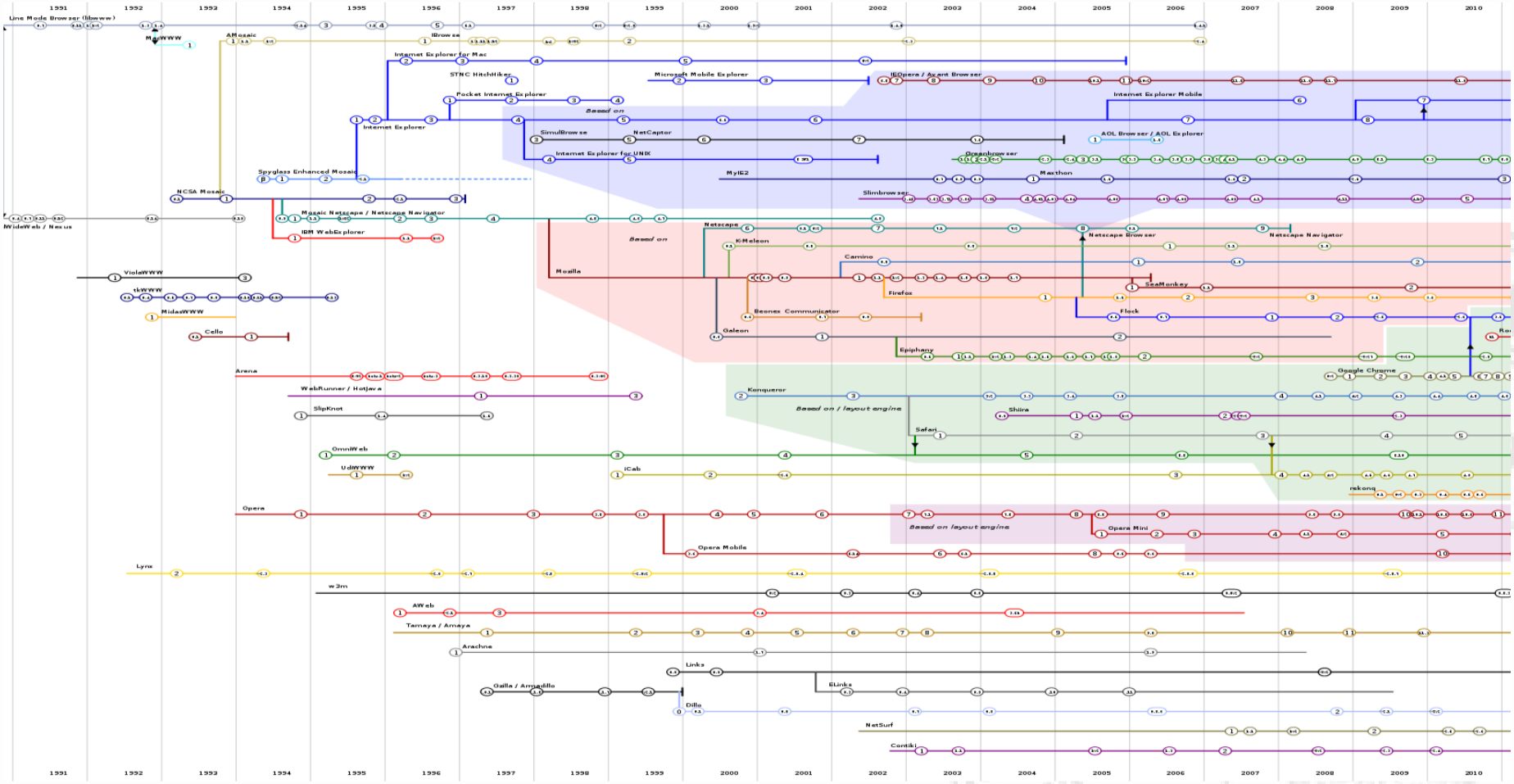
- Basados en estándares *Web*: librerías para el desarrollo de sitios web, principalmente usada para aplicaciones web con bastante interacción con el cliente, donde se hace un uso intensivo de Javascript, Ajax, etc. Te permite desarrollar en Javascript atendiendo a patrones, con una variante del paradigma MVC y facilitan las pruebas
  - *Angular.js* en 2009 de *Google*
  - *Backbone.js* de **Ashkenas, J.** de *opensource* en 2010
  - *Ember.js* de **Katz** del *MIT* en 2011

1.7. Herramientas de Desarrollo Web

1. Clientes Web
2. Servidores Web
3. Entornos Integrados de Desarrollo
4. Editores de Lenguajes de Marcado
5. Gestores de Contenidos

1.7.1. Clientes Web

- Desde la aparición de la Web se han desarrollado multitud de Navegadores Web



1.7.1. Clientes Web

- *Nexus* desarrollado por el *CERN* entregado en 1990 disponible sobre *NeXT STEP*. Primer navegador Web y originalmente llamado World Wide Web con funcionalidades muy avanzadas (WYSIWYG, hojas de estilo, postcript, news, video, …).
- *ViolaWWW* desarrollado por la *Universidad de California* entregado en 1992 disponible sobre *XWindows* de *Unix*. Incorpora los conceptos precursores del lenguaje de script para dar dinamizar las páginas y embeber objetos y programas.
- *Mosaic* desarrollado por *NCSA* en 1992. Primer navegador disponible sobre plataformas *Windows*, *Macintosh* y *Unix*.

1.7.1. Clientes Web

- *Navigator* desarrollado por *Netscape* entregado en 1994, basado en *Mosaic*. Primer navegador comercial que incorpora *Java Script* y nuevas etiquetas que no respetaban el estándar de *HTML*. Desde 1997 está liberado como software libre.
- *Opera* desarrollado por *Opera Software* entregado en 1994. Gratuito desde el 2000. Para plataformas *Windows, Mac OS X, GNU/Linux* y móviles *BlackBerry, Symbian, Windows Mobile, Androido iOS*. Incorporó la navegación por pestañas, corrector ortográfico, bloqueo de ventanas emergentes (phishing) y otras funcionalidades asumidas posteriormente por otros.
- *Internet Explorer* desarrollado por *Microsoft* entregado en 1995, basado en *Spyglass* que a su vez se basó en *Mosaic*. Forma parte de los sistemas operativos *Windows* de pago. Sobre *Windows* y *MacOS*

1.7.1. Clientes Web

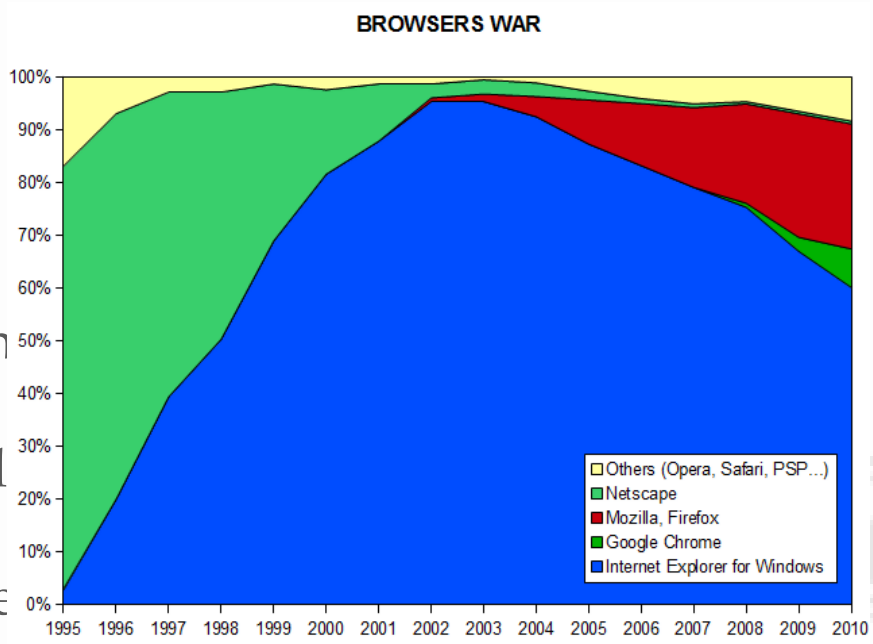
- *Mozilla Firefox* desarrollado por la *Fundación Mozilla* entregado en 2002, basado en *Mozilla* que a su vez se basó en la liberación de *Navigator*. Disponible sobre *Windows, Linux y MacOS*. Liberado como software libre de *Mozilla* y *GNU*.
- *Safari* desarrollado por *Apple* entregado en 2003. Disponible sobre *MacOS, iOS(móviles)* y *Windows*.
- *Chrome* desarrollado por *Google* entregado en 2008, basado en el proyecto *Chromium* de software libre. Disponible sobre *Windows*, *Linux* y *MacOS*. Incorpora geolocalización, búsqueda instantanea, …

1.7.1. Clientes Web

- Existen otros muchos navegadores Web:
  - *Flock*, *Midori*, *Arora*, que comparten con *Safari* y *Chrome* el motor de visualización (render) *WebKit* basado en el motor *KHTML* de navegador *Konqueror*.
  - *AOL Explorer*, … que comparte el motor *Trident* con *IExplorer*.
  - *Camino*, *Galeon*, … que comparten el motor *Gecko* con *Mozilla Firefox*.

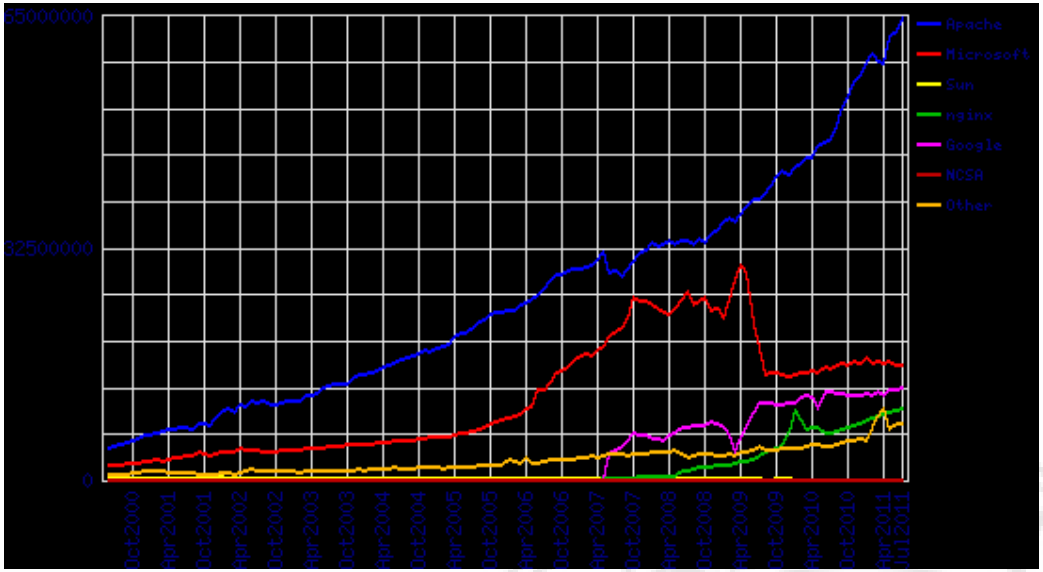
1.7.1. Clientes Web

- La lucha por el mercado ha desencadenado dos famosas batallas:
  - I Batalla entre *Navigator* de *Netscape* e *IExplorer* de *Microsoft* en los 90'. Inicialmente ganada por *Navigator* y con resolución judicial a favor de *Netscape* por carácter monopolista de *Microsoft* pero éste finalmente surgió ganador en el mercado.
  - II Batalla actual desde el 2003 entre los "3 grandes", *IExplorer*, *Firefox*, *Chrome*, o los "5 grandes", incluyendo *Safari* y *Opera*. Domina *Iexplorer* pero cada vez con menos distancia.



1.7.2. Servidores Web

- *CERN httpd* del *CERN* en 1990
- *Netscape Enterprise Server* de *Netscape* en 1994. Precursor de *SunONE Web Server, iPlanetWeb Server* y actualmente *Oracle Iplantet Web Server*
- *Internet Information Services* de *Microsoft* en 1995.
- *Apache HTTP Server* de *Apache Foundation* en 1996



Totals for Active Sites Across All Domains.June 2000 -July 2011. NetCraft

1.7.3. Entornos de Desarrollo Integrado

- Aplicación que ofrece facilidades a los programadores de desarrollo de aplciacionesincluyendo editores, compiladores, depuradores, …Casi todos son multilenguaje y extensibles con *pluggins*:
  - *VisualAge* de *IBM* en 1980.
  - *VisualStudio* de *Microsoft* en 1995.
  - *Netbeans* de *Sun Microsystems* 1999.
  - *Eclipse* de *IBM* en 2004 y actualmente *opensource*.

1.7.4. Editores de Lenguajes de Marcado

- Construcción y edición de sitios Web incluyendo HTML, xHTML, CSS, javascript, …:
  - Modo texto: *UltraEdit*, *Notepad*++, …
  - Modo gráfico:
    - *Netscape Composer* de *Netscape* en 1997.
    - *DreamWeaver* de *Macromedia* en 1997 y después de *Adobe System* basándose en *Adobe Flash*
    - *FrontPage* de *Microsoft* en 1997. Actualmente *Word Expression* desde 2006

1.7.5. Gestores de Contenidos

- *CMS (content managmentsystem)*. Sistema que permite crear una estructura de soporte (*framework*) para la creación y administración de contenidos, principalmente en páginas web, por parte de los administradores, editores, participantes y demás roles.
  - Consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio web.
  - Permite manejar de manera independiente el contenido y el diseño.
  - Para portales, wikis, groupware, blogs, …

1.7.5. Gestores de Contenidos

- Gestores de contenidos genralistas:
  - *Drupal* de *Buytearten* 2001. Desarrollo con *PHP* sobre *MySQL* bajo licencia GPL
  - *WordPress* de **Mullenweg, M.** en 2005 con PHP, MySQL y Apache bajo licencia GPL
  - *Joomla*! de *Open Source Mattersen* 2005. Desarrollo con *PHP* sobre *MySQL* bajo licencia GPL
- Gestores para redes sociales:
  - *Elgg* por **Elggy Toshen** 2004.Desarrollo con PHP sobre MySQLbajo licencia GPL
- Gestores para plataformas de enseñanza:
  - *Moodle* por *Dougiamasen* 2006. Desarrollo con *PHP* sobre *MySQL* o *PostgreSQL* bajo licencia GPLv2