CAPÍTULO III APLICACIÓN WEB

3.1. Introducción

Con la aparición de Internet y de la web en concreto, se han abierto infinidad de posibilidades en cuanto al acceso a la información desde casi cualquier sitio. Este sistema de información es conocido como *World Wide Web* (WWW).

La web en sus orígenes fue pensada como un medio para desplegar información, ésta se encuentra contenida en servidores, denominados servidores web. La manera de acceder a las páginas web es a través de un navegador o *browser*, el cual realiza peticiones valiéndose del protocolo HTTP (*HyperText Transfer Protocol*). La dirección que localiza la información dentro de Internet se denomina URL: es el Localizador Uniforme de Recursos (*Uniform Resource Locator*).

Las características de la web son las siguientes:

- Global: Se puede acceder a él desde cualquier tipo de plataforma, usando cualquier navegador y desde cualquier parte del mundo.

- Pública: Toda su información está distribuida en miles de ordenadores que ofrecen su espacio para almacenarla. Esta información es pública y toda puede ser obtenida por el usuario.

- Dinámica: La información, aunque esta almacenada, puede ser actualizada por quién la publico sin que el usuario deba actualizar su soporte técnico.

La facilidad de comunicación que proporciona Internet conjuntada con la necesidad de acceso remoto a aplicaciones sin necesidad de instalaciones en la máquina del usuario ha hecho evolucionar este concepto. La comunicación ya no se basa simplemente en la carga de una página estática, sino que ésta puede ser el resultado de la ejecución en el servidor de alguna lógica de programación, es decir, interacción dinámica entre usuario y servidor.

Esto representa un desafío a los desarrolladores de aplicaciones, ya que los avances en tecnología demandan cada vez aplicaciones más rápidas, ligeras y robustas que permitan utilizar la web.

3.2. Tecnologías para el desarrollo de aplicaciones web

Para el desarrollo de aplicaciones web se han generado múltiples tecnologías entre las que se encuentran:

- **CGI**. *Common Gateway Interface* fue la primera técnica utilizada para que el contenido de las páginas web se generará de manera dinámica, es común encontrar en los diferentes servidores web el modulo que soporta la ejecución de CGIs. De manera resumida se puede decir que el CGI es un mecanismo de

comunicación entre el servidor web y una aplicación externa, esta aplicación puede estar desarrollada en casi cualquier lenguaje, este solo debe cumplir la condición de ser soportado por el servidor http, es común encontrar que la mayoría de las aplicaciones CGIs se encuentren desarrolladas con el lenguaje PERL.

Este mecanismo tiene deficiencias que evita su uso a gran escala, la más conocida es en cuanto a rendimiento, ya que por cada petición que se realice en el servidor se crea un nuevo proceso, lo cual tiene un costo muy alto en lo que a recursos del sistema se refiere.

- **Fast-CGI.** Esta es una solución similar al CGI mencionado anteriormente, solo que propone la creación de un solo proceso persistente por cada programa FastCGI en lugar de por cada solicitud del cliente. Es una solución viable pero también tiene inconvenientes de proliferación de procesos en el caso de peticiones concurrentes.

- **Paginas dinámicas en servidor.** Con la aparición de esta tecnología se entra a una nueva forma de trabajo, la cual esta orientada al trabajo del diseñador web, quien no necesariamente conoce de lenguajes de programación. Este nuevo enfoque consiste en insertar pequeños fragmentos de lógica de programación en la estructura HTML de la página, al contrario de lo que se hacia en los CGIs, que era en el lenguaje de programación utilizar sentencias de impresión para generar salidas HTML. En este sentido se conocen diferentes alternativas, entre ellas mencionar PHP, ASP, JSP, entre otros.

- **Java**. Java es un lenguaje de programación orientado a objetos desarrollado por la compañía Sun Microsystems. Está construido a partir de lenguajes orientados a objetos anteriores, como C++, pero no pretende ser compatible con ellos sino ir mucho más lejos, añadiendo nuevas características como recolección de basura, programación multihilos y manejo de memoria a cargo del lenguaje.

- **Java DataBase Connectivity.** JDBC es una interfaz que provee comunicación con bases de datos. Consiste en un conjunto de clases e interfaces escritas en Java, que proveen una API (Interfaz de Programación de Aplicación) estándar para desarrolladores de herramientas de base de datos, permitiendo independizar la aplicación de la base de datos que utiliza.

  La API JDBC es la interfaz natural a las abstracciones y conceptos básicos de SQL (Lenguaje de Consultas Simple): permite crear conexiones, ejecutar sentencias SQL y manipular los resultados obtenidos.

- **Servlets.** El *servlet* se puede considerar como una evolución de los CGIs desarrollada por SUN Microsystems como parte de la tecnología Java. Son programas Java que proveen la funcionalidad de generar dinámicamente contenidos web.

  A diferencia de los *applets*, no poseen restricciones en cuanto a seguridad. Tienen las propiedades de cualquier aplicación Java y pueden acceder a los

archivos del servidor para escribir y leer, cargar clases, cambiar propiedades del sistema, etc.

Del mismo modo que las aplicaciones de programas Java, los *servlets* están restringidos por los permisos del sistema. De forma general consiste en la ejecución de aplicaciones Java en el motor de *servlets* (*Servlet engine*) el cual hace parte del servidor web, algo que lo hace ventajoso con respecto a los CGIs es que por cada petición de usuario no se crea un proceso sino un hilo, el cual es mucho mas económico para el sistema.

Esta tecnología hace parte de la arquitectura propuesta por SUN en su plataforma J2EE (*Java 2 Enterprise Edition*).

.

- **Java Server Pages.** JSP provee a los desarrolladores de web de un entorno de desarrollo para crear contenidos dinámicos en el servidor usando plantillas HTML y XML (*eXtensible Markup Language*) en código Java, encapsulando la lógica que genera el contenido de las páginas.

  Cuando se ejecuta una página JSP es traducida a una clase de Java, la cual es compilada para obtener un *servlet*. Esta fase de traducción y compilación ocurre solamente cuando el archivo JSP es llamado la primera vez, o después de que ocurran cambios.

- **eXtensible Markup Language** La familia XML es un conjunto de especificaciones que conforman el estándar que define las características de un mecanismo independiente de plataformas desarrollado para compartir datos. Se puede considerar a XML como un formato de transferencia de datos multi-plataforma.

  XML ha sido diseñado de tal manera que sea fácil de implementar. No ha nacido sólo para su aplicación en Internet, sino que se propone como lenguaje de bajo nivel (a nivel de aplicación, no de programación) para intercambio de información estructurada entre diferentes plataformas.

  XML hace uso de etiquetas (únicamente para delimitar datos) y atributos, y deja la interpretación de los datos a la aplicación que los utiliza. Por esta razón se van formando lenguajes a partir del XML, y desde este punto de vista XML es un metalenguaje.

  El conjunto de reglas o convenciones que impone la especificación XML permite diseñar formatos de texto para los datos estructurados, haciendo que se almacenen de manera no ambigua, independiente de la plataforma y que en el momento de la recuperación se pueda verificar si la estructura es la correcta.

  Para comprobar que los documentos estén bien formados se utiliza un DTD (*Document Type Definition*). Se trata de una definición de los elementos que pueden incluirse en el documento XML, la relación entre ellos, sus atributos, posibles valores, etc. Es una definición de la gramática del documento, es decir, cuando se procesa cualquier información formateada mediante XML, el primer

paso es comprobar si está bien formada, y luego, si incluye o referencia a un DTD, comprobar que sigue sus reglas gramaticales.

- **eXtensible Stylesheet Language.** XSL es una especificación desarrollada para aplicar formato a los documentos XML de forma estandarizada. Aunque se ha establecido un modo para que puedan usarse hojas de estilo CSS (Hojas de Estilo en Cascada) dentro de documentos XML, es lógico pensar que para aprovechar las características del nuevo lenguaje hace falta tener un estándar paralelo y similar asociado a él.

  La XSL permite añadir lógica de procesamiento a la hoja de estilo. La idea es asociar al documento XML con una hoja de estilo y a partir de esto visualizar el documento XML en cualquier plataforma: PalmPC, PC, Internet Explorer, Netscape, etc. y con el aspecto (colores, fuentes, etc.) que se quiera utilizar.

- **Applets de Java.** Un *applet* es un componente de software que corre en el contexto de otro programa, por ejemplo un navegador web. El *applet* debe correr en un contenedor, que es proporcionado por un programa anfitrión, mediante un plugin o en aplicaciones como teléfonos celulares que soportan el modelo de programación por *applets*.

  A diferencia de un programa, un *applet* no puede correr de manera independiente, ofrece información gráfica y a veces interactúa con el usuario, típicamente carece de sesión y tiene privilegios de seguridad restringidos. Un *applet* normalmente lleva a cabo una función muy específica que carece de uso independiente.

3.3. Arquitectura web

La idea fundamental es que los navegadores, *browsers*, presentan documentos escritos en HTML, que han obtenido de un servidor web. Estos documentos HTML habitualmente presentan información de forma estática, sin más posibilidad de interacción con ellos.

El modo de crear los documentos HTML ha variado a lo largo de la corta vida de las tecnologías web pasando desde las primeras páginas escritas en HTML almacenadas en un fichero en el servidor web hasta aquellas que se generan al vuelo como respuesta a una acción del cliente y cuyo contenido varía según las circunstancias.

Así mismo, el modo de generar páginas dinámicas ha evolucionado, desde la utilización del CGI, hasta los *servlets* pasando por tecnologías tipo JSP. Todas estas tecnologías se encuadran dentro de aquellas conocidas como *Server Side*, ya que se ejecutan en el servidor web.

Otro aspecto que completa el panorama son las inclusiones del lado del cliente, *Client Side*, que se refieren a las posibilidades de que las páginas lleven incrustado código que se ejecuta en el cliente, como por ejemplo *JavaScript* y programas Java (*Applets*).

El esquema general de la situación se puede ver en la figura 3.1, donde se muestran cada tipo de tecnología involucrada en la generación e interacción de documentos web.

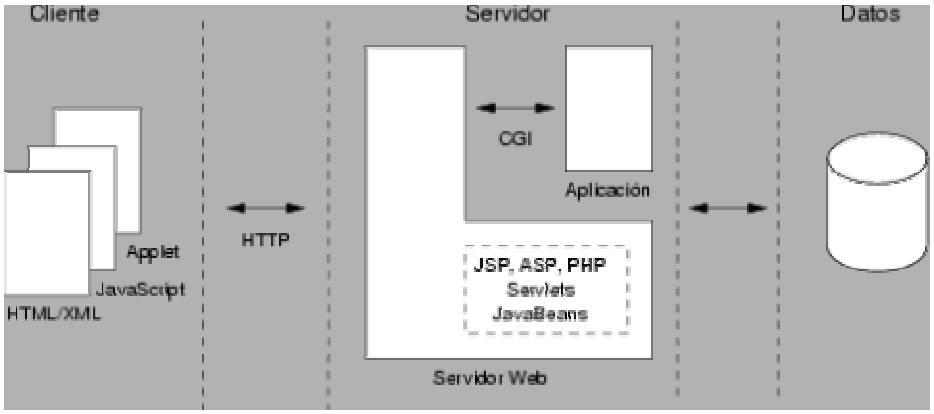

Figura 3.1**.** Esquema general de las tecnologías web

3.3.1. Navegador web o Browser

El navegador puede considerarse como una interfaz de usuario universal. Dentro de sus funciones están la petición de las páginas web, la representación adecuada de sus contenidos y la gestión de los posibles errores que se puedan producir.

Para poder cumplir con todas estas funciones, los navegadores tienen la posibilidad de ejecución de programas de tipo *script*, con modelos de objetos que permiten manipular los contenidos de los documentos. Estos lenguajes de programación son *VBScript*, *JScript* (ambas de Microsoft) y *JavaScript* (de Netscape), y proporcionan las soluciones llamadas del lado del cliente, *client side* y permiten realizar validaciones de datos recogidos en las páginas antes de enviarlos al servidor proporcionando un alto grado de interacción con el usuario dentro del documento.

Otras de las posibilidades de los navegadores es la gestión del llamado HTML dinámico (DHTML). Éste está compuesto de HTML, hojas de estilo en cascada, (*Cascade Style Sheets*, CSS), modelo de objetos y *scripts* de programación que permiten formatear y posicionar correctamente los distintos elementos HTML de las páginas web, permitiendo un mayor control sobre la visualización de las páginas.

En esta línea, los navegadores han ido un poco más allá y permiten la visualización de documentos XML después de haber sido transformado adecuadamente a HTML por las hojas de estilo extensibles XSL. De esta manera se puede elegir visualizar ciertos elementos y otros no, dependiendo de las circunstancias.

Además, los navegadores permiten la ejecución de aplicaciones dentro de los documentos mostrados. Las dos posibilidades más populares son la tecnología *ActiveX* y los *applets* Java. Los *applets* Java son pequeños programas que se descargan del servidor web y se ejecutan en la JVM (*Java Virtual Machine)* del navegador.

3.3.2. Servidor web

El servidor web es un programa que corre sobre el servidor que escucha las peticiones HTTP que le llegan y las satisface. Dependiendo del tipo de la petición, el servidor web buscará una página web o bien ejecutará un programa en el servidor. De cualquier modo, siempre devolverá algún tipo de resultado HTML al cliente o navegador que realizó la petición.

El servidor web es fundamental en el desarrollo de las aplicaciones del lado del servidor, *server side applications*.

3.3.3. Aplicaciones Multinivel

Al hablar del desarrollo de aplicaciones web resulta adecuado presentarlas dentro de las aplicaciones multinivel. Los sistemas típicos cliente/servidor pertenecen a la categoría de las aplicaciones de dos niveles. La aplicación reside en el cliente mientras que la base de datos se encuentra en el servidor. En este tipo de aplicaciones el peso del cálculo recae en el cliente, mientras que el servidor hace la parte menos pesada, y eso que los clientes suelen ser máquinas menos potentes que los servidores. Además, está el problema de la actualización y el mantenimiento de las aplicaciones, ya que las modificaciones a la misma han de ser trasladada a todos los clientes.

Para solucionar estos problemas se ha desarrollado el concepto de arquitecturas de tres niveles: interfaz de presentación, lógica de la aplicación y los datos.

La capa intermedia es el código que el usuario invoca para recuperar los datos deseados. La capa de presentación recibe los datos y los formatea para mostrarlos adecuadamente. Esta división entre la capa de presentación y la de la lógica permite una gran flexibilidad a la hora de construir aplicaciones, ya que se pueden tener múltiples interfaces sin cambiar la lógica de la aplicación.

La tercera capa consiste en los datos que gestiona la aplicación. Estos datos pueden ser cualquier fuente de información como una base de datos o documentos XML.

Convertir un sistema de tres niveles a otro multinivel es fácil ya que consiste en extender la capa intermedia permitiendo que convivan múltiples aplicaciones en lugar de una sola (figura 3.2).

La arquitectura de las aplicaciones web suelen presentar un esquema de tres niveles (figura 3.3). El primer nivel consiste en la capa de presentación que incluye no sólo el navegador, sino también el servidor web que es el responsable de dar a los datos un formato adecuado. El segundo nivel está referido habitualmente a algún tipo de programa o *script*. Finalmente, el tercer nivel proporciona al segundo los datos necesarios para su ejecución.

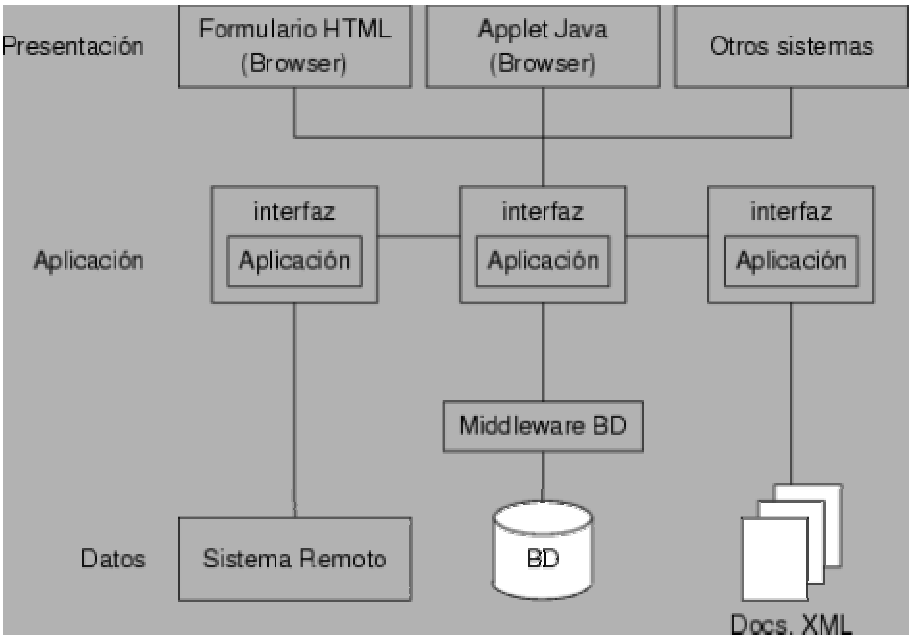

Figura 3.2. Arquitectura Multinivel

Una aplicación web típica, recogerá datos del usuario (primer nivel), los enviará al servidor, que ejecutará un programa (segundo y tercer nivel) y cuyo resultado será formateado y presentado al usuario en el navegador (primer nivel otra vez).

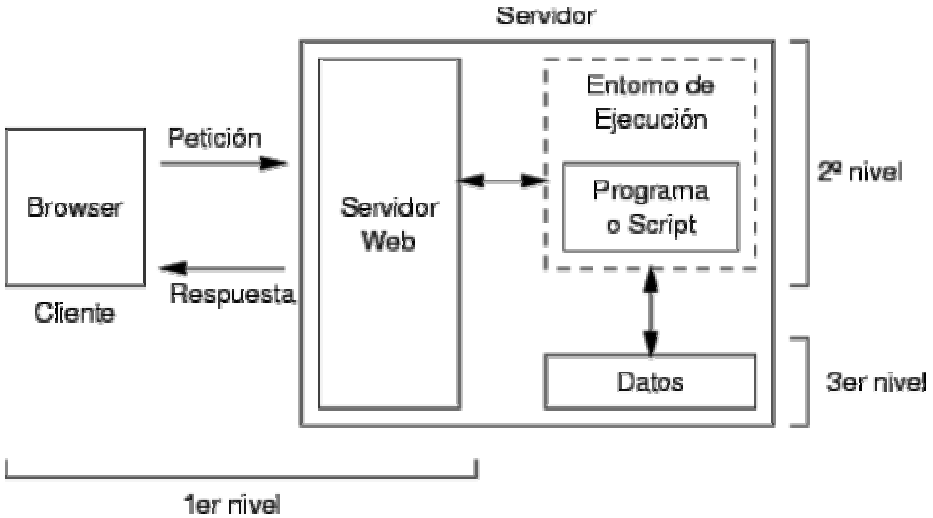

Figura 3.3. Arquitectura web de tres niveles

El viejo CGI ha cumplido con el propósito de añadir interactividad a las páginas web pero sus deficiencias en el desarrollo de aplicaciones y en la escalabilidad de las mismas ha conducido al desarrollo de APIs específicos de servidor como *Active Server Pages*, ASP, y PHP, que son más eficientes que su predecesor CGI.

Para aprovechar el potencial de estas tecnologías y ofertar una solución de servidor más extensible y portable, Sun ha desarrollado la tecnología llamada *servlet*. Los *servlets* Java son muy eficientes, debido al esquema de *threads* en el que se basan y al uso de una arquitectura estándar como la JVM.

Otra nueva tecnología viene a sumarse a las que extienden la funcionalidad de los servidores web, llamada *JavaServer Pages*, JSP. Los JSP permiten juntar HTML, aplicaciones Java, y componentes como las *JavaBeans* creando una página web especial que el servidor web compila dinámicamente en un *servlet* la primera vez que es llamada.

3.4. Introducción al lenguaje Java

El lenguaje Java fue desarrollado en 1991 por un grupo de ingenieros de Sun Microsystems con el fin de desarrollar software para el control de pequeños dispositivos electrónicos (TV interactiva, microondas, tostadora, etc.), lo que llevo a desarrollar un lenguaje sencillo capaz de generar código de reducido tamaño.

Debido a la existencia de multitud de tipos de electrodomésticos y a continuos cambios en los mismos, era necesaria una herramienta que no dependiera del tipo de electrodoméstico utilizado. Se desarrolló por tanto un código "neutro", independiente del tipo de electrodoméstico, el cual se ejecutaba sobre una "máquina virtual" denominada *Java Virtual Machine* (JVM), la cual interpretaba el código neutro convirtiéndolo en código particular del electrodoméstico.

Este lenguaje no fue acogido por las empresas de electrodomésticos, siendo en 1995 cuando de la mano de Netscape, Java se introdujo como lenguaje de programación para ordenadores. La incorporación de un intérprete Java en la versión 2.0 de Netscape Navigator supuso una gran revolución en Internet.

Java fue diseñado para que la ejecución de código a través de la red fuera segura, para lo cual fue necesario deshacerse de herramientas de los lenguajes anteriores como C tales como los punteros. También se han eliminado aspectos que demostraron ser mejores en la teoría que en la práctica, tales como sobrecarga de operadores, que todavía está en discusión, y herencia múltiple.

La programación en Java se basa en un gran número de clases preexistentes, algunas comerciales, otras hechas por el propio usuario, pero todas forman parte del propio lenguaje (API *Application Programming Interface de Java).*

La ejecución de programas desarrollados en Java tiene muchas posibilidades: ejecución como aplicación independiente (*Stand-alone Application*), ejecución como *applet*, ejecución como *servlet*, etc. La ejecución como aplicación independiente es análoga a los programas desarrollados en otros lenguajes (C, C++, etc.). Un *applet* es una aplicación que se ejecuta dentro de un navegador o browser (*Internet explorer o Netscape Navigator)* al cargar una página HTML desde un servidor web, por tanto se descarga del servidor no necesitando instalación. Un *servlet* es una aplicación sin interfaz grafica que se ejecuta en un servidor de Internet.

Java presenta las siguientes características:

- Java es pequeño: Los programas son rápidos de descargar desde una página web.
- Java es seguro: Evita programas que dañen a los computadores.
- Java es portable: Permite ser ejecutado en Windows, Macintosh y otras plataformas sin modificación alguna.

Como consecuencia de estas características, de haber sido desarrollado recientemente, y por un único equipo, muchos expertos opinan que Java es el lenguaje ideal para aprender la informática moderna.

Las versiones de Java van desde *Java1.0*, *Java1.1* que introdujo sustanciales mejoras, *Java 1.2* bautizada como *Java2* y actualmente existen versiones de *Java1.4.*

3.5. Entorno de desarrollo Java

Para desarrollar código Java se requiere algún paquete de programación Java. La compañía Sun Microsystems, creadora de Java distribuye gratuitamente el *Java(tm) Development Kit* (JDK), o llamado *Standard Development Kit* (SDK). Se trata de un conjunto de programas y librerías que permiten desarrollar, compilar y ejecutar programas en Java.

Es posible descargar la última versión del kit de desarrollo Java visitando el sitio oficial de Sun en: http://java.sun.com

Hay diversas plataformas sobre las que correr programas Java:

- J2EE (*Java2 Enterprise Edition*) especialmente pensada para crear aplicaciones web.
- J2SE (*Java 2 Standard Edition*) es el entorno de desarrollo de aplicaciones Java orientado a las aplicaciones solitarias y los *applets*.
- JRE (*Java Runtime Environment*), versión reducida del JDK, destinada únicamente a ejecutar código Java, no permitiendo compilar.

Java WSDP (*Web Services Developer Pack*) es un conjunto de herramientas integradas que permite a los desarrolladores de la plataforma Java desarrollar, probar y desplegar aplicaciones XML, aplicaciones web y servicios web, proporcionando implantaciones Java estándar de servicios web estándar como WSDL (*Web Services Definition Language*), SOAP (*Simple Object Access Protocol*), ebXML (*Electronic Business using eXtensible Markup Language*) y UDDI (Universal *Description Discovery and Integration*).

El JDK/SDK incorpora una herramienta para generar el código para compilar: genera una clase (*.*class*) a partir de código java (*.java) mediante *javac*.exe, y detectar

errores basada en la utilización de una consola (ventana de comandos de *MSDOS*) bastante pesada de utilizar. Por tanto el JDK al ejecutar programas en Java que no ofrece un ambiente de trabajo óptimo para proyectos complejos, es decir para compilar una o dos clases quizás el comando *javac* ofrecido en los JDK es suficiente, pero si el proyecto está compuesto por 100 o 200 clases, *javac* sería muy deficiente. La opción más adecuada para proyectos complejos es a través de los denominados entornos integrados.

Otra herramienta incluida en el JDK es *Appletviewer* que permite ver el comportamiento de *applets* sin necesidad de la utilización de un navegador.

Los IDE (*Integrated Development Environment)*, son entornos de desarrollo integrados, en un mismo programa es posible escribir el código Java, compilarlo y ejecutarlo sin cambiar de aplicación. Los IDE's ofrecen un ambiente gráfico en los que se tiene acceso a mayor número de herramientas no ofrecidas en los JDK's: *Debuggers* más elaborados, *check-points* dentro de la compilación, creación de WAR'S (*Web-Archives*), *Wizards* para acelerar desarrollo, entre otras cosas.

Algunos IDE'S son:

- NetBeans Open-Source
- Eclipse Open-Source
- Forte de Sun
- JBuilder de Borland
- Visual Cafe de Symantec
- Visual Age de IBM
- JDeveloper de Oracle

Alguno de los cuales precisan tener instalado JRE, como por ejemplo JDeveloper.

Es importante comenzar instalando JDK para comprender el funcionamiento básico del compilador en que se apoyan las herramientas integradas. Una vez abordados programas sencillos con pocas clases, conviene recurrir a una herramienta integrada. JBuilder 9 es una de las herramientas más completas a la hora de generar parte del código Java automáticamente. IntelliJ IDEA es otra de las herramientas más potentes, de interfaz muy sencilla y que no consume demasiados recursos de la computadora.

3.6. Variables Class y ClassPath

El desarrollo y ejecución de aplicaciones en Java exige que las herramientas para compilar (*javac.exe*) y para ejecutar (*java.exe*) se encuentren accesibles. El ordenador, desde una ventana de comandos MSDOS, solo es capaz de ejecutar programas indicados en la variable PATH. Por tanto se debe asignar a esta variable el directorio donde se encuentra instalado el JDK:

- Para Windows XP: abrir el Sistema dentro del panel de control, para ello seguir la secuencia: *Inicio -> Panel de Control -> Sistema*, seleccionando a continuación *opciones avanzadas* y después *variables de entorno*. Se asigna a la variable PATH el directorio donde se encuentra instalado el JDK, que será algo parecido a esto: PATH: C:\j2sdk1.4\bin.

Java utiliza otra variable de entorno denominada CLASSPATH, que determina donde buscar las clases o librerías Java (el API) además de otras clases de usuario. A partir de la versión 1.1.4 del JDK no es necesario especificar esta variable, salvo que se deseen añadir clases de usuario, o para indicar la ruta de directorios o ficheros **.jar* o **.zip* (archivos comprimidos con varias clases; en el caso de **.jar* mediante la herramienta *jar.exe*) en los que se encuentren los ficheros **.class.*

- La variable CLASSPATH se establece de modo análogo a la PATH.

3.7. Applets de Java

Una manera de incluir programas complejos en el ámbito de una página web, es a través de los *applets* de Java. Estos, se programan en Java beneficiándose de la potencia de este lenguaje para la Red.

Es otra manera de incluir código a ejecutar en los clientes que visualizan una página web. Se trata de pequeños programas hechos en Java, que se transfieren con las páginas web y que el navegador ejecuta en el espacio de la página.

Una pregunta frecuente es diferencia existente de los *applets* de Java respecto de los lenguajes *scripts*. Los *applets* de Java están programados en Java y precompilados, es por ello que la manera de trabajar de éstos varía un poco con respecto a los lenguajes de *script* como *JavaScript*. Así mismo, los *applets* son más difíciles de programar que los *scripts* y requerirán unos conocimientos medios del lenguaje Java.

La principal ventaja de utilizar *applets* consiste en que son mucho menos dependientes del navegador que los *scripts* en *JavaScript*, incluso independientes del sistema operativo del ordenador donde se ejecutan. Además, Java es más potente que *JavaScript*, por lo que el número de aplicaciones de los *applets* podrá ser mayor.

Como desventajas en relación con *JavaScript* cabe señalar que los *applets* son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos. Es por ello que con los *applets* de Java no se puede hacer directamente cosas como abrir ventanas secundarias, controlar *Frames*, formularios, capas, etc.

El hecho de que los *applets* se descarguen en el cliente para ejecutarse, lo que se denomina cliente pesado, hace la aplicación web dependiente de la velocidad de procesamiento del usuario, por lo que no es una de las técnicas más usadas a nivel de empresa. Es la programación con *servlets* la más utilizada, donde la ejecución de procesos se realiza en el servidor, siendo el cliente (cliente ligero) el que se encarga del interfaz.

La ventaja principal de la programación en *applet* es precisamente liberar al servidor de la ejecución del programa, la cual se lleva a cabo por parte del cliente. Si la aplicación es utilizada por muchos usuarios al mismo tiempo, se requiere un servidor muy potente para poder atender todas las peticiones, cliente- servidor. Por lo que la programación en *applet* a pesar de ralentizar al cliente, en ocasiones es una solución