Tema 1

Generación dinámica de páginas Web. Arquitecturas y herramientas de Desarrollo

Objetivos

- Identificar el modelo de arquitectura de software Cliente/Servidor.
- Considerar la arquitectura del software de una aplicación Web como un caso particular de la arquitectura Cliente/Servidor.
- Conocer la agrupación de funcionalidades del modelo de desarrollo de software de tres capas en los entornos basados en una arquitectura de software Cliente/Servidor.
- Comprender los mecanismos de ejecución de código en un entorno Cliente/Servidor en función de las diferentes tecnologías de desarrollo para la Web.
- Reconocer las ventajas de la generación dinámica de páginas Web.
- Identificar y caracterizar las distintas tecnologías y herramientas de desarrollo existentes para el desarrollo de aplicaciones Web con procesamiento en el lado del servidor.
- Comprender los aspectos básicos de la tecnología de desarrollo ASP.NET de Microsoft.
- Conocer las características y el uso del entorno de desarrollo Microsoft Visual Studio.

Introducción

En este primer tema se presentan los conceptos básicos que permiten comprender el contexto en el que se realiza el desarrollo de las aplicaciones Web que incluyen procesamiento en el servidor Web. Se introducen conceptos importantes relacionados con: el modelo de arquitectura del software cliente/servidor, el modelo de capas para el desarrollo de software en los entornos Web y el mecanismo de procesamiento en el entorno del servidor Web. Por último, se presenta una panorámica actual de las tecnologías y herramientas de desarrollo más significativas que pueden aplicarse, en especial en el lado del servidor, para el desarrollo de aplicaciones Web interactivas.

Índice

1.1 ASPECTOS PREVIOS: EL SERVICIO DE INFORMACIÓN WEB

El servicio de información Web *(World Wide Web)* representa un universo de información accesible a través de Internet. El servicio de información Web está formado por un conjunto de recursos de información interconectados que conforman buena parte del conocimiento humano actual. Su funcionamiento es posible debido a la coexistencia de una serie de elementos lógicos (software) y físicos (hardware). Dentro de un contexto enfocado hacia el desarrollo Web en entorno servidor se consideran exclusivamente los elementos lógicos intervinientes.

El servicio de información Web puede definirse como un **servicio hipermedia de acceso a la información a través de internet** que está basado en:

- El uso de la infraestructura de Internet basada en la familia de protocolos TCP/IP.
- La utilización de documentos hipertexto para facilitar el acceso y la navegación entre los recursos solicitados mediante enlaces o *links*.
- El modelo de arquitectura del software Cliente/Servidor que incorpora un esquema de petición-respuesta para el intercambio de información entre ambos.
- El sistema *Uniform Resource Identifier* (URI) para la identificación genérica de los recursos en la red. Un identificador URI referencia unívocamente un recurso de la red mediante una cadena de caracteres, empleando la siguiente sintaxis: esquema:autoridad/ruta/recurso. Por ejemplo: ftp://ftp.iesmarenostrum.com/pub/Tema1.doc. En el caso concreto de referencias a direcciones Web, se suele emplear el identificador particular del servicio de información Web que se denomina *Uniform Resource Locator* (URL). Así, el concepto URL se ha incorporado al concepto más general que es URI. Un ejemplo de una URL, o mejor de una URI para la Web, sería: http://www.iesmarenostrum.com/informatica/pub/inicio.htm.
- El protocolo *HyperText Transfer Protocol* (HTTP) como mecanismo de transmisión de los recursos solicitados. Se trata del protocolo de comunicaciones usado para el intercambio de información en cada transacción entre el cliente y el servidor. Especifica la sintaxis, la semántica y la sincronización que utilizan los elementos de software del servicio de información Web, cliente y servidor, para comunicarse entre sí mediante un esquema de intercambio petición-respuesta, o lo que es lo mismo, solicitud-envío.

El término hipermedia que aparece más arriba en la definición del servicio de información Web debe entenderse en el sentido de multimedia, es decir, como un servicio que puede presentar información procedente de diversos medios de comunicación: texto, gráficos, sonido, videos, etc.

La dinámica de funcionamiento del servicio de información Web, desde el punto de vista más básico, se representa en la Ilustración 1.1 que se muestra a continuación. En primer lugar, el programa navegador Web (cliente) utiliza una URL, o mejor una URI, para conectarse con un programa servidor Web (servidor) y solicitarle un recurso que contiene la información multimedia requerida (documento HTML, archivo de texto, imagen, video, etc.). Establecida la conexión con el servidor Web, en segundo lugar, éste localiza el recurso solicitado dentro de su sistema de almacenamiento asociado y lo envía al cliente como respuesta a través de la infraestructura de Internet. Finalmente, conforme va llegando el contenido del recurso solicitado, éste va siendo interpretado por el cliente, reconstruyéndose la información en la ventana de visualización del navegador Web.

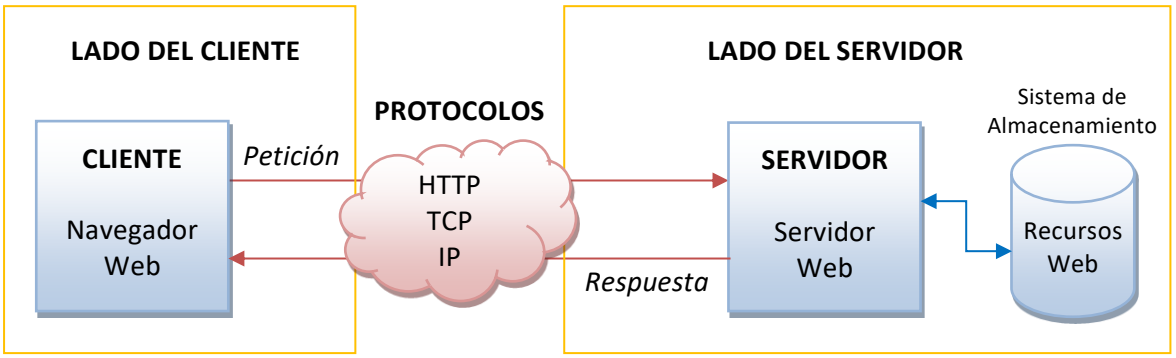


Ilustración 1.1. Dinámica de funcionamiento del servicio de información Web.

1.2 MODELOS DE DESARROLLO EN ENTORNO CLIENTE/SERVIDOR

En el apartado anterior se han expuesto las características más significativas del servicio de información Web en el ámbito del desarrollo de software. En este contexto, es necesario incidir en la idea de que en el desarrollo de software en entornos Web debe tenerse siempre en cuenta la distribución de los diversos elementos y la función que tiene cada uno de ellos. De manera que, para el futuro desarrollador Web es básico comprender que no será posible producir soluciones software adecuadas, si durante su desarrollo no se han considerado las características, distribución y función de los diferentes elementos que conforman el entorno del servicio de información Web.

El modelo de desarrollo de software en entornos Web se fundamenta en el modelo de arquitectura de software Cliente/Servidor. Este modelo de arquitectura se basa en la idea de servicio: el cliente es un proceso consumidor de servicios y el servidor es un proceso proveedor de servicios. La relación entre el cliente y el servidor se establece en base al intercambio de mensajes a través de un protocolo de comunicaciones, como el único elemento de acoplamiento entre ambos. El protocolo de comunicaciones es un conjunto de reglas o estándar que definen la sintaxis, semántica y sincronización de la comunicación entre el cliente y el servidor. Así, los elementos intervinientes en el modelo de arquitectura del software Cliente/Servidor son tres: el cliente, el servidor y el protocolo de comunicaciones. La Ilustración 1.2 muestra la dinámica más general de este proceso y que es el siguiente: el cliente es el que inicia el intercambio de información, solicitando datos al servidor que responde enviando al cliente un flujo de datos de respuesta. Además de la transferencia de la información solicitada por el cliente, este intercambio de información suele incluir otra información administrativa adicional, como puede ser: el nombre del recurso a transferir, las direcciones de red y el nombre de los dispositivos del cliente y del servidor, la fecha y hora de la solicitud, etc.

Ilustración 1.2. Vista general del modelo de arquitectura de software Cliente/Servidor.

El modelo de arquitectura del software del sistema de información Web puede ser considerado como un caso particular de un modelo de arquitectura del software Cliente/Servidor, en el cual los elementos software intervinientes están constituidos, fundamentalmente, por software estándar.

- Cliente: cualquier navegador Web (Edge, Chrome, Firefox, Safari, Opera, etc.) ejecutado desde cualquier dispositivo con acceso a internet (ordenador, tablet, teléfono móvil, etc).
- Servidor: cualquier servidor Web (Internet Information Services (IIS), Apache, Nginx, etc.)
- Protocolos de comunicación: HTTP (servicio de información Web) y TCP/IP (Internet).

Modelo de desarrollo de software de tres capas

Cuando se desarrolla una aplicación informática basada en el modelo de arquitectura del software Cliente/Servidor, y también en el caso particular de desarrollar una aplicación para la Web, las funcionalidades del software suelen quedar agrupadas en diferentes capas, cada una de ellas especializada en la gestión de un determinado aspecto del servicio. Los modelos de desarrollo del software para el servicio de información Web se pueden clasificar en función de en qué lugar, si en el cliente (navegador Web) o en el servidor (Servidor Web), se sitúan cada una de estas capas. La decisión de dónde se sitúa cada una de estas capas dependerá del propio diseño del software, del entorno de ejecución utilizado y de las tecnologías de desarrollo empleadas. Si bien es posible encontrarse con divisiones más o menos especializadas, se suelen identificar tres tipos de capas fundamentales: la capa de presentación, la capa de la lógica de negocio y la capa de persistencia.

- **Capa de presentación:** esta capa corresponde con la interfaz de usuario. Una interfaz gráfica integrada en el navegador presenta el contenido del recurso solicitado y sirve para recoger su interacción con el usuario. La capa de presentación es interpretada por el cliente (navegador Web). El desarrollo de esta capa se centra en el formateo de la información enviada por el servidor y la captura de las acciones interactivas realizadas por el usuario en el cliente, a través del lenguaje HTML. Un aspecto importante a tener en cuenta cuando se desarrolla la capa de presentación para el servicio de información Web es que el navegador Web (cliente) únicamente es capaz de interpretar documentos HTML.

- **Capa de lógica del negocio:** es la capa que soporta y gestiona las funcionalidades que se esperan de la aplicación Web. Estas funcionalidades se relacionan con la representación de la lógica o reglas de negocio que implementa la solución software. Habitualmente es donde se reciben las peticiones del usuario y desde donde se envían las respuestas apropiadas tras el procesamiento de la información proporcionada por el cliente. Al contrario que la capa de presentación, la lógica de negocio puede ser desarrollada tanto en el entorno cliente (navegador Web) como en el entorno servidor (servidor Web).

- **Capa de persistencia o de datos:** es la capa donde residen los datos que maneja la aplicación Web y es la encargada de acceder a los mismos. Normalmente, está formada por uno o más servidores de bases de datos que reciben solicitudes de manipulación (almacenamiento o recuperación) y de administración de la información almacenada desde la capa de negocio. El resultado del procesamiento de información solicitado es enviado hacia la capa de negocio desde la capa de persistencia.

Capa	Presentación	Lógica del negocio	Datos
Tareas	Interfaz de usuario	Lógica del procesamiento de los datos	Acceso a datos
Software	Navegador Web (Cliente)	Navegador Web (Cliente) y/o Servidor Web (Servidor)	Servidor de Base de Datos

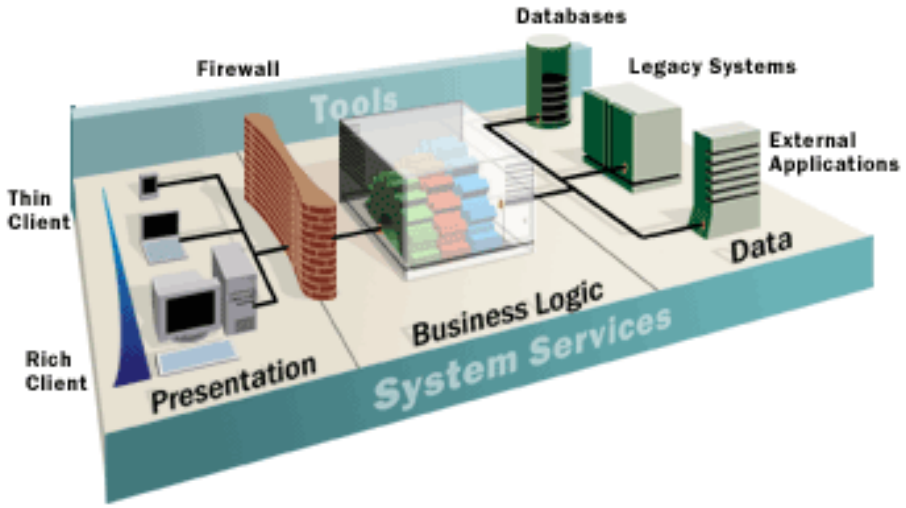


Ilustración 1.3. Modelo en tres capas de la arquitectura de software Cliente/Servidor.

Las propuestas de arquitecturas para el desarrollo de aplicaciones Web se suelen basar en un modelo de desarrollo de software en tres capas. Ahora bien, dependiendo de la especialización de las capas, de las tecnologías empleadas y del reparto de las funcionalidades agrupadas en capas, pueden encontrarse otras configuraciones con más o menos capas. Así, por ejemplo, en caso de que las funcionalidades de las capas de lógica de negocio y de datos se agruparan en una única capa, entonces se dispondría de un modelo en dos capas. Y si, además del servidor Web, se incorporará un servidor de aplicaciones para procesar los componentes de la lógica del negocio, entonces se dispondría de un modelo en cuatro capas. El diseño de las capas determina el número de capas que se utilizarán. Por este motivo, generalizando, este modelo de desarrollo suele denominarse **modelo de desarrollo de software en capas**, o también **modelo de desarrollo de software de n capas**.

Las tecnologías de desarrollo Web actuales permiten gestionar y distribuir las responsabilidades de cada una de las funcionalidades de las capas del software entre el cliente y el servidor, todo ello utilizando un modelo de desarrollo de software en capas. El diseño y desarrollo de cada una de las capas influye a la hora de seleccionar el elemento de la arquitectura Web que soportará más o menos carga de procesamiento. Así, se suele hablar de configuraciones cliente rico *(Rich Client)*, cuando el mayor peso de la aplicación se ejecuta en el cliente. Y, se suele hablar de configuraciones de cliente ligero *(Thin client)*, cuando el mayor peso recae en el servidor y la funcionalidad asociada al cliente se limita a presentar la información enviada por el servidor. En el ámbito del desarrollo de aplicaciones Web en el entorno del servidor, se suelen emplear configuraciones de cliente ligero.

Algunas de las tecnologías de desarrollo Web y los lenguajes de programación que se emplean en el desarrollo de aplicaciones Web en el entorno del cliente y en el entorno del servidor son:

- **En el lado del cliente**, se suelen emplear algunas de las siguientes tecnologías Web: lenguaje de marcas **HTML** para estructurar la presentación de la información, hojas de estilo en cascada **CSS** para definir la apariencia y presentación visual de los elementos, **objetos embebidos** que muestran diversos tipos de contenido, **scripts de cliente** para incluir código procedural mediante un lenguaje de script interpretado por el cliente (Javascript) y, **AJAX** que facilita la creación de aplicaciones Web interactivas desde el cliente a través conexiones asíncronas con el servidor para realizar cargas dinámicas de contenidos.

- **En el lado del servidor**, es posible utilizar **scripts de servidor** embebidos en el código HTML (PHP, ASP, JSP, Perl, Python, etc.), enlaces a **ejecutables externos** (CGI), **componentes en objeto** (EJB de Oracle) que encapsula la lógica del negocio en componentes precompilados y **tecnologías de código subyacente** (ASP.NET de Microsoft) que separan la presentación de la página de la lógica de negocio.

El lado del servidor suele incorporar otras capacidades, además del propio servidor Web, como pueden ser: servidor de base de datos, servidor de aplicaciones, servidor de objetos o componentes, servidores de transacciones, etc. Se denomina **entorno del servidor Web** al conjunto de estas capacidades que forman parte del lado del servidor, incluido el servidor Web.

1.3 GENERACIÓN DE PÁGINAS WEB DE FORMA DINÁMICA

Se hace referencia al concepto de **aplicaciones Web**, o de **aplicaciones para la Web**, como **aquellas aplicaciones informáticas cuya interfaz de usuario se construye utilizando páginas Web que son interpretadas por un navegador Web**, en lugar de construirse sobre ventanas y controles específicos de un sistema operativo concreto. Como cualquier otro tipo de aplicaciones informáticas, las aplicaciones Web están constituidas por un conjunto de procesos, representados por páginas Web que pueden incluir procesamiento en el cliente o en el servidor (capas de presentación y de lógica del negocio) y, por un conjunto de datos almacenados en Bases de Datos (capa de datos).

Las aplicaciones Web pueden clasificarse según diversos criterios: según la tecnología de desarrollo utilizada para su construcción, dependiendo del modelo arquitectónico utilizado o considerando la capacidad de interacción de los usuarios con la interfaz Web que se les presenta. Este último criterio da lugar a la **clasificación funcional de las aplicaciones Web** en estáticas, dinámicas e interactivas:

- **Aplicaciones Web estáticas:** son aquellas aplicaciones Web en las que el usuario recibe solo páginas Web pasivas, donde la interacción del usuario no produce ningún tipo de acción, ni en las propias páginas, ni generan respuesta alguna por parte del servidor. Este tipo de aplicaciones Web son procesadas (interpretadas) solo por el cliente (navegador Web) y se construyen empleando los lenguajes **HTML** y **CSS** exclusivamente, que se utilizan para especificar la presentación de la información y la organización visual. Un ejemplo típico sería un sitio Web corporativo de cualquier organización, en el que los contenidos presentados se modifican a través de tareas de mantenimiento de las páginas Web que lo forman.

- **Aplicaciones Web dinámicas:** son aquellas aplicaciones Web en las que la interacción del usuario con el recurso recibido (página Web) produce algún cambio en relación con la visualización del mismo, es decir, produce cambios en la capa de presentación. Estos cambios dinámicos, o a lo largo del tiempo, se refieren a modificaciones de formato, ocultación de partes del documento, creación de elementos nuevos, etc. que pueden ser ejecutados tanto en el cliente como en el servidor, o incluso en ambos conjuntamente. Las tecnologías Web más significativas involucrados en el desarrollo de las aplicaciones Web dinámicas son: **HTML**, **CSS, scripts de cliente (JavaScript), objetos Flash u otros objetos embebidos**, **controles ActiveX**, **controles de servidor, etc**.

- **Aplicaciones Web interactivas:** al contrario que ocurre en los tipos de aplicaciones Web anteriores, en las aplicaciones Web interactivas la interacción del usuario produce algún cambio en la información contenida en el recurso recibido. Las aplicaciones Web interactivas se basan en que dicha interacción hace que se genere un intercambio de información o diálogo entre el cliente y el servidor. Desde el punto de vista del modelo de desarrollo y según cada aplicación concreta, la lógica del procesamiento asociada al inicio y a la gestión de dicho diálogo puede ser ejecutada tanto en el entorno del cliente como en el entorno del servidor, o en ambos conjuntamente.

Las aplicaciones Web interactivas son las que más interés están despertando actualmente y en las que más han evolucionado las tecnologías disponibles en los últimos años. El interés por el desarrollo de aplicaciones Web interactivas se relaciona con la necesidad de integrar los datos almacenados en bases de datos con el servicio de información Web. **Las aplicaciones Web interactivas permiten representar funcionalidades complejas relacionadas con poder presentar información de naturaleza cambiante a lo largo del tiempo y/o ajustarse a las necesidades de información propias de cada usuario**. Por ejemplo, para desarrollar una aplicación Web que presente la información sobre las películas que se proyectan en los cines de una ciudad, es necesario desarrollar una aplicación Web interactiva que presente esta información cambiante que estará almacenada en una base de datos. Y mucho más cuando se precise incorporar funcionalidades como la presentación de los horarios de inicio de las sesiones o la compra de entradas en sesiones numeradas. Otro ejemplo sería el desarrollo de una aplicación Web para presentar la información de las cotizaciones de los valores bursátiles en tiempo real. La naturaleza dinámica, o cambiante a lo largo del tiempo, de este tipo de información determina su almacenamiento en bases de datos y, su posterior consulta y presentación Web a través de los procesamientos adecuados. Otros ejemplos pueden ser el desarrollo de aplicaciones Web para la gestión de correo electrónico vía Web o de una red social. En estos dos últimos casos, además de considerar la naturaleza cambiante de la información, se precisa seleccionar y personalizar la información a presentar para cada usuario concreto del servicio.

Existe una gran diversidad de tecnologías de desarrollo involucradas en el desarrollo de aplicaciones Web interactivas y, además, varían mucho en función de si el procesamiento se realiza en el entorno del cliente o en el entorno del servidor. En la actualidad, una de las tecnologías de referencia en el entorno del cliente para el desarrollo de aplicaciones Web interactivas es **AJAX** *(Asynchronous JavaScript And XML)*. En el entorno del servidor existe una gran variedad de tecnologías de desarrollo para la construcción de aplicaciones Web interactivas, cuya panorámica actual se estudiará en los siguientes apartados. El ámbito de estudio de este módulo se refiere al desarrollo de aplicaciones Web interactivas empleando tecnologías de procesamiento en el entorno del servidor Web.

Dinámica del procesamiento de código en el servidor Web

La Ilustración 1.4 muestra una representación genérica de la dinámica del procesamiento de código en el entorno del servidor Web. Como puede comprobarse, el mecanismo de procesamiento de código en el servidor Web queda integrado en el contexto básico de la dinámica de funcionamiento del servicio de información Web que se muestra en la Ilustración 1.1.

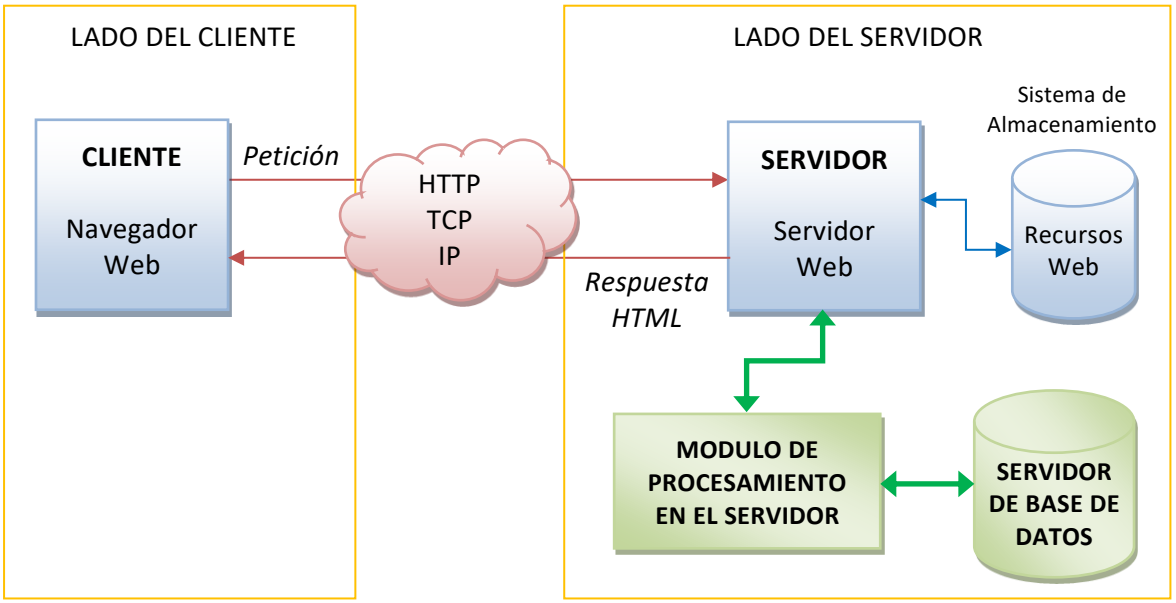


Ilustración 1.4. Dinámica general del mecanismo de procesamiento en el entorno del Servidor Web.

En primer lugar, en el navegador Web (cliente) se realiza la solicitud de un recurso que contiene código de procesamiento en el servidor Web. Existen diversos métodos y tecnologías para poder especificar el código escrito o la funcionalidad que ejecutará el servidor Web. Establecida la conexión con el servidor Web, en segundo lugar, éste localiza el recurso solicitado dentro de su sistema de almacenamiento y, al detectar que el recurso incluye código que especifica un procesamiento en el servidor Web, realiza una llamada al módulo de procesamiento en el servidor. En tercer lugar, el módulo de procesamiento en el entorno del servidor ejecuta el procesamiento solicitado. En el caso de que el procesamiento a ejecutar solicite un acceso a un servidor de base de datos, se establece la conexión con la base de datos y se ejecuta el comando de acceso a datos incluido que puede estar especificado empleando SQL. Finalizada la ejecución, en cuarto lugar, el servidor Web envía el resultado del procesamiento al cliente a través de la infraestructura de Internet. Un aspecto muy importante que se debe tener en cuenta es que **el resultado de un procesamiento en el servidor Web es un documento HTML válido** (página Web) que incorpora dinámicamente en su contenido la información resultante de la ejecución en el servidor Web. Finalmente, conforme va llegando el contenido del documento HTML resultante del procesamiento en el servidor (página de respuesta), va siendo interpretado por el navegador Web, reconstruyéndose en la ventana de visualización.

A resaltar que el resultado del procesamiento en el servidor Web produce siempre una página HTML de respuesta, que se genera dinámicamente incorporando en ella los resultados de la ejecución.

1.4 TECNOLOGÍAS DE DESARROLLO WEB EN EL SERVIDOR

Se entiende por tecnología de desarrollo Web en el entorno del servidor a aquella tecnología de desarrollo Web cuyo código, ya sea como código interpretado o bien como código objeto precompilado, es ejecutado por un componente específico de software en el entorno del servidor Web. La dinámica del procesamiento que suelen emplear este tipo de tecnologías sigue el esquema genérico que se muestra en la ilustración 1.4 y que ha sido estudiado en el apartado anterior. El código ejecutado en el servidor representa la capa de la lógica del negocio y se emplea para proporcionar funcionalidades complejas a la aplicación Web, como puede ser el acceso a datos almacenados. Las tecnologías de desarrollo Web en el entorno del servidor se aplican solo para construir aplicaciones Web interactivas.

Existe una gran variedad de tecnologías de desarrollo que facilitan la ejecución de código en el lado del servidor Web. A continuación, se realiza una panorámica actual de las tecnologías de desarrollo Web en el entorno del servidor más conocidas y que se pueden clasificar en tres grandes grupos: tecnologías basadas en lenguajes de script de servidor Web, tecnologías basadas en enlaces a programas y componentes ejecutables y tecnologías de código subyacente.

Tecnologías basadas en lenguajes de script de servidor Web

Este tipo de tecnologías se basa en utilizar un lenguaje de script de servidor, cuya característica principal es que el código escrito es interpretado por el servidor Web. El código escrito utilizando el lenguaje de script de servidor se entremezcla o intercala con el código HTML estático, para conformar una página de respuesta que incluya el resultado del procesamiento en el servidor. El código HTML especifica la estructura básica de la página de respuesta que se envía al cliente. Las porciones de código escrita en un lenguaje de script son ejecutadas por el servidor Web para obtener y presentar los resultados del procesamiento en el servidor sobre la página de respuesta. Para que estas porciones de código script sean ejecutadas, el servidor Web debe tener instalado un módulo de procesamiento específico, denominado *scripting engine*. Este módulo es el encargado de reconocer e interpretar el lenguaje de script utilizado para escribir las porciones de código que se han insertado en la página Web y que especifican el procesamiento a ejecutar por el servidor Web.

Actualmente, existe una gran diversidad de lenguajes de script de servidor Web, así como de *Frameworks*. En la actualidad, es habitual abordar el desarrollo de las aplicaciones Web interactivas partiendo de un *Framework* por razones de productividad. En la siguiente tabla se muestran algunos de los lenguajes de script de servidor Web y *Frameworks* asociados más conocidos y extendidos:

Lenguajes de script de servidor	**Frameworks**	**Fabricante**
C#, Basic, etc.	**Active Server Pages (ASP)**	Microsoft
Java	**JavaServer Pages (JSP)** **JavaServer Faces (JSF)** **Grails**	Oracle Corporation Java Community Process OCI
Perl	**Catalyst** **Dancer** **Jifty**	Catalyst Foundation Alexis Sukrieh Best Practical Solutions, LLC

Lenguajes de script de servidor	Frameworks	Fabricante
Ruby	**Ruby on Rails** **Sinatra**	Rails Core Team Blake Mizerany
PHP	**Laravel** **Symfony** **CakePHP** **CodeIgniter** **Yii**	Taylor Otwell Symfony SAS Cake Software Fundation, Inc Ellislab Yii Software, LLC
CFML	**ColdFusion**	Adobe
Python	**Django** **Pyramid**	Django Software Foundation Agendaless Consulting, Inc
Javascript	**Node.js** **Express.js**	Joyent y Google Node.js Foundation

Tecnologías basadas en enlaces a programas y componentes ejecutables

Una alternativa a los lenguajes de script en el servidor es utilizar enlaces a programas y componentes ejecutables, que reciban ciertos parámetros de entrada y devuelvan como resultado el contenido HTML que se enviará el cliente como respuesta. Este tipo de tecnologías de desarrollo Web en el entorno del servidor se basan en desarrollar unidades de software (programas o componentes precompilados) externos que son ejecutados de forma independiente del servidor Web. Algunas de las tecnologías más representativas que se basan en esta idea se comentan a continuación.

La forma más simple de generar páginas Web de forma dinámica en el servidor es delegar su creación a un programa externo. La tecnología **CGI *(Common Gateway Interface)*** se basa, precisamente, en este comportamiento. Dado que el programa externo ejecutable que genera la página Web de respuesta es un fichero compilado, pueden utilizarse diversos lenguajes de programación para especificar el código fuente de la lógica del procesamiento en el servidor. La localización del programa externo a ejecutar se indica en la URL que forma la petición HTTP del cliente. La principal desventaja es el bajo rendimiento a la hora de responder a múltiples peticiones CGI simultáneas, puesto que para cada una de ellas se crea un proceso nuevo en el servidor.

Otra alternativa más actual avanza en esta idea simple, pero utilizando componentes precompilados estándar que son ejecutados por un servidor de aplicaciones independiente que forma parte del entorno del servidor Web. Existen diversas tecnologías que implementan la lógica de negocio de la aplicación Web en componentes precompilados en objeto y que utilizan el lenguaje de programación Java y las características de la plataforma de desarrollo **J2EE *(JavaTM 2 Platform, Enterprise Edition)*** de Oracle Corporation. Una de estas tecnologías se basa en utilizar componentes **Enterprise JavaBeans (EJB)** que proporcionan un modelo de componentes distribuido estándar en el lado del servidor Web. Los componentes EJB son componentes precompilados que están desarrollados en Java. La plataforma de desarrollo J2EE se encarga de proveer los componentes EJB, que se ejecutan en un servidor de aplicaciones independiente, al propio servidor Web. El uso de estas tecnologías basadas en la ejecución de componentes precompilados en un servidor de aplicaciones independiente está indicado en los casos que se requiera altas prestaciones de transaccionalidad,

concurrencia y seguridad. La tecnología basada en el uso de EJB forma parte de plataforma J2EE y está también desarrollada por Oracle Corporation. Otras tecnologías similares basadas en componentes precompilados, que también son compatibles y funcionan bajo la plataforma J2EE, son **Struts** de Apache Software Foundation y **Spring** de Rob Johnson.

Tecnologías de código subyacente

Otras estrategias de desarrollo en el lado del servidor, denominadas tecnologías de código subyacente o *code-behind*, se basan en separar, por un lado, el código de marcado para presentación de la página y, por otro, el código de la lógica del procesamiento en el servidor que expresa la lógica del negocio de la aplicación Web. Esta misma separación también está presente en las tecnologías basadas en componentes precompilados en objeto. Sin embargo, la diferencia reside en que, las tecnologías de código subyacente mantienen la lógica de procesamiento en el servidor en librerías precompiladas independientes que residen en el propio servidor Web. Y, por tanto, no se necesita emplear un servidor de aplicaciones independiente para ejecutar el procesamiento en el servidor Web. Se trata de una tecnología intermedia entre el procesamiento basado en scripts de servidor, que puede sobrecargar los servidores Web, y el procesamiento de componentes precompilados en un servidor independiente, que puede producir pérdidas de rendimiento y de fiabilidad.

La tecnología de código subyacente más representativa es **ASP.NET *(Active Server Pages .NET)*** de Microsoft que utiliza las características de la plataforma de desarrollo **.NET** de Microsoft. Una aplicación Web construida con ASP.NET se precompila al lenguaje nativo de .NET. Posteriormente, en cada solicitud se ejecuta el código nativo a través de un módulo de software específico, denominado *Runtime*, lo que permite mejorar el rendimiento. Otra ventaja derivada del proceso de precompilación es que se trata de una tecnología independiente del lenguaje de programación empleado para escribir el código fuente. Así, en la práctica, se pueden desarrollar aplicaciones Web interactivas utilizando cualquier lenguaje de programación orientado a objetos y manejado por eventos compatible con .NET, lo que facilita la adaptación a las características del desarrollador y aumenta la productividad de las tareas de desarrollo. El uso de la tecnología ASP.NET está indicada en los casos que se requiera desarrollar aplicaciones Web interactivas de altas prestaciones de fiabilidad, transaccionalidad, concurrencia, seguridad, rendimiento y productividad.

En la actualidad, las tecnologías de desarrollo Web más tecnológicamente avanzadas para construir aplicaciones Web interactivas con procesamiento en el servidor y acceso a datos almacenados de altas prestaciones son las tecnologías de código precompilado: ASP.NET de Microsoft y las tecnologías Enterprise JavaBeans (EJB) y Struts, ambas bajo la plataforma J2EE de Oracle.

1.5 INTEGRACIÓN CON SERVIDORES WEB

Tal como se ha estudiado anteriormente, pueden utilizarse diferentes modelos, tecnologías y lenguajes para el desarrollo de aplicaciones Web interactivas. En cualquier caso, la posibilidad de responder a una petición de un cliente depende de las capacidades que tenga el servidor Web.

Un servidor Web es un **programa cuya misión es servir información en forma de documentos HTML**. Así, la función básica de un servidor Web es proveer de contenido a un cliente (navegador Web) que

ha realizado una petición. Para poder cumplir con esta funcionalidad es necesario que el servidor Web pueda entender la petición del navegador Web. La petición debe ser correcta formalmente, es decir, debe constar de unos elementos concretos y especificados en un orden determinado. Las direcciones de las peticiones deben cumplir con los requisitos del sistema de identificación de recursos en la red, denominado URI. Este sistema de identificación define los elementos que debe contener la petición: el protocolo (HTTP, FTP, etc.), la dirección del servidor Web y la descripción del recurso solicitado en forma de ruta.

El desarrollo de **aplicaciones Web interactivas** se basa en que la interacción del usuario pueda producir un cambio en la información contenida en el recurso recibido tras el procesamiento en el servidor. Esta interacción se consigue a través de un intercambio de información o diálogo entre el cliente y el servidor a través de mensajes de solicitud y de respuesta. Por un lado, la solicitud proporciona la descripción del recurso que debe procesar el servidor, así como la información necesaria para ello y, por otro, la respuesta provee al cliente de la información solicitada en formato HTML. Este diálogo hace posible que el contenido del recurso solicitado quede adaptado de forma dinámica a la realidad del momento en que se realiza la petición, como consecuencia del procesamiento en el servidor de la información enviada desde el cliente. Para poder establecer este diálogo, es fundamental la información que se envía desde el cliente hacia servidor a través de **formularios Web** junto con la petición HTTP. Existen dos métodos de envío de la información contenida en los campos de un formulario Web hacia el servidor Web que son los siguientes:

- **Método GET**: es un método de envío en el cual la información que se envía desde el cliente hacia el servidor forma parte de la cadena que define la propia dirección URI. Este método tiene limitado el tamaño del envío a 256 caracteres. A continuación, se muestran algunos ejemplos de peticiones Web desde el cliente empleando el método de envío get.

  http://www.servidor3.com?nombre=jose&email=jose@infor.es
  http://www.morea.com?nombre=juan&dni=21212121-H&direccion=C/cisne,3

  Los ejemplos anteriores muestran como la información que se envía desde el cliente al servidor forma parte de la cadena que especifica la URI, poniendo en riesgo la protección de la información de tipo personal durante la interacción con el servidor.

- **Método POST**: es el método recomendado para enviar información a un servidor Web, por razones de seguridad y protección de la información. Los identificadores y valores de los campos del formulario son enviados desde el cliente hacia el servidor de forma transparente al usuario y al proceso de interacción durante el envío. Además, el envío se realiza empleando un sistema de encriptación. A pesar de todo ello, las especificaciones Web siguen manteniendo el método de envío get como opción por defecto en el envío de formularios, por lo que es necesario incluir la definición del valor post en el atributo method de la etiqueta <FORM> correspondiente.

Independientemente del método de envío utilizado, el servidor Web recibirá la información enviada desde el cliente a través de un formulario, realizará el procesamiento considerando la información recibida y, finalmente, devolverá al cliente el resultado del procesamiento, que ha realizado como consecuencia de esa petición concreta, en formato HTML.

Infraestructura del servicio en el entorno del servidor Web

El desarrollo y la explotación de las aplicaciones Web en el entorno del servidor se fundamenta en la utilización de una serie de lenguajes y tecnologías que son ejecutadas, en el lado del servidor, por uno o más servidores, entre ellos el servidor Web. La forma en la cual se distribuye la infraestructura del servicio en el entorno del servidor se basa en una separación de elementos atendiendo a la funcionalidad que aporta cada uno de ellos, pudiéndose crear tantos niveles como funcionalidades hayan involucradas en el procesamiento y el acceso a la información almacenada.

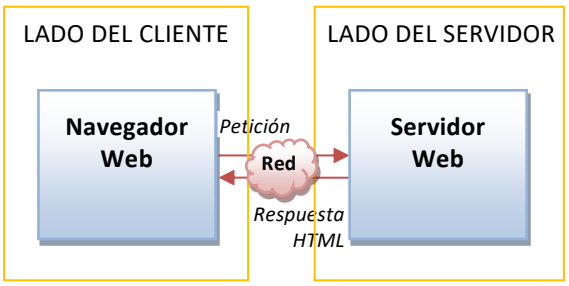


Ilustración 1.5. Arquitectura de dos niveles.

La Ilustración 1.5 representa la arquitectura de dos niveles que equivale al modelo de arquitectura Cliente-Servidor más tradicional.

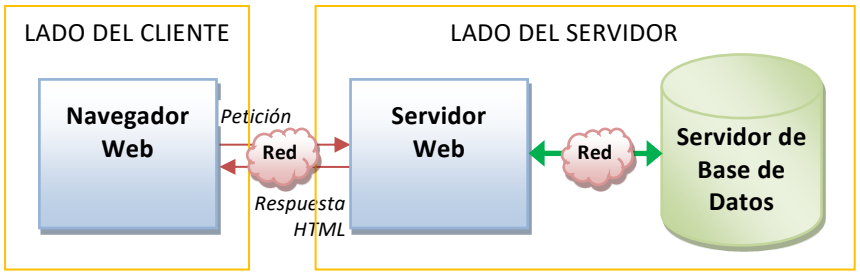


Ilustración 1.6. Arquitectura de tres niveles.

La Ilustración 1.6 representa la arquitectura de tres niveles en la que un servidor Web se encarga de aceptar y contestar peticiones Web y un servidor de base de datos se encarga de gestionar el acceso a datos, disminuyendo la carga de trabajo del servidor Web. Es la arquitectura más habitual.

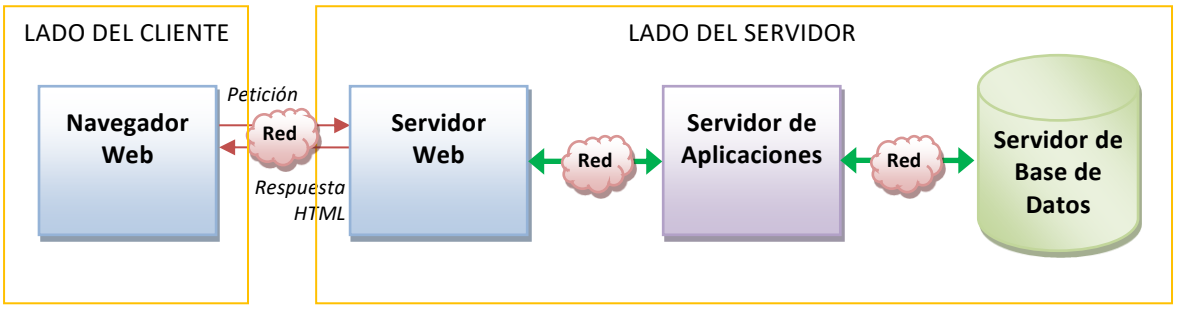


Ilustración 1.7. Arquitectura de cuatro niveles.

La Ilustración 1.7 representa otro caso de arquitectura que pone de manifiesto cómo la división en niveles corresponde a una división funcional. En este caso, se incorpora un servidor de aplicaciones que atiende a los servicios de ejecución de componentes precompilados externos para la generación dinámica de páginas Web en el entorno del servidor. Concretamente, esta arquitectura se refiere al uso de la tecnología J2EE.

La gestión del flujo de procesamiento y **la arquitectura del entorno del servidor depende, principalmente, de la tecnología de desarrollo que se utilice en cada caso para construir las aplicaciones Web interactivas**. En menor medida, también puede depender de la distribución de la lógica del negocio de la aplicación Web y de la forma de interactuar con la información que proviene del cliente. En efecto, cada tecnología de desarrollo requiere de una arquitectura y configuración determinada del entorno del servidor y, en algunos casos, de la instalación de componentes software específicos que se encarguen de la ejecución de las páginas dinámicas de servidor. La elección y configuración del entorno del servidor Web vienen determinadas por la tecnología de desarrollo a utilizar para construir las aplicaciones Web.

Para comprender el proceso de interacción entre el navegador (cliente) y el servidor Web (servidor) es necesario tener en cuenta cómo se produce la comunicación entre un cliente y un servidor. El intercambio de datos entre ambos se hace por medio del protocolo HTTP. Por defecto, un servidor Web se mantiene a la espera de peticiones HTTP realizadas por parte de los clientes escuchando un puerto de comunicaciones, que en el caso del servicio Web es el puerto 80. En un primer paso, el navegador solicita, como cliente DNS *(Domain Names System)*, la traducción de una URL a la dirección IP del servidor Web. Una vez que ha recibido la dirección IP desde el servidor DNS, el navegador realiza la petición HTTP al dispositivo cuya IP es la obtenida. Este dispositivo deberá tener instalado el componente software que representa el servidor Web. Es necesario tener siempre presente que se hace referencia a cliente y servidor como componentes software, refiriéndose siempre al programa navegador y al programa servidor Web, respectivamente. Puesto que HTTP es un protocolo sin estado, cada petición de un cliente a un servidor no está influenciada por las transacciones anteriores. Esto quiere decir que el servidor tratará cada petición como una operación totalmente independiente del resto. En el siguiente paso, el servidor procesa la solicitud realizada por el cliente y, tras ejecutar en el entorno del servidor Web el código asociado al recurso solicitado a través de la URI, responde al cliente enviando el código HTML resultante del procesamiento y que conforma la página Web de respuesta. Finalmente, el navegador del cliente, cuando recibe el código, lo interpreta y lo muestra en la ventana de visualización.

Entorno de ejecución de desarrollo y de explotación

Por razones de rentabilidad, el proceso de desarrollo del software de una aplicación Web suele realizarse en un entorno de ejecución diferente al que se llevará a cabo la ejecución efectiva de la aplicación Web. Una vez finalizado completamente el proceso de desarrollo de una aplicación Web, debe ser transferida a un entorno de explotación donde se producirá la utilización efectiva de la aplicación Web por parte de los usuarios finales. De manera que pueden diferenciarse entre:

- **Entorno de desarrollo**. Es un entorno de ejecución que permite realizar las tareas de construcción de la aplicación Web. Este entorno de ejecución incorpora únicamente las herramientas de desarrollo necesarias para implementar de la Aplicación Web.

- **Entorno de pruebas o preproducción**. Es un entorno de ejecución que se utiliza para realizar las pruebas finales de integración de la aplicación Web, una vez que se ha finalizado su desarrollo. El entorno de pruebas debe recrear unas condiciones idénticas a las que tendrá el entorno de ejecución en el que se producirá la utilización efectiva de la aplicación Web.

- **Entorno de explotación o producción**. Es el entorno de ejecución final que usan los usuarios finales para la ejecución y utilización efectiva de la aplicación Web. Este entorno de ejecución incorpora todas las prestaciones y características necesarias para que la ejecución de la aplicación web se realice con información real de manera adecuada, una vez que ésta haya sido totalmente construida y probada.

La operación por la cual se realiza la transferencia de la aplicación Web a un entorno de explotación, o de pruebas, una vez finalizado completamente el proceso de desarrollo, se denomina **despliegue de la aplicación Web**.

Características de los servidores Web más utilizados

A continuación, se presenta una panorámica actual de algunos de los servidores Web más utilizados que incluye una breve referencia a sus características más significativas.

- **Apache HTTP Server** de Apache Software Foundation. Está diseñado para ser utilizado en múltiples plataformas y sistemas operativos. Se trata de un servidor Web robusto, que implementa los últimos estándares y protocolos de red y que cuenta con capacidad de modularización. Es un servidor Web de código abierto y cuenta con una comunidad de desarrolladores amplia que proporcionan documentación sobre su uso y configuración para diferentes propósitos. Su diseño altamente modular, flexible y configurable permite a los administradores de sitios Web elegir las características que van a ser soportadas por el servidor y se pueden seleccionar los módulos que van a estar disponibles cuando se ejecute el servidor Web. Incorpora soporte para gestionar el acceso a bases de datos, establecer páginas protegidas por contraseña, personalizar las páginas de error devueltas por el servidor, generar registros de actividad en múltiples formatos, etc. Ofrece soporte a diferentes lenguajes y tecnologías de desarrollo como pueden ser: Perl, Python, PHP, Ruby, componentes precompilados basados en la tecnología J2EE, ASP.NET, Node.js, etc.

- **Internet Information Services (IIS)** de Microsoft. Es un servidor Web de alto rendimiento, flexible, robusto, altamente seguro, fiable y fácil de gestionar que permite alojar cualquier tipo de contenido. Destaca, principalmente, por dar soporte a lenguajes de programación y tecnologías de desarrollo como ASP y **ASP.NET de forma nativa**. Dado que el modelo de desarrollo de aplicaciones .NET es independiente del lenguaje de programación empleado para la construcción de la aplicación Web, permite utilizar cualquier lenguaje de programación como PHP, Java, Perl, Python, etc. Incorpora servicios de despliegue de aplicaciones, acceso a bases de datos o administración remota del servidor. Ofrece soporte a diferentes lenguajes y tecnologías de desarrollo como pueden ser: ASP.NET, Node.js, componentes precompilados basados en la tecnología J2EE, Ruby, etc.

- **Ngnix** de Nginx, Inc**.** En la actualidad, es ampliamente utilizado. Se trata de un servidor Web de código abierto y cuenta con una arquitectura modular de alto rendimiento, además de permitir funcionar como servidor proxy para otros protocolos de Internet como IMAP o POP3 (correo electrónico). Sus principales características son la estabilidad, la facilidad de configuración y la capacidad para ser ejecutado en múltiples plataformas consumiendo relativamente pocos recursos. Soporta el lenguaje de programación **PHP de forma nativa**, sin necesidad de tener que instalar un módulo como es el caso de Apache. Es el servidor Web utilizado por compañías como Wordpress, Netflix, GitHub, SourceForge, etc. Ofrece soporte a diversos lenguajes y tecnologías de desarrollo como pueden ser: PHP, JSP, ASP.NET, componentes precompilados basados en la tecnología J2EE, Node.js, etc.

Pueden consultarse datos sobre la implantación de servidores Web en http://news.netcraft.com

Como puede comprobarse en la lista anterior, en algunos casos, estos servidores Web soportan de forma nativa diferentes tecnologías o lenguajes de programación del lado del servidor, como es el caso de la tecnología ASP.NET con el servidor Web IIS de Microsoft. Sin embargo, en otros casos, se precisa de la instalación y configuración de módulos o extensiones específicas, como es el caso del módulo de soporte de PHP para el servidor Web de Apache. El soporte nativo de las tecnologías Web suele proporcionar una mejora significativa de las prestaciones del servicio en cuanto a rendimiento, fiabilidad y seguridad, por lo que suele ser la opción recomendada.

La tecnología basada en componentes J2EE precisa, además del servidor Web, de la participación de un servidor de aplicaciones para poder ejecutar componentes precompilados de los estándares EJB, Struts, Spring, etc. Algunos de los servidores de aplicaciones para J2EE que actúan en combinación con Apache son los siguientes: Oracle WebLogic Server (Oracle), JBoss (Redhat), WildFly (RedHat), Glassfish (Oracle), WebSphere (IBM).

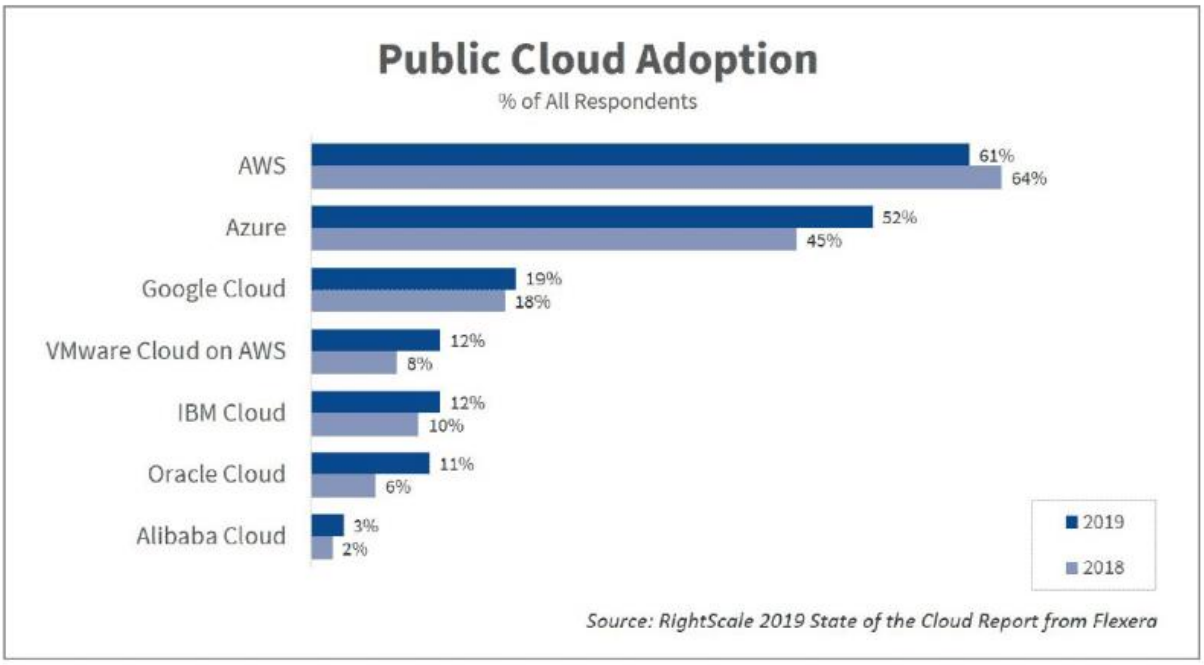
Como curiosidad, cabe comentar que el servidor Web **Google Web Server** (GWS) de Google LLC es utilizado por la compañía Google para dar servicio a su propia infraestructura Web. A pesar de que es utilizado exclusivamente por Google para el acceso a sus propios dominios y sitios Web, es uno de los servidores Web que recibe más peticiones HTTP recibe actualmente.

Servicios de despliegue de aplicaciones Web en la nube *(Cloud Computing)*

La tendencia actual es trasladar las necesidades de procesamiento a la nube *(Cloud Computing)*. El **procesamiento en la nube** se basa en la idea de utilizar la infraestructura de soporte a las aplicaciones como un servicio **(IaaS, *Infrastructure as a Service*)**. La razón principal de este cambio de planteamiento que se ha producido en las empresas, y en las organizaciones en general, viene provocado por la elevada reducción de costes que supone el uso de servicios de procesamiento en la nube frente al uso de infraestructura física propia. Esta tendencia que afecta, en general, a todo el software, también está afectando, de manera muy importante al uso de la infraestructura tecnológica del entorno del servidor Web. Por estos motivos, actualmente, se suelen contratar **servicios de despliegue de aplicaciones Web en la nube** para implementar el entono de explotación o producción de las aplicaciones Web interactivas, así como del entorno de pruebas. Los servicios de despliegue de aplicaciones Web en la nube no suelen utilizarse para implementar el entorno de desarrollo que suelen implementarse utilizando herramientas de desarrollo específicos.

Este tipo de servicios en la nube proveen servidores Web virtuales y otros tipos de servicios que se puedan necesitar como, por ejemplo: servidores virtuales de acceso a bases de datos, servidores de aplicaciones, etc. Y, además, los servicios en la nube también proporcionan servicios de gestión del despliegue de aplicaciones Web, tales como ***Docker*** o ***Kubernetes***. Estos gestores de despliegue integrados en la nube automatizan todas las tareas de virtualización necesarias para dejar la aplicación Web preparada para su uso efectivo o explotación.

Los proveedores de servicios en la nube más utilizados actualmente son: Amazon Web Services (AWS), Microsoft Azure, y Google Cloud.



Fuente: https://www.flexera.com/blog/cloud/2019/02/cloud-computing-trends-2019-state-of-the-cloud-survey/

Ilustración 1.8. Modelo de desarrollo de software .NET.

En el año 2019, AWS continúa liderando el mercado de los servicios informáticos que ofrecen proveedores externos a través de la Internet pública (Nube Pública o *Public Cloud*). Sin embargo, puede observarse que su cuota de mercado se ha reducido y que otros servicios en la nube crecen más rápidamente.

La cuota de mercado de Azure crece del 45% al 52%, lo que reduce la brecha de mercado con AWS, tal como ha venido produciéndose en los años anteriores.

Por su parte, Google Cloud ha mantenido su tercera posición en el mercado, aumentando su cuota de mercado ligeramente, del 18% ha pasado al 19%.

En la ilustración anterior, puede apreciarse que VMware Cloud subió a la cuarta posición este año, aumentando al 12% desde el 8% y superando a IBM Cloud en cuota de mercado.

1.6 HERRAMIENTAS DE DESARROLLO

En el desarrollo de las aplicaciones Web interactivas pueden intervenir diferentes tecnologías de desarrollo Web: HTML, CSS, framework de diseño Web, scripts, programas escritos en diferentes lenguajes, componentes precompilados, bases de datos, recursos de diferentes formatos, etc. Por este motivo, para la selección del entorno de desarrollo en que se llevará a cabo la construcción de una aplicación Web interactiva debe tener en cuenta, entre otros aspectos más concretos: el conjunto de tecnologías de desarrollo y lenguajes de programación, el tipo de servidor, los módulos y extensiones del servidor necesarios, los permisos de acceso y ejecución, la localización de los datos utilizados por la aplicación Web y las herramientas de desarrollo que se van a emplear.

A la hora de proporcionar criterios para poder seleccionar las herramientas de desarrollo Web más adecuadas que se emplearán en el desarrollo de una aplicación Web interactiva concreta, es necesario hacer una primera clasificación, en función de sus capacidades, de las herramientas de desarrollo que pueden intervenir:

- **Navegadores Web.** Son los programas que constituyen el entorno del cliente, que permiten capturar las interacciones del usuario y presentar la información resultante en forma de documentos escritos en lenguaje HTML. Algunos de los navegadores Web más conocidos son: Microsoft Edge, Google Chrome, Mozila Firefox, Safari, Opera, etc.

- **Editores de código.** Se trata de editores de texto que permiten escribir el código empleando diversos lenguajes de programación, tales como: HTML, CSS, Javascript, PHP, etc. sin ningún tipo de ayuda ni facilidad adicional. Algunos de los editores de código más conocidos son: Notepad++, UltraEdit, Kompozer, Sublime Text, Brackets, Atom, Visual Studio Code, etc.

- **Herramientas específicas de diseño Web.** Son herramientas visuales específicas de ayuda a creación de páginas y sitios Web con visualización dinámica sobre entorno gráfico. Algunas de las herramientas más conocidas son: Adobe Dreamweaver, Google Web Designer, etc.

- **Entornos de desarrollo (IDE).** Son entornos integrados de desarrollo de software que permiten editar, compilar, ejecutar y probar el código editado empleando diferentes lenguajes y tecnologías de desarrollo de aplicaciones Web. Algunos de los entornos integrados de desarrollo más utilizados son: Visual Studio de Microsoft, NetBeans de Oracle, Eclipse de Eclipse Foundation, etc.

- **Herramientas de edición de contenidos multimedia.** El servicio de información Web es, por definición, multimedia. La mayoría de los sitios Web incorporan contenidos de naturaleza gráfica o multimedia (imágenes, víadeos, sonidos, etc.) que, por regla general, es necesario adecuar a las características de visualización y transmisión por la red mediante el uso de este tipo de herramientas que facilitan si tratamiento.

- **Herramientas para la creación, manipulación y administración de bases de datos.** Se trata de herramientas para la definición, manipulación y control de la información almacenada en las bases de datos que utilizará la aplicación Web.

1.7 ASPECTOS BÁSICOS DE ASP.NET

El modelo de desarrollo de software **.NET** de Microsoft constituye una plataforma para desarrollar aplicaciones de escritorio para Windows, aplicaciones Web, Servicios Web *(Web Services)* y aplicaciones para dispositivos móviles.

La infraestructura básica de la tecnología .NET se denomina **.NET Standard** que unifica las tres implementaciones específicas que la componen: **.NET Framework**, **.NET Core** y **Xamarin**. La implementación clásica .NET Framework facilita el desarrollo de aplicaciones de escritorio para Windows (WinForms y WPF) y aplicaciones Web mediante ASP.NET. El enfoque más moderno .NET Core es una implementación de .NET Standard para uso general, modular, multiplataforma y de código abierto que permite desarrollar aplicaciones de escritorio (UWP) y aplicaciones Web (ASP.NET Core) sobre plataformas Windows, Linux y Mac OS. La implementación Xamarin facilita el desarrollo de aplicaciones móviles para iOS de Apple, Android de Google y OS X de Apple. Las aplicaciones desarrolladas con .NET se desarrollan utilizando un conjunto de herramientas y código comunes, pudiendo ser integradas fácilmente entre sí.

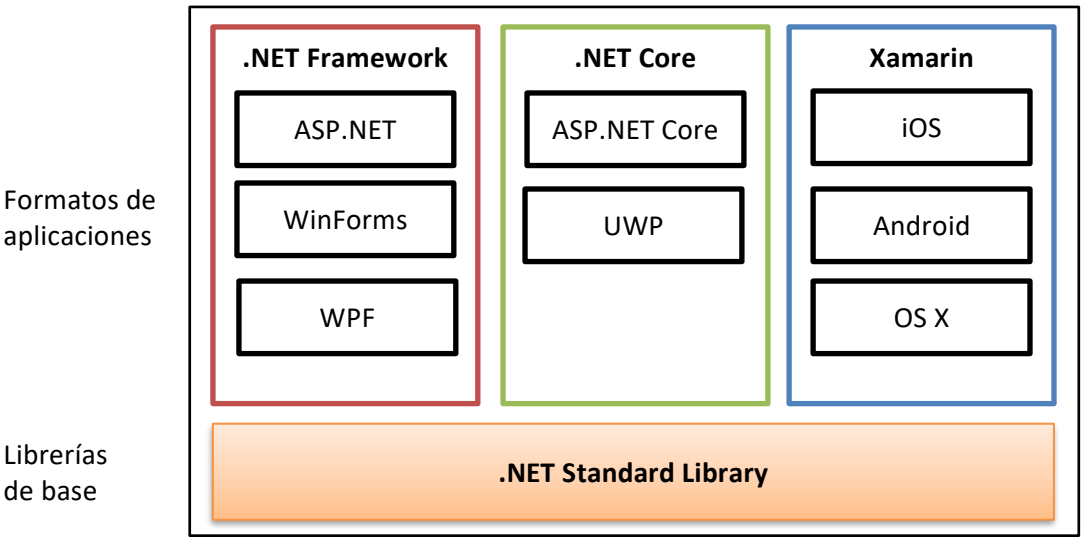


Ilustración 1.9. Modelo de desarrollo de software .NET.

Un componente básico del modelo de desarrollo .NET es ***Common Language Runtime*** **(CLR)** que proporciona el entorno de ejecución administrado del código nativo de .NET Framework. El componente CLR ejecuta el código precompilado, que se conoce como código administrado, lo que facilita la integración con una gran variedad de lenguajes de programación. Los compiladores que son compatibles con .NET traducen el código fuente escrito utilizando un lenguaje de programación al código administrado que ejecuta el componente CLR de .NET. Por este motivo la tecnología de desarrollo .NET es independiente del lenguaje de programación empleado para el desarrollo de las aplicaciones. En la actualidad existe una gran variedad de compiladores compatibles con .NET disponibles para multitud de lenguajes de programación.

La tecnología de desarrollo ASP.NET configura un modelo de desarrollo unificado que incluye los servicios necesarios para crear aplicaciones Web empresariales de altas prestaciones. Como forma

parte de .NET, al desarrollar aplicaciones Web con esta tecnología se tiene acceso a toda la funcionalidad que proporciona el uso de las clases de .NET Standard Library. Las características más significativas de la tecnología ASP.NET para el desarrollo de aplicaciones Web son los siguientes:

- Proporciona un modelo de programación orientado a objetos y manejado por eventos de aplicaciones Web basado en la plataforma .NET.

- Es independiente del lenguaje de programación empleado para expresar la lógica del negocio de la aplicación. Se pueden desarrollar aplicaciones Web en cualquiera de los lenguajes compatibles con .NET, como: C#, Visual Basic, PHP, J#, etc.

- Es independiente del navegador Web empleado como cliente.

- Es independiente de los dispositivos empleados: smartphone, tablet, PC, etc.

- Garantiza la separación de la capa de presentación de la capa de lógica del negocio, a través de la utilización de los archivos de código subyacente *(code-behind)*.

- Puede utilizar todas las funcionalidades que proporciona la librería de clases .NET para el desarrollo las aplicaciones Web, como por ejemplo el uso de las clases de la librería ADO.NET para acceso a datos almacenados y a servicios de datos.

- Se trata de una tecnología de desarrollo compilada, no interpretada.

- Conserva el contenido de los formularios (Web Forms) después de que sean enviados al servidor. Esta funcionalidad recibe el nombre de mantenimiento del estado de la página.

El proceso de compilación y ejecución de ASP.NET se describe a continuación. Cuando el cliente envía una solicitud de una página de ASP.NET al servidor Web, este localiza la página mediante su URL. El servidor Web analiza el código de la página solicitada para comprobar si ha sido modificado respecto de la última compilación. Si el código fuente ha sido modificado, se inicia el proceso de compilación, que incluye las llamadas al analizador *(parser)*, al compilador *(compiler)* y al ensamblador *(assembly cache)* para generar el código compilado. Si el código fuente no ha sido modificado desde la última compilación de la página de ASP.NET solicitada, el entorno de ejecución CLR *(Runtime)* del servidor carga y ejecuta el código administrado (precompilado) y, envía la respuesta al cliente. En las siguientes peticiones de la misma página .aspx, el CLR carga y ejecuta directamente el código administrado, siempre que no se hayan producido modificaciones en el código fuente de esa página. Finalmente, después de la ejecución del código administrado, la respuesta HTML resultante del procesamiento en el servidor Web es enviada al cliente para su presentación. La ventaja principal de este proceso de compilación y ejecución en el entorno del servidor Web consiste en que las modificaciones realizadas en el código de las aplicaciones Web puedan realizarse directamente sobre el entorno de producción durante la fase de explotación de la aplicación Web, sin necesidad de tener que detener del servidor Web para poder incorporarlas.

La Ilustración 1.9 muestra el proceso de compilación y ejecución de ASP.NET.

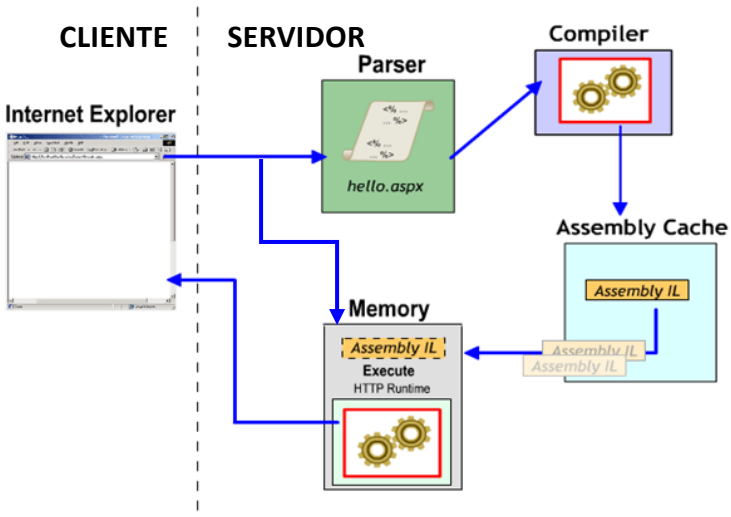


Ilustración 1.10. Proceso de compilación y ejecución de ASP.NET.

Los elementos más significativos que forman una aplicación Web basada en ASP.NET de la implementación .NET Framework son los siguientes:

- **Web Forms.** Proporcionan la interfaz de usuario para la aplicación Web, por lo que representan la capa de presentación de la aplicación Web. El término Web Form se emplea para referirse a las páginas de ASP.NET. Cada Web Form queda definido en un archivo cuya extensión es .aspx. Una aplicación Web está constituida por varios Web Forms a los que se accede mediante enlaces o *links* dispuestos en los propios Web Forms para facilitar el acceso a los procesamientos de la aplicación Web.

- **Archivos de código subyacente o code-behind.** Contienen el código lógico de servidor que representa la lógica del negocio de la aplicación. Cada archivo de código subyacente está asociados al Web Forms correspondiente a través de su nombre. La extensión del nombre de los archivos de código subyacente depende del lenguaje de programación empleado. Por ejemplo, la extensión de un archivo de código subyacente resuelto con C# será .aspx.cs.

- **Archivos de configuración con formato XML.** Los archivos de configuración son archivos XML que definen diversos aspectos de configuración de la aplicación Web y del servidor Web. Algunos de los archivos de configuración XML más significativos son el archivo Web.config y el archivo Machine.config que definen las características de configuración de la aplicación Web y del servidor Web, respectivamente.

- **Archivo global.asax.** El archivo global.asax contienen código de respuesta a los eventos a nivel de aplicación y a nivel de sesión.

- **Conectividad a datos.** Los mecanismos de conectividad facilitan el establecimiento de conexiones con servidores de bases de datos, documentos XML y servicios de datos para acceder a la información que maneja desde las aplicaciones Web basadas en ASP.NET.

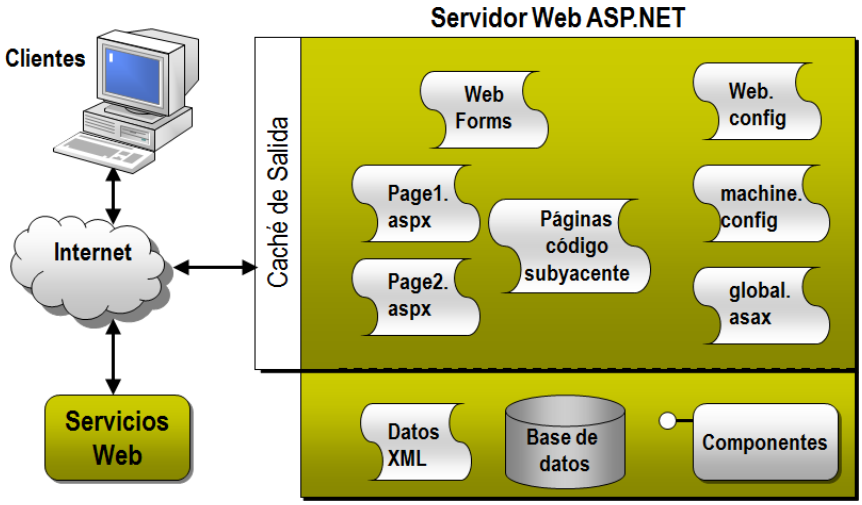


Ilustración 1.11. Elementos que forman una aplicación Web basada en ASP.NET (.NET Framework).

En la Ilustración 1.11 puede apreciarse la evolución de la tecnología .NET Framework a lo largo del tiempo. En todos los casos, las aplicaciones Web desarrolladas con ASP.NET mantienen una compatibilidad ascendente. El entorno de desarrollo Visual Studio .NET incorpora utilidades para la gestión y conversión de las aplicaciones Web a las nuevas versiones de .NET Framework.

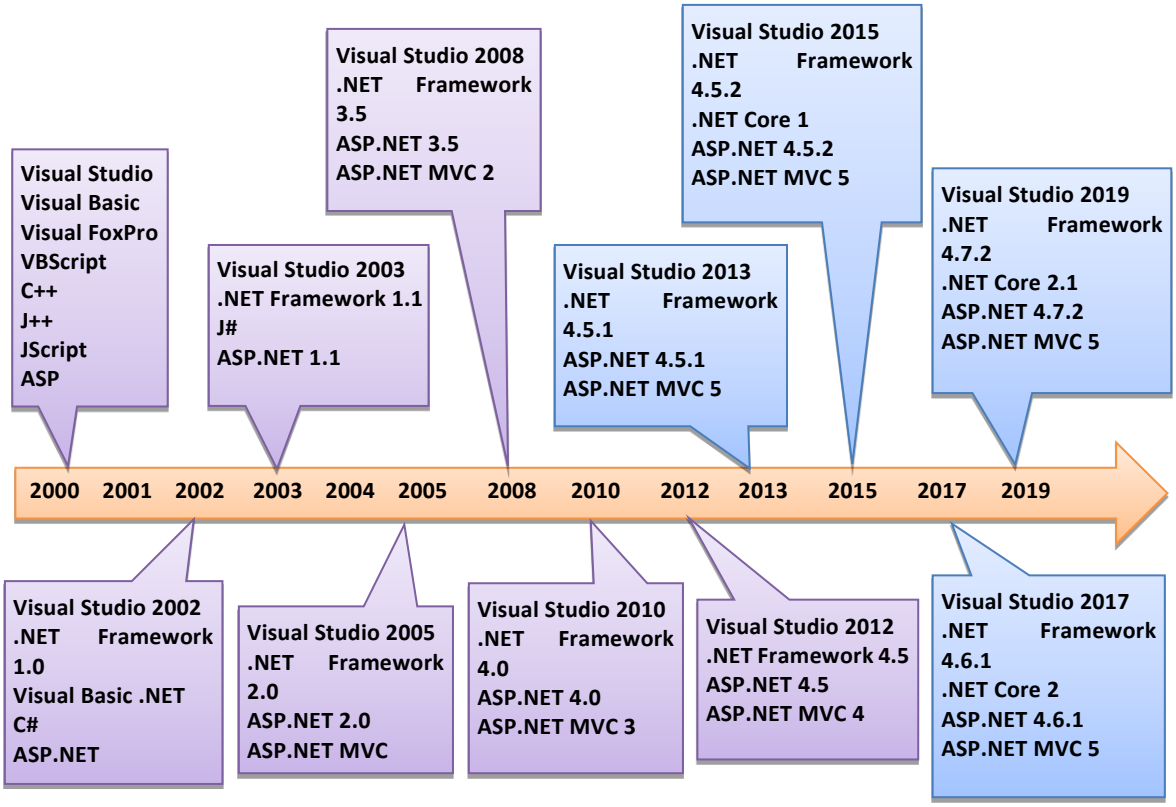


Ilustración 1.12. Línea de tiempo de ASP.NET.

1.8 USO DE MICROSOFT VISUAL STUDIO

El modelo de desarrollo de .NET comparte un único Entorno Integrado de Desarrollo (IDE) denominado Visual Studio. Este Entorno Integrado de Desarrollo se compone de varios elementos: la barra de menús, las barras de herramientas, varias ventanas que se acoplan u ocultan automáticamente a la izquierda, en la parte inferior y a la derecha, así como el espacio del editor. Las ventanas de herramientas, menús y barras de herramientas se adaptan al tipo de proyecto o de archivo que se esté desarrollando.

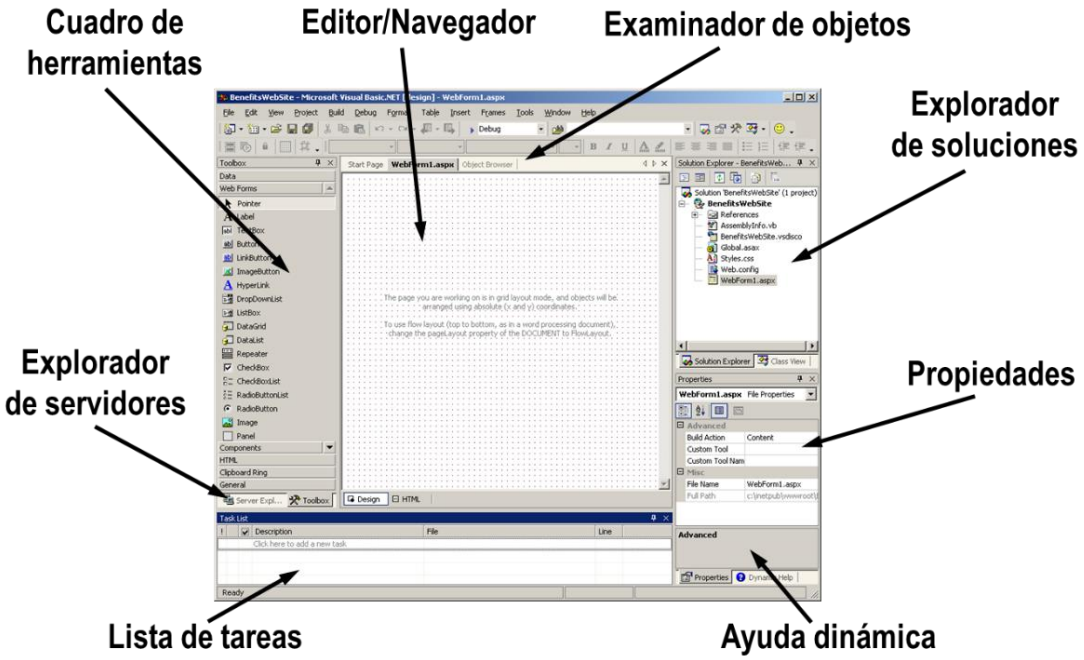


Ilustración 1.13. Entorno Integrado de Desarrollo (IDE) Visual Studio .NET.

Las características más significativas de Visual Studio en relación con el desarrollo de aplicaciones Web basadas en la tecnología de desarrollo ASP.NET son las siguientes:

- Soporta múltiples tipos de proyectos de desarrollo de aplicaciones de escritorio, aplicaciones Web, Servicios Web y aplicaciones para dispositivos móviles.
- Pueden emplearse múltiples lenguajes de programación en un mismo proyecto.
- Incluye herramientas para gestionar y administrar el acceso a datos de las aplicaciones.
- Incorpora navegador Web, servidor Web de desarrollo y servidor de bases de datos SQL Server integrados y adaptados al entorno de desarrollo, lo que configura un entorno de desarrollo completo para el desarrollo de aplicaciones Web.
- Incorpora herramientas y utilidades que facilitan la compilación de los Proyectos de aplicación Web basadas en ASP.NET, así como el despliegue en el entorno de producción.

- Da soporte a la realización pruebas unitarias y conjuntas e incorpora herramientas de depuración.

- Dispone de editores visuales y de código para el desarrollo de aplicaciones Web.

El IDE de Visual Studio incorpora herramientas y utilidades que facilitan las tareas a realizar por desarrollador durante todas las fases del proceso de desarrollo de software. La Ilustración 1.13 muestra un esquema del conjunto de las fases que constituyen el proceso de desarrollo de las aplicaciones informáticas.

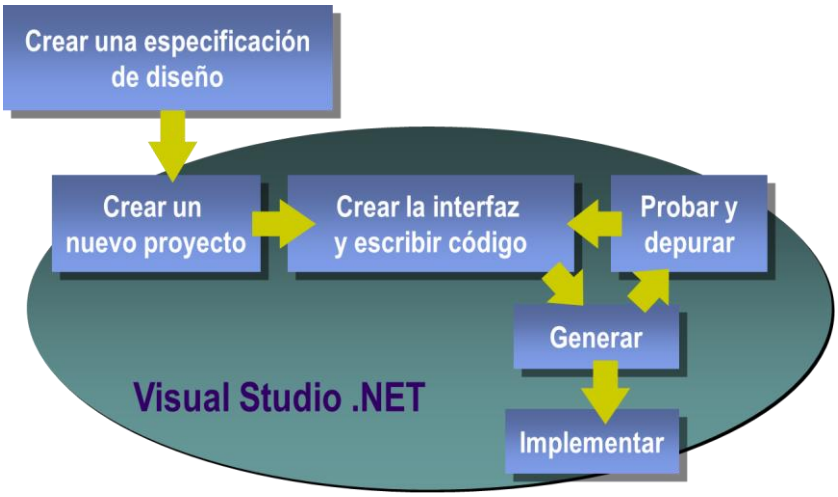


Ilustración 1.14. Fases del proceso de desarrollo de software.

Las Soluciones y los Proyectos de Visual Studio contienen elementos en forma de referencias, conexiones de datos, carpetas y archivos necesarios para crear la aplicación. Una Solución es un contenedor que encapsula uno o varios Proyectos. A su vez, un Proyecto es un contenedor que encapsula el conjunto de elementos que conforman la aplicación que se está desarrollando. Esta estructura de contenedores facilita la organización de los elementos que se están desarrollando y la realización de las tareas comunes del desarrollo. El resultado de un Proyecto puede ser un programa ejecutable (.exe), un archivo de biblioteca de vínculos dinámicos (.dll) o un módulo, según el tipo de proyecto que se esté desarrollando. La estructura de los archivos que forman parte del desarrollo de una aplicación Web basada en ASP.NET de .NET Framework puede contener los siguientes elementos más significativos:

- Archivos de Solución (.sln, .suo)
- Archivos de Proyecto (.vbproj, .csproj)
- Web Forms o páginas de ASP.NET (.aspx)
- Archivos de código subyacente (aspx.cs, aspx.cs, etc.)
- Archivos de definición de hojas de estilo en cascada (.ccs)
- Archivo de configuración de la aplicación (Web.config)
- Archivo de clases globales de aplicación (global.asax)
- Archivo de ensamblado del proyecto (.dll)

Glosario de términos

Computación en la nube *(Cloud Computing)*. Conocida también como Servicios en la nube, Informática en la nube, Nube de cómputo, o simplemente «la nube», es un paradigma que permite ofrecer servicios de computación a través de una red, que usualmente es Internet.

CSS *(Cascading Style Sheets).* Lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML, y por extensión en XHTML, a través de la especificación de las características de estilo asociadas a cada uno de los elementos contenidos en el documento HTML o XML. La idea subyacente que se encuentra detrás del uso de las Hojas de Estilo en Cascada (CSS) es separar el contenido y la estructura de un documento HTML o XML de su presentación.

Framework. Traducido como Marco de trabajo o mejor como Entorno de trabajo, es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar. En la disciplina de desarrollo de software, un *Framework* es una estructura conceptual y tecnológica de asistencia definida, normalmente, con módulos de software, que puede servir de base para la organización y el desarrollo de soluciones software más concretas. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto de software.

HTML *(HyperText Markup Language).* Lenguaje de marcado predominante para la elaboración de páginas Web que se utiliza para especificar la presentación y la información contenida. El código HTML se escribe en forma de etiquetas o marcas, rodeadas por corchetes angulares (<, >).

Infraestructura como servicio *(IaaS, Infrastructure as a Service)*. Se trata de un concepto del nuevo paradigma de computación en la nube *(Cloud Computing)*. La infraestructura como servicio (IaaS) es una infraestructura informática inmediata que se aprovisiona y administra a través de Internet. Permite reducir o escalar verticalmente los recursos para ajustarlos a la demanda y se basa en concepto de pago por uso. Evita el gasto y la complejidad que suponen la compra y administración de sus propios servidores físicos y el resto de la infraestructura física de un centro de datos. Cada recurso se ofrece como un componente de servicio aparte, de modo que solo se contrata cada recurso concreto durante el tiempo que se necesite. El proveedor de servicios en la nube administra la infraestructura, mientras que el usuario del servicio puede instalar, configurar y administrar su propio software (sistemas operativos, middleware y aplicaciones).

Interfaz de usuario *(User Interface)*. Procesamiento (programa informático) que actúa empleando un conjunto de imágenes y objetos para presentar la información y las acciones disponibles que puede realizar el usuario. Su principal uso, consiste en proporcionar un entorno visual que facilite la comunicación del usuario con una aplicación informática o con el sistema operativo, de modo que transformen un sistema automatizado no humano en una herramienta convivencial. En la actualidad, dado que la mayor parte de las interfaces de usuario son de tipo gráfico, éstas suelen recibir la denominación de G*raphical User Interface* (GUI) o interfaz gráfica de usuario.

Protocolo de comunicaciones. Conjunto de reglas y estándares que gobiernan el intercambio de información entre entidades de una red. Existen muchos tipos de protocolos de comunicación, cada uno de los cuales se ha especificado con un propósito específico: FTP, POP3, SMTP, ICMP, etc.

Protocolo HTTP *(HyperText Transfer Protocol)*. Protocolo de la capa de aplicación usado en cada transacción Web. Cada petición realizada mediante HTTP implica una conexión con el servidor que es liberada al término de la misma, es decir, no tiene estado.

Protocolo IP *(Internet Protocol)*. Protocolo de comunicaciones no orientado a conexión, usado tanto por el origen como por el destino para la comunicación de datos, a través de una red de paquetes conmutados no fiable y de mejor entrega posible sin garantías. El protocolo IP no provee ningún mecanismo para garantizar la recepción del paquete en su destino. La fiabilidad en la transmisión de datos es proporcionada por otros protocolos de la capa de transporte, como TCP. Los datos en una red basada en IP son enviados en bloques conocidos como paquetes o datagramas. Los aspectos principales del protocolo son el direccionamiento y el enrutamiento. El direccionamiento se refiere a la forma como se asigna una dirección IP y cómo se estructuran las subredes de equipos. El enrutamiento, que es realizado por enrutadores fundamentalmente, consiste en encontrar un camino que conecte una red con otra. Los enrutadores *(routers)* son unos dispositivos especializados en recibir y enviar paquetes por diferentes interfaces de red y, para proporcionar opciones de seguridad, redundancia de caminos y eficiencia en la utilización de los recursos.

Protocolo TCP *(Transmission Control Protocol)*. Protocolo de comunicaciones orientado a conexión y fiable del nivel de transporte que da soporte a la creación de conexiones entre los ordenadores de una red a través de las cuales puede enviarse un flujo de datos. Muchas de las aplicaciones más populares de Internet (navegadores, intercambio de ficheros, clientes FTP, etc.) y los protocolos de la capa de aplicación HTTP, SMTP, SSH y FTP se basan en protocolo TCP.

Runtime. También denominado *Runtime system* o *Runtime environment.* Componente software diseñado para apoyar la ejecución de programas escritos en algún lenguaje de programación. Contiene implementaciones básicas de comandos de bajo nivel y puede ejecutar comandos de nivel superior. En la mayoría de los casos puede apoyar la depuración, la verificación de tipos y la optimización del código. Proporciona una capa de abstracción que oculta la complejidad o las variaciones en los servicios ofrecidos por el sistema operativo. En algunos casos, el *runtime* puede ser una máquina p-code o máquina virtual que oculta incluso el conjunto de instrucciones del procesador. Este es el enfoque adoptado por lenguajes de programación como Java que están destinados a ser compilado en *bytecode* o pseudocódigo independiente de la máquina. El sistema de apoyo a la ejecución de programas basado en *runtime* simplifica la tarea de implementación del lenguaje y su adaptación a diferentes máquinas y sistemas operativos, al permitir que el mismo programa pueda ser ejecutado en cualquier máquina o sistema operativo sin recompilación. En estos casos, tan sólo es necesario instalar el *runtime* adecuado para esa máquina y sistema operativo, antes de lanzar la ejecución de programa ya compilado con ayuda del *runtime* instalado.