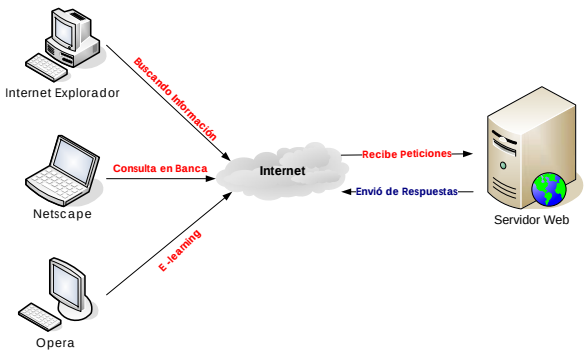
CAPITULO I

Aplicaciones en N-Capas



1. Aplicaciones N -Capas
2. J2EE
3. Comparativa J2ee y Microsoft .Net

1.1 APLICACIONES N-CAPAS

1.1.1 Introducción

Después del gran apogeo que tuvieron las aplicaciones ***cliente/servidor***, el aparecimiento del ***Internet*** y los ***sistemas distribuidos*** multiplataforma han representado la punta del iceberg del desarrollo una nueva generación de aplicaciones, incluso las ***bases de datos*** y las herramientas de desarrollo están migrando hacia esta arquitectura dada las limitaciones de la cliente/servidor o dos capas.

El modelo n-capas ha emergido como la arquitectura predominante para la construcción de aplicaciones ***multiplataforma*** en la mayor parte de las empresas, cuando hablamos de este modelo aparece también involucrado otros elementos entre ellos componentes de la aplicación en diferentes servidores: entregar los datos, validar las normas del negocio y asegurarse de que las transacciones se procesen de la manera debida, generar los reportes, o los formularios de entrada, etcétera.

La incorporación a esta arquitectura del Internet permite crear aplicaciones distribuidas en el Internet tales como: ***e-comercio, e-bussines*** *y* ***e-learning,*** lo que se facilita por la esencia del modelo ya que se permite la separación de capas, manteniendo cada componente tan separado del contexto global como sea posible. **[Lib001]**

1.1.2 Características de las Arquitectura N-capas

Las aplicaciones n-capas proporcionan una gran cantidad de beneficios para las empresas que necesitan soluciones flexibles y fiables para resolver complejos problemas inmersos en cambios constantes.

Entre las principales características de las arquitecturas n-capas tenemos:

Clientes ligeros

Todas las aplicaciones basadas en n-capas permitirán trabajar con clientes ligeros, tal como navegadores de Internet, WebTV, Teléfonos Inteligentes, **PDA**s (Asistentes Personales Digitales) y muchos otros dispositivos preparados para conectarse a Internet.

Red

Las arquitecturas basadas en n-capas permiten a los componentes de negocio correr en una **LAN, WAN** o Internet. Esto significa que cualquiera con un ordenador y conexión a la Red posee toda la funcionalidad que tendría si se encontrase delante de su sistema de escritorio.

Subdivisión de sistemas

Los sistemas de n-capas subdivididos ayudan a facilitar el desarrollo rápido de aplicaciones y su posterior despliegue, con beneficios incrementales fruto de los esfuerzos

del desarrollo en paralelo coordinado y del outsourcing inteligente, resultando un enorme decremento del tiempo de desarrollo y de sus costes.

La arquitectura de n-capas provee flexibilidad, rendimiento y seguridad en el diseño así como soporte para estándares de desarrollo abiertos (independientemente de base de datos, lenguaje o sistema operativo).

1.1.3 Estructura de la Arquitectura N-Capas

La arquitectura n-capas forma parte también de un revolucionario proceso basado en la aplicación de estas nuevas tecnologías (componentes y estándares de Internet). Estas tecnologías son los bloques para crear Software de Negocio y Sistemas de Información adaptables que ayuden a las empresas a integrar todos sus sistemas de Tecnologías de la Información, así como las inversiones realizadas en éstos, mientras que obtienen una ventaja clara en el uso de Internet.

La separación de la presentación, lógica de negocio y datos es realizada en un número indefinido de capas lógicas, permitiendo a cada capa ser desarrollada, mejorada, gestionada y desplegada de forma independiente. Esta es precisamente la base para el modelo de informática de red en n-capas. Las plataformas multicapa funcionan consistentemente a lo largo de un variado conjunto de hardware, permitiendo escalar las operaciones del negocio desde un simple portátil, hasta un ***DataCenter***, desde el dispositivo más simple hasta el más complejo de los ***Mainframes***.

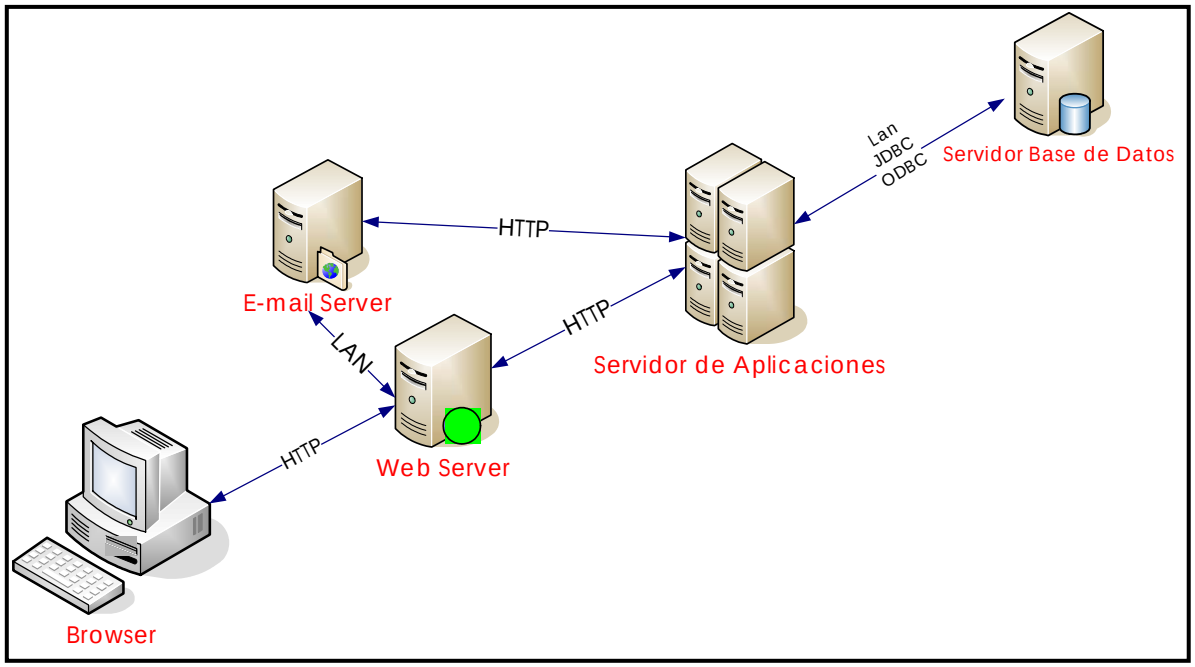


Figura1.1: Arquitectura N-Capas

En la figura1.1 podemos ver un ejemplo de la estructura n-capas donde se aprecia la capa de presentación, lógica de negocios y acceso a datos.

1.1.3.1 Capa de Presentación

Es la encargada de los servicios de presentación, proporciona la interfaz necesaria para presentar información y reunir datos. También aseguran los servicios de negocios necesarios para ofrecer las capacidades de ***transacciones*** requeridas e integrar al usuario con la aplicación para ejecutar un proceso de negocios. **[www021]**

Los servicios de presentación generalmente son identificados con la interfaz de usuario, y normalmente residen en un programa ejecutable localizado en la estación de trabajo del

usuario final. Aún así, existen oportunidades para identificar servicios que residen en componentes separados.

El cliente proporciona el contexto de presentación, generalmente un navegador como Microsoft Internet Explorer o Netscape Navigator, que permite ver los datos remotos a través de una capa de presentación **HTML**, o también una aplicación WIN32 como son los formularios de Visual Basic.

La capa de aplicaciones cliente se compone de aplicaciones cliente (como un pedido o mantenimiento de productos) las cuales se crean a partir de componentes de aplicaciones cliente.

La capa de presentación es responsable de:

- ✓ Obtener información del usuario.
- ✓ Enviar la información del usuario a los servicios de negocios para su procesamiento.
- ✓ Recibir los resultados del procesamiento de los servicios de negocios.
- ✓ Presentar estos resultados al usuario.

1.1.3.2 Capa de Negocios

Se encarga de los servicios de negocios, son el "puente" entre un usuario y los servicios de datos. Responden a peticiones del usuario (u otros servicios de negocios) para ejecutar una tarea. Cumplen con esto aplicando procedimientos formales y reglas de negocio a los datos relevantes. Cuando los datos necesarios residen en un servidor de bases de

datos, garantizan los servicios de datos indispensables para cumplir con la tarea de negocios o aplicar su regla. Esto aísla al usuario de la interacción directa con la base de datos.

Una tarea de negocios es una operación definida por los requerimientos de la aplicación, como introducir una orden de compra o imprimir una lista de clientes. Las reglas de negocio son políticas que controlan el flujo de las tareas.

Como las reglas de negocio tienden a cambiar más frecuentemente que las tareas específicas de negocios a las que dan soporte, son candidatos ideales para encapsularlas en componentes que están lógicamente separados de la lógica de la aplicación en sí.

La capa del servidor de negocios se compone de servidores de negocios (como el proceso de órdenes y el manejo del almacén) la cual se crea a partir de componentes de aplicaciones de servidor de negocios. **[www021]**

El nivel de servicios de negocios es responsable de:

- ✓ Recibir la entrada del nivel de presentación.
- ✓ Interactuar con los servicios de datos para ejecutar las operaciones de negocios para los que la aplicación fue diseñada a automatizar (por ejemplo, la preparación de impuestos por ingresos, el procesamiento de ordenes y así sucesivamente).
- ✓ Enviar el resultado procesado al nivel de presentación.

1.1.3.3 Capa de Acceso a Datos

La capa del servidor de datos se compone de servidores de datos (como órdenes y productos) que se crean a partir de componentes de servidores de datos. En esta capa es donde van ha residir los datos, es también utilizada en la arquitectura cliente servidor. **[www021]**

El nivel de servicios de datos es responsable de:

- ✓ Almacenar los datos.
- ✓ Recuperar los datos.
- ✓ Mantener los datos.
- ✓ La ***integridad*** de los datos.

En una arquitectura tradicional, una capa puede comunicarse sólo con otra directamente arriba o abajo de ella. En este caso los servicios de usuarios, de negocios y de datos pueden comunicarse con ellos mismos. Este modelo se conoce como el modelo de servicios, dado que, lejos del comportamiento de un modelo de capas, cualquier servicio puede invocar a otro dentro de su capa.

Lo que realmente es nuevo en el modelo de n-capas es la posibilidad de distribuir objetos independientes sobre el número de capas que sean necesarias y enlazarlas dinámicamente, cuando sea necesario, para proporcionar una flexibilidad ilimitada a la aplicación.

1.2 JAVA 2 EDICIÓN EMPRESARIAL (J2EE)

1.2.1 Introducción

Las aplicaciones para las empresas han evolucionado por la influencia de la tecnología del lado del servidor lo que ha aumentado la velocidad, seguridad, y fiabilidad. Este fenómeno de cambio rápido en el mundo exigente de e-comercio, e-información y la tecnología, ha obligado a que las aplicaciones de las empresas se construyan con la mayor velocidad y con menos recursos.

J2EE es una ***plataforma estándar*** para desarrollar y desplegar aplicaciones empresariales, maneja modelos de la aplicación re-usable como los componentes, una seguridad unificada el mando de la transacción y servicios Web, el apoyo a través de los datos integrados por medio del ***XML***.

J2EE no es solo un conjunto de ***APIs*** para el desarrollo de aplicaciones distribuidas, sino también una infraestructura dentro de la cuál se ejecutan dichas aplicaciones: esta infraestructura la proporciona un tipo especial de aplicaciones, llamadas Servidores de Aplicación (Application Servers), tales como ***WebSphere*** de IBM, Inprise Application Server de Borland, ***Tomcat- Apache***.

J2EE, plataforma creada por SUN en el año 1997 es la que ofrece mejores perspectivas de desarrollo para empresas que quieran basar su arquitectura en productos basados en ***software libre***.

1.2.2 Que es J2EE

J2EE resuelve el problema del costo y la complejidad en el desarrollo de servicios multicapa, que sean escalables, de alta disponibilidad, seguros y eficientes. Consigue esto proporcionando una arquitectura de estándar abierto a través de la Plataforma J2EE. Esta plataforma permite a los desarrolladores enfocarse en la ***lógica de negocio*** mientras que J2EE maneja los detalles de bajo nivel. Con J2EE, los servicios son fácilmente mejorables y rápidamente desarrollados, permitiendo a los negocios reaccionar rápidamente ente los cambios competitivos. **[www003]**

J2EE es un entorno abierto para desarrollar y desplegar servicios multi-capa donde pequeñas aplicaciones cliente invocan lógica de negocio que se ejecuta en un servidor de aplicaciones. Comprende un conjunto de servicios, protocolos e interfaces de programación. El lenguaje ***Java***, la máquina virtual Java y los componentes ***Java Beans*** son la base de J2EE.

1.2.3 Características

Una aplicación distribuida de cierto tamaño, además de dar respuesta a las necesidades concretas para la que ha sido diseñada, necesitará enfrentarse y resolver toda una serie de cuestiones técnicas que contribuyen significativamente a aumentar la dificultad del desarrollo, entre están:

- ✓ Soporte para distribución de objetos.
- ✓ La necesidad de guardar y recuperar objetos o persistencia (típicamente utilizando una Base de Datos).
- ✓ El soporte para ***concurrencia*** y ***seguridad***.
- ✓ El soporte para transacciones.

El soporte para poder encontrar objetos o recursos distribuidos (para encontrar una impresora color, o para poder encontrar a los Clientes, incluso aunque se cambie el servidor donde residen). **[www002]**

1.2.3.1 Soporte de Distribución

Dentro de la arquitectura J2EE la comunicación entre objetos en distintas máquinas resulta bastante transparente, y el código fuente raramente tendrá que tener en cuenta esto: si el programador hubiese de codificar el mecanismo de distribución, esto haría el desarrollo prácticamente imposible.

1.2.3.2 Persistencia

Uno de los elementos importantes dentro de la plataforma J2EE es el soporte para guardar y recuperar objetos (a la capacidad de un objeto para guardarse/leerse de un dispositivo de almacenamiento se le llama persistencia).

El soporte de ***persistencia*** dentro de J2EE se puede llevar a cabo de varios modos.

En primer lugar, algunos servidores de aplicación proporcionan soporte de modo más o menos automático es decir que permiten especificar que un ***EJB*** **(**Enterprise Java Bean) se guarde como un registro de una tabla determinada, indicando a qué columna del registro va cada campo del objeto.

También es posible hacer que sea el propio objeto el que se responsabilice de su persistencia: para esto se podría utilizar por ejemplo el API ***JDBC***, que encapsula el acceso a bases de datos relacionales de diversos fabricantes.

1.2.3.3 Seguridad

J2EE permite limitar el acceso a partes sensibles de un sistema de forma excepcionalmente sencilla.

Para cada método de un Enterprise Bean es posible especificar qué roles tienen acceso a dicho método. Cada vez que se llama a un método de un EJB se verifica si la persona que llama a dicho método tiene alguno de los roles autorizados, de modo que si no es así se elevará una excepción.

Para indicar los roles que tienen permiso para acceder a cada método no es necesario escribir ningún código fuente, sino que esto se especifica en un archivo en formato XML, llamado descriptor de despliegue.

Podemos notar que este sistema de roles es muy sencillo de utilizar. Cuando una aplicación se distribuya, la persona encargada de instalarla en un servidor de aplicaciones tendrá simplemente que decidir qué personas tienen cada rol. Si en lugar de trabajar a nivel de roles se trabajara a nivel de usuario, la persona encargada de la instalación tendría que decidir para cada método que usuarios tienen permisos.

1.2.3.4 Soporte para Concurrencia

El uso de sistemas distribuidos implica que muchos usuarios pueden estar accediendo a la información (objetos) al mismo tiempo. J2EE proporciona este soporte sin tener la necesidad de bloquear un objeto mientras es utilizado por un cliente, lo que permite que otro cliente pueda utilizar el mismo objeto.

El soporte de concurrencia es mas optimo debido a que los EJB no crean hilos de control en ninguno de sus métodos. Adicionalmente, el servidor de aplicaciones puede detectar casos en los que se podría producir un bloqueo, y elevará una excepción.

1.2.4 Arquitectura de J2EE

En un modelo multicapa el cliente que normalmente es un navegador Web o una aplicación Java invoca a la lógica del negocio de una o más capas medias que se están ejecutando sobre hardware dedicado, que a su vez acceden a los datos desde el Servidor de Base de Datos en la tercera capa. **[www006]**

Desarrollar un servicio multicapa requiere aplicaciones cliente, lógicas de negocio y de presentación (las aplicaciones que obtienen, actualizan y presentan los datos) y código de infraestructura. La infraestructura son componentes de bajo nivel del sistema que accede a varias bases de datos, recursos del sistema y proporcionan seguridad.

En la capa media, la lógica de negocio se implementa como componentes Enterprise Java Beans (EJB), mientras que la lógica de presentación se implementa como ***Java Server Pages*** (JSP) y ***Servlets***. Los Servlets y las JSPs permiten la separación del procesamiento de la solicitud de su lógica de presentación. La capa de presentación del modelo permite fácilmente acceder a las funciones de negocio de la capa media. La tecnología JSP permite a los desarrolladores presentar páginas Web creadas dinámicamente. Los servlets permiten a los desarrolladores crear presentaciones dinámicas para los usuarios completamente en lenguaje Java.

A continuación se ilustra los componentes de J2EE con los cuales se puede realizar aplicaciones n-capas.

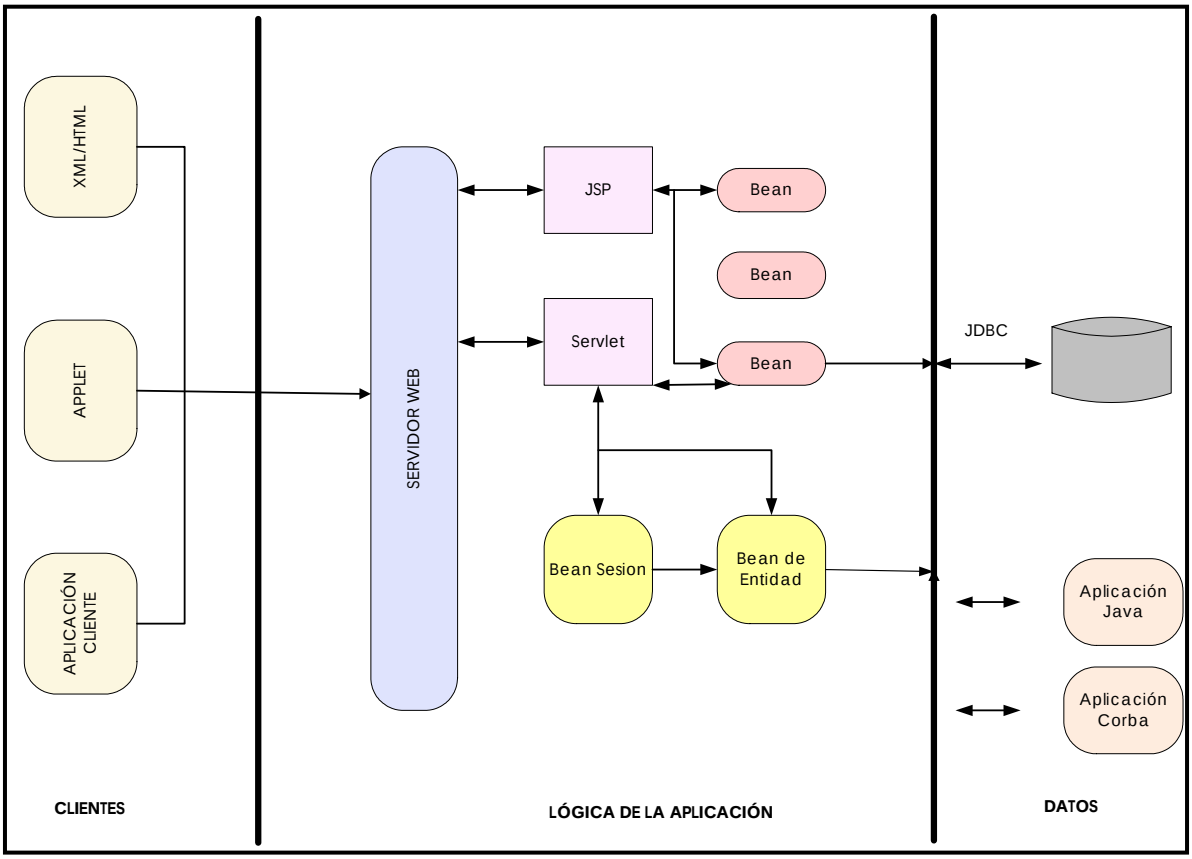


Figura 1.2: Entorno J2EE

En la figura 1.2 se presenta una aplicación donde interactúan los componentes de J2EE los cuales son:

- ✓ Servlets.- sirve para crear aplicaciones Web dinámicas; a diferencia de JSP este utiliza únicamente Java.
- ✓ JavaServer Pages (JSP).- similar a los script del lado de servidor que permite generar paginas Web dinámicas.

- ✓ Enterprise JavaBeans (EJB).- control de sesión del lado del servidor, que encapsula la lógica de negocios y abstracción para acceder a datos persistentes.
- ✓ Java Database Connectivity (JDBC).- un API que describe una librería estándar Java para acceder a fuentes de datos.
- ✓ Transaction Support(***JTA***).- transacciones declarativas para componentes donde las transacciones pueden expandir componentes y procesos.
- ✓ Java Naming and Directory Interface (***JNDI***).- una interfaz abstracta para servicios de búsqueda de uniones de nombres y directorios.
- ✓ Remote Method Invocation (***RM/IIOP***).-una tecnología que permite la comunicación entre objetos distribuidos.
- ✓ ***CORBA*** Compatible.- CORBA complementa a Java proporcionando un marco de trabajo de objetos distribuidos, servicios para soportar ese marco de trabajo e interoperabilidad con otros lenguajes.

En el capitulo II se describirá a detalle los componentes de J2EE, así como también su funcionamiento.

Aplicaciones Multicapa con J2EE

La plataforma de J2EE usa un modelo de aplicación de multicapas distribuidas para la aplicación de la empresa. La lógica de la aplicación es dividida en los componentes según funcione, y los varios componentes de la aplicación que constituyen una aplicación de J2EE se instalan en máquinas diferentes que dependen de la capa. Las capas que considera J2EE son:

- ✓ Capa Cliente componentes corridos en la máquina del cliente.
- ✓ Capa Web componentes corridos en el servidor de J2EE.
- ✓ Capa Negocio componentes corridos en el servidor de J2EE
- ✓ Capa en el servidor de base de datos

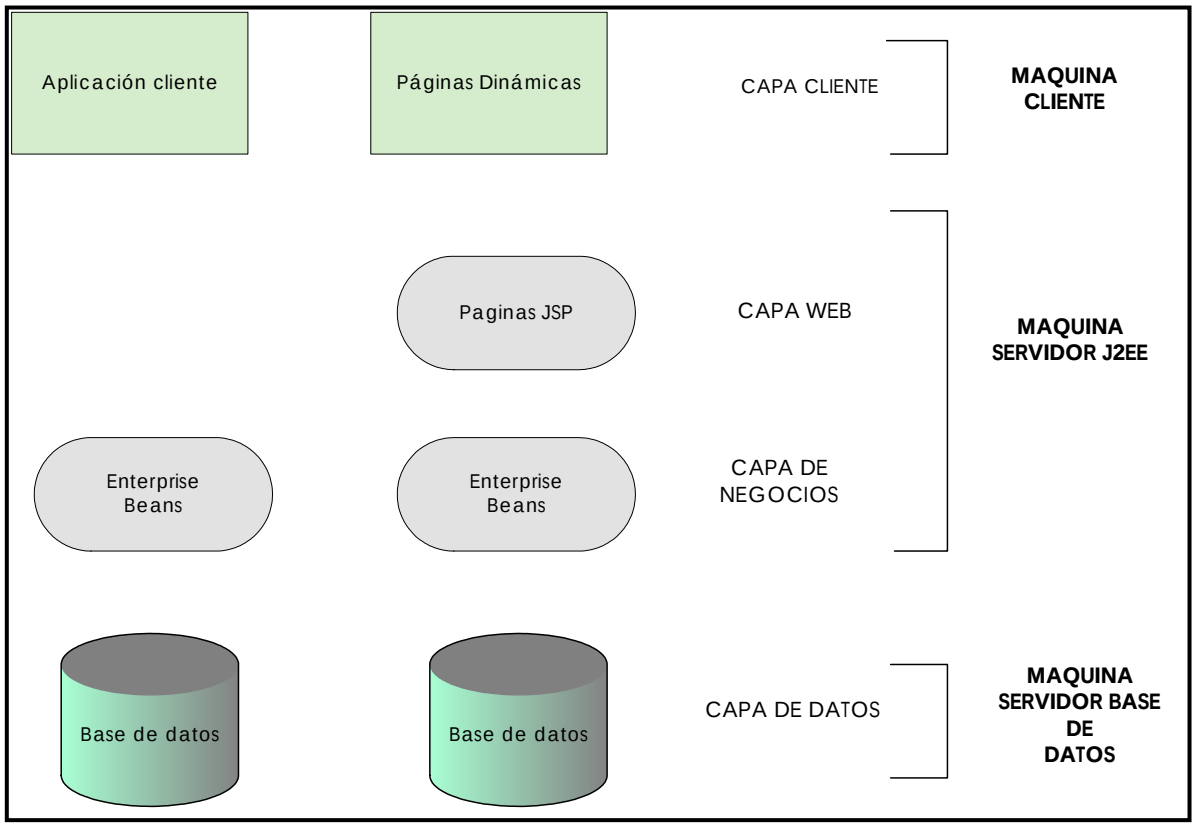


Figura 1.3: Aplicaciones Multicapas

La arquitectura de J2EE, que está basada en componentes, hace muy sencillo el desarrollo de este tipo de aplicaciones porque la lógica de negocios está organizada dentro de ***componentes reutilizables*** y el servicio subyacente lo proporciona el J2EE en la forma de un contenedor por cada tipo de componente. Pensemos en un contenedor como una interfaz entre el componente y la funcionalidad de bajo-nivel que soporta el componente. Por lo tanto, antes de poder ejecutar un componente de una aplicación cliente, debe configurarse como un servicio J2EE y desplegarse dentro de su contenedor. **[www006]**

1.3 COMPARATIVA J2EE Y MICROSOFT.NET

En la construcción de Sistema Web hay que tomar en consideración no solo la presentación sino además los servicios de Web que posean estas características: fiables, disponibilidad, sin errores, escalables. Estas necesidades no son diferentes que las de cualquier otra aplicación de la empresa.

J2EE y el DOT NET son las evoluciones de tecnología de servidor de aplicación existente la visión compartida entre J2EE y .NET es que hay una cantidad increíble de recursos para construcción ***aplicaciones Web***, como la interoperabilidad de XML, equilibrio de carga y transacciones. En lugar de escribir toda la aplicación, usted puede escribir una aplicación que corre dentro de un contenedor que mantiene esos servicios.

1.3.1 Plataforma de J2EE

La plataforma Java 2 Enterprise Edition (J2EE) fue diseñada para simplificar los problemas complejos con el desarrollo, despliegue, y dirección de soluciones de empresa de multicapas. J2EE es una norma de industria, y es el resultado de una iniciativa llevado por Sun Microsystems.

Es importante comprender que J2EE es una norma, no un producto. Por lo tanto no se puede descargar a J2EE, únicamente se descarga un archivo que contenga la descripción de la arquitectura y contenedores. Así J2EE, pueden desplegarse y desarrollarse en un variedad de ambientes de contenedores.

La meta de J2EE es proporcionar al cliente la capacidad de elegir entre los productos y herramientas de varios vendedores, para que el cliente escoja la mejor opción y de esa manera aumentar la competencia. Además, para una venta segura **Sun** colaboró con otros vendedores de plataformas e-Business como ***Oracle,*** BEA***, IBM***. Otro factor importante de J2EE es que es constantemente mejorado a través de ***Java Community Process*** que recibe ideas y comentarios tanto de las empresas desarrolladoras así como de desarrolladores independientes.

1.3.1.1 J2EE y Servicios de Web

J2EE es una arquitectura utilizada para la construcción de aplicaciones del lado del servidor. Puede usarse además para la construcción de sitios Web Tradicionales, componentes de software o puede empaquetar aplicaciones. J2EE se ha extendido a la construcción de Sitios Web basados en XML, eso le permite a los servicios interactuar con otros servicios que no hayan sido desarrollados con J2EE. **[www004]**

Modelo de desarrollo Web Con J2EE en Figura 1.4

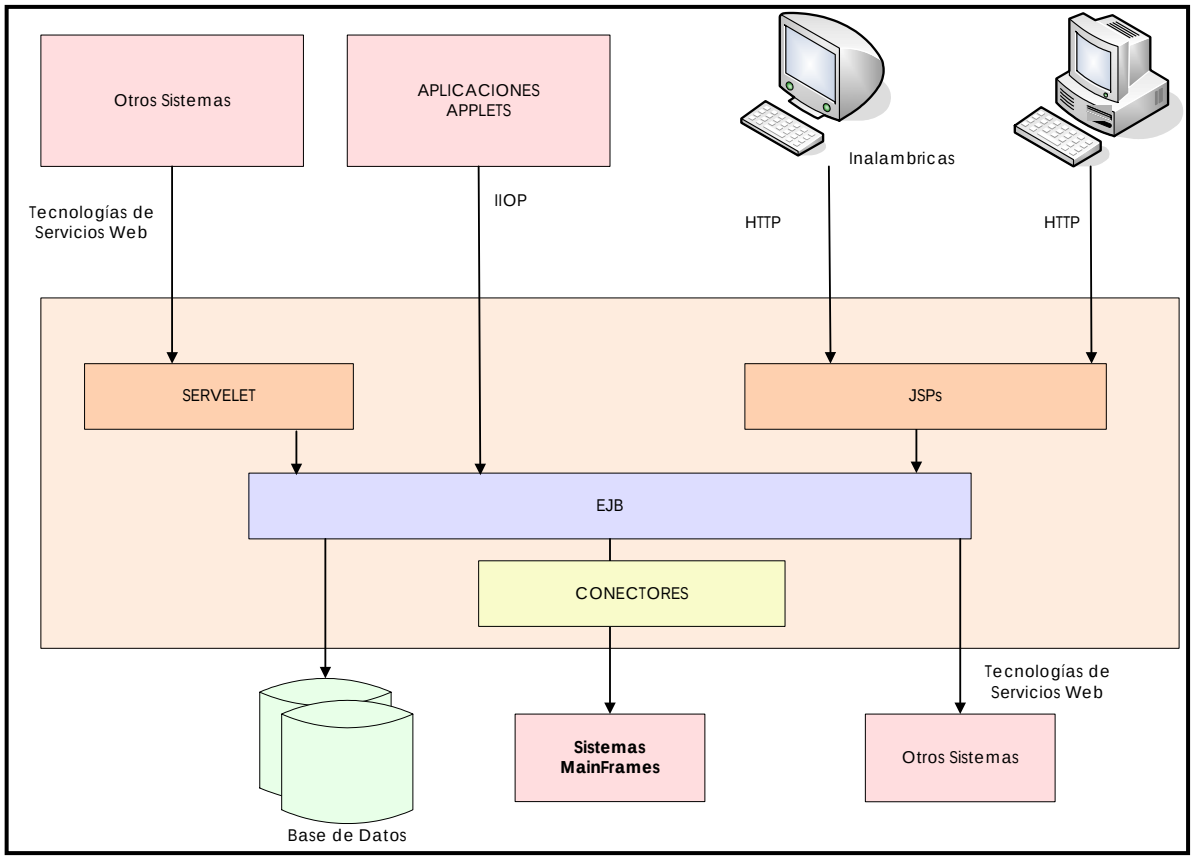


Figure 1.4: Servicios de Web desarrollo con J2EE

La figura 1.4 esta dividida en tres partes que se las explica a continuación:

La aplicación de J2EE se organiza dentro de un contenedor que mantiene los servicios necesarios para las aplicaciones de la empresa como las transacciones, seguridad, y servicios de persistencia.

La capa de negocios realiza el proceso comercial y lógica de datos. En las aplicaciones de J2EE de gran potencia, la lógica de negocios se construye usando componentes Entreprise Java Beans (EJB). Se conecta a las bases de datos por medio de un JDBC o ***SQL/J***. También puede conectarse a otras empresas que usan tecnologías Servicios Web(***SOAP, UDDI, WSDL,*** XML) a través de los API de Java para XML.

Los compañeros comerciales pueden también conectarse con las aplicaciones de J2EE a través de tecnologías de Servicios Web (SOAP, UDDI, WSDL, XML). Un servlet que es una petición/respuesta orientada objetos de Java, puede aceptar las demandas del servicio de Web de los compañeros comerciales. El servlet usa los API de JAX para realizar el funcionamiento de los Servicios Web.

Los clientes como ***applets*** o aplicaciones se conectan directamente a la capa de EJB a través del ***IIOP*** en lugar de Servicios Web, desde que generalmente los clientes son escritos por la misma organización que la de la aplicación de J2EE, y no hay la necesidad de la colaboración de Servicios Web basados en XML.

Los navegadores de Web y los dispositivos inalámbricos se conectan a las Páginas de JSP qué da la interfaz del usuario en HTML, ***XHTML*** o ***WML.***

1.3.2 Plataforma Microsoft. NET

Microsoft .NET es una colección de productos que permite a las organizaciones construir los Servicios Web de la empresa. La diferencia fundamental es que .NET es un conjunto de productos, en cambio J2EE es una norma para crear productos.

Microsoft .NET se basa en Windows ADN que era la plataforma anterior de Microsoft para aplicaciones de la empresa en vías de desarrollo. Windows ADN incluye muchas tecnologías como: el Microsoft Transacción Server (***MTS***) y ***COM+***, Microsoft Messenger Queve (***MSMQ***), y Microsoft SQL Server. El nuevo ***Framework*** de ***.NET*** reemplaza estas tecnologías, e incluye un Servicio Web que mejora el apoyo del lenguaje.

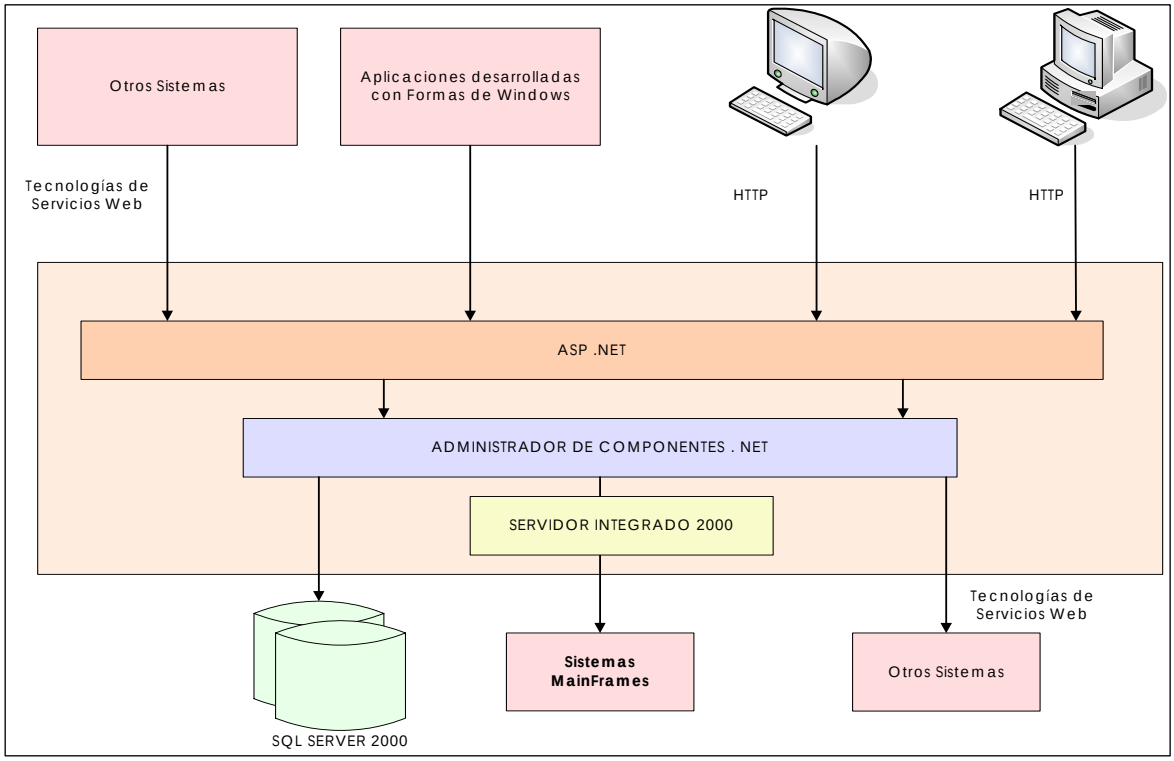


Figura 1.5: Microsoft .Net

A continuación se explica brevemente la figura 1.5.

La aplicación .NET se organiza dentro de un contenedor que mantiene los servicios necesarios; las aplicaciones de la empresa como las transacciones, seguridad, y servicios de mensajería.

La capa de negocios que usa .NET maneja los componentes. Esta capa realiza el proceso comercial y lógica de datos. Se conecta a las bases de datos por medio de Active Data Object(ADO.NET) y a sistemas existentes que usan los servicios proporcionados por Microsoft Anfitrión Integration Server 2000, como COM Transaction Integrator (el COM TI). También puede conectar a otras aplicaciones que usan tecnologías Servicios Web (SOAP, UDDI, WSDL). **[www004]**

Y también otras aplicaciones pueden conectar con la aplicación NET a través de tecnologías de Servicios Web (SOAP, UDDI, WSDL, BizTalk).

Clientes tradicionales, los navegadores de Web, los dispositivos inalámbricos se conectan a las Páginas del Servidor Activas (***ASP.NET***) qué da la interfaz de usuario en HTML, XHTML, o WML. La interfaz del usuario se construye usando los Formularios de Windows.

1.3.2.1 Framework de .Net

Microsoft.NET ofrece independencia del lenguaje e interoperabilidad, este es uno de los aspectos más intrigantes y fundamentales del la plataforma NET. Un solo componente NET puede escribirse, por ejemplo, parcialmente en VB.NET, ***C.***

El código fuente se traduce en el Microsoft Lenguaje Intermedio (***MSIL***). Este código de IL es el lenguaje neutral, y es análogo al ***bytecode*** de Java.

El código de IL necesita ser interpretado entonces y traducido en un ejecutable nativo. El. Framework .NET incluye el Lenguaje Común Runtime, análogo al Java Runtime Environment (***JRE***). El CLR es el intermediario de Microsoft entre los diseñadores .NET, el código fuente, el hardware subyacente, y todos los códigos NET corren finalmente dentro del CLR.

Este CLR no proporciona muchos rasgos excitantes disponible en las versiones más tempranas de Windows ADN, como la colección de basura automática, el manejo de excepciones, herencia del lenguaje y la ejecución de versiones diferentes del mismo componente .NET.

1.3.2.2 Servidores .NET

La plataforma .NET incluye lo siguiente:

- ✓ **SQL Server 2000.** Es la base de datos relacional de Microsoft.
- ✓ ***Microsoft Exchange 2000 Server.*** Es un sistema de mensajería y plataforma de colaboración útil, desarrollando y corriendo en el centro de los servicios comerciales y se integra herméticamente con Windows 2000.
- ✓ ***Commerce Server 2000.*** Oferta el desarrollo más rápido y menos complicado y despliegue de las soluciones del comercio electrónico en línea personalizable.
- ✓ ***Application Center Server 2000.*** Permite manejar los servidores cluster
- ✓ ***Host Integration Server 2000.*** Da acceso al legado seleccionado de sistemas que corren en otras plataformas.

1.3.3 Comprendiendo J2EE y .Net por la Analogía

A través de la siguiente analogía entre J2EE y .NET, se permite entender las similitudes y diferencias de las mismas.

CARACTERÍSTICAS	J2EE	.NET
Tipo De Tecnología	Estándar	Producto
Vendedores del ***Middleware***	Mas de 30	Microsoft
Interprete	JRE	CLR
Paginas Web Dinámicas	JSP	ASP.NET
Componentes de la Capa Media	EJB	Administrador de Componentes .NET
Acceso a Base de Datos	JDBC, SQL/J	ADO.NET
SOAP, WSDL, UDDI	Si	Si
Middleware Implícito (Carga Balanceada, etc.)	Si	Si

Tabla 1.1: Cuadro Comparativo J2EE y .Net

1.3.4 Ventajas y Desventajas de J2EE vs .Net

Para obtener una comparativa apegado a la realidad de las dos plataformas se las ha comparado en algunos aspectos esenciales incluyendo el mercado donde se desenvuelven. **[www004]**

Tiempo en el mercado

Al crear sistemas, el tiempo es un factor importante, por lo que se debe escoger una plataforma que permite el desarrollo rápido de aplicaciones. Esto les permite a diseñadores escribir y mantener el código rápidamente.

Sun J2EE y Microsoft .Net proporcionan mecanismos a diseñadores del software de las dependencias particulares como son el JRE y CLR, además de esto:

- ✓ J2EE ofrece varios rasgos que aceleran tiempo-a-mercado en que no se encuentra en el .NET. Como son los servicios de administración de estado, servicios de persistencia, transacciones pragmáticas y la creación de etiquetas personalizadas que contribuyen a la creación rápida de aplicaciones y a más de ello le proporcionan al diseñador una gran libertad de utilización de código.
- ✓ Además de estos rasgos J2EE se complementa con rasgos de dirección de proceso de negocios, integración con XML y colaboración ***B2B*** mejorada. Pero lamentablemente todos estos rasgos limitan la portabilidad de los sistemas debido a que no todos los casos sirven para todas las personas.
- ✓ Microsoft .Net ofrece una variedad de rasgos que no se encuentran en J2EE. El más notable es ASP.NET el cual es independiente del dispositivo del cliente, permite varias interfaces de usuario sin necesidad de reescribir el código. Microsoft

Ofrece una cola de componentes que son superiores a los MessageDriven Bean, debe notarse que se ha intentado simplificar la programación del lado del servidor. Proporciona además dirección de procesos comerciales y capacidades de comercio electrónico que están disponibles en las aplicaciones de J2EE pero no en todas.

En conclusión J2EE y .Net tienen en los aspectos analizados diferencias menores por lo que es difícil hacer un pronunciamiento a favor de la una o la otra

Soluciones de un único vendedor

Cuando se construyen Servicios Web, es preferible escoger las soluciones que presenta un solo vendedor ya que se más fiable que utilizar varias soluciones de varios vendedores.

Una gran ventaja de J2EE es que dispone de una gran cantidad de herramientas, productos y aplicaciones que proporcionan una gran funcionalidad. Sin embargo, esta ventaja también es una desventaja. Las herramientas de J2EE a menudo no son interoperables, debido a las imperfecciones de J2EE a la portabilidad. Se debe entonces escoger las soluciones de un solo vendedor como las que presenta IBM, Oracle, BEA e iPlanet cada una con un conjunto completo de herramientas.

A diferencia de J2EE, .Net presenta una solución bastante completa de un solo vendedor, Microsoft. A esta solución le pueden faltar algunos rasgos que dispone J2EE, pero en general, es un conjunto de herramientas completo y funcional.

Otra visión de una solución de un solo vendedor es la perspectiva de legado. Muchos sistemas han sido desarrollados por vendedores de J2EE, como IBM. Por lo que existe una integración entre las herramientas y aplicaciones desde una versión inferior a una superior.

Apoyo para sistemas existentes

La mayoría de las empresas todavía mantienen el código existente escritos en una variedad de lenguajes y tienen varios sistemas anteriores como ***COBOL, C++.*** Es vital que las empresas den un camino eficaz y rápido para re-usar y conservar estas inversiones debido a que muchas veces no se dispone de los fondos ni el tiempo para reinventar todos los sistemas existente. Esta integración de legado es el rasgo más desafiante de las tareas que superar para construir un Servicio Web.

Tanto J2EE y .Net deben dar soporte para la reutilización de estos sistemas.

J2EE tiene varias maneras de lograr la integración de legado, incluyendo:

- ✓ Java Messenge Services (***JMS***) para integrar con los sistemas de la mensajería existentes
- ✓ Servicios Web para integrar con cualquier sistema
- ✓ CORBA por unir con el código escrito en otros idiomas que pueden existir en las máquinas remotas.
- ✓ NI para las bibliotecas nativas cargantes y llamándolos localmente.

Pero la parte más importante de J2EE es el ***JCA***. El Java Connector Architecture (JCA) es una especificación de una serie de adaptadores que permiten la comunicación con

sistemas existentes. Si los adaptadores no existen, se pueden escribir los propios adaptadores. Estos adaptadores son reusables en cualquier contenedor JCA.

.Net También ofrece una integración de legado a través del Host Integration Server 2000. COM Transaction Integrator (COM TI) puede usarse para colaborar transacciones a través de sistemas mainframes. Microsoft Message Queue (MSMQ) puede integrarse con sistemas construidos en IBM ***MQSerie***s. Finalmente ***BizTalk*** Server 2000 puede integrarse con sistemas basados en protocolos B2B.

En conclusión se establece que los rasgos ofrecidos por J2EE son superior a los de .Net. El mercado de JCA está produciendo adaptadores que aliviarán la integración de aplicaciones.

Percepción del Mercado

No siempre la tecnología más buena gana en el mercado sino la que mejor tiene un sistema de mercadeo y venta.

J2EE tiene más de 50 vendedores surtidos en todo el mundo. Esta red forma un verdadero ente comercial como si fuera uno solo, y el resultado es una percepción fantástica por J2EE.

Pero en cambio el sistema de mercadeo que dispone Microsoft para sus productos y en este caso .Net, convierten a este el ganador de la percepción del mercado ante J2EE.

La madurez de la Plataforma

Cuando una organización adopta un Servicio Web debe considerar la madurez de la plataforma que adopte. Una plataforma menos madura presentará más pronto errores y problemas.

J2EE es en si una plataforma muy madura, pero todavía tiene un par de problemas críticos como la persistencia en los EJB, la implantación de JCA y el apoyo de Servicios Web son todavía inmaduros.

En Microsoft .Net a pesar de ser relativamente reciente dispone de una gran madurez ya que mucha de su tecnología esta basado en Windows ADN. Pero también existen sectores críticos como son: el nuevo CLR, el C#, y el apoyo de Servicios Web.

Lógicamente en esta comparación el ganador es J2EE, su trayectoria es mucha más amplia que la de .Net, pero tampoco se puede decir que .Net no tenga suficiente madurez para ser una herramienta de desarrollo de Servicios Web. **[www004]**