DESARROLLO DE UNA APLICACIÓN CLIENTE/SERVIDOR BASADA EN CORBA SISTEMA DE RESERVACIONES DE VUELO PARA UNA AEROLÍNEA

Eric Guagua Alvarado, Ph.D. Sixto García Aguilar
Facultad de Ingeniería en Electricidad y Computación (FIEC)
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
profesorguagua@hotmail.com

Escuela Superior Politécnica del Litoral (ESPOL), Ph.D. en Ingeniería, sgarcia@espol.edu.ec

Resumen

El proyecto realizado involucra el desarrollo de una aplicación de un sistema de reservaciones de vuelo para una aerolínea ficticia, en la que se aplicó una tecnología que permite hacer uso de objetos distribuidos. Se llama CORBA, la arquitectura común de corredores de solicitudes de objetos. Esta aplicación tiene una parte administrativa de tipo standalone (independiente) para uso interno de la compañía, y una parte para los clientes de la misma de tipo dependiente de browser (un applet de java) para brindar el servicio a través del web. El sistema permite hacer las siguientes transacciones: reserva, cancelación de reservas, ingreso y cancelación de vuelos; ingreso de recorridos, ingreso de ubicaciones y consultas.

Palabras Claves: *Aplicación, tecnología, Objetos distribuidos, CORBA, cliente, browser, applet, java*

Abstract

The project involves the development of an application of a flight reservation system for a fictitious airline, which allows the use of distributed object technology. CORBA is called the common architecture of object request brokers. This application has a standalone administrative part for internal use of the company ,an another part the use of the customers with the browser, and some customers for the same type of browser dependent (a Java applet) to provide the service through the web. The system allows the following transactions: booking, reservation cancellations, income and cancellation of flights; entry routes, locations and consultations income.

Keywords*: Application, technology, distributed objects, CORBA, client, browser, applet, java*

Introducción

La computación ha evolucionado muchísimo en las dos últimas décadas, tanto a nivel de hardware como a nivel de software. En cuanto al software, las maneras de construir sistemas han cambiado desde que se introdujeron nuevas metodologías de diseño como la programación orientada a objetos y la arquitectura cliente/servidor. Esta última ha sido fundamental para el desarrollo realmente distribuidos haciendo que los programadores e ingenieros de software dejen a un lado viejos esquemas, como el de file server y más aún el de multiusuario y centren sus miradas en esta nueva forma de diseño. Por otro lado, la programación orientada a objetos ha suplantado otro tipo de programación conocida como programación estructurada encapsulando código y datos en un solo ente inteligente conocido como objeto.

Cualquier persona relacionada con las computadoras sabe que los objetos son maravillosos, y que sencillamente no podríamos vivir sin ellos. Hoy en día, existe un sin número de objetos mucha bibliografía relacionada a los conceptos como el encapsulado, la herencia y el polimorfismo. Nunca la programación orientada a objetos y la arquitectura cliente/servidor se complementaron tanto como en la actualidad.

En los primeros días los expertos en la industria de la computación se daban cuenta de las bondades que ofrecían los objetos y se preguntaban con insistencia ¿Qué pueden hacer los objetos en una red? ¿Cómo pueden apoyar estos al esquema cliente/servidor? Para decirlo brevemente ellos necesitan comprender como extender la tecnología de objetos para manejar los

complejos asuntos inherentes a la creación de robustos sistemas cliente/servidor. Ahora, cuando iniciamos un nuevo siglo, conocemos las respuestas a estas preguntas y sabemos con precisión qué pueden hacer los objetos por los sistemas con arquitectura cliente/servidor. Esta es la intención del presente proyecto, demostrar que en lo subsecuente la siguientes generaciones de sistemas cliente/ servidor estarán basados en objetos distribuidos.

Convencido de que está es el área en la que los objetos desarrollaran todo su potencial. También creo que internet, las intranets y las extranets necesitan de los objetos distribuidos. Obviamente pues, la computación en internet no sustituye de ninguna manera a la arquitectura cliente/ servidor, y esto se debe a que ya es en sí misma cliente/servidor.

Con el fin de cumplir con el propósito que mencionamos anteriormente, se ha desarrollado un "Sistema de Reservaciones de Vuelo" para una aerolínea ficticia donde hemos empleado una tecnología que permite hacer uso de objetos distribuidos. Se llama CORBA, la arquitectura común de corredores de solicitudes de objetos. Esta aplicación tiene una parte administrativa de tipo standalone para uso interno de la compañía, y una parte para los clientes de la misma de tipo dependiente de browser (un applet de java) para brindar el servicio a través del web. Con este sistema se busca facilitarle el trabajo al administrador en muchos campos del negocio y también el sistema proporciona algunos beneficios al cliente como el ahorro de tiempo.

1. Metodología

El desarrollo rápido de aplicaciones o RAD (acrónimo en inglés de rapid application development) es un proceso de desarrollo de software, desarrollado inicialmente por James Martin en 1980. Permite construir sistemas utilizables en poco tiempo, el método comprende el desarrollo interactivo, la construcción de prototipos y el uso de ingeniería de software asistida por computadora o sea utilidades CASE (Computer Aided Software Engineering). ya definidas. . El Desarrollo Rápido de Aplicaciones es una adaptación a "Alta velocidad del modelo lineal secuencial" en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso RAD permite al equipo de desarrollo crear un "sistema completamente funcional" dentro de periodos cortos de tiempo. Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución. Mientras menos tiempo transcurre en el desarrollo del sistema menos habrán cambiado las necesidades de los usuarios.

El Modelo RAD (Figura 1) comprende las siguientes etapas:

Modelado de gestión. Este modelo se basa en dar respuesta a las siguientes preguntas:
¿Qué información conduce el proceso de gestión?
¿Qué información genera?
¿A dónde va la información?
¿Quién la procesa?
Modelado de datos. En este modelo se definen los almacenes de datos y cómo se relacionan los almacenes entre sí.
Modelado del proceso. Se utiliza para añadir, modificar, suprimir o recuperar un objeto de datos.
Generación de aplicaciones. Para esto se utiliza una herramienta de cuarta generación que permite crear el software y facilitar la construcción del programa.
Pruebas y entrega. El proceso de desarrollo finaliza realizando pruebas de calidad del software diseñado con la herramienta RAD, posteriormente se realiza la implementación de la aplicación

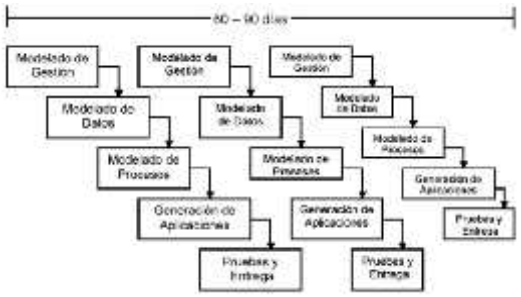

Figura 1. Fases del modelo desarrollo rápido de aplicaciones (RAD)

En este entorno de programación se desarrolló el presente proyecto.

2. Especificaciones

Las especificaciones se detallan a continuación:

Debe ser desarrollado con arquitectura cliente/servidor haciendo uso de un esquema de tres capas

Debe hacer uso de la tecnología de objetos distribuidos y demostrar su importancia para la nueva generación de sistemas con arquitectura cliente/servidor de aquel entonces.

Debe emplear la tecnología que nos ofrece la arquitectura CORBA y lo conveniente que es para el desarrollo de sistemas cliente/servidor de tres capas

Debe mostrar la aplicación que tiene un DBMS cliente/servidor en un ambiente tres capas

Requerimientos funcionales

Se desea crear una aplicación para que sea accesible desde el Web, para lo cual se utilizará applets de Java. Los applets de Java podrán ser ejecutados desde distintas maquinas clientes. El tema principal del

proyecto es el desarrollo de un sistema cliente/servidor para una aerolínea ficticia usando tecnología de Internet. Las maquinas clientes deben contar con la existencia de browsers los cuales descargaran del servidor un applet de java en el cual se brindan las opciones de los distintos tipos de transacciones con los que contara el proyecto.

Las disposiciones para realizar este proyecto son las de hacer uso del lenguaje de programación Java y el uso de CORBA como middleware de objetos.

Del análisis del sistema se determina que el proyecto ha sido concebido para ejecutarse en una intranet o internet, por lo que el programa deberá utilizar applets y ejecutarse en un browser. Además este applet debe proveer la interfaz gráfica necesaria para brindarle al usuario la posibilidad de realizar las siguientes cinco transacciones que de lo analizado y de la información recogida, se identificaron las cuales son las metas "primordiales" del sistema.

1. Reserva
2. Cancelación de reservas
3. Ingreso y cancelación de vuelos
4. Ingreso de recorridos
5. Ingreso de ubicaciones y consultas

Requerimientos del proceso servidor

1. Utilizar el modelo de procesamiento paralelo Pool de objetos de servidor para implementar el proceso servidor y que éste pueda atender los requerimientos de los procesos clientes
2. Implementar parte de la lógica de la aplicación en procedimientos almacenados (stored procedures) usando un manejador de bases de datos (DBMS) Cliente/servidor.
3. Emplear CORBA usando los ORB (Object Request Broker) que es el bus de comunicación de objetos de Inprise Visibroquer como middleware en nuestro ambiente Cliente/Servidor

Requerimientos para el administrador

1. Crear las ciudades que serán tomadas en cuenta en los recorridos de los respectivos vuelos
2. Realizar la respectiva consulta de las ciudades
3. Crear los recorridos para los vuelos que seguirán los respectivos aviones ingresando la distancia en kilómetros entre los puntos origen y destino y el respectivo costo
4. Realizar la respectiva consulta de recorridos
   Crear los vuelos escogiendo el recorrido apropiado y adicionalmente ingresando el número de asientos disponibles y la fecha y hora de partida y llegada del vuelo
5. Realizar la respectiva consulta de vuelos

Requerimiento para el cliente de la aerolínea

1. Deberá registrarse la primera vez que desee hacer una reservación, la siguiente ocasión que desee usar el sistema bastara con que valide su ingreso en la opción de autenticación
2. Podrá realizar las respectivas reservaciones escogiendo el vuelo que desee tomar y adicionalmente ingresando el número de asientos que desee reservar. El sistema devolverá el costo que debe pagar
3. Existe la opción de poder cancelar las reservaciones.

Restricciones del sistema

Para la implementación del sistema lo mínimo necesario en cuanto a software es:

- Utilizar cualquier sistema operativo Windows
- Un Browser Java – enabled
- Un compilador java jdk 1.1.7
- Microsoft SQL Server 7.0 para la Base de Datos
- Inprise Visibroker 3.1 para los ORBs de CORBA

En cuanto al hardware lo mínimo necesario es:

Procesador

Pentium de 100 MHz (en el año 1999) como mínimo pero Pentium II 200 MHz recomendable para las computadoras donde van a correr tanto los procesos clientes como el proceso servidor

Memoria

24 MB (en el año 1999) donde corra el proceso servidor como mínimo, pero recomendable 32 Mb y 16Mb en aquellas computadoras donde corren los procesos clientes

El sistema esta implementado en java que es un lenguaje que permite hacer aplicaciones para la web así como aplicaciones de red en general, lo que significa que es idóneo para el desarrollo de sistemas distribuidos con arquitectura cliente/servidor.

La base de datos fue implementada en el producto SQL Server 7.0 de Microsoft, que posee un poderoso motor de base de datos que inclusive tiene soporte de terabytes y un sinnúmero de características que facilitan su administración, además de un fabuloso esquema de seguridad.

3. Esquema de la Arquitectura de la Aplicación

El Sistema de Reservaciones de Vuelo posee una arquitectura Cliente/Servidor de Tres Capas. En la que se debía aplicar la tecnología que permite hacer uso de objetos distribuidos llamada arquitectura común de corredores de solicitudes de objetos CORBA (por sus siglas en inglés Common Object Request Broker Arquitecture)

Las tres capas consisten en una capa de la Presentación, otra capa de la lógica de la aplicación y otra capa de la base de datos.

¿Qué es cliente/servidor?

Clientes y servidores son entidades lógicas independientes que operan en conjunto ya sea a través de una red o en la misma computadora para realizar una tarea específica. A continuación presentamos una definición más formal:

Cliente/Servidor es una arquitectura de diseño de software que subdivide la aplicación en un conjunto de procesos servidores, generalmente especializados, que pueden ejecutarse en variadas plataformas (hardware+software), que proveen servicios (datos, información, procesamiento, etc.) a un conjunto de procesos clientes, que pueden ejecutarse en diferentes plataformas (hardware+software), a través de redes de área local o de redes de área extendida, utilizando uno o varios protocolos de comunicación.

¿Qué es CORBA?

CORBA (Common Object Request Broker Arquitecture) es un estándar para objetos distribuidos que ha sido desarrollado por el OMG (Object Management Group), una asociación de empresas y usuarios de las más grandes que existen actualmente, que agrupa a unas 800 entidades que se dedican básicamente al desarrollo y comercialización de software para sistemas informáticos, y, que en su mayoría, se encuentran actualmente desarrollando aplicaciones que soportan este estándar o comercializando productos que ya lo hacen.

CORBA proporciona los mecanismos a través de los cuales los objetos hacen peticiones y reciben respuestas, definidas como ORBs (Object Request Broker) de forma transparente para el sistema. El ORB de CORBA es entonces, una aplicación que proporciona interoperabilidad entre diferentes objetos posiblemente programados en diferentes lenguajes, corriendo bajo sistemas operativos distintos y en general en diferentes máquinas. Este es el punto básico en el que se basa la arquitectura CORBA

¿Qué es ORB?

El ORB (Object Request Broker), intermediador de requerimientos de servicios, es el middleware que permite la comunicación entre cliente y servidor por medio de objetos. Usando un ORB, un cliente puede transparentemente invocar un método en un objeto servidor, el cual puede estar en una misma máquina o en algún lugar de la red. El ORB intercepta la llamada y es el responsable de encontrar un objeto que pueda implementar el requerimiento, pasar los parámetros, invocar el método, y retornar los resultados. El cliente no necesita conocer donde está localizado el objeto, su lenguaje de programación, sistema operativo, o algún otro aspecto del sistema que no sea parte de la interface del objeto. El ORB es simplemente un proceso cuya función es localizar los procesos que proveerán los servicios requeridos. Una vez localizados el cliente puede conectarse con el servidor. Los objetos en el ORB pueden actuar tanto como cliente o servidor, dependiendo de la ocasión.

La comunicación entre los ORBs se lo realiza por medio del protocolo IIOP (Internet Inter-ORB Protocol).

IIOP (Internet Inter-ORB Protocol)

OMG ha definido un conjunto de reglas que dan formato a los datos que se transfieren, llamado CDR (Common Data Representation), además de un conjunto de tipos de mensajes. Juntos, los CDR y los tipos de mensajes, constituyen un protocolo abstracto llamado GIOP (General Inter-ORB Protocol). Para lograr verdadera interoperabilidad entre objetos distribuidos que residen en ambientes heterogéneos sobre Internet, se necesita que ORBs envíen mensajes a través de TCP/IP, porque es el protocolo de transporte orientado a conexión estándar para Internet. En pocas palabras, al unir GIOP + TCP/IP da como resultado el protocolo IIOP.

Para otros ambientes que no son TCP/IP, se utiliza el protocolo ESIOP (Environment Specific Inter-ORB Protocol).

Los objetos publican sus identidades y ubicaciones utilizando las referencias de objetos. CORBA especifica un formato común para las referencias de objetos llamado IOR (Interoperable Object Reference), el cual contiene perfiles que describen como los clientes pueden encontrar y enviar requerimientos a1 servidor usando un protocolo particular.

Todos los IORs, deben tener por lo menos un perfil IIOP, lo cual asegura que donde sea que se encuentre la referencia de objeto, cualquier ORB que cumpla con CORBA, será capaz de localizar el servidor y enviarle los requerimientos. El perfil IIOP tiene la dirección Internet del servidor y un valor clave usado por el servidor para encontrar el objeto especifico descrito por la referencia.

IIOP brinda un ambiente en el cual los programadores definen e invocan cualquier método que necesitan, y transparentemente se realiza la comunicación, es decir, los programadores no tienen que escribir programas IIOP, nunca requieren interactuar con IIOP de ninguna forma, simplemente es invisible para ellos.

Procesos Cliente y Servidor

El proceso servidor y el proceso cliente deberán ejecutarse en computadoras cuyo sistema operativo es Windows. La comunicación entre ambos se realizará utilizando el mejor middleware de nuestra industria actual: los ORB (Object Request Broker) de CORBA y además un pool de objetos de servidor será

implementado para encargarse de atender los requerimientos de cada proceso cliente (Figura 2). El lenguaje en que han sido implementados tanto el proceso cliente como el proceso servidor, es el lenguaje de programación Java.

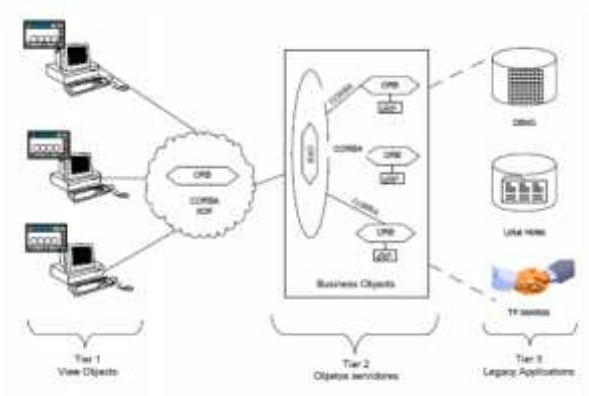

Figura 2. Arquitectura del proyecto con CORBA three tier

4 RESULTADOS OBTENIDOS

4.1 Diseño de Interfaces Gráficas de Usuario

Se analizó varias posibilidades de cómo hacer el cliente, ya que podría ser de varios applets sencillos que contengan una transacción a la vez. Pero se tuvo una dificultad con este tipo de cliente. Otra posibilidad fue la que hemos escogido. Se lo realizo con el enfoque de un Applet que pueda contener varios paneles y de esta manera superar el inconveniente de poder mantener información de una transacción anterior de un cliente sin tener la necesidad de accesarlo otra vez. El applet del Cliente fue realizado con la ayudad de Visual Café permitiendo tener un Tab de paneles para todas las opciones de transacción. El cliente solo con dar un click del ratón sobre botones en la interfaz gráfica llama a funciones que se encuentran en objetos ubicados en el servidor. Por medio de la interface se accesa a estos objetos. El código de la interacción con los objetos Corba no presenta ninguna dificultad para el programador ya que se trata a los objetos servidores como si fueran objetos locales. En el Applet se realizaron varios mensajes de Error que sirven para evitar realizar las transacciones en forma incorrecta. Estos mensajes de errores, en su mayoría no llaman a ningún objeto sino que comprueba su veracidad dentro del propio Applet. Los errores que detecta son de formato, nombre, incompletos, campos nulos, etc. Los Paneles que contiene este Applet son de reserva, cancelación de reservas, ingreso y cancelación de vuelos; ingreso de recorridos, ingreso de ubicaciones y consultas.

Applet Principal es el nombre que se le ha dado al applet, encargado de facilitar al Cliente una interfaz gráfica para el manejo de las transacciones en una manera amigable. Es el encargado de llamar a la función que nos conecta con el servidor atravez de su interface ORB. La Interfaz gráfica consta de 2 partes que se encuentran dentro de un applet La primera es un panel que hemos denominado Principal, La segunda es el TAB panel, el mismo que contiene seis Paneles y consta de las seis transacciones

4.1.1 Registro de clientes

El cliente deberá escoger la opción de registro (Figura 3) e ingresar la información que se lo solicita: Apellidos, nombres, usuario, clave, país, ciudad, dirección y su e-mail. Una vez introducidos todos estos datos, el cliente deberá hacer “click” en el botón Registrar.

El sistema en la barra de status, le indicara si el proceso de registro fue exitoso o si hubo problemas

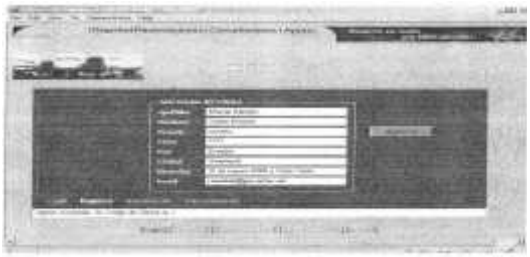

Figura 3 Registro de clientes

4.1.2 Autenticación

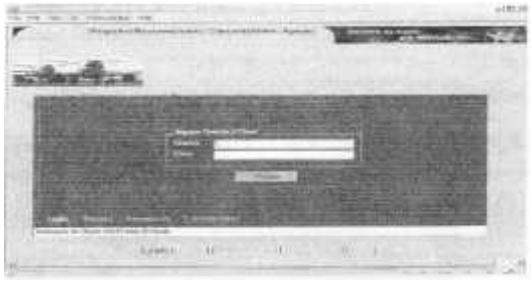

Figura 4. Autenticación

Si el cliente ha usado el sistema en anteriores oportunidades, deberá estar registrado (Figura 4), por lo que puede ingresar al mismo haciendo ·log _in”. De igual manera indicará en la barra de status si la autenticación fue exitosa reconociendo al usuario del sistema mediante su información de usuario y clave.

4.1.3 Reservaciones de vuelo

Podrá realizar las respectivas reservaciones escogiendo el vuelo que desee tomar y adicionalmente ingresando el número de asientos que debe reservar (Figura 5). El sistema devolverá el costo que deberá pegar.

El cliente debe seleccionar el vuelo a tomar e ingresar el número de reservas.

Figura 5. Reservas de Vuelo

4.1.4 Cancelación de Reversas de Vuelo

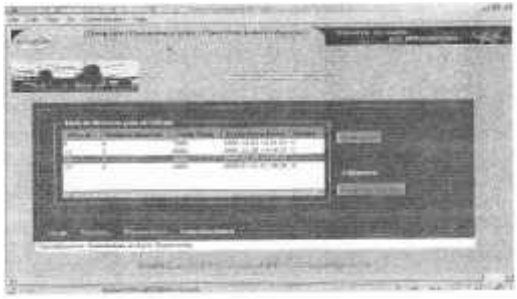

Figura 6. Cancelación de Reservas de Vuelo

Existe la opción de poder cancelar las reservaciones (Figura 6). El usuario deberá seleccionar la reserva, dentro de la lista de reservas para el cliente, y luego cancelaria presionando el botón "Cancelar Reserva"
El sistema indicará al usuario, mediante la barra de status, el resultado de la cancelación de la reserva de vuelo

5. Mejoras en el sistema

A continuación se detallarán las mejoras de implementar el sistema de reservaciones de vuelo:

- **Permitir al administrador y a los clientes:** Manejar el sistema con una interfaz gráfica fácil de usar:
- **Agilita las reservaciones:** El sistema permite que los clientes de la aerolínea reserven sus vuelos mediante el web sin necesidad de acercarse físicamente a las oficinas de la empresa.
- **Información en línea:** El sistema brinda toda la información tal como vuelos recorridos y ubicaciones de tal forma que los usuarios del web siempre estén actualizados de cualquier nueva actividad
- **Privacidad al hacer las Reservaciones:** Cada cliente tiene su propio identificador, de manera tal que nunca un cliente puede consultar o cancelar las reservaciones de otros clientes.
- **Clientes Concurrentes:** Debido a que el proceso servidor implementa un "Pool de objetos de Servidor", el sistema tiene la capacidad de atender a un ¨"ilimitado" número de procesos clientes concurrentes. La única limitación del sistema es todo el ambiente de hardware en el que se desenvuelve
- **Fácil y eficiente revisión de Reservaciones por parte del Administrador:** El administrador podrá revisar las reservaciones de todos los clientes y verificar su estado, de tal manera que se podrá detectar quienes son los que más reservaciones hacen y con qué frecuencia cancelan sus reservas

6. Limitaciones

- No existe la opción de escoger el número de asiento que el pasajero desea al momento de hacer reservas, además todos los asientos tienen igual jerarquía, es decir, no existe primera clase, segunda clase, tercera clase, etc.
- Si un cliente hace más de una reservación, todas las reservaciones son asignadas a él y no se permite que se ingresen los nombres de las otras personas que van a viajar con él, en ese caso tendría que físicamente acercarse y dar sus nombres a la hora de pagar sus boletos y retirarlos.
- El sistema no permite vuelos compuestos, es decir, vuelos con más de un segmento entre destinos finales
- el costo para un vuelo es fijo, lo que significa que no varían de acuerdo a las temporadas. Esto se menciona debido a que las aerolíneas reales brindan unos costos en temporada alta y otros en temporada baja.

En general existen beneficios de implementar aplicaciones en un sitio Web, hoy en día los usuarios buscan más que información en un sitio Web. Desean tener sistemas a su disposición para satisfacer rápidamente y en cualquier momento alguna necesidad especifica. Por otra parte muchas empresas obtienen grandes beneficios proveyendo estas soluciones a sus clientes y usuarios consiguiendo reducir costos, aumentar ventas, mejorar la imagen de la empresa, conservar clientes, etc. Se debe pensar en la utilidad que ofrecen las aplicaciones web.
Algunas aplicaciones que puede incluir en un sitio Web:
Comercio electrónico
Seguimiento online de operaciones
Consulta de estados de cuenta
Envío masivo de información
Webmail

7. CONCLUSIONES Y RECOMENDACIONES

Al culminar la fase de diseño del proyecto se pudo constatar lo siguiente:

Conclusiones

1. Para toda la industria de la computación con la notable excepción de Microsoft corporation

(en el año 1999) la generación de middleware era CORBA

2. El desarrollo de aplicaciones distribuidas con esquemas 3 tier se facilita cuando empleamos una tecnología que aporte con un bus de objetos y que aplique los conceptos de ORB
3. La computación de internet no reemplaza a la arquitectura cliente/servidor ya que ésta es en sí misma cliente/servidor.

4. Los DBMS cliente/servidor tiene una participación importante dentro de todo el ambiente cliente/servidor en el que se desenvuelve la aplicación distribuida y muchas veces es un componente infaltable
5. Las siguientes generaciones de sistemas distribuidos estarán basados en objetos distribuidos y componentes
6. En un contexto distribuido es donde los objetos brindaran todo su potencial

7. CORBA es una especificación. No es un software o aplicación. Hay un gran número de implementaciones de CORBA. Estas son conocidas como Object Request Broker (ORB).

8 Se nota también que muchas tecnologías están en constante desarrollo y maduración, lo cual implica un minucioso estudio previo de muchos factores antes de apostar por alguna tecnología en especial

9 En todo proyecto es recomendable realizar una correcta planificación, por ende, seleccionar una metodología que permita minimizar los riesgos, aumentar la calidad y acortar el tiempo de implementación ayuda en gran medida al equipo de trabajo.

10 CORBA es hoy en día una opción tecnológica aceptada por la industria del software en lo referente al desarrollo de soluciones distribuidas. A CORBA le surgen competidores, especialmente desde el mundo de Microsoft y desde el mundo de Java, aunque CORBA presenta las ventajas de no ser propietario, no estar ligado a ningún lenguaje de programación concreto y además ser capaces de integrarse sus competidores mediante las pasarelas adecuadas, en un claro ejemplo de interoperabilidad entre sistemas heterogéneos.

11 Java como herramienta también integra interoperabilidad con CORBA siempre y cuando los objetos estén usando un ORB compatible con las especificaciones y que se apoyen con IIOP como protocolo de comunicaciones.

Recomendaciones

1 Se recomienda explorar y conocer bien el estándar CORBA y sus servicios antes el posible diseño de un proyecto bajo esta herramienta. Esto ahorrará tiempo y problemas.
2 Se recomienda dar a conocer el presente proyecto para futuras investigaciones de implementaciones con CORBA y el lenguaje de programación JAVA
3 Se recomienda continuar con el uso de este estándar CORBA y hacer revisiones periódicas de nuevos especificaciones e implementaciones de componentes
4 La construcción y utilización de clases reutilizables es una meta que los desarrolladores siempre deben tener presente, aunque tampoco se ha de convertir en una obsesión.
5. Durante las fases de planificación y diseño de los sistemas informáticos se debe considerar si el sistema a desarrollar debe o no ser internacionalizado ya que actualmente la demanda de acceso a la información exige romper las limitantes del idioma y así el sistema será más escalable.
6. Tal vez sería interesante, que en un futuro, se profundizara más sobre IDL en este ramo, ya que sin duda CORBA representa un estándar que no se puede dejar de lado.

8. BIBLIOGRAFÍA

[1] ARNOLD Y GOSLING, El lenguaje de programación Java, Addison – Wesley/Domo, 1997.

[2] MICROSOFT CORPORATION, mastering Distribuited Application Design, Student Workbook Microsoft,1998

[3] ORFALI Y HARKEY Y EDWARDS, Cliente/Servidor Guía de supervivencia, segunda edición, 1997

[4] LEMAY LAURA Y PERKINS CHARLES. Aprendiendo Java 1.1 en 21 días. Segunda edición, Prentice Hall, 1998

[5] 0. GRAF, A. KOTZEN, 0. TAKAGIWA & U. WAHLI, "VisualAge for Java Enterprise Version 2: Data Access Beans - Servlets - CICS Connector", 1era. Edición: New York: Red book de IBM, http://www.redbooks.ibm.com, 1998.

[6] BOOCH GRADY. Análisis y Diseño Orientado a Objetos con Aplicaciones. Segunda edición, Addison-Wesley/Díaz de Santos, 1996.

[7] GRIFFITH & CHAN & F.ISA1, 1001 Tips para Programar con Java, 1era edición en español México, Mc Graw Hill, 1998