APLICACIÓN DE LA TECNOLOGIA CLIENTE/SERVIDOR EN TRES CAPAS CON OBJETOS DISTRIBUIDOS EN LA RESERVACION DE HABITACIONES DE UN HOTEL

Juan Cruz Rodriguez[1], Italo Galarza Espinoza[2], Fabricio Echeverria[3]

[1]Ingeniero Eléctrico en Computación 2005; email: jccruz@mazda.ec
[2]Ingeniero Eléctrico en Computación 2005; email: igalarza@rp3.com.ec
[3]Director de Tópico. Ingeniero Eléctrico en Computación, Escuela Superior Politécnica del Litoral, 1998, Diplomado Ecuador, Escuela de Postgrado en Administración de Empresas, 2002. Profesor de ESPOL desde 2000., email: pechever@hotmail.com

RESUMEN

Las aplicaciones de múltiples-capas, requieren de entender y conocer la forma de trabajar con objetos distribuidos, sobre todo para maximizar la utilidad y decrementar la complejidad, en esquemas de este tipo la ubicación física de los procesos servidores y de los datos no es importante, se requiere de una implementación completa del Directorio de Servicios, el mismo que aísla a la aplicación de saber donde se encuentran los procesos y los datos.

Se implementará una aplicación, con la cual se pueda realizar las siguientes transacciones: reservas de habitaciones, check-in, check-out y emisión de factura, esta aplicación se basa en arquitectura de tipo Cliente/Servidor en múltiples capas, para demostrar la funcionalidad de CORBA, JDBC y Applets, y la obtención de recursos distribuidos en la Internet. Los clientes Applets invocan a objetos en el servidor (objetos CORBA), a través del protocolo IIOP, toda la lógica del negocio se encuentra en los objetos servidores y estos almacenan los datos en la base de datos SQL con soporte JDBC.

The applications of multiple-layers, require of to understand and to know the form of working with distributed objects, mainly to maximize the utility and decrease the complexity, in outlines of this type the physical location of the processes servants and of the data it is not important, it is required of a complete implementation of the Directory of Services, the same one that isolates to the application of knowing where they are the processes and the data.

An application was implemented, with which can be carried out the following transactions: reservations of rooms, check-in, check-out and invoice emission, this application is based on type architecture Client/Server in multiple layers, to demonstrate the functionality of CORBA, JDBC and Applets, and the obtaining of resources distributed in the Internet. The clients Applets invokes to objects in the servant (objects CORBA), through the protocol IIOP, the whole logic of the business is in the objects servants and these they store the data in the database SQL with support JDBC.

INTRODUCCION

Debido a la necesidad de desarrollar sistemas distribuidos, aparece la tecnología Cliente/Servidor con varios componentes que distribuyen los elementos básicos de un sistema: Front-End, Middleware, Back-End, en el tiempo esta tecnología ha evolucionado desde esquemas de Dos-Tiers hasta esquemas de Múltiples-Tiers.

Dentro del esquema Múltiples-Tiers se tiene una tecnología desarrollada por OMG la cual se la definió como CORBA (Common Objects Request Broker Arquitecture), debido a que esta se convirtió en una de las impulsadoras de la comunicación entre servicios de diferentes programas y plataformas; cuyas principales funcionalidades son responder a: necesidades de interoperabilidad y sobre todo permitir que las aplicaciones se comuniquen entre ellas sin importar su localidad física, esto además de que CORBA, fue la que permitió el uso de la metodología OOA (Análisis Orientado a Objetos) y OOD (Diseño Orientada a Objetos) provocando que se cambien las disciplinas de análisis, diseño, desarrollo, y de implementación, brindándonos la flexibilidad para reutilización, facilidad para el mantenimiento, y utilización de integración con otros componentes internos o externos.

Para el desarrollo del proyecto, se revisó diferentes páginas Web de varios hoteles de la ciudad de Guayaquil, para revisar los requerimientos básicos y comunes de cada uno de ellos.

Como siguiente paso se hizo el análisis y diseño orientado a objetos utilizando OMT (Tecnología de Modelamiento Orientado a Objetos /Rumbaugh). Y por último el proyecto fue implementado haciendo uso de las siguientes herramientas: Eclipse, SQL Server 2000, Windows Server 2000, Windows XP, VisiBroker for Java, y Apache Web Server.

CONTENIDO

1.1 Tecnología Cliente/Servidor

El concepto cliente/servidor es eminentemente técnico. Su principio básico es muy sencillo: se tienen aplicaciones en un computador que están "conversando" con aplicaciones en otro computador. A partir de ese momento se establece un diálogo cooperativo entre los dos computadores. Y en su forma básica deben existir por lo menos dos componentes, el proceso servidor el mismo que puede ser ejecutado en las diversas plataformas existentes en el mercado, y el/los procesos clientes; estos procesos clientes se comunican en la Network usando uno o varios protocolos de LAN o WAN. La idea no hace referencia a un tipo específico de hardware o sistema administrador de base de datos; no solo funciona para aplicaciones accediendo bases de datos, sino que existen otras áreas de la computación, como por ejemplo el correo electrónico - entre otras - que pueden ser susceptibles a la implementación de la tecnología.

1.2. Características deseables del esquema Cliente/Servidor

1. **Transparencia de localización.-** El servidor es un proceso que puede residir en la misma maquina del cliente o en una maquina diferente que pertenezca a la red, el software Cliente / Servidor usualmente oculta la localización del servidor a los clientes pero direccionando las llamadas a los servicios si es necesario. Un programa puede ser cliente, servidor o ambos.

2. **Transparencia de Plataforma.-** El software ideal Cliente/Servidor es independiente del Hardware o de la plataforma donde se ejecuta (Sistema Operativo). El software tiene que ser capaz de trabajar entre plataformas heterogéneas.

3. **Escalabilidad.-** Los sistemas cliente servidor pueden ser escalados Horizontalmente o Verticalmente. EL escalamiento horizontal principalmente se trata de agregar o quitar estaciones cliente, provocando un impacto de desempeño menor. El escalamiento vertical se trata de migrar a maquinas servidoras mas rápidas y robustas.

1.3 Arquitectura de Tres-Tiers

Referida también como arquitectura de múltiples-tier; emerge para cubrir las limitaciones de la arquitectura de Dos-Tiers, ya que ahora se agrega una capa intermedia la cual se añade entre el ambiente de la interfase cliente y el ambiente servidor que administra la base de datos. Esta capa intermedia se puede implementar de muchas maneras, como monitor de procesos transaccionales, mensajes del servidor, o servidores de aplicación. La capa intermedia puede desempeñarse como: encolado, ejecución de aplicaciones, and database staging. Por ejemplo, Si esta capa provee encolado, el cliente envía su requerimiento a la capa intermedia para que esta acceda a los datos y retorne la respuesta al cliente. Además esta capa agrega calendarizacion y priorizacion de los trabajos. Las tres capas en la arquitectura Cliente/Servidor ha sido diseñada para mejorar el desempeño de una gran cantidad de usuarios (en el orden de los miles) y brindar flexibilidad.

1.3.1 Arquitectura de Tres-Tiers con tecnología de Monitor de Procesos Transaccionales.

El tipo mas básico de arquitectura de Tres-Tiers consiste de Transaction Processing (TP) monitor. La base de la tecnología TP es encolar mensajes, candalerizar mensajes, y priorizar servicios, donde el cliente se conecta al TP monitor (capa intermedia) en vez de conectarse al servidor de base de datos. La transacción es aceptada por el monitor, la cual la encola y toma la responsabilidad para administrar la transacción y culminarla, librando al cliente de esta transacción. Cuando esta capacidad esta embebida en la DBMS (puede ser considerada como una arquitectura de dos-tiers. La tecnología TP monitor provee de:

- La habilidad de actualizar múltiples DBMSs diferentes en una sola transacción
- Conectividad a una variedad de fuentes de datos inclusive archivos planos, DBMS no relacionales.
- La habilidad de dar prioridades a las transacciones.
- Seguridad robusta.

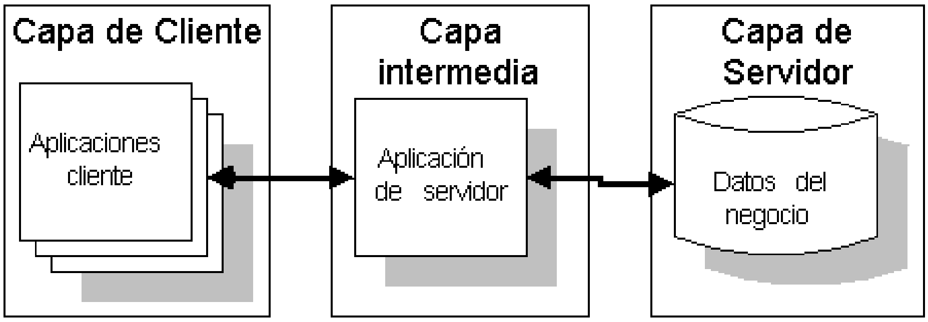

Fig. 1 Arquitectura Cliente/Servidor en tres- tiers

1.3.2 Arquitectura de Tres-Tiers con ORB.

Actualmente la industria esta trabajando en el desarrollo de estándares para proveer interoperabilidad y determinar lo que el Object Request Broker (ORB) va ha ser. La gran promesa es un sistema de desarrollo cliente/servidor usando tecnologías que soporte objetos distribuidos, como estas tecnologías soportan interoperabilidad a través de lenguajes y plataformas, también permiten aumentar el mantenimiento y adaptabilidad del sistema. Hay actualmente dos promesas en tecnología de objetos distribuidos:

- Commond Object Request Broker Arquitecture (CORBA)
- COM/DCOM

La industria esta trabajando en estándares para proveer interoperabilidad entre CORBA y COM/DCOM. EL Object Management Group (OMG) ha desarrollado un mapeo entre CORBA y COM/DCOM y es soportado por algunos productos.

2. Descripción general del proyecto

El aplicativo esta basado bajo la arquitectura Cliente/Servidor, haciendo uso de CORBA, JDBC, y Applets, se creo una aplicación de tres tiers, siendo usada desde el Web, en definitiva el sistema presenta los Applets que serán invocados desde una pagina HTML que es cargada desde el Web Server.

Los clientes Applets invocan a objetos en el servidor (objetos CORBA), a través del protocolo IIOP, toda la lógica del negocio se encuentra en los objetos servidores, para acceder a los datos los objetos servidores se comunican con el objeto controlador de acceso a la base de datos (SQL SERVER) con soporte JDBC.

Las paginas HTML y los Applets deben residir en el servidor donde reside el Web Server, los objetos servidores CORBA pueden encontrarse en el mismo servidor o en otro en la LAN, la base de datos puede residir de igual manera en el Web Server o en otro equipo en la LAN.

El sistema facilita la reservación de una habitación en un hotel, para permitir las siguientes transacciones principales:

- **Reservación de una habitación:** Se ingresa un nombre, numero de cedula/pasaporte, tipo de habitación, fecha de check-in y check-out, y el numero de personas. El sistema devuelve el valor que se debía facturar.
- **Cancelación de la reservación:** Las habitaciones dentro de esta reservación se liberaban, para estar disponibles a nuevas reservaciones o check-in.
- **Chech-In:** Se verificaba en el sistema las habitaciones disponibles si no había reservación previa, de existir se asignaba la/s habitaciones reservadas.
- **Check-Out:** El usuario al devolver la habitación, y de acuerdo a los días en las que estuvo en la misma, el sistema calcula el costo y emite la respectiva factura indicando el idusruaio, nombre, el detalle de las habitaciones, la fecha de check-in y check-out.

El análisis y diseño del sistema fue realizado con la metodología de Análisis y Diseño Orientación a Objetos, utilizando una combinación de las técnicas de OMT/Rumbaugh, Booch y Jacobson, conocidas como UML

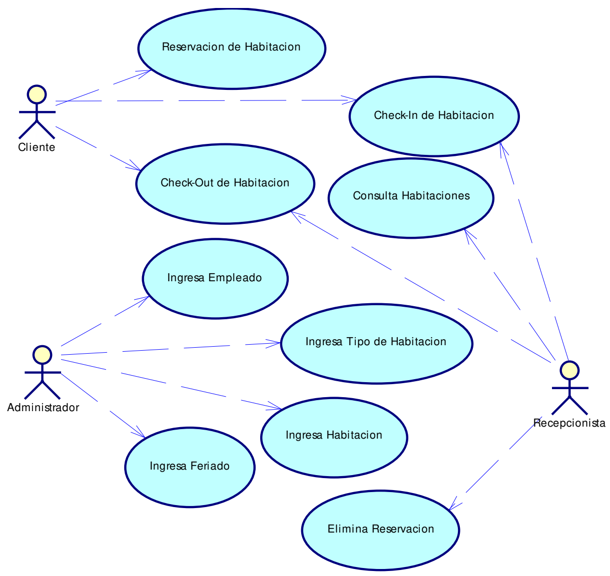

Fig 2. Diagrama de Casos de Uso

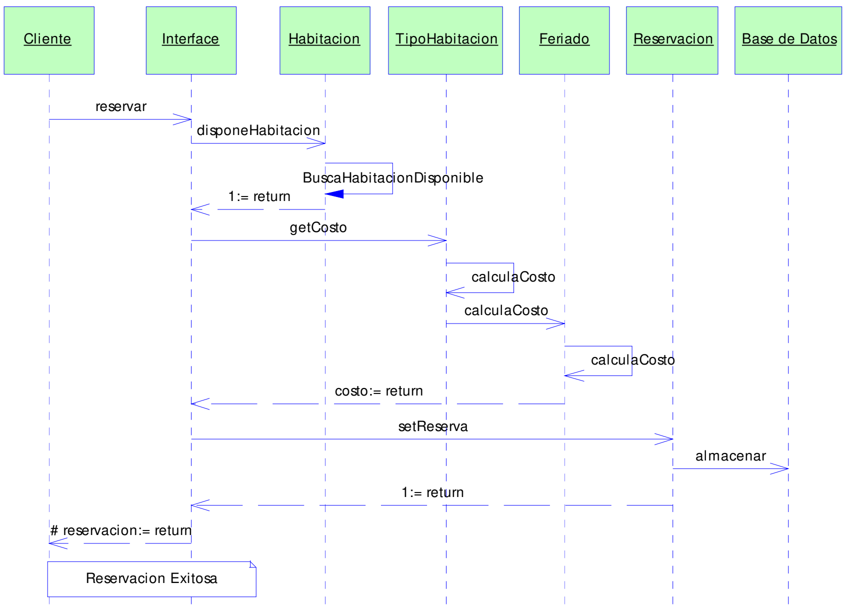

Fig 3. Diagrama de Interacción de Objetos

CONCLUSIONES

No hay una única posibilidad a la hora de distribuir las reglas de negocio dentro de un esquema Cliente/Servidor. Sin embargo, sí hay ciertas pautas que se pueden tener en cuenta a la hora de tomar una decisión.

CORBA nos permite: Facilidad de implementación de las interfaces, facilidad de ampliación de funcionalidad, transparencia total respecto a la distribución (incluso excepciones), potencial enorme: multilenguaje, multiarquitetcura, multiprotocolo, multiplataforma.

Interfaz dinámica frente a estáticas.

La adopción de un diseño distribuido de aplicaciones empresariales, aumenta la reusabilidad, reduce la cantidad de recursos, y los costos necesarios de desarrollo y mantenimiento.

En la actualidad el ambiente tecnológico es muy variante, las aplicaciones distribuidas brindan la facilidad de escalar o soportar cambios dinámicos. Un ejemplo de esto en nuestra aplicación, ya que si deseamos cambiar de DBMS lo podemos hacer y solo tenemos que heredar del objeto de acceso a la base datos e implementar para la nueva DBMS, sin realizar cambios en los objetos del negocio.

Este nuevo enfoque de diseño pone en manos de los desarrolladores no solo la funcionalidad que demandan las aplicaciones, sino también la seguridad, rapidez y flexibilidad.

REFERENCIAS

a) Libro con edicion

1. Robert Orfali, Dan Harkey, Client/ server Programimng with JAVA and CORBA (Second Edition, Jhon Wiley & Sons. Inc. Publishing, 1998)

b) Libro con edicion

2. Robert Orfali, Dan Harkey, Jeri Edwards, The Essential Client/Server Survival Guide (Second Edition, Wiley Computer Publishing, 1996)

c) Libro

3. Jeri Edwards, Devorah DeVoe, 3-Tier Client/Server At Work, Wiley Computer Publishing, 1997, pp 19-24, 41-45

d) Libro

4. Richard Monson-Haefel, Enterprise JAVA BEANS, Guide (Second Edition, O'Reilly, 2000), pp 14-16, 19, 39-72, 390-393, 405-408

e) Libro

5. Wendy Boggs, Michael, Mastering UML with Rational Rose 2002, SYBEX Inc, 2002

f) Libro

6. Kenneth S. Rubin, Developing Object-Orineted Software An Experience-Based Approach, IBM Object-Orinted Technology Center, Prentice Hall PTR, New Jersey, 1997

g) Tutorial

7. Santiago Comella-Dorda, John Robert, Robert Seacord, Kurt Wallnau, Enterprise Java Beans: A COSTS Architecture for Modern Enterprise Systems, (The 99 Software Enginnering Symposium, Carnegie Mellon University 1999), http://www.sei.cmu.edu/cbs/cbs_slides/99symposium/032tut.pdf, pp 1-80

h) Referencias de Internet

8. Darleen Sadoski, Frank Rogers, 2 August 97, Client/Server Software Architectures- An Overview, http://www.sei.cmu.edu/str/descriptions/clientserver_body.html

9. Darleen Sadoski, Santiago Comella-Dorda , 16 Feb 2000, Three Tier Software Architectures http://www.sei.cmu.edu/str/descriptions/threetier.html#34492

10. Darleen Sadoski, 2 January 97, Two Tier Software Architectures http://www.sei.cmu.edu/str/descriptions/twotier.html