

Tecnologías de Desarrollo de Sistemas Distribuidos basados en Objetos

Resumen

Debido al auge que se ha venido dando últimamente en el uso de las redes, se ha incrementado el crecimiento de los entornos distribuidos y heterogéneos. El desarrollo de aplicaciones distribuidas enfrenta diferencias de arquitectura, tales como: el hardware, el sistema operativo, el ambiente de desarrollo, el lenguaje de programación e incluso el paradigma de programación, por todo esto ha sido necesario utilizar diferentes tecnologías y mecanismos de desarrollo. La programación distribuida hace uso de distintas tecnologías e incluso puede mezclarlas para generar nuevas. Este artículo presenta tres tecnologías para la programación distribuida, las denominadas Tecnologías de Desarrollo de Sistemas Distribuidos basados en Objetos.

Palabras clave: Aplicaciones distribuidas, CORBA, COM /DCOM/ActiveX, JavaBeans.

1. Introducción

El presente artículo presentará tres de las tecnologías utilizadas para el desarrollo de aplicaciones distribuidas, las cuales han generado un nuevo paradigma en el desarrollo de aplicaciones distribuidas, denominado, aplicaciones basadas en "Plataformas de Componentes Distribuidos". Primeramente se hablará de CORBA (Common Object Request Broker Architecture), que es una tecnología de integración que define un marco estándar para interoperabilidad entre objetos con independencia del lenguaje y de forma transparente al programador. Posteriormente se describirá COM/DCOM/ActiveX, que son mecanismos de comunicación entre procesos diseñados principalmente para los sistemas Windows. Y por último se hablará de los JavaBeans, que son un modelo de componentes que

favorece la reutilización y que son visualizados en un entorno Java.

2. CORBA

CORBA (Common Object Request Broker Architecture), es una arquitectura de objetos distribuidos, que con el patrocinio del grupo OMG (Object Management Group) compuesto por compañías como American Airlines, Canon, Data General, HP, Philips Telecomunicaciones, Sun, 3Com, Microsoft y Unisys entre otros; se ha convertido en un estándar y gracias a ello permite a aplicaciones de software implementadas incluso en diferentes lenguajes comunicarse entre sí, a través de sistemas de cómputo, que a su vez pueden estar conformados por hardware, sistemas operativos distintos y que forman parte de alguna red. Además, la ejecución de objetos remotos se puede lograr sin la necesidad de un servidor Web.^[6]

CORBA al ser un estándar cuenta con un conjunto de especificaciones, sobre las cuales los vendedores de implementaciones de CORBA, conocidas como Object Request Broker (ORB) se apegan, para facilitar la comunicación con la implementación de otro vendedor.^[9]

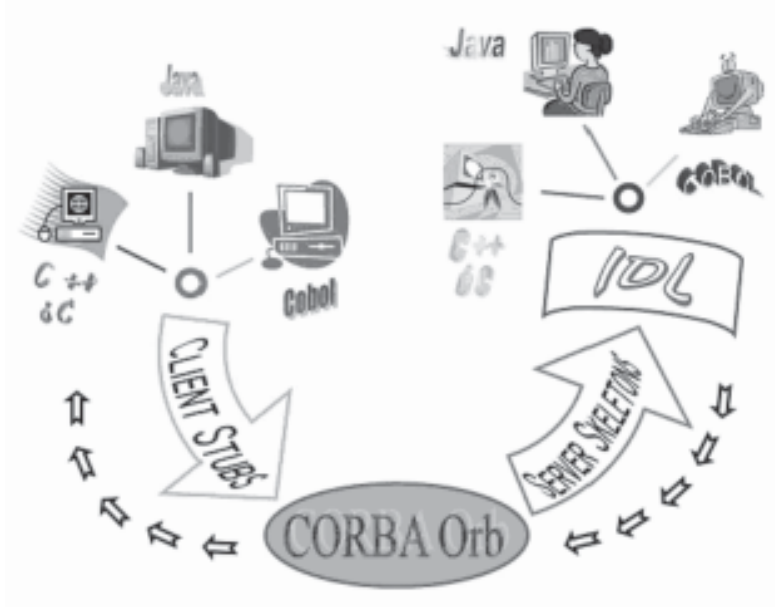
CORBA cuenta con tres elementos principales en los cuales se basa: El lenguaje de definición de interfaces IDL (Interface Definition Language), el ORB (Object Request Broker) y el protocolo GIOP (General Inter-ORB Protocol).^[5]

En cuanto al modelado de los objetos, CORBA hace uso del modelo cliente/servidor para el manejo de los mensajes y así establecer la comunicación entre ellos, cuando un objeto en una aplicación cliente requiere ejecutar los métodos de un objeto remoto en una aplicación servidor, hace uso del ORB que es específico para el lenguaje y la plataforma de cada aplicación, el

cual traduce la llamada del cliente a un formato neutro, totalmente independiente, que puede transportarse sobre cualquier medio para el cual exista un protocolo de comunicación.^[9]

Un esquema conceptual de la arquitectura de CORBA se muestra en la figura 1.

FIGURA 1: ESQUEMA DE CORBA



Como se observa en la figura 1, se tienen tres elementos importantes: El Cliente Stub, un Servidor Skeleton y el ORB.

El Cliente Stub es una entidad de programa que invoca una operación sobre la implementación de un objeto remoto a través de un Stub cuyo propósito es lograr que la petición de un cliente llegue hasta el ORB Core. Logrando el acoplamiento entre el lenguaje de programación en que se escribe el cliente y el ORB Core. El stub crea y expide las solicitudes del cliente.

Un Servidor Skeleton es la implementación de un objeto CORBA en algún lenguaje de programación, y define las operaciones que soporta una interface IDL CORBA. Puede escribirse en una gran variedad de lenguajes como C, C++, Java, Ada o Smalltalk. Y a través del skeleton entrega las solicitudes procedentes del ORB a la implementación del objeto CORBA.

La función del ORB consiste en conectar las dos partes: cliente y servidor. Presumiblemente estas partes se ejecutan sobre plataformas distintas y funcionan con diferentes sistemas operativos. Esto significa que pueden existir diferencias en tipos de datos, el orden de

los parámetros en una llamada, el orden de los bytes en una palabra según el tipo de procesador, etc. Es misión del ORB efectuar los procesos conocidos como marshaling y unmarshaling. En caso de que el método invocado devuelva un valor de retorno, la función de los ORB del cliente y servidor se invierte. Éste realiza el marshaling de dicho valor y lo envía al ORB del cliente, que será el que realice el unmarshaling y finalmente facilite el valor en formato nativo.^[12]

Existe una gran variedad de implementaciones CORBA. En las siguientes páginas web:

- <http://www.puder.org/corba/matrix/>
 - <http://adams.patriot.net/~tvalesky/freecorba.html>
- se encuentran implementaciones basadas en CORBA y se describen sus características, algunas son propietarias y otras más son libres.

3. COM/DCOM/Activex

COM (Component Object Model) es un estándar que permite la creación de objetos que ejecuten tareas que resuelven problemas específicos pero comunes a varias aplicaciones que puedan desear hacer uso de ellos. Estos pueden ser invocados por diferentes programas que los requieran, tanto OLE como ActiveX están basados en esta tecnología.

La idea es tener un mundo de objetos independientes de un lenguaje de programación. Por ello COM proporciona un estándar para las comunicaciones entre componentes, de tal forma, que una aplicación puede utilizar características de cualquier otro objeto de la aplicación, o del sistema operativo, y permite actualizar el software de un componente sin afectar a la operación de la solución global.[1]

COM soporta comunicación entre objetos de equipos de cómputo distintos, en una LAN, WAN, o incluso en Internet.

DCOM extiende el estándar COM de objetos remotos, para su utilización en redes. Inicialmente se desarrolló para Windows NT 4.0, y posteriormente para Solaris 2.x y Macintosh, así como para diferentes versiones UNIX.

Se encarga de manejar los detalles muy bajos de protocolos de red, por lo que el desarrollador se puede centrar en la realidad de los negocios, proporcionando así mejores soluciones a los clientes.

La arquitectura define cómo los componentes y sus clientes interactúan entre sí. Esta interacción es definida

de tal manera que el cliente y el componente puede conectarse sin la necesidad de un sistema intermedio. El cliente llama a los métodos del componente sin tener que preocuparse de niveles más complejos.

DCOM olvida completamente la localización de los componentes, no importando que estén en el mismo proceso que el cliente o en una máquina en cualquier lugar del mundo. En cualquier caso, la forma en la que el cliente se conecta a un componente y llama a los métodos de éste, es idéntica. No es sólo que no necesite cambios en el código fuente, sino que además no necesita que el programa sea recompilado. Una simple reconfiguración cambia la forma en la que los componentes se conectan entre sí.

La independencia de localización en DCOM simplifica enormemente las tareas de los componentes de aplicaciones distribuidas para alcanzar un nivel de funcionamiento óptimo. Supongamos, por ejemplo, que cierto componente debe ser localizado en una máquina específica en un lugar determinado. Si la aplicación tiene numerosos componentes pequeños, se puede reducir la carga de la red situándolos en la misma LAN, en la misma máquina, o incluso en el mismo proceso. Si la aplicación está compuesta por un pequeño número de grandes componentes, la carga de red es menor y no es un problema, por tanto se pueden poner en las máquinas más rápidas disponibles independientemente de donde estén situadas.

Es completamente independiente del lenguaje. Casi cualquier lenguaje puede ser utilizado para crear componentes COM, y estos componentes puede ser utilizado por muchos más lenguajes y herramientas. Java, Microsoft Visual C++, Microsoft Visual Basic, Delphi, PowerBuilder, y Micro Focus COBOL interactúan perfectamente con DCOM.

Puede utilizar cualquier protocolo de transporte, como TCP/IP, UDP, IPX/SPX y NetBIOS, y proporciona un marco de seguridad a todos estos protocolos.

Los desarrolladores pueden utilizar las características proporcionadas por DCOM y asegurar que sus aplicaciones son completamente independientes del protocolo.[11]

DCOM está pensado para que el sistema pueda funcionar bajo cualquier tipo de red, ya sea LAN, WAN o Internet, de forma que se solucionen los múltiples problemas que añaden estos entornos.

La denominada tecnología ActiveX desarrollada por Microsoft hizo su aparición en Internet con el navegador Internet Explorer 3.0. Su objetivo es similar al de los plug-ins, insertar objetos de diferente tipo en una página Web, aunque va mucho más allá al añadir mayores posibilidades de interacción y comunicación con programas externos, funciona de una manera similar al mecanismo de Microsoft OLE que usa el sistema operativo Windows, lo realmente novedoso es la aplicación de esta tecnología al WWW. Guarda parecido con los objetos para plug-ins y con los applets Java, aunque presenta algunas mejoras con respecto a ambos. Como ocurría con los plug-ins, los controles y documentos de ActiveX pueden ser insertados en una página Web, sin embargo no requieren de un pequeño programa para cada tipo de objeto ActiveX, esta nueva tecnología ha sido denominada como auto-contenida porque cada objeto tiene suficiente información para ejecutarse él mismo sin ayuda de ninguna aplicación.

Tal y como ocurre con los applets podemos crear nuestros propios controles o usar los creados por otros programadores. Si se opta por programarlos existen herramientas que lo facilitan, entre ellas destacan las creadas para este fin por Microsoft y que distribuye a través de su Web.^[3]

ActiveX está compuesto por dos tipos de objetos:

- *Controles Activos (Active controls)*
- *Documentos Activos (Active documents)*

Estos últimos permiten insertar documentos con formato PDF, DOC, etc.^[5]

La tecnología ActiveX constituye una interesante aportación para aumentar la interactividad y capacidad de representación de los documentos Web. La filosofía de su aplicación es sencilla: Dentro de una página se insertan componentes (pequeñas aplicaciones) capaces de interactuar con el usuario, realizar cálculos o representar datos. Se insertan dentro de un documento con las etiquetas <OBJECT> y <PARAM>, estas corresponden a las operaciones de inserción del componente y paso de sus parámetros de ejecución. Como se puede notar, no hay grandes diferencias con los *plug-ins* de Netscape.^[3]

Cabe mencionar que los controles ActiveX que no estén firmados pueden generar un alto riesgo de seguridad, ya que pueden contener código malicioso que puede tomar el control de la computadora de manera remota o dañar archivos en el disco duro.^[5]

4. JavaBeans

Los JavaBeans traen la tecnología de componentes a la Plataforma Java. Un componente software es una parte básica para la construcción de una aplicación, con las siguientes características que lo diferencian del resto del código de un proyecto: Independencia de la plataforma, independencia del lenguaje, encapsulación, basados en un modelo estándar permitiendo a los componentes interactuar entre sí, JavaBeans les da a sus componentes estas propiedades. JavaBeans es un modelo de componentes software que ofrece flexibilidad, reutilización y que puede ser visualizado por un lenguaje de programación Java.

Los beans pueden ser aplicaciones que se pueden personalizar. El API de bean incluye clases e interfaces para personalizar en tiempo de diseño y en tiempo de ejecución. Se pueden personalizar sus propiedades. Las propiedades con tipos de datos simples no tienen problema, en cambio hay otro tipo de propiedades más complejas que se entrecruzan y que necesitan para ser personalizadas un editor de propiedades que ayuda a tal tarea.^[10]

La especificación del API JavaBeans es una descripción completa de los JavaBeans, ésta se puede hallar en la página web:

<http://java.sun.com/products/javabeans/docs/spec.html>

Los beans deben entenderse entre sí de alguna forma, debe haber comunicación entre ellos para que la aplicación que forman realicen su misión. Los eventos son como los mensajes de C++ para la comunicación entre los objetos. JavaBeans sigue el modelo de eventos de Java.

Las propiedades de un objeto determinan su estado y lo diferencian del resto, el estado debe ser conocido para que se configuren sus características así como para que sus cambios se transmitan a otros beans. Hay ocasiones en las que no se puede, ni tampoco interesa, cambiar ni examinar el valor de ciertas propiedades. Dados varios *beans*, es tarea del diseñador conectar los beans y construir una aplicación coherente. Para ello es necesario que conozca sus propiedades y que se ajuste a las necesidades de la aplicación completa.^[7]


Por último JavaBeans no tiene un soporte para persistencia, más bien usa un método primario como la serialización. Pero la persistencia abarca además la seguridad de los beans y la compatibilidad de versiones.

5. Conclusiones

Las tres tecnologías tienen como base la arquitectura cliente-servidor, utilizan un protocolo de transporte para enviar mensajes a través de las computadoras en una red y además usan un tipo de invocación de método remoto.

CORBA es un estándar que ha sido ampliamente probado y cuyo objetivo es facilitar la creación de aplicaciones distribuidas permitiendo la interoperabilidad entre componentes de software, de forma independiente a los lenguajes en que hayan sido desarrolladas, así como de la plataforma hardware y sistemas operativos donde se ejecutan, es una tecnología más madura que las otras dos citadas en este artículo, se hallan disponibles diversas implementaciones de este estándar, el desarrollo de aplicaciones es más complejo que en DCOM y los componentes Java, proporciona un conjunto más completo de servicios distribuidos. Se le puede llamar una tecnología de integración.

Por su parte las tecnologías COM/DCOM y ActiveX también permiten el desarrollo de aplicaciones distribuidas, sin embargo al principio fueron duramente criticadas por ser un estándar para servir únicamente al sistema operativo de Microsoft. Al presentar problemas de seguridad con el uso de los Controles ActiveX en Internet, Microsoft trató de resolverlos con el desarrollo del framework Net. Gracias a su arquitectura Net ha permitido que los programadores experimentados en lenguajes como el VisualBasic puedan desarrollar aplicaciones distribuidas incluso para dispositivos PDA colocándose en el mercado actual como una tecnología de vanguardia para este tipo de desarrollos.

Los JavaBeans se utilizan solamente con lenguaje Java, aunque Java IDL se utiliza en diversos lenguajes, se han visto bastante restringidos en cuanto a funcionalidad en aplicaciones grandes, de aquí que Sun ha proporcionado nuevas tecnologías para apoyar en el desarrollo de aplicaciones empresariales y distribuidas con el desarrollo del EJB. Java esta creciendo con API's para proporcionar mejores servicios distribuidos. JavaBeans se conoce como una tecnología de programación 

Referencias.

- [1] APPLEMAN DAN
(2000) Desarrollo de componentes COM/ActiveX con Visual Basic 6
- [2] BLUM ADAM
(1997) ActiveX web programming ISAPI, controls, and scripting
- [3] BROWN MATTHEW E.
(1997) 10 [ten]minute guide to ActiveX control pad
- [4] CUENCA JIMÉNEZ PEDRO MANUEL
(1997) Programación en Java, ANAYA Multimedia
- [5] ERNST WARREN
(1997) Introducción a ActiveX
- [6] HENNING, MICH
(2002) Programación avanzada en CORBA con C++, Pearson Educación
- [7] MONSON-HAEFEL RICHARD
(2001) Enterprise JavaBeans, Sebastopol O'Reilly
- [8] Nicolás Cedric
(2000) JAVA Cliente-Servidor: JDK1.1, JavaBeans, JDBC, Corba/RMI, MarimbaCastenet, Barcelona Eyrolles
- [9] ORFALI ROBERT
(1997) Client/Server programming with Java and CORBA, Wiley
- [10] PIROZ MOHSENI, TOM STEWART
(1997) Guía de desarrollo de JavaBeans, ANAYA Multimedia
- [11] SESSIONS ROGER
COM and DCOM: Microsoft's Vision for Distributed Objects
- [12] ZAHAVI RON
(2000) Enterprise application integration with CORBA: component and Web-based solutions, Wiley

Everth Rocha Trejo¹

Imelda Vertti Guzmán²

¹Universidad Tecnológica de la Mixteca

²Instituto Tecnológico de Toluca