ESCUELA SUPERIOR
POLITECNICA DEL LITORAL

**Facultad de Ingeniería en
Electricidad y Computación**

"Tecnología Cliente Servidor con Arquitectura CORBA"

**Simulación del Proceso de
Comercialización Agrícola**

Proyecto de Tópico de Graduación
Previa a la obtención del Título de
INGENIERO EN COMPUTACION

GUAYAQUIL - ECUADOR
1998

Presentado por:

Iván Cabrera Salazar

Gina Carvajal Mata

Pedro Echeverría Briones

Gleiston Guerrero Ulloa

Alfonso Guijarro Rodriguez

Jaime Holguín Salazar

Fabricio Júpiter Mera

Harvey Marín Avalos

William Reyes Aguilar

Jenny Sabando Mendoza

David Sánchez Rodriguez

Paul Seminario López

Víctor Soria Heredia

Alfredo Toasa Peñafiel

Galo Valverde Landívar

Moisés Vera Solórzano

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestos en esta tesis, nos corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL"

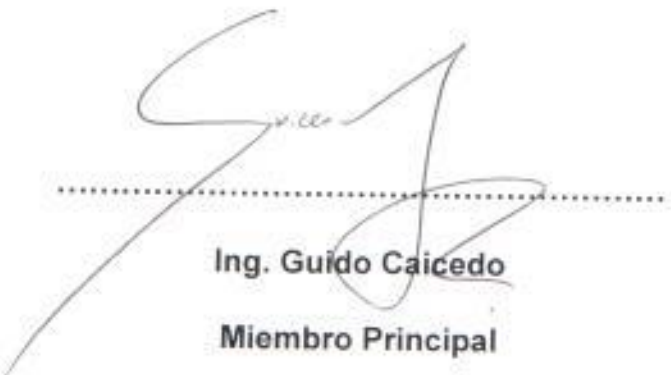
(Reglamento de Exámenes y Títulos profesionales de la ESPOL)

.....

TRIBUNAL DE GRADO

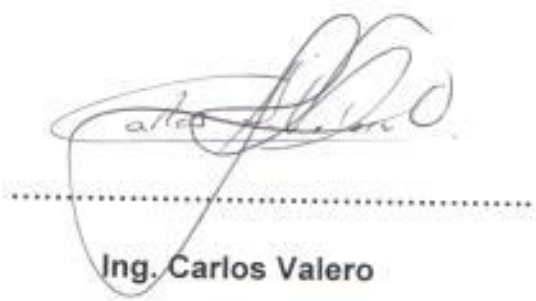
Ing. Servio Lima

Miembro Principal



Ing. Guido Caicedo

Miembro Principal



Ing. Carlos Valero

Profesor de Tópico

Ing. Armando Altamirano

Subdecano de la FIEC

RESUMEN

CORBA (Common Object Request Broker Arquitecture), es una nueva arquitectura para el desarrollo de sistemas distribuidos. Su aplicación es conveniente en situaciones en las que las herramientas tradicionales no ofrecen suficiente funcionalidad ni confiabilidad.

Para demostrar su aplicabilidad, en este trabajo se ha elaborado un sistema de comercialización de productos agropecuarios, el mismo que implementa un mecanismo de negociación que se realiza en la Bolsa Nacional de Productos.

Sin embargo, el desarrollo de una aplicación basada en esta arquitectura, demanda que su diseño también obedezca a la tecnología de orientación a objetos (diseño orientado a objetos).

Se encontraron e implementaron tres subsistemas bien definidos, a saber: Membresía de Corredor, Membresía de Producto, Negociación de Producto.

TABLA DE CONTENIDO

INTRODUCCION

Antecedentes

La tecnología Cliente/Servidor, ha evolucionado mucho en tiempos recientes. Así, mientras en sus inicios esta tecnología empleaba un esquema conocido como "Two-Tier" (dos niveles o capas), hoy en día se emplea esquemas mas complejos, conocidos como "Three-Tier" o "Multi-Tier".

La arquitectura CORBA, utilizada en este trabajo, justamente obedece a estos nuevos esquemas.

CORBA basa su gran funcionalidad en la tecnología de orientación a objetos; con ello aprovecha todas las ventajas que esta tecnología ofrece, tales como la independencia del lenguaje de desarrollo, implementación de objetos, y la reusabilidad de tales objetos.

Si bien con esto se logra una verdadera independencia del lenguaje de desarrollo, en la práctica es muy conveniente escoger uno solo, por lo que en este trabajo hemos seleccionado el lenguaje Java, el mismo que se acopla perfectamente a la arquitectura CORBA.

Cabe señalar que en nuestro medio los esquemas "Three-Tier" o "Multi-Tier" no se encuentran muy difundidos, siendo el modelo "Two-Tier" el más común en la mayoría de sistemas con arquitectura Cliente-Servidor. Por lo tanto, este trabajo tiene el carácter de pionero en nuestro medio.

Objetivos

- Crear un Sistema Cliente - Servidor, para la simulación del proceso de comercialización de productos agropecuarios, en el que se muestre claramente el uso de la Arquitectura CORBA.
- Demostrar la funcionalidad y practicidad del uso de Internet como medio de transmisión de información en aplicaciones que requieran cobertura a gran escala (nacional y/o internacional).

Hipótesis

Partimos de tres hipótesis:

1. El empleo de la arquitectura CORBA es conveniente en aplicaciones distribuidas, y especialmente para sistemas que quieran utilizar a la Internet como medio de transmisión de información.
2. El análisis y diseño orientado a objetos es capaz de modelar procesos de negociación.
3. La implementación de la interfaz gráfica basada en applets de Java, proporciona un ambiente lo suficientemente amigable y funcional.

Metodología

1) Levantamiento de información directamente con el usuario; por medio de:
   a) Entrevistas con los Funcionarios de la Bolsa Nacional de Productos Agropecuarios.
   b) Visitas de campo a almaceneras de la ciudad, para solicitar información.
2) Análisis y diseño orientado a objetos, utilizando una combinación de las metodologías de Rumbaugh y Fusion.

3) Proceso de desarrollo basado en modelos de implementación de interfaces con clases AWT de Java.
4) Implementación de objetos IDL de OMG.

Procedimientos Utilizados

Se presentan dos puntos de vista para proceder

Externos:

Instalación y configuración de los diferentes dispositivos de Redes de Area Local.

Selección de los sistemas operativos, requerimientos mínimos de hardware, herramientas de desarrollo y manejo de datos como lo son: JDK en versión 1.1.5, Visibroker 3.1, Visual Cafe 2.1, NetScape Communicator 4.04, Internet Explorer 4.01.

Preparación de las diferentes plataformas para el desarrollo del sistema.

Selección de la estrategia de diseño.

Búsqueda de la herramienta adecuada para la implementación del sistema.

Documentación para el soporte del sistema.

Internos:

Análisis basado en la factibilidad del sistema.

Desarrollo basado en modelos:

- Identificar y describir los modelos a construir.
- Definir las razones para describir los modelos.
- Definir las relaciones entre los modelos.
- Capturar la información en una Visión General.

GLOSARIO

API. Application Protocol Interface. Conjunto de funciones en la capa de aplicaciones que permiten iteractuar con la capa de protocolo.

Applets Clase o subrutina que encapsula su funcionalidad, escrita en Java. Puede ser usada bajo cualquier sistema operativo.

AWT Clase de Java que permite implementación de una interfaz gráfica

Back-end Denominación de la contraparte servidora en una plataforma Cliente-Servidor

BOA Basic Object Adapter. Proceso Servidor de Visibroker que activa y desactiva la definición de los objetos en la red.

Browser Programa que sobre el protocolo HTTP, recupera un documento desde un servidor Web, interpreta el código HTML, y presenta el mismo a manera gráfica, en nuestro caso con soporte para Java.

Compilador Programa encargado de convertir a instrucciones de bajo nivel, un programa diseñado con instrucciones de alto nivel.

Concurrencia Acceso simultáneo a un mismo recurso

CORBA Common Object Request Broker Architecture. Plataforma de implementación de ORB.

DCOM Arquitectura Microsoft para implementación de objetos distribuidos

DII Dynamic Invocation Interface. Interfaz para invocar los objetos, métodos y parámetros de un objeto ORB.

DLL Dynamic Link Library. Conjunto de librerías bajo la arquitectura Microsoft para iteractuar con aplicaciones.

DSI Dynamic Skeleton Interface. Interfaz de acceso a métodos para el lado servidor.

Enlaces dinámicos Direccionamiento lógico a un servicio o servidor bajo el protocolo http.

Front-end Denominación de la contraparte cliente en una plataforma Cliente-Servidor

Framework Entorno operativo, de comunicación o trabajo, para el intercambio de información

Gatekeeper Servicio que permite a usuarios Web accesar a servicios provistos por objetos que son parte de un sistema interno.

Hardware Componentes electrónicos de un equipo de computación

Hipervínculo Direccionamiento lógico a un servicio o servidor bajo el protocolo http.

HTML Codificación aplicada a archivos de texto que permite su acceso como páginas con formato.

IDL Interface Definition Lenguage. Lenguaje algoritmico para definir los objetos base en la arquitectura ORB.

IIOP Inter Internet Object Protocol. Protocolo de intercambio de objetos.

Interface Repository Servicio de Visibroker que sirve como repositorio de interfaces para sus invocaciones desde los clientes.

Interfaz Representación gráfica para iteracción de los actores de un sistema.

Interproceso Intercambio de información entre servicios internos a un sistema.

Intertareas Intercambio de información entre procesos internos a un sistema.

JAR Java Arc Repository. Archivos de clases Java empaquetados con funcionalidad implícita.

Java Lenguaje de programación orientado a objetos

JavaBeans Objetos de tipo multimedia creados con Java.

JDBC Driver de conexión a bases de datos por medio de Java.

LAN Local Area Network. Redes de area local.

MAN Metropolitan Area Network. Redes de cobertura metropolitana.

Multired Entorno de operación sobre cualquier tipo de red.

Multiservicio Entorno que proporciona múltiples servicios.

Multiservidor Entorno operativo para soporte a un grupo de servidores.

Multitarea Sistema que permite ejecutar varias tareas paralelas a la vez.

Multiusuario Acceso múltiple a un conjunto de usuarios a un mismo servidor.

Naming Services Servicio Visibroker que permite a las aplicaciones cliente atar objetos usando nombres significativos y lógicos sin tener que direccionar por medio de cualquier convención de nombre de una plataforma específica

OLE Object Linking and Embeding. Arquitectura básica de Microsoft para invocación e iteracción entre objetos.

OMG Object Management Group. Grupo encargado de establecer estándares para la invocación de objetos.

ORB Object Request Broker. Arquitectura de soporte a aplicaciones orientadas a objetos bajo los estándares de la OMG.

OS Agent Servicio Visibroker para atender requerimientos de objetos CORBA.

Peer-to-peer Comunicación de datos a un mismo nivel de servicio.

Plug-ins Programas que se añaden para proporcionar un mejor soporte a una característica inicial de una aplicación.

Protocolo Códigos y procedimientos que hacen posible que se intercambie información entre dos nodos.

Proxy Servidor de administración de los paquetes de una red en contraparte de una red externa.

Semáforos Bits de señalización y coordinación para procesos.

Site Dominio de servicios bajo un ambiente particular.

Skeletons Esqueletos. Estructura inicial de operatibilidad del lado de un servidor CORBA.

Software Componentes lógicos de un equipo de computación.

SQL Structured Query Lenguage. Lenguaje estructurado de petición de requerimientos estándares a servidores de bases de datos.

Stored Procedures Procedimientos internos a una base de datos que proporcionan coherencia.

Stubs Estructura inicial de operatibilidad del lado de un cliente CORBA.

TAN Total Area Network. Redes de cobertura a escala corporativa.

Tareas colaborativas Conjunto de procesos que establecen un intercambio de eventos común.

Threads Hilos de ejecución de procesos.

Tier Capa lógica de servicios intermediarios entre cliente y servidor

TP-Monitor Servicio de monitoreo de transacciones contra un servidor.

WAN Wide Area Network. Redes de cobertura a gran escala.

WEB Forma de tener acceso a información a través de una red Internet.

Capítulo 1

TECNOLOGÍA CLIENTE-SERVIDOR

1.1 Definición

La tecnología Cliente/Servidor nos permite crear sistemas de software distribuido, en la cual la parte servidora (back-end) puede proveer servicios a muchos clientes (front-end).

Clientes y Servidores, son dos entidades lógicamente separadas, que operan juntas sobre una red de trabajo para complementar una tarea, por medio de un grupo de servicios intermediarios denominados middleware (figura 1).

Los requerimientos de desarrollo de aplicaciones Cliente/Servidor, actualmente incluyen procesos y transacciones, diseño de bases de datos, experiencia en comunicaciones y una interfaz gráfica amigable. Las aplicaciones más avanzadas requieren un conocimiento de objetos distribuidos e Internet

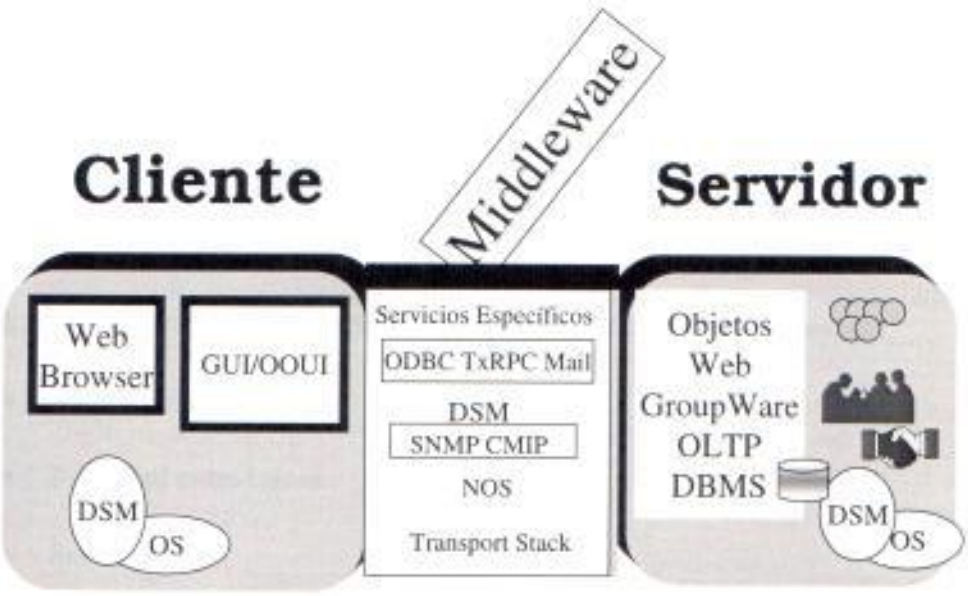


Figura 1. Arquitectura Cliente/Servidor .

La tecnología cliente/servidor es la alternativa completa en el desarrollo de aplicaciones en las que se debe maximizar el uso de los recursos, tanto de software como de hardware en una red de trabajo.

1.2 Características del Sistema Operativo

Los servidores requieren un alto nivel de concurrencia, debido a que, idealmente, una tarea separada debe ser asignada a cada uno de los clientes.

La Multitarea es la manera natural de simplificar la codificación de aplicaciones complejas que pueden ser divididas en una colección de tareas concurrentes y lógicamente distintas, a las que se denominan co-rutinas o hilos de ejecución (threads).

Las siguientes son las características o requerimientos necesarios:

- Tareas colaborativas.
- Prioridad entre tareas.
- Semáforos.
- Comunicaciones ínter – proceso, tanto Local como Remotas.
- Threads.
- Protección Intertareas.
- Sistema de archivo de alto desempeño y multiusuario.
- Administración de memoria eficiente.
- Instrucciones de ejecución con enlaces dinámicos.

1.3 Middleware

Es el término que cubre todas las necesidades de software distribuido para soportar las interacciones entre clientes y servidores. Es lo que hace posible la comunicación entre ambos.

Middleware no incluye el software que provee los servicios comunes, es decir lo propio del servidor; tampoco incluye la interfaz de usuario ni la lógica de la aplicación, es decir lo que pertenece al cliente.

1.3.1 Comunicación peer-to-peer

El término, peer - to - peer indica que los dos lados de un enlace de comunicación usan el mismo protocolo de interfaz para manejar una conversación en red. Cualquier computadora puede iniciar una conversación con cualquier otra. El protocolo es simétrico, por esto es a veces llamado program-to-program.

1.3.2 Objetos Distribuidos

Un sistema distribuido es una colección de objetos, donde cada objeto satisface y cumple una función determinada. Además, cada objeto encapsula funcionalidad aplicable a una entidad en particular.

Si analizamos básicamente a un objeto, podemos notar que es modificable tan sólo a través de su interfaz pero, para acceder a un objeto, es necesario tener una referencia al mismo; es decir, la referencia identifica el objeto.

1.3.3 Java Applets

Un applet no es una aplicación, sino más bien un componente que corre en un ambiente de browser. Estos permiten crear aplicaciones tamaño componente que los servidores puedan cargar sobre clientes usando páginas HTMLs.

1.3.4 Three Tier

El negociar con objetos, crea soluciones generales escalables para Cliente/Servidor Three Tier.

- El Primer Tier representa el aspecto visual del objeto negociado.
- En el Segundo Tier están los objetos servidores, que representan los datos persistentes y las funciones lógicas de negociación.
- En el Tercer Tier se encuentran las bases de datos y las aplicaciones servidoras tradicionales.

1.3.5 Two-Tier vs Three-Tier

En los sistemas clientes/servidor "Two-Tier" la lógica de aplicación está inmersa dentro de la interfaz del usuario, del lado del cliente, en el lado del servidor o en ambos. En los sistemas cliente/servidor "Three-Tier" la lógica de la aplicación está en la capa intermedia, está separada de los datos y de la interfaz del usuario.

En teoría, los sistemas Cliente/Servidor Three-Tier son más robustos, escalables y flexibles. En suma, ellos pueden integrar datos desde múltiples fuentes. Los ejemplos de este tipo sistemas pueden ser los TP-Monitor, objetos distribuidos y el WEB. Ejemplos de sistemas clientes Two-Tier típicos son: servidores de archivo, servidores de base de datos con Store Procedures.

CORBA

2.1 ORB

Un ORB (Object Request Broker), es el middleware que permite la comunicación entre Clientes y Servidores por medio de objetos.

Con ORB, un applet Java ordinario puede invocar directamente métodos en objetos CORBA usando el protocolo IIOP sobre el Internet.

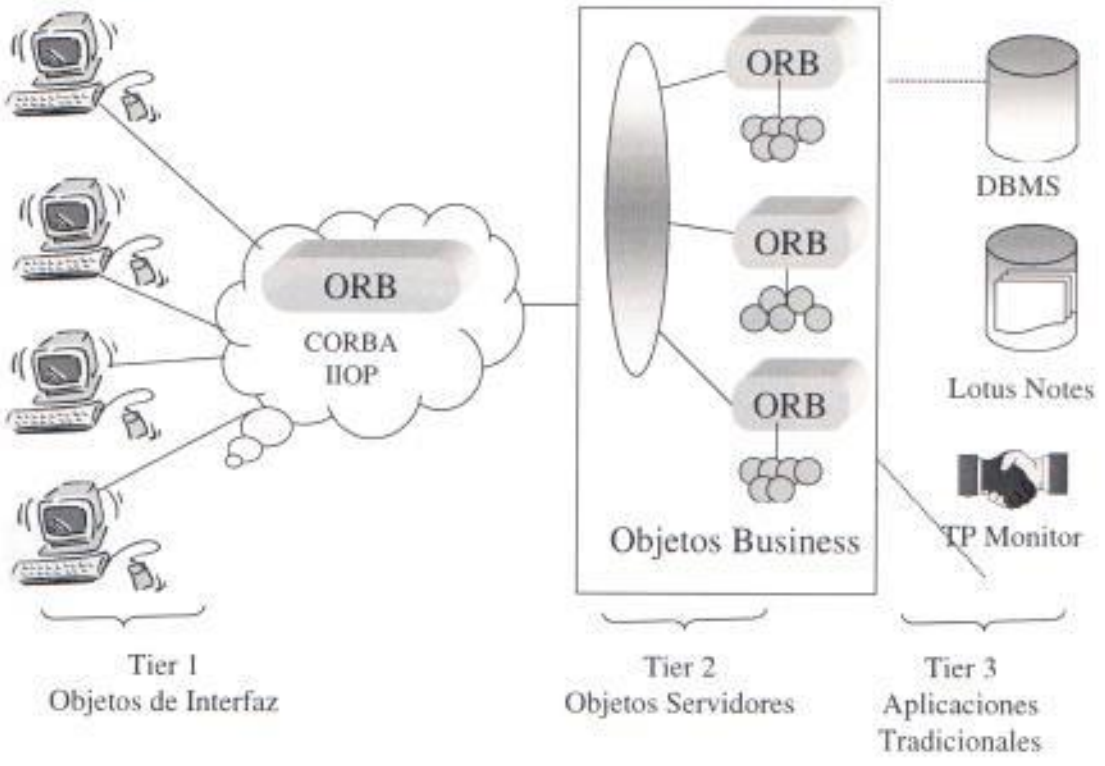
Una de las principales ventajas del empleo de ORB's es que el Cliente no tiene que saber dónde están localizados los objetos, ni su lenguaje de programación, ni su sistema operativo ni ningún otro aspecto de sistema. Simplemente requiere conocer la interfaz del objeto (figura 2).

Usando un ORB, un objeto Cliente transparentemente puede invocar un método en un objeto Servidor, que puede ser en la misma máquina o a través de una red. El ORB

intercepta las llamadas y es responsable de localizar un objeto que pueda atender la petición, pasa los parámetros, invoca su método y retorna los resultados.

Cabe recalcar que los roles del Cliente/Servidor son solamente usados para coordinar las interacciones entre dos objetos. Los objetos en el ORB pueden actuar tanto como Cliente o Servidor, dependiendo de la ocasión.

Figura 2. Modelo de un Objeto Web basado en Cliente Java y CORBAS ORBS



2.2 Características

Toda implementación de CORBA tiene las siguientes características (figura 3):

1. Sigue el modelo de objetos.
2. Utiliza la semántica y la sintaxis del IDL de OMG.
3. Implementa los siguientes componentes:
    - DII: Dynamic Invocation Interface.
    - DSI: Dynamic Skeleton Interface
    - Interface Repository.
    - ORB Interface.
    - Basic Object Adapter.

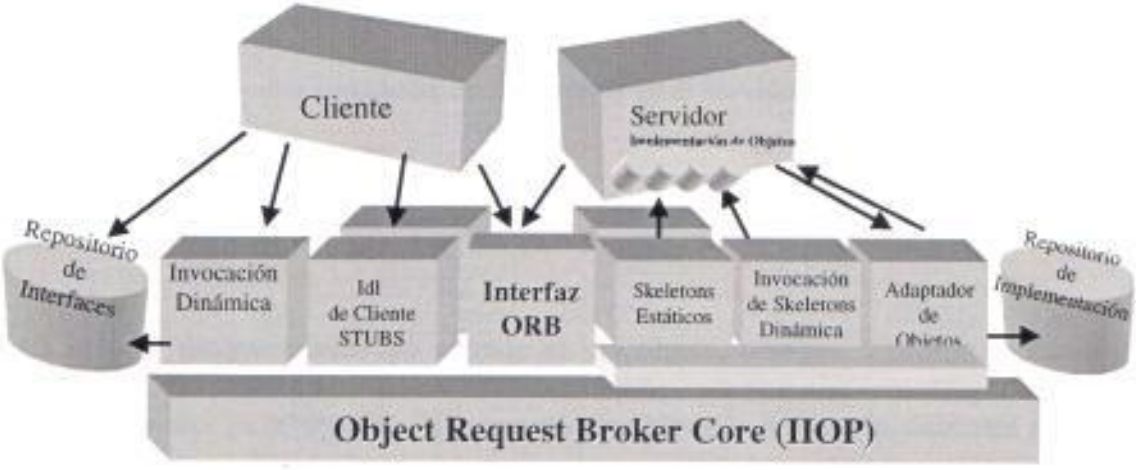


Figura 3. La Estructura de un ORB CORBA.

2.3 Invocaciones estáticas

Una invocación estática es similar a cualquier invocación a un método ordinario. Sin embargo, transparentemente para el usuario, es una invocación remota por medio del ORB. Una vez que obtenemos la referencia sobre los métodos del objeto interfaz, se usan para recuperar los datos de los objetos.

Para poder hacer invocaciones estáticas es necesario seguir los siguientes pasos: Definir los objetos por medio de IDL's (lenguaje de definición de interfaces), producir "skeletons" (esqueletos) del lenguaje para implementar las clases, añadir código de implementación a los "skeletons", compilar el código, referenciar las definiciones de las clases en el Repositorio de Interfaces, registrar los objetos en el Repositorio de Implementación y, crear instancias de los objetos en el servidor.

2.4 Invocaciones dinámicas

Para invocar dinámicamente un método en un objeto, primero debemos encontrar el objeto y obtener su referencia. Una vez que tengamos la referencia, debemos usar ésta para recuperar las interfaces de los objetos y dinámicamente construir la petición. Se debe especificar la petición del método que usted quiera ejecutar y sus parámetros.

2.5 Interfaces de invocaciones dinámicas

Los servicios que se necesitan para invocar dinámicamente un objeto (figura 4), son parte del núcleo de CORBA. Se necesitan 4 interfaces para usar invocación dinámica:

CORBA:: Objeto: Es una interfaz seudo-objeto, que define operaciones que todo objeto Corba puede soportar. Esta es la interfaz raíz para todos los objetos CORBA.

CORBA:: Requerimiento: Es una interfaz seudo-objeto, que define las operaciones en un objeto remoto.

CORBA:: NVList: Es una interfaz seudo-objeto, que ayuda a construir una lista de parámetros. La interfaz NVList, define operaciones que ayudan a manipular una lista.

CORBA:: ORB: Es una interfaz seudo-objeto, que define propósitos generales de los métodos. Se pueden invocar estos métodos en un seudo-objeto ORB de una implementación Cliente o Servidor.

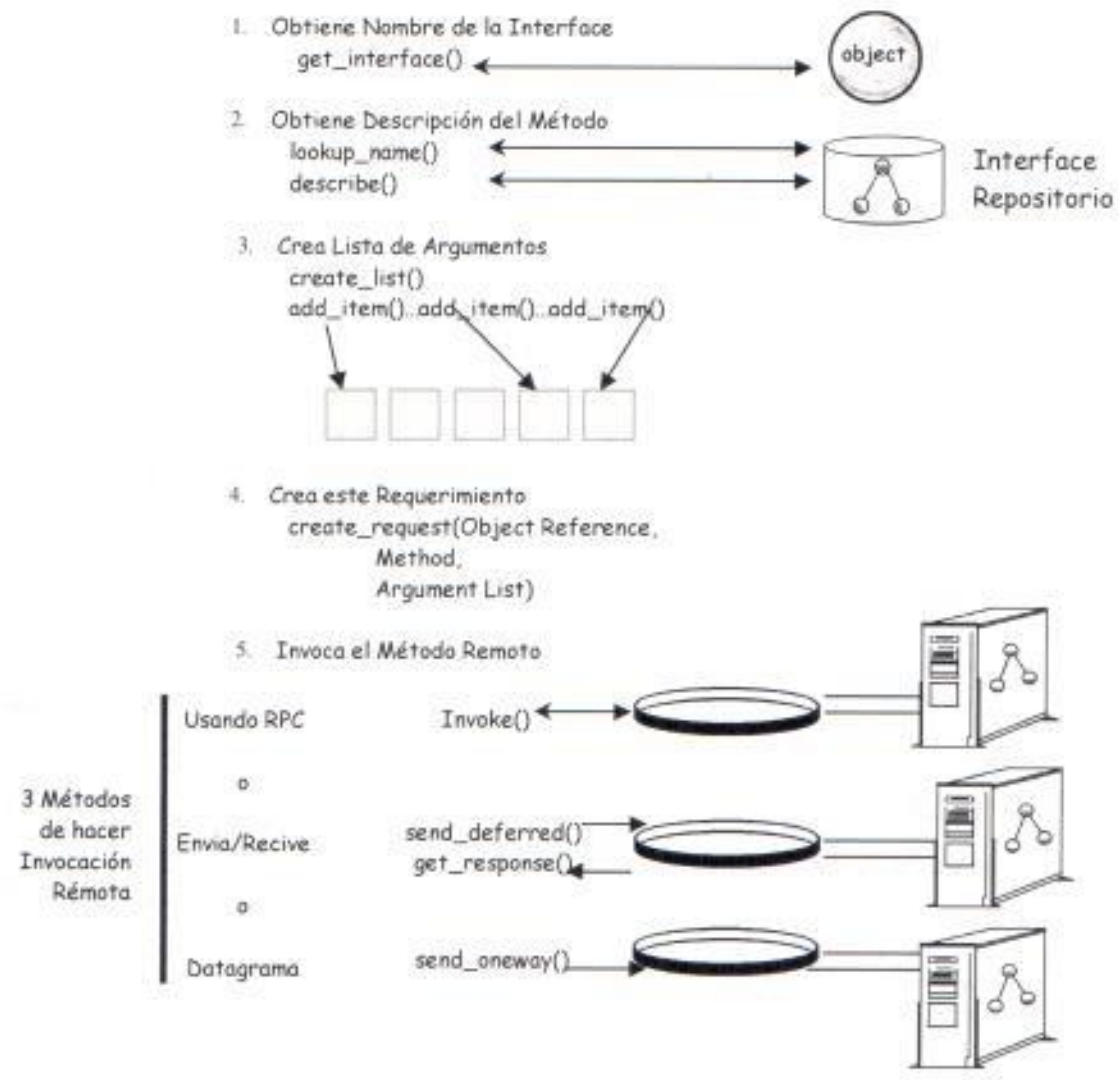


Figura 4. Procesos de invocaciones dinamicas.

2.6 DCOM vs CORBA

CORBA es visto como un middleware orientado a objetos, que se escoge cuando se necesita sistemas grandes, para soportar aplicaciones distribuidas que corren en una variedad de sistemas operativos. CORBA tiende a ser usado como una abstracción general de objetos y procesos de negociación; además, la conectividad de productos de CORBA permite una mezcla de plataformas (figura 5).

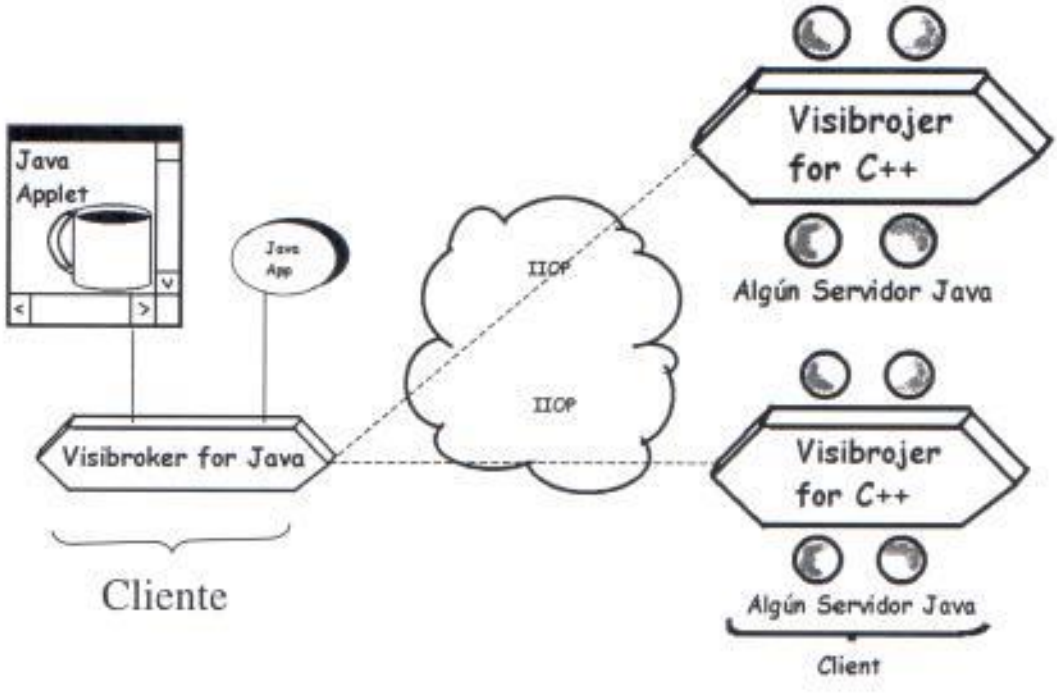


Figura 5. Visibroker: Comunicación Cliente/Servidor usando IIOP.

CORBA puede proveer requerimientos a una gran escala de frameworks o niveles de trabajo, orientados a objetos que soportan sistemas críticos a gran escala; y una infraestructura de comunicación de objetos de negociación proporcionado por el ORB.

Por otro lado, DCOM tiene su origen en las estaciones de trabajo y en la programación sobre Windows; esto incluye librerías DLL´s, su modelo binario y, el mecanismo RPC delineado para distribución de tipo COM, íntimamente ligado con OLE.

Filosóficamente la interfaz y el camino de acceso a clientes y servidores, difiere significativamente entre DCOM y CORBA.

Un objeto CORBA es una interfaz que es comprimida de todas las interfaces de las subclases. En contraste, un objeto DCOM tiene múltiples interfaces; un cliente de un objeto CORBA tiene acceso total a las interfaces de los objetos, por medio de referencias a los mismos. El código del cliente puede ser accesado a un método habilitado públicamente y a las propiedades que el objeto expone. En el otro caso todo acceso a un objeto DCOM es un puntero a la interfaz, la cual solo accesa al cliente con un particular grupo de funciones a la vez.

Capítulo III

APLICACIÓN DE LA TECNOLOGÍA CLIENTE SERVIDOR

3.1 Descripción del Proyecto

3.1.1 Descripción de la Bolsa Nacional de Productos Agropecuarios

La Corporación Bolsa Nacional de Productos Agropecuarios o Bolsa de Productos, inició sus operaciones en abril 25 de 1986. Es una institución con personería jurídica de derecho privado sin fines de lucro, con domicilio principal en la ciudad de Guayaquil y sucursales en las ciudades de Quito, Quevedo y Babahoyo, y tendrá una duración indefinida.

La Bolsa de Productos no compra ni vende productos agropecuarios; es el punto de encuentro, de reunión o información de oferentes y demandantes, que permite a escala regional, otras en el ámbito nacional y hasta internacional, facilitar las transacciones de productos agropecuarios.

3.1.2 Funciones y Objetivos

La Bolsa Nacional de Productos Agropecuarios tiene como objetivos:

1.Facilitar comercialización de productos de origen y destino agropecuarios

2.Aproximar la Oferta y la Demanda

3.Favorecer la libre concurrencia, la competencia y la transparencia de mercado

Para el cumplimiento de sus objetivos la Corporación tiene las siguientes funciones principales:

1.Constituir y ofrecer a los corredores de Bolsa el sitio oficial de reunión para tratar todas las clases de negocios autorizados por la misma; estas son: compra y venta de certificados de depósitos de productos agrícolas,

2.Cesión de contratos de comercialización de productos agropecuarios, agrícolas y agroindustriales.

3.Mantener el funcionamiento del mercado bursátil.

4.Establecer normas, procedimientos y montos de garantías adecuados para las transacciones.

5.Buscar los funcionarios que cumplan con las disposiciones legales para evitar manipulaciones y garantizar el buen funcionamiento del mercado

6.Resolver las controversias que surgieren sobre el cumplimiento de las obligaciones

7.Inscribir los títulos y contratos cotizables en la Bolsa y mantener en sus registros las operaciones que se realicen con determinación del producto, su calidad, cantidad, precios, contratantes y cualquier otra observación

8.Cancelar la inscripción de los títulos y contratos que no ofrezcan las respectivas garantías y seguridades

9.Informar al público permanentemente de las operaciones que se realicen, sus precios, y características.

3.2 Arquitectura General de la Aplicación

La aplicación esta basada en la Arquitectura Cliente/Servidor, que opera en una Red de Area Local y centralizada, con capacidad de crecer a un ambiente distribuido. El corazón del sistema es un Servidor Web que incluye varios subsistemas, con diversos niveles de acceso.

Cada subsistema fue diseñado como una interfaz gráfica de applets, que operan por medio de objetos CORBA (figura 6).

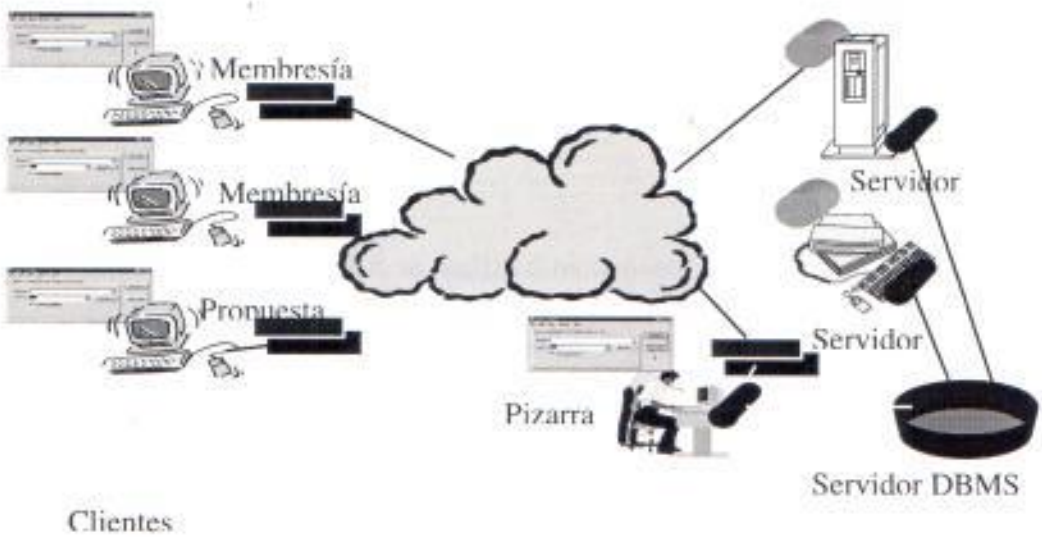


Figura 6. Esquema General de la Aplicación.

3.3 Herramientas Utilizadas

3.3.1 Visual Café

Symantec Visual Café 2.1 es la segunda generación de herramientas de programación que incluyen JavaBeans.

Esta versión incluye más de 100 JavaBeans, el compilador JIT 3.0, y un entorno de desarrollo visual, que facilitan el desarrollo de interfaces en applets java.

La facilidad de uso para aquellos programadores acostumbrados a ambientes de desarrollo gráfico, tipo Visual Basic, permite que la curva de aprendizaje se reduzca considerablemente.

La invocación a objetos CORBA se realiza directamente dentro del código de la clase; es decir, no existe programación visual para ello. Como parte de la iniciación, el applet instancia la clase CORBA, luego hace una llamada al método ORB necesario.

Posteriormente se utiliza un HTML para correr el applet, en él se especifica la clase, el archivo JAR, y los parámetros necesarios para encontrar el Naming Service.

3.3.2 Visibroker para Java

Es una herramienta para el desarrollo de aplicaciones Cliente/Servidor basadas en la arquitectura CORBA, que emplea Java como su principal lenguaje de programación (figura 7).

El Visibroker para Java soporta métodos de invocaciones CORBA tanto estáticos como dinámicos. Los métodos en el Servidor pueden ser invocados por una aplicación Cliente Java o por applets dentro de un browser.

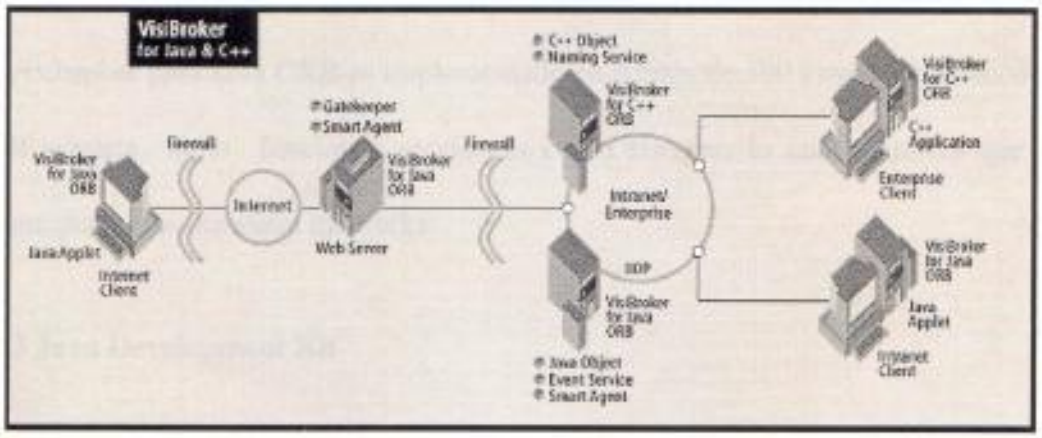


Figura 7. Esquema de ambiente de Visibroker para Java.

El Visibroker para Java incluye un completo repositorio de interfaces CORBA, escritos en Java. El compilador IDL Visibroker genera tanto el código skeleton para los objetos servidores en C++ o Java; como los stubs Java para el lado del Cliente.

Todo Visigenic ORBs implementa completamente el protocolo IIOP, lo cual hace más fácil para los objetos C++ invocar métodos en objetos Java y vice-versa.

El Visibroker para Java viene con un servicio de nombramiento de objeto llamado OSAgent. Múltiples OSAgents corriendo en la misma red se localizan unos a otros y automáticamente particionan el espacio de nombres entre ellos. Esto les permite replicar y cargar objetos entre diferentes máquinas servidoras. En el caso de una excepción al sistema, el ORB redirecciona las llamadas de los clientes a una replica.

El Visibroker para Java ORB es implementado en menos de 100 kbytes de bytecodes. El ORB soporta tanto funciones servidoras como clientes, lo cual significa que algún cliente puede implementar callbacks.

3.3.3 Java Development Kit

Java 1.1 representa la implementación base de JavaSoft, para hacer que el lenguaje Java trabaje más eficientemente y seguro en ambientes distribuidos.

Sus principales características que lo diferencian son: Mejoras al lenguaje, especialmente en la parte del AWT (Interface gráfica); los archivos JAR, que combinan archivos de los applets en uno solo comprimido; el API de seguridad y, el JDBC, que es el driver de conexión a bases de Datos.

3.3.4 Object Domain

Es una herramienta de asistencia en el análisis y diseño orientado a objetos; cuyas principales características son las de diagramar los distintos modelos que se pueden utilizar en el proceso.

Soporta todos los diagramas estáticos y dinámicos; además, provee de herramientas para generación de código y documentación.

3.3.5 FrontPage

Es la más reciente herramienta de Microsoft para Internet, que esta compuesto de dos elementos: Explorador y Editor.

El Explorador permite diseñar el sitio WEB que alberga el proyecto de aplicación. Esto es, crear páginas, visualizar hipervínculos, distribuir los archivos componentes en una estructura jerarquizada de carpetas y navegar a través de las páginas que constituyen el Sitio.

El Editor permite diseñar una página WEB. Esto es, establecer hipervínculos, añadir applets java, animaciones y cualquier tipo de imágenes.

Capítulo IV

SIMULACION DEL PROCESO DE COMERCIALIZACION

4.1 Antecedentes del Análisis y Diseño Orientados a objetos

Es una abstracción y descripción precisa de lo que hace el sistema, que permite crear un modelo que define los detalles de implementación, en armonía con las decisiones de alto nivel de la arquitectura general de la aplicación.

Fue necesaria una descripción precisa para entender como usar la metodología en el desarrollo posterior del sistema.

4.2 Modelo de Requerimientos

Entendiéndolo como la descripción del comportamiento externo esperado de un sistema, cuando este sea puesto en operación; se llegó a que debería:

"Efectuar una simulación del Proceso de comercialización de los productos agropecuarios, que permita optimizar tiempo y recursos".

4.2.1 Requerimientos Funcionales

Siendo el comportamiento observable de lo que hará el sistema, se definió un conjunto de subsistemas con sus respectivos casos de uso, para documentar los requerimientos operacionales del sistema.

Los casos de uso son secuencias de transacciones en el sistema cuya tarea es producir un resultado con valor mensurable para un actor del sistema.

4.2.1.1 Lista de Casos de Uso

Secuencia de transacciones en el sistema cuya tarea es producir un resultado con valor mensurable para un actor del sistema.

Sistema 1. Membresía de Corredor

Casos de Uso:

1.1 Inscripción de Corredor

1.2 Modificación de Membresía

Sistema 2. Membresía de Producto

Casos de Uso:

2.1 Inscripción de Producto

2.2 Mantenimiento de Producto

Sistema 3. Negociación de Producto

Casos de Uso:

3.1 Operación de Negociación

4.2.1.2 Descripción de Casos de Uso

Sistema 1. Membresía de Corredor

Nombre: Inscripción de Corredor

Descripción: Persona a la cual es aprobada ser funcionario tipo corredor para operar en la Bolsa Nacional de Productos.

Nota:

- Siempre y cuando no hay sido eliminado en el arbitraje
- Membresía es vitalicia o hasta que decida vendérsela a otra persona

Nombre: Modificación de Membresía

Descripción: Cuando un corredor desea ceder o vender su membresía a otro

Nota:

- Equivale a cambio de corredor
- El pago era entregado al corredor anterior

Sistema 2. Membresía de Producto

Nombre: Inscripción de Producto

Descripción: Cuando un producto queda habilitado para poder ser negociado

Nota: Debe pasar por estudios de factibilidad y normas de calidad INEN

Nombre: Mantenimiento de Producto

Descripción: Reflejar cualquier cambio en las características del producto ingresado

Nota:

- Tipos de productos por categoría
- Se contempla modificación y eliminación

Sistema 3. Negociación de Producto

Nombre: Operación de Negociación

Descripción: Cuando existen corredores que realizan la negociación

Nota:

- Debe ser aprobada por un funcionario
- Debe ser publicada igual que una operación directa

4.2.1.3 Descripción de Actores

Sistema 1. Membresía de corredor

1.1 Inscripción de corredor

Nombre: Persona natural

Descripción: Aspirante a ser corredor

Nota: Puede ser solo de compra, solo de venta o de ambos

Nombre: Funcionario o comité

Descripción: Persona que autoriza la licencia

Nota: Grupo de Personas habilitadas por BNP

1.2 Modificación de Membresía

Nombre: Corredor que cede

Descripción: Persona que quiere salir de la Bolsa

Nota: Corredor registrado

Nombre: Corredor que recibe

Descripción: Persona que quiere ingresar

Nota: Persona Natural no registrado

Nombre: Funcionario o comité

Descripción: Persona que autoriza el cambio de membresia.

Nota: Grupo de personas habilitadas por la BNP

Sistema 2. Membresía de producto

2.1 Inscripción de productos

Nombre: Corredor

Descripción: Funcionario que solicita

Nota: Persona habilitada por la BNP para negociar

Nombre: Funcionario o comité

Descripción: persona que califica al producto.

Nota: Grupo de personas habilitadas por la BNP

2.2 Mantenimiento de productos

Nombre: Corredor

Descripción: Funcionario que solicita

Nota: Persona habilitada por la BNP para negociar

Nombre: Funcionario o comité

Descripción: Persona que califica al producto.

Nota: Grupo de personas habilitadas por la BNP

Sistema 3. Negociación de producto

3.1 Operación de negociación

Nombre: Corredores

Descripción: Funcionario con poder para ejercer negociación

Nota: Persona habilitada por la BNP para negociar

Nombre: Funcionario de la Bolsa

Descripción: Funcionario que otorga poder sobre la propuesta

Nota: Persona habilitada por la BNP para finiquitar una negociación

4.2.1.4 Diagrama de Casos de Uso

Este diagrama complementa la lista de casos de uso especificando gráficamente la relación de los mismos con los actores.

(figura 8)

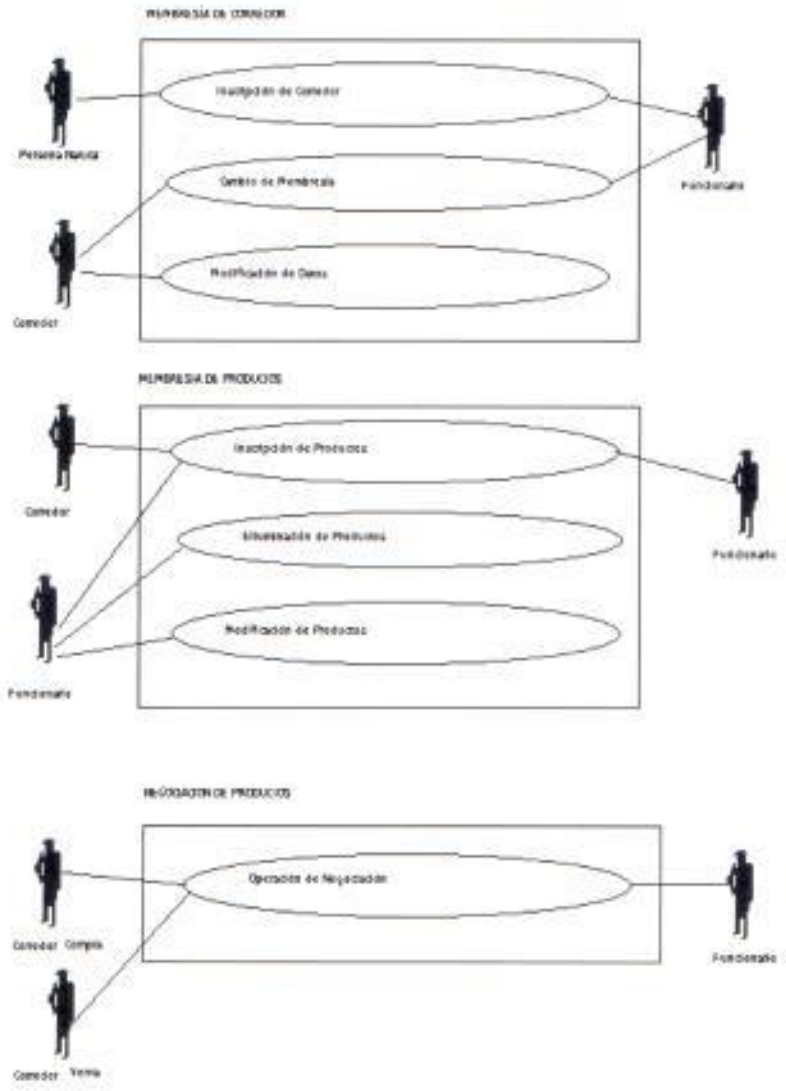


Figura 8. Diagrama de Casos de Uso.

4.2.1.5 Escenarios

MEMBRESÍA DE CORREDOR

Inscripción de corredor

Procedimiento: Consiste en registrar a una persona natural como corredor autorizado de la bolsa de producto. Para esto se da una membresía al corredor.

Asumpciones:

- La solicitud se la envía por correo
- La respuesta a la solicitud se la da a las 24 horas de haber recibido.
- Tarjeta del corredor

Salidas: Asignación de Membresía.

Cambio de Membresía

Procedimiento: Consiste en quitar la membresía a un corredor y asignarla a otro.

Asumpciones:

- La solicitud la envía por correo el corredor que cede la membresía, además especifica a quien se la va a ceder.
- La respuesta a la solicitud se la da a las 24 horas de haberla recibido.

Salidas: Quitar y Asignar Membresía.

Cambio de datos del corredor

Procedimiento: Consiste en modificar los datos personales del corredor.

Asumpciones:

- La solicitud la envía por correo el corredor que desea cambiar los datos o por decisión de un funcionario.

Salidas: Quitar y Asignar Membresía.

MEMBRESÍA DE PRODUCTOS

Inscripción de productos

Procedimiento: Consiste en incluir un producto dentro de la Bolsa para negociación.

Asumpciones:

- La solicitud la envía por correo un corredor o un funcionario.
- La respuesta a la solicitud se la da a las 52 horas de haber recibido.

Salidas: Inclusión o negación del Producto

Modificación de productos

Procedimiento: Consiste en modificar los datos del producto.

Asumpciones:

- La solicitud la envía por correo el corredor o funcionario que requiere la modificación.
- La respuesta a la solicitud se la da a las 52 horas de haberla recibido.

Salidas: Modificar datos.

Eliminación de productos

Procedimiento: Consiste en excluir un producto para negociación.

Asumpciones:

- La solicitud la envía por correo el corredor o funcionario que requiere la modificación.
- La respuesta a la solicitud se la da a las 52 horas de haberla recibido.

Salidas: Descalificar el Producto.

NEGOCIACION DE PRODUCTOS

Operación directa con modificación y fallida

Procedimiento: Un corredor de compra/venta ingresa una propuesta y después la modifica, otro corredor venta/compra ingresa su propuesta, entonces se realiza una negociación y después uno de los dos pide una revisión de la negociación a un comité y espera un fallo de éste

Asumpciones:

- Propuesta es modificada
- Ambas propuestas son de productos válidos
- Se realiza una negociación

Salidas:

El comité emite un fallo, el cual es publicado y al infractor se le pone una multa

Operación directa con modificaciones y exitosa

Procedimiento: Un corredor de compra/venta ingresa una propuesta y después la modifica, otro corredor venta/compra ingresa su propuesta, entonces se realiza una negociación.

Asumpciones:

- Propuesta es modificada
- Ambas propuestas son de productos válidos
- Se realiza una negociación

Salidas:

Se realiza una negociación exitosa

Operación directa sin modificación y fallida

Procedimiento:

Un corredor de compra/venta ingresa una propuesta, otro corredor venta/compra ingresa su propuesta, entonces se realiza una negociación, y después uno de los dos pide una revisión de la negociación a un comité y espera un fallo de éste.

Asumpciones:

- Ambas propuestas son de productos válidos
- Se realiza una negociación

Salidas:

El comité emite un fallo, el cual es publicado y al infractor se le pone una multa

Operación directa sin modificación y exitosa

Procedimiento:

Un corredor de compra/venta ingresa una propuesta, otro corredor venta/compra ingresa su propuesta, entonces se realiza una negociación.

Asumpciones:

- Ambas propuestas son de productos válidos
- Se realiza una negociación

Salidas:

Se realiza una negociación exitosa

Operación cruzada con modificación y fallida

Procedimiento:

Un corredor realiza la propuesta de compra y venta a la vez, luego la modifica, se realiza una negociación, después se pide a un comité la revisión de esta negociación por incumplimiento de la misma

Asumpciones:

- Corredor hace ambas propuestas, la de compra y venta.
- Ambas propuesta contiene productos válidos
- Se realiza la negociación

Salidas:

El comité emite un fallo, el cual es publicado y al infractor se le pone una multa

Operación cruzada con modificaciones y exitosa

Procedimiento:

Un corredor realiza la propuesta de compra y venta a la vez, luego la modifica, se realiza una negociación.

Asumpciones:

- Corredor hace ambas propuestas, la de compra y venta.
- Ambas propuesta contiene productos válidos
- Se realiza la negociación

Salidas:

Se realiza negociación exitosa

Operación cruzada sin modificación y fallida

Procedimiento:

Un corredor realiza la propuesta de compra y venta a la vez, se realiza una negociación después se pide a un comité la revisión de esta negociación por incumplimiento de la misma

Asumpciones:

- Corredor hace ambas propuesta, la de compra y venta.
- Ambas propuestas contienen productos válidos
- Se realiza la negociación

Salidas:

El comité emite un fallo, el cual es publicado y al infractor se le pone una multa

Operación cruzada sin modificación y exitosa

Procedimiento:

Un corredor realiza la propuesta de compra y venta a la vez, se realiza una negociación.

Asumpciones:

- Corredor hace ambas propuesta, la de compra y venta.
- Ambas propuestas contienen productos válidos
- Se realiza la negociación

Salidas:

Se realiza negociación exitosa

Operación cancelada

Procedimiento:

Corredor ingresa una propuesta y después es cancelada

Asumpciones:

- Corredor hace una propuesta válida
- No se realiza ningún tipo de negociación

Salidas:

La propuesta es cancelada y no puede entrar en negociación

4.2.1.6 Plan de Pruebas

Nombre del Grupo de Prueba: Inscripción de corredor.

Número de Prueba: 1.1

Pre requisitos: Carlos desea ser corredor de la BNPA, emite la solicitud para ello, con todos sus datos requeridos para ser revisados por los funcionarios de la BNPA.

Instrucciones de configuración: La BNPA tiene varias solicitudes a corredores incluyendo la de Carlos, el Funcionario revisa los datos de Carlos.

Carlos cumple con todos los requisitos.

Carlos paga el valor por la membresía.

Instrucciones de Prueba: El Funcionario recibe la solicitud de membresía de corredor.

El Funcionario acepta la solicitud por parte de Carlos.

El Funcionario emite el Número de membresía a Carlos.

Comportamiento aceptable:

El sistema ingresa a la base de datos los del nuevo corredor.

El sistema imprime un comprobante de aceptación.

Nombre del Grupo de Prueba: Cambio de Membresía

Número de Prueba: 1.2

Pre-requisitos: Carlos es corredor de la BNPA, su Número de membresía es 981285, y desea vender su membresía.

Instrucciones de configuración: La BNPA tiene 200 corredores incluyendo a Carlos con Número de membresía 981285 (98 indica el año de afiliación), el sistema tiene toda la información perteneciente a Carlos.

Carlos no tiene negociaciones pendientes.

Carlos no tiene deudas de inscripción.

Instrucciones de Prueba: El corredor ingresa sus datos y los datos del nuevo corredor

El Funcionario recibe la solicitud de cambio.

El Funcionario acepta al nuevo corredor, y el cambio de membresía por parte del corredor anterior.

Comportamiento aceptable:

El sistema desactiva como corredor al anterior.

El sistema actualiza la base de datos con los datos del nuevo corredor.

El sistema imprime un comprobante de cambio.

Nombre del Grupo de Prueba:Modificación de datos de un corredor

Número de Prueba: 1.3

Pre-requisitos: Carlos es corredor de la BNPA, su Número de membresía es 981285, y desea modificar su cuenta de e-mail de cc@satnet.net a cc@ecuanet.net y además quiere cambiar el Número de tarjeta de crédito de VISA # 123456 a Diners # 7891074.

Instrucciones de configuración: La BNPA tiene 200 corredores incluyendo a Carlos con Número de membresía 981285 (98 indica el año de afiliación), el sistema tiene toda la información perteneciente a Carlos.

Carlos no tiene negociaciones pendientes.

Carlos no tiene deudas de inscripción.

Instrucciones de Prueba: El corredor ingresa los datos que desea cambiar (como dirección e-mail, etc.).

El Funcionario recibe la solicitud de modificación.

El Funcionario acepta los cambios realizados (solicitados) por el corredor.

Comportamiento

Aceptable El sistema modifica los datos del corredor.

El sistema imprime un comprobante de modificación.

El sistema actualiza la información de la base de datos.

Nombre del Grupo de Prueba: Inscripción de productos

Número de Prueba: 2.1

Pre-requisitos: Un corredor de la BNPA, con Número de membresía es desea ingresar un producto para la negociación.

Instrucciones de configuración: La BNPA tiene en negociaciones muchos productos, y solicitudes hechas para ello.

El producto es conocido y bueno.

El producto cumple con las especificaciones.

Instrucciones de Prueba: El corredor ingresa los datos del producto que desea ingresar para la negociación.

El Funcionario recibe la solicitud Inclusión del producto

El Funcionario acepta o rechaza el producto.

Comportamiento aceptable: El sistema ingresa los datos del producto.

El sistema imprime un comprobante de inclusión.

El sistema actualiza la información de la base de datos de los productos.

Nombre del Grupo de Prueba: **Modificación de datos de producto**

Número de Prueba: 2.2

Pre-requisitos: El arroz es un producto negociable, en este año el precio definido por la BNPA, no es suficiente ya que los gastos fueron excesivos. Se desea cambiar algunos datos de este producto como por ejemplo para que sea aceptado con precios mayores.

Instrucciones de configuración: La BNPA tiene muchos productos negociables incluyéndolo al arroz, con sus precios mínimo y máximo, el sistema tiene toda la información perteneciente al arroz.

El arroz cumple con las normas INEN establecidas.

Instrucciones de Prueba: El corredor ingresa los datos que desea cambiar (precios, calidad, etc.).

El Funcionario recibe la solicitud de modificación.

El Funcionario acepta los cambios realizados (solicitados) por el corredor interesado.

Comportamiento aceptable: El sistema modifica los datos del producto.

El sistema imprime un comprobante de modificación.

El sistema actualiza la información de la base de datos.

Nombre del Grupo de Prueba: Eliminación de productos.

Número de Prueba: 2.3

Pre-requisitos: La cebolla colorada es un producto negociable en la BNPA, tiene características para que sea aceptable; pero la cosecha de este año no cumple con las especificación y se decide no negociarla.

Instrucciones de configuración: La BNPA negocia la cebolla colorada. El sistema tiene toda la información acerca de la cebolla colorada.

La cebolla colorada no cumple con los requisitos.

Instrucciones de Prueba: El corredor ingresa los datos del producto a eliminar

El Funcionario recibe la solicitud de modificación.

El Funcionario acepta los cambios realizados (solicitados) por el corredor.

Comportamiento aceptable: El sistema desactiva como producto negociable

El sistema actualiza la información de la base de datos.

Nombre del Grupo de Prueba: **Operación directa fallida**

Número de Prueba: 3.1.

Pre-requisitos: Carlos y Julio como corredores cierran una negociación y firman un contrato.

Instrucciones de configuración: La BNPA tiene varios contratos cada cual con un número diferente, el sistema tiene toda la información perteneciente a los contratos y a los corredores.

Carlos firmó el contrato de la negociación.

Julio firmó el contrato de la negociación.

Instrucciones de Prueba: Carlos emite una carta de incumplimiento de contrato por parte de Julio.

El Funcionario recibe la carta.

El Funcionario delega al comité de arbitraje.

Comportamiento aceptable:

El comité estudia el caso.

El comité emite el fallo.

El fallo es publicado para conocimiento de los interesados, para ver quien paga la multa.

El sistema guarda la información pertinente en la base de datos.

Nombre del Grupo de Prueba: Operación directa exitosa

Número de Prueba: 3.2.

Pre-requisitos: Carlos y Julio como corredores cierran una negociación y firman un contrato.

Instrucciones de configuración:

La BNPA tiene varios contratos cada cual con un número diferente, el sistema tiene toda la información perteneciente a los contratos y a los corredores.

Carlos firmó el contrato de la negociación.

Julio firmó el contrato de la negociación.

Instrucciones de Prueba: Ambas partes estan de acuerdo con el cumplimiento del contrato.

Comportamiento aceptable: El sistema guarda la información del contrato.

No hay problemas se da la operación exitosa.

Nombre del Grupo de Prueba: Operación cruzada

Número de Prueba: 3.1.

Pre-requisitos: Carlos es un corredor, dos clientes le encomiendan el uno vender y el otro comprar el mismo producto. Presenta su propuesta. Firma el contrato correspondiente.

Instrucciones de configuración: La BNPA tiene varios contratos cada cual con un número diferente, el sistema tiene toda la información perteneciente a los contratos y a los corredores.

Carlos firmó el contrato de la negociación como vendedor.

Carlos firmó el contrato de la negociación como comprador.

Instrucciones de Prueba: No hay ningún reclamo, ya que es el mismo como comprador y vendedor.

Comportamiento aceptable: La negociación se da exitosamente.

4.2.2 Requerimientos No Funcionales

Estos son las limitaciones en el sistema y/o criterios aceptables como rendimiento, confiabilidad, utilidad, costo, utilidad.

Los criterios establecidos abarcan los siguientes puntos:

- No es una aplicación tradicional.
- Considerar un proceso de negociación rígido (no hay regateo).
- Requerimientos de hardware de desarrollo muy sofisticados
- Especificaciones de software muy rígidas.

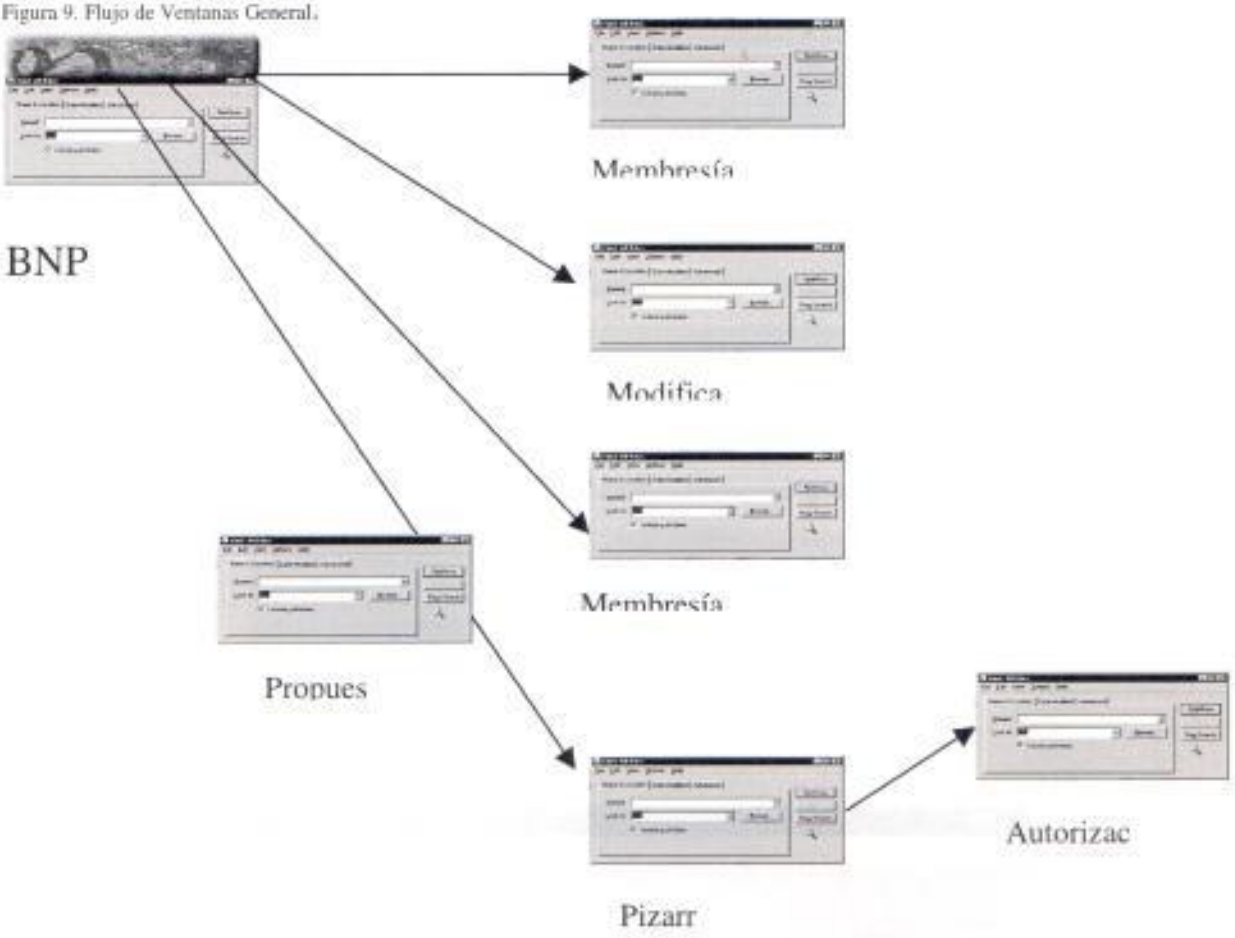
4.3 Modelo de Análisis y Diseño

Entendiendo que el sistema es una entidad encapsulada, cuyo comportamiento y requerimientos operacionales pueden ser descritos mediante los casos de uso.

4.3.1 Modelo Dinámico

4.3.1.1 Flujo de Ventanas y Layouts

Figura 9. Flujo de Ventanas General.



4.3.1.1.1 Layout de Membresía

Figura 10: Layout de Membresía

4.3.1.1.2 Layout de Autorización

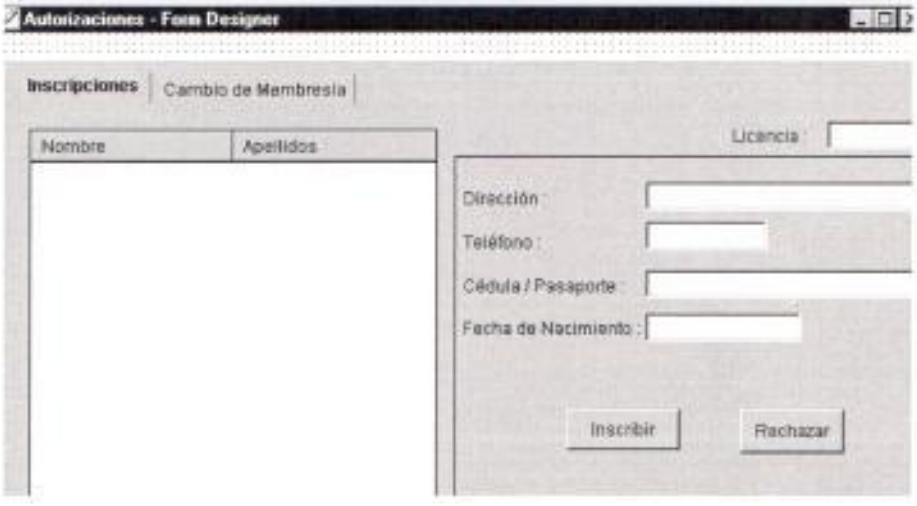


Figura 11: Layout de Autorización

4.3.1.1.3 Layout de Pizarra

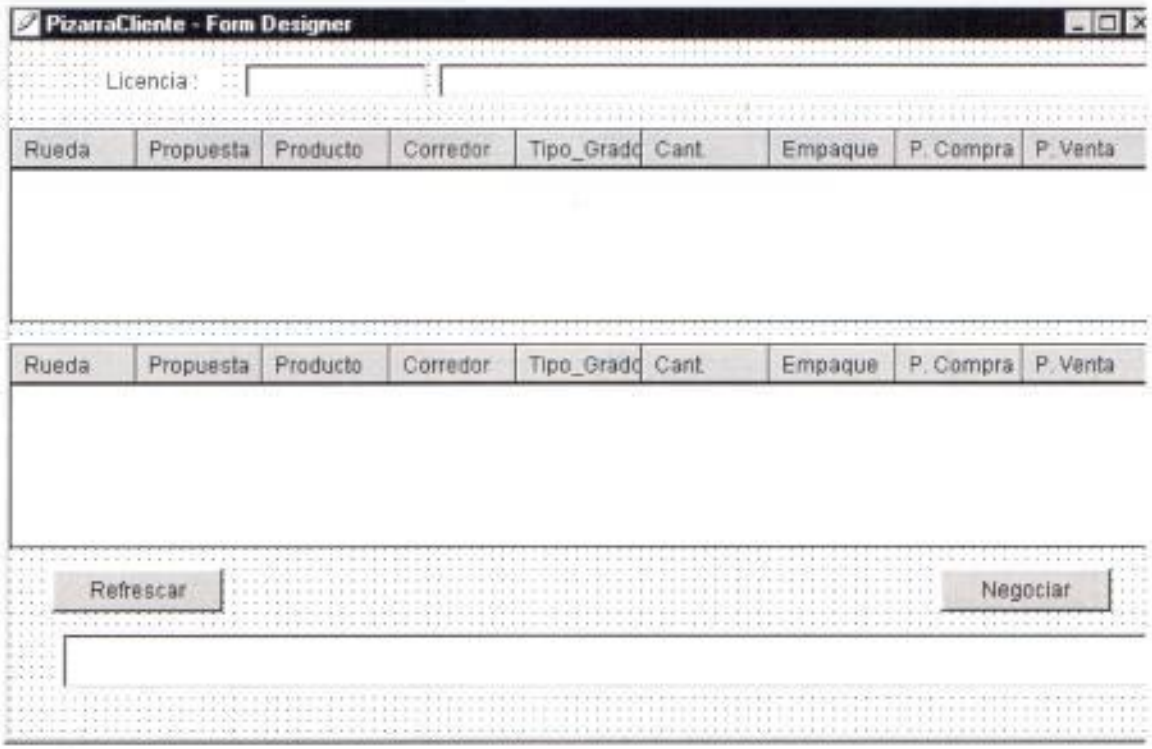


Figura 12: Layout de Pizarra

4.3.1.2 Diagramas de Análisis de Interacción de Objetos

4.3.1.2.1 Inscripción de Corredor Exitosa

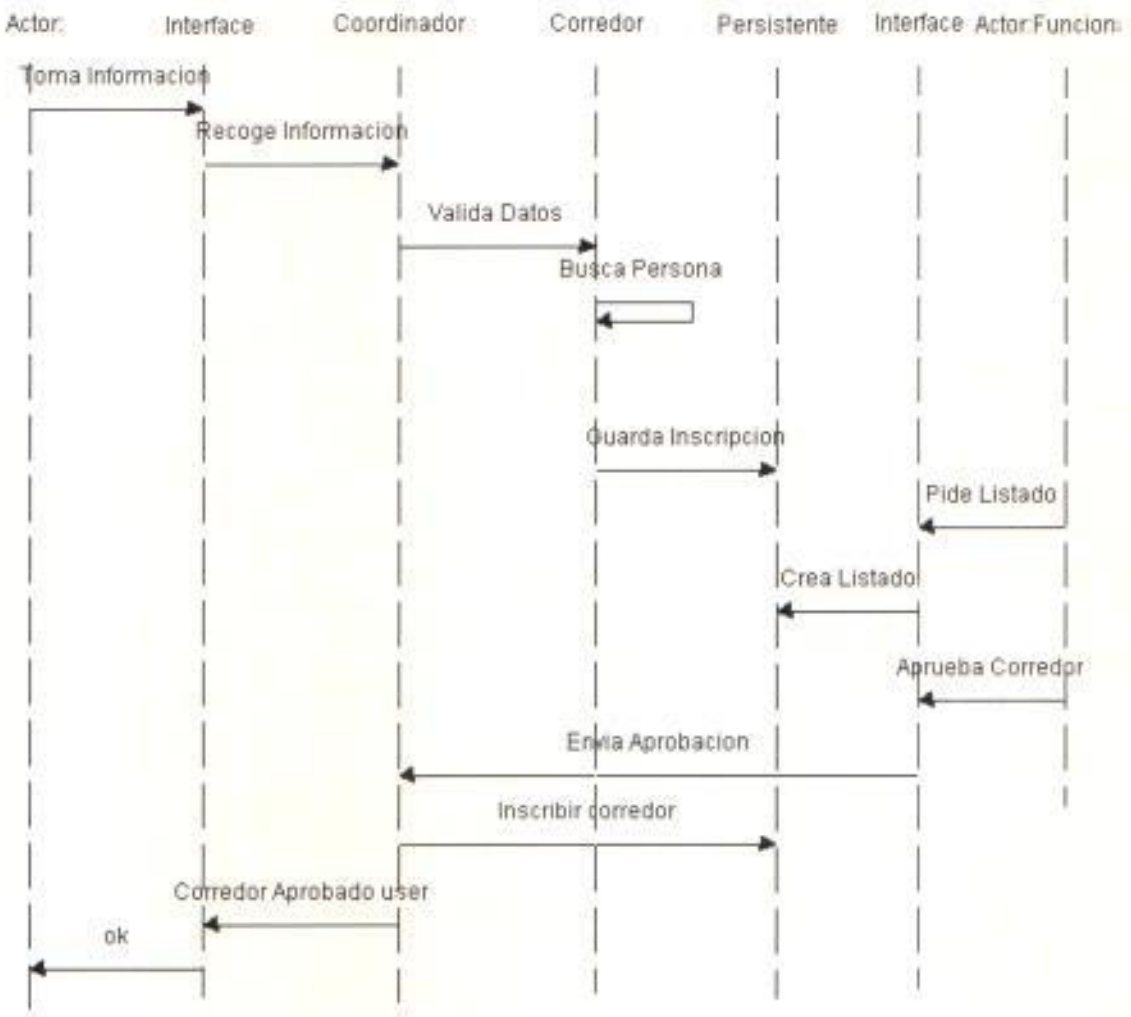


Figura 13: Diagrama de Iteracción de Objetos de Inscripción de Corredor Exitosa

4.3.1.2.2 Cambio de Membresía Exitosa

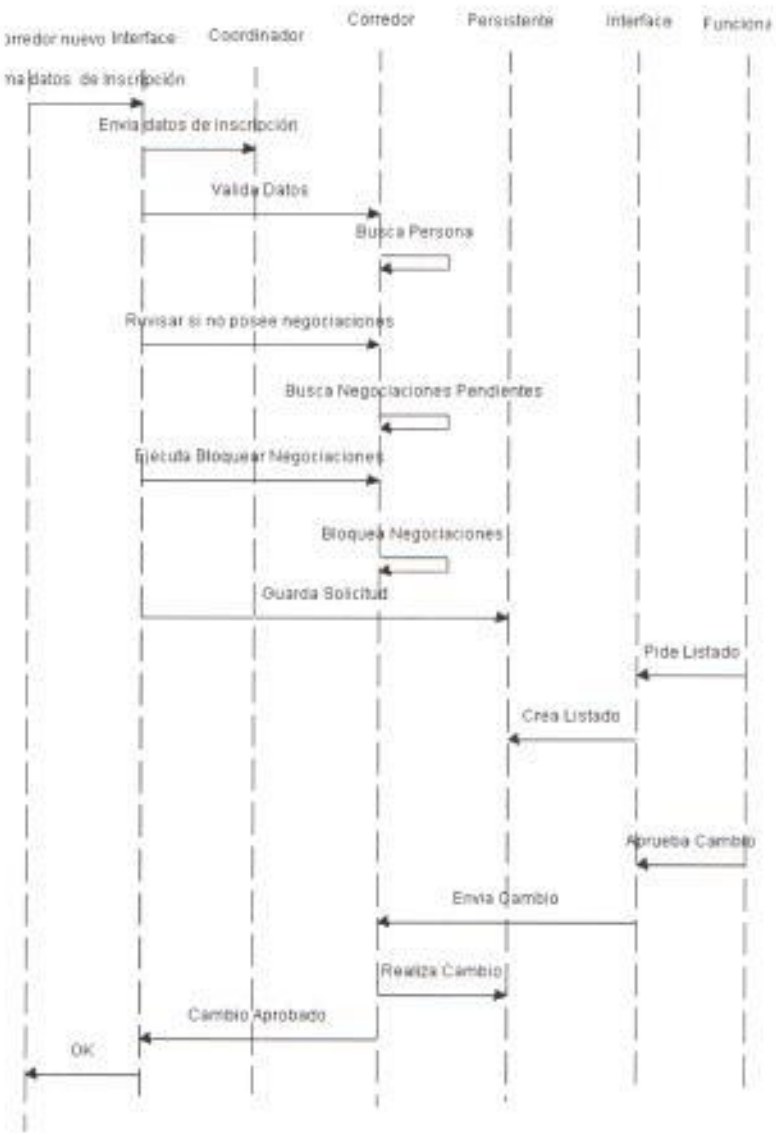


Figura 14: Diagrama de Iteracción de Objetos de Cambio de Membresía Exitosa

4.3.1.2.3 Modificación de Datos de Corredor Exitosa

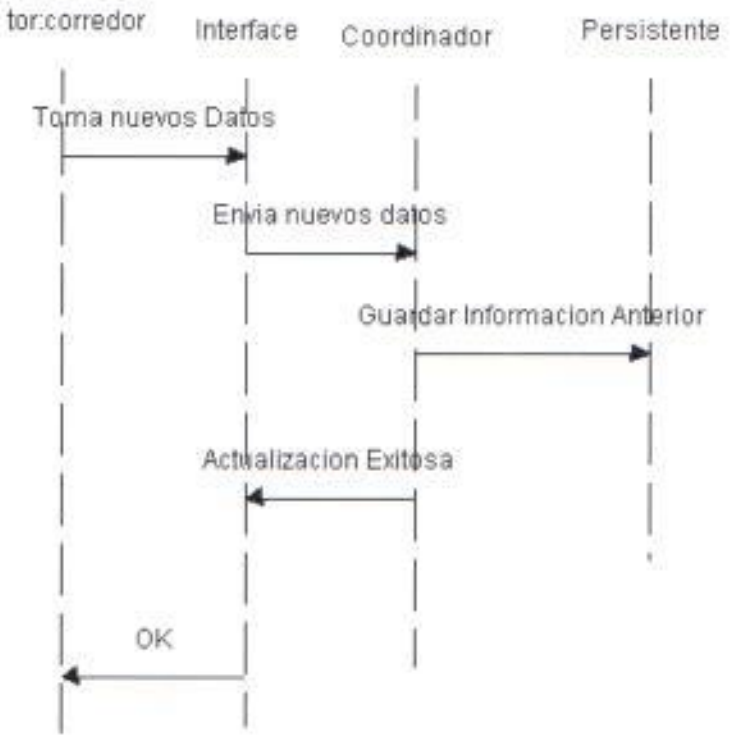


Figura 15: Diagrama de Iteracción de Objetos de Modificación de Datos de Corredor Exitosa

4.3.1.2.4 Inscripción de Productos por Corredor Exitosa

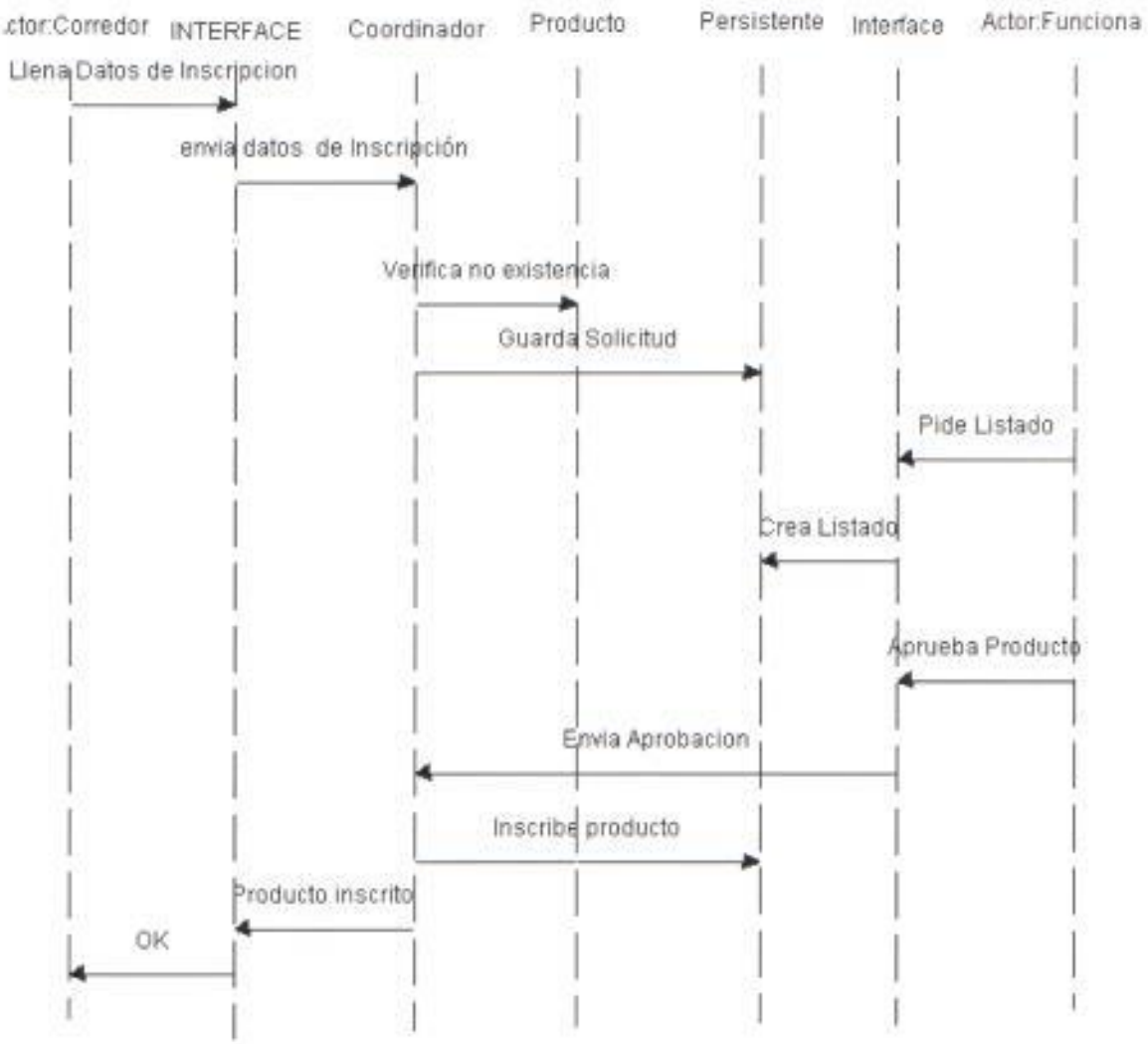


Figura 16: Diagrama de Iteracción de Objetos de Inscripción de Productos por Corredor Exitosa

4.3.1.2.5 Eliminación de Productos Exitosa

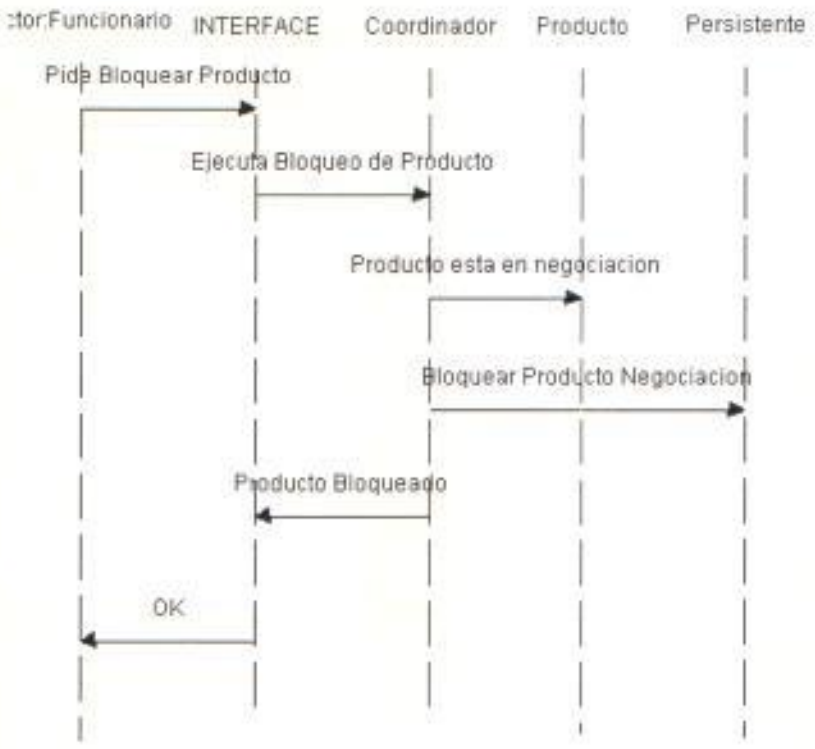


Figura 17: Diagrama de Iteracción de Objetos de Eliminación de Productos Exitosa

4.3.1.2.6 Modificación de Productos Exitosa

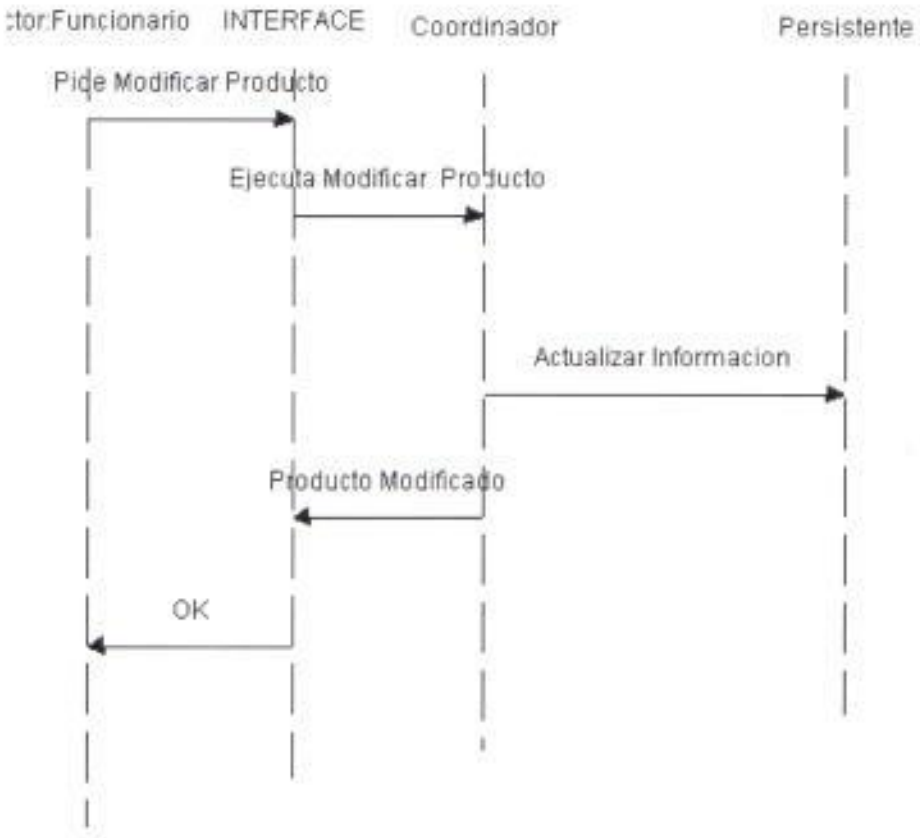


Figura 18: Diagrama de Iteracción de Objetos de Modificación de Productos Exitosa

4.3.1.2.7 Operación directa con modificación y fallida

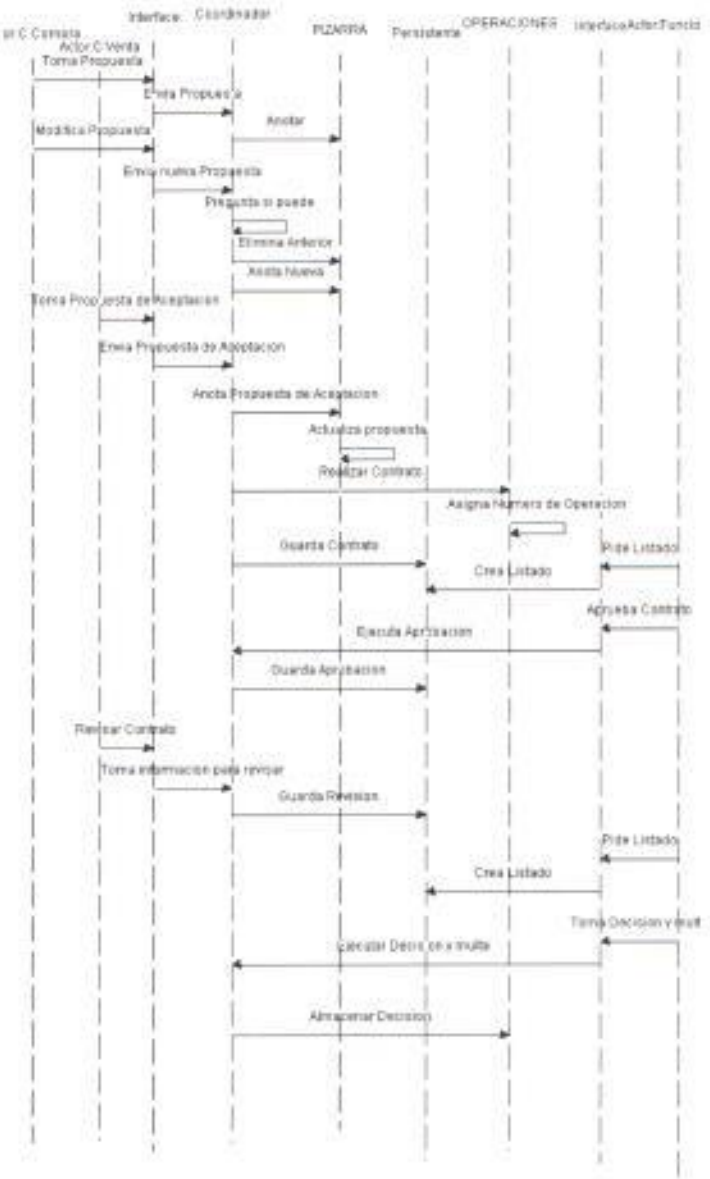


Figura 19: Diagrama de Iteracción de Objetos de Operación directa con modificación y fallida

4.3.1.2.8 Operación directa con modificación y exitosa

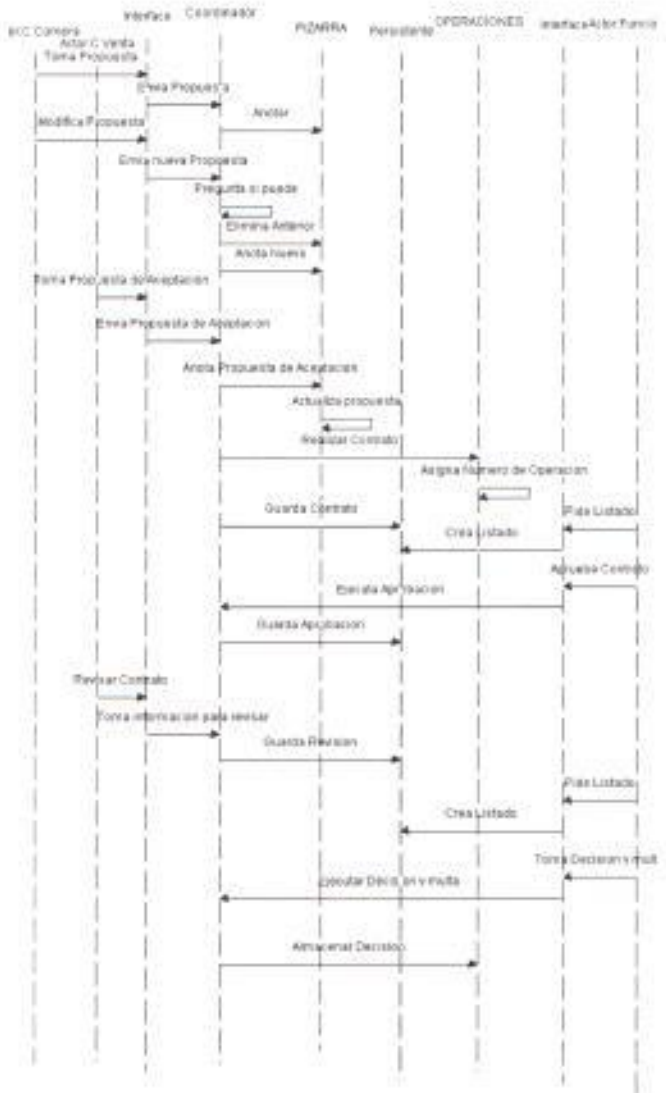


Figura 20: Diagrama de Iteracción de Objetos de Operación directa con modificación y exitosa

4.3.1.2.9 Operación directa sin modificación y fallida

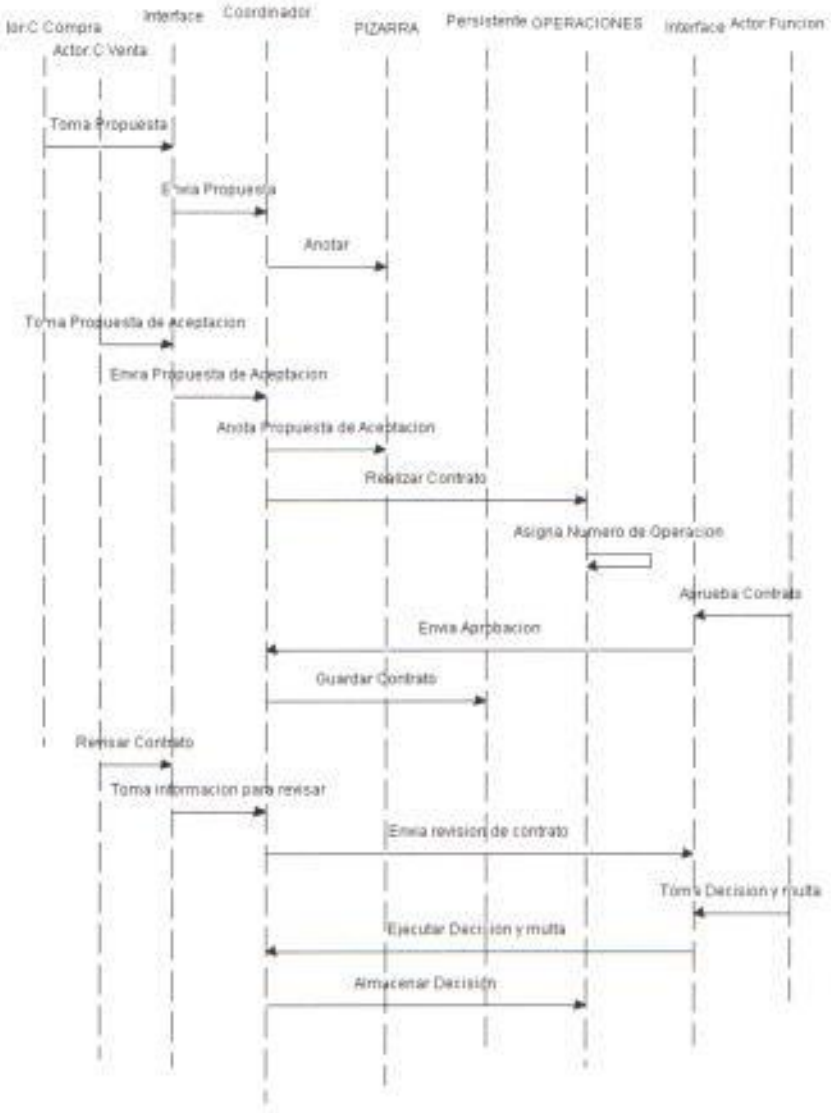


Figura 21: Diagrama de Iteracción de Objetos de Operación directa sin modificación y fallida

4.3.1.2.10 Operación directa sin modificación y exitosa

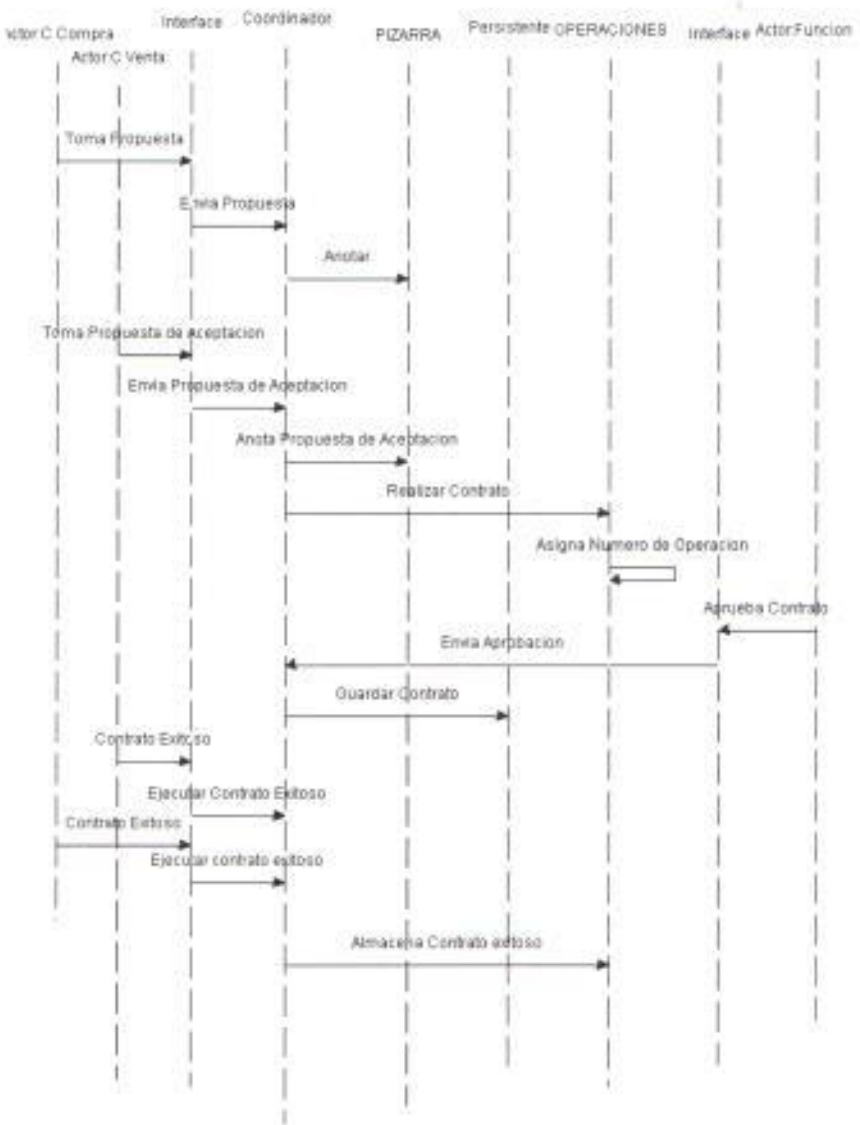


Figura 22: Diagrama de Iteracción de Objetos de Operación directa sin modificación y exitosa

4.3.1.2.11 Operación cruzada con modificación y fallida

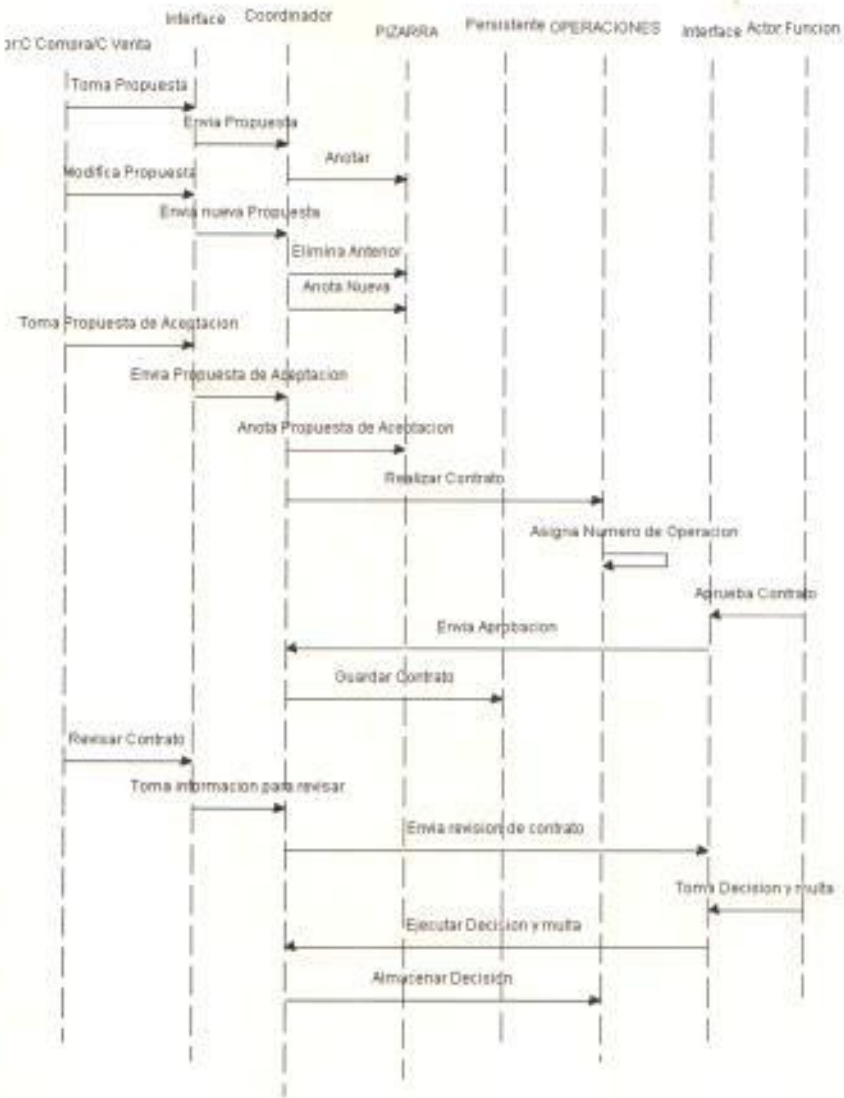


Figura 23: Diagrama de Iteracción de Objetos de Operación cruzada con modificación y fallidâ

4.3.1.2.12 Operación cruzada con modificación y exitosa

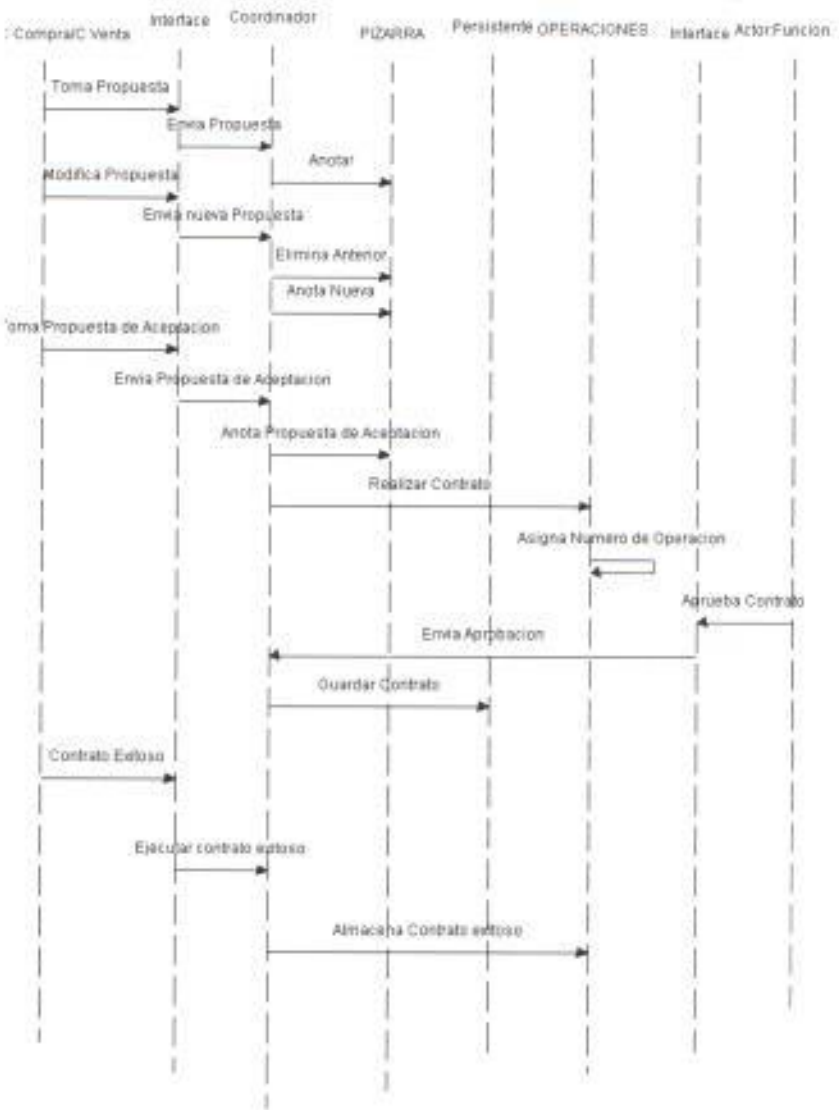


Figura 24: Diagrama de Iteracción de Objetos de Operación cruzada con modificación y exitosa

4.3.1.2.13 Operación cruzada sin modificación y fallida

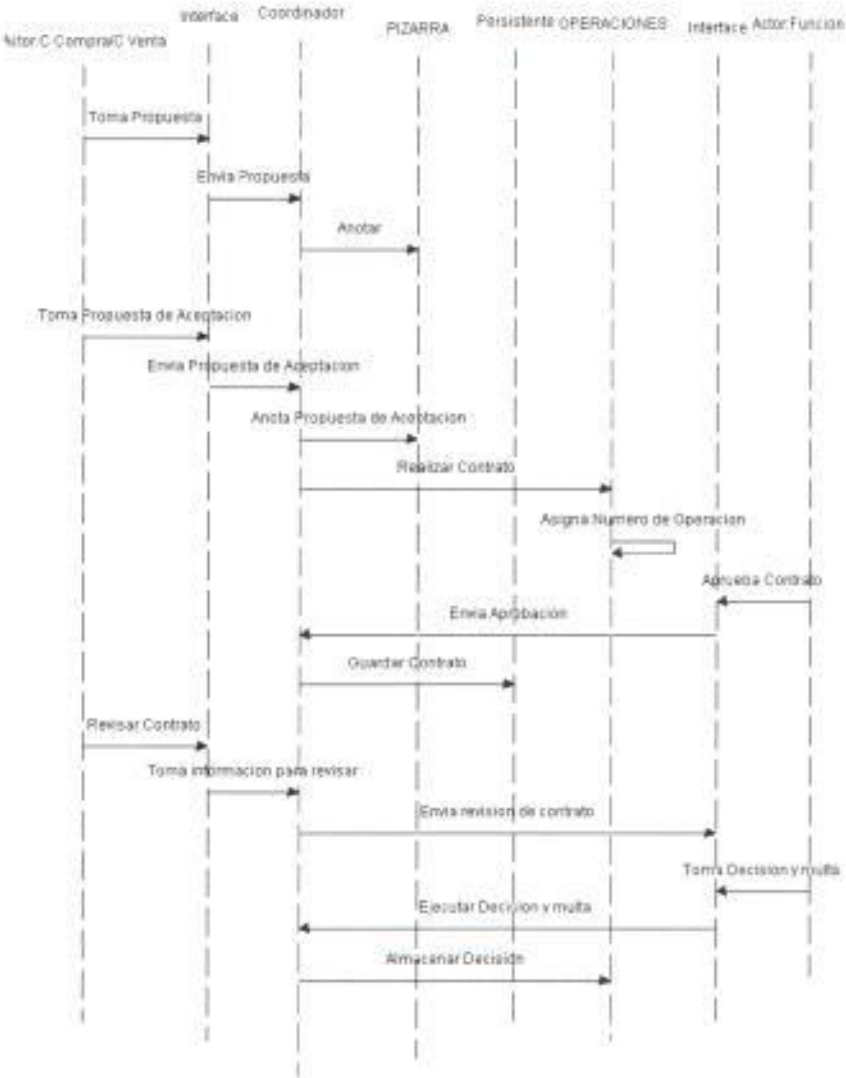


Figura 25: Diagrama de Iteracción de Objetos de Operación cruzada sin modificación y fallida

4.3.1.2.14 Operación cruzada sin modificación y exitosa

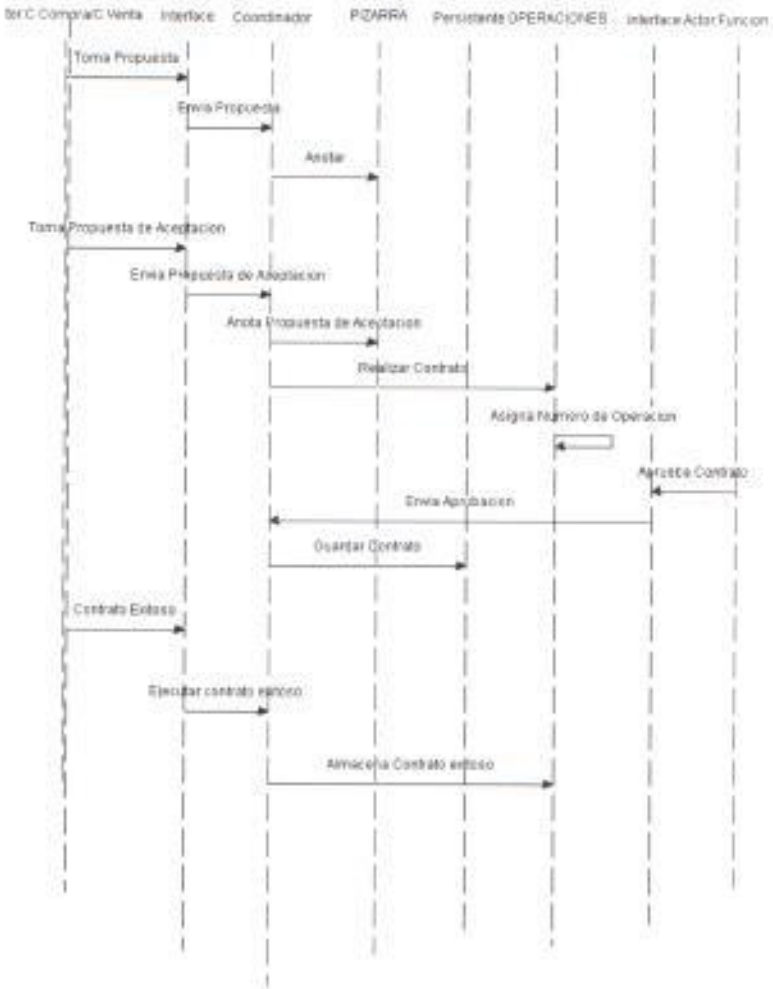


Figura 26: Diagrama de Iteracción de Objetos de Operación cruzada sin modificación y exitosa

4.3.1.2.15 Operación cancelada

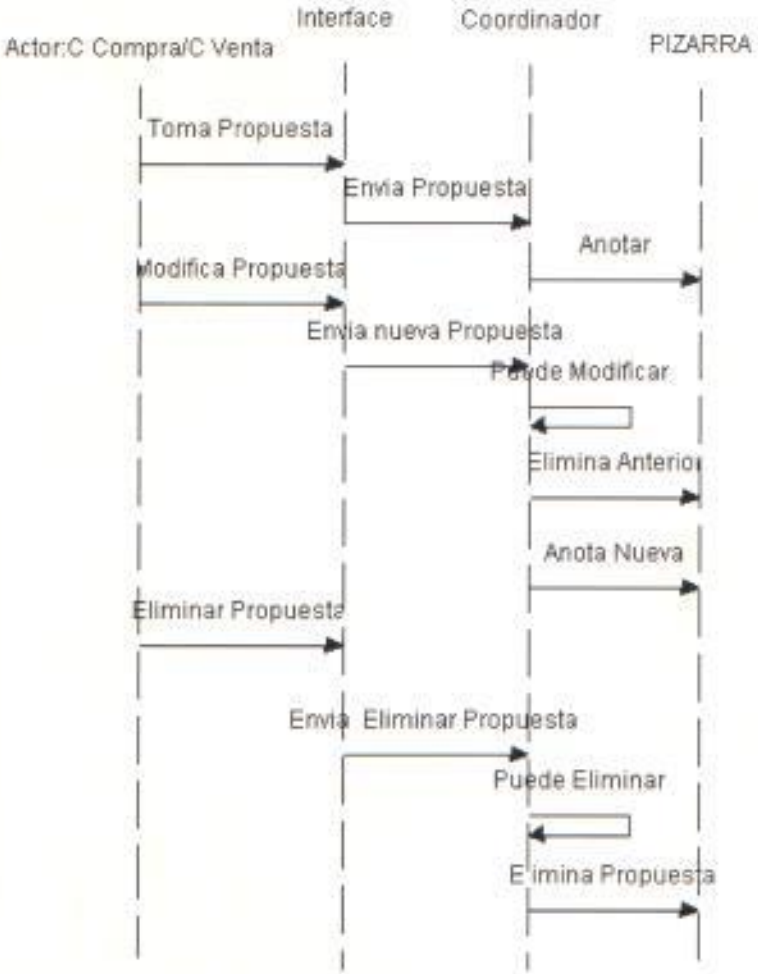


Figura 27: Diagrama de Iteracción de Objetos de Operación cancelada

4.3.1.3 Diagramas de Diseño de Interacción de Objetos

4.3.1.3.1 Inscripción de Corredor Exitosa

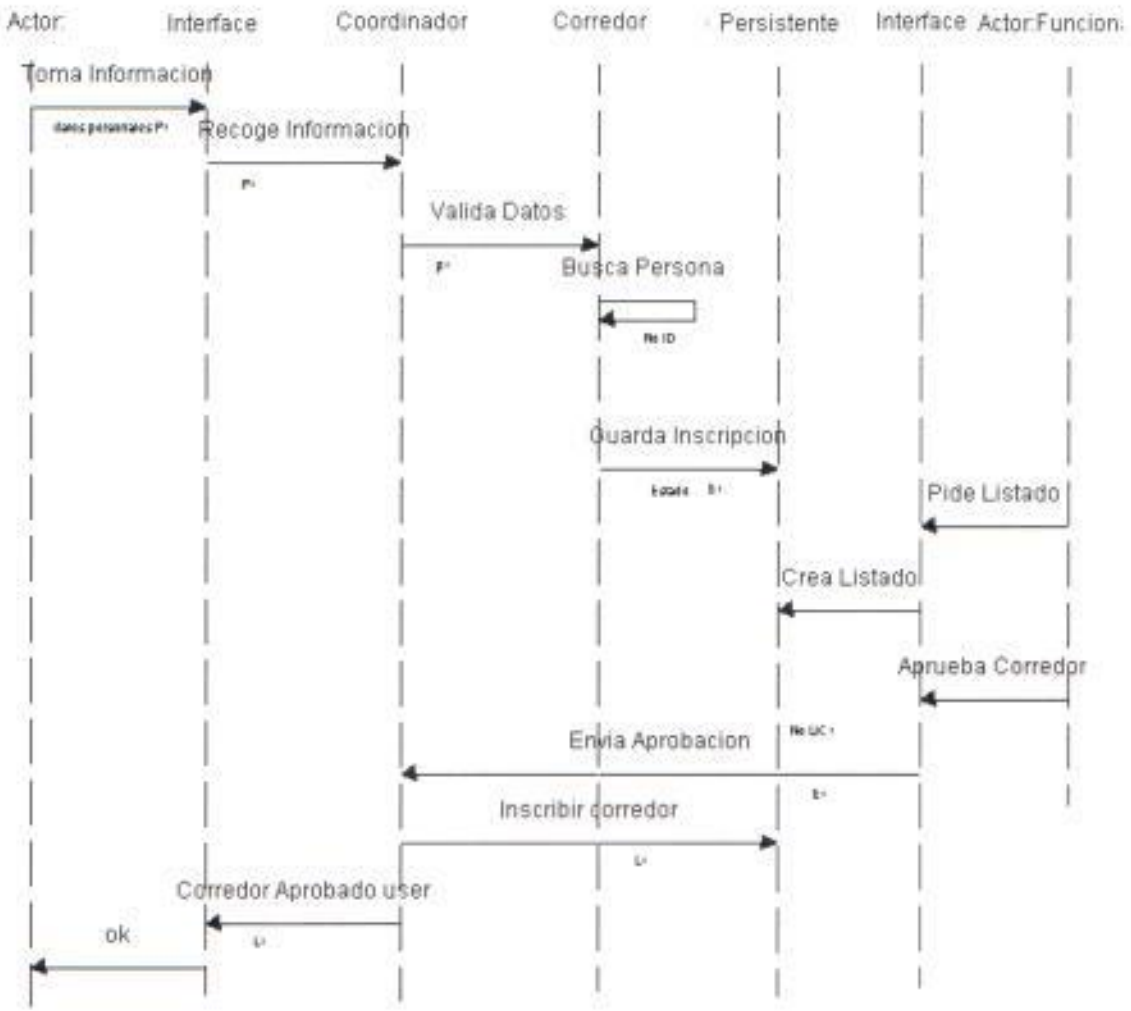


Figura 28: Diagrama de Iteracción de Objetos de Inscripción de Corredor Exitosa

4.3.1.3.2 Cambio de Membresia Exitosa

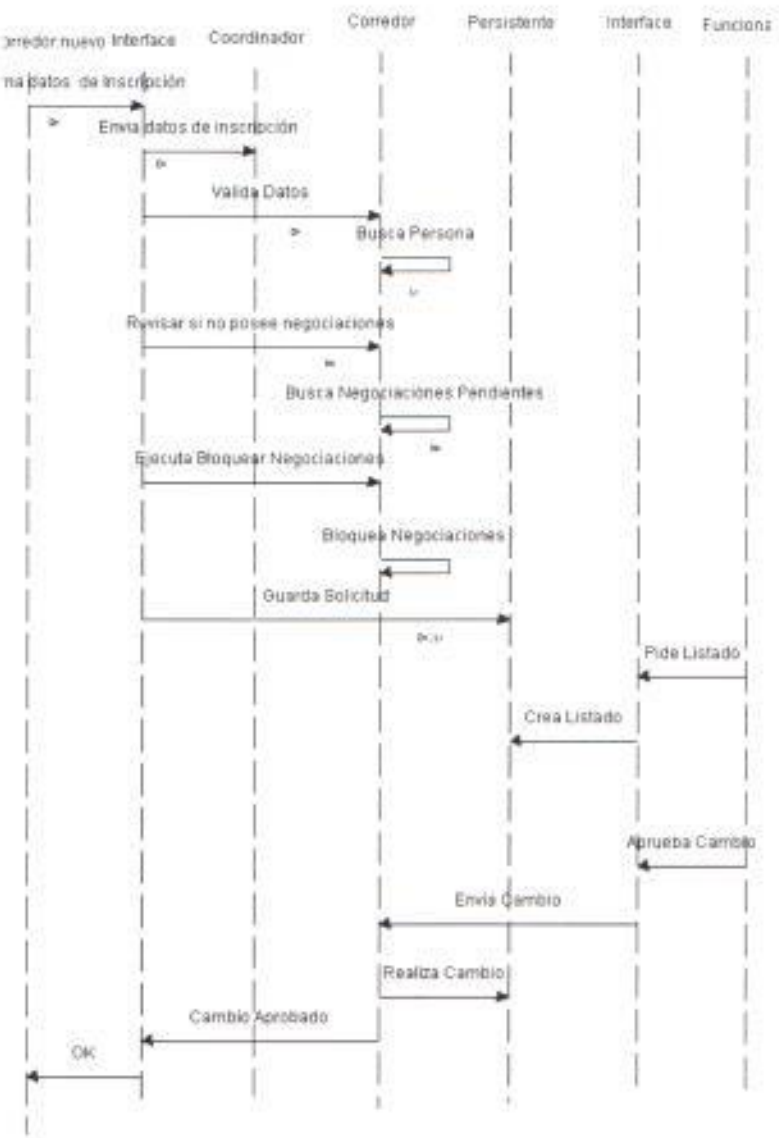


Figura 29: Diagrama de Iteracción de Objetos de Cambio de Membresia Exitosa

4.3.1.3.3 Inscripción de Productos por Corredor Exitosa

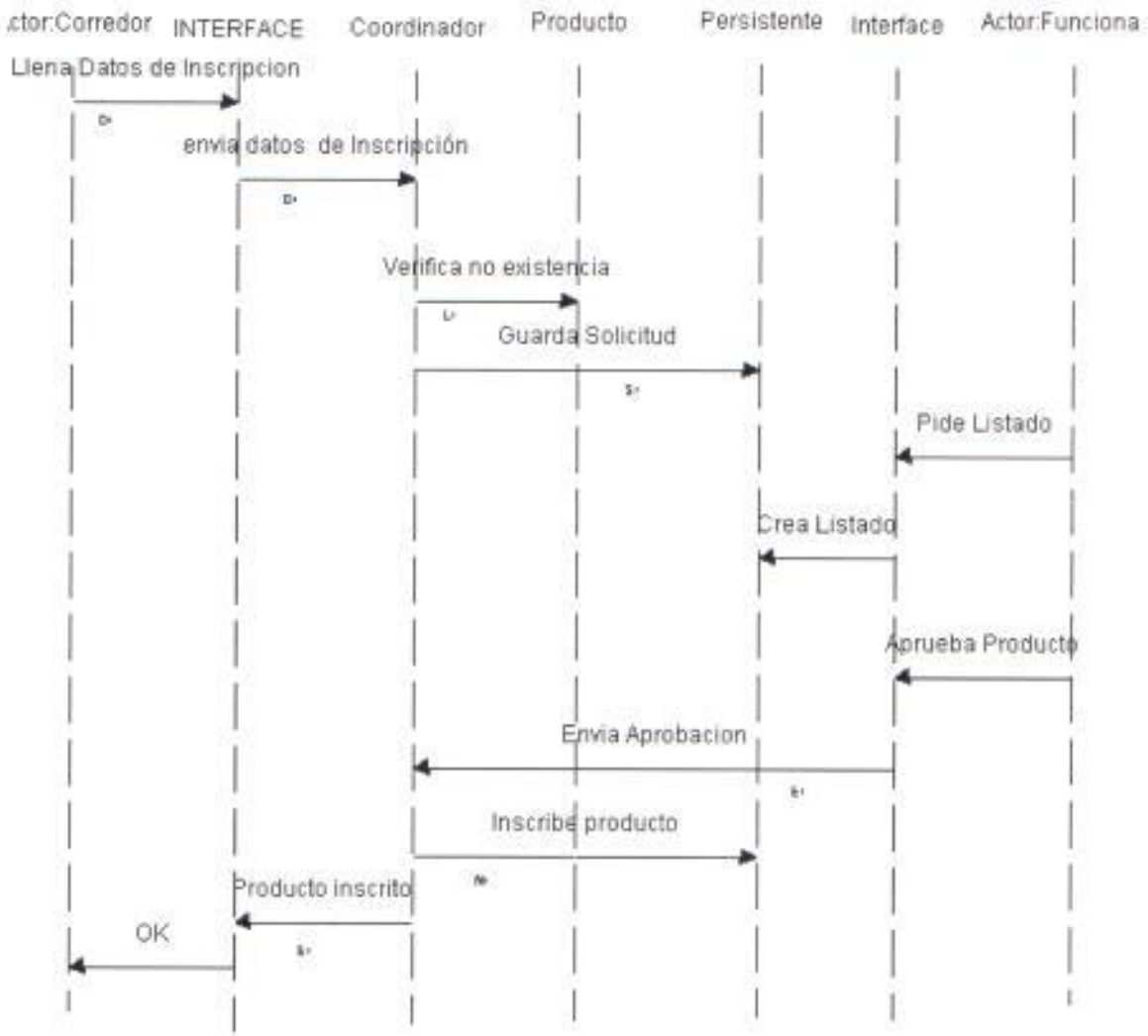


Figura 30: Diagrama de Iteracción de Objetos de Inscripción de Productos por Corredor Exitosa

4.3.1.3.4 Operación directa con modificación y exitosa

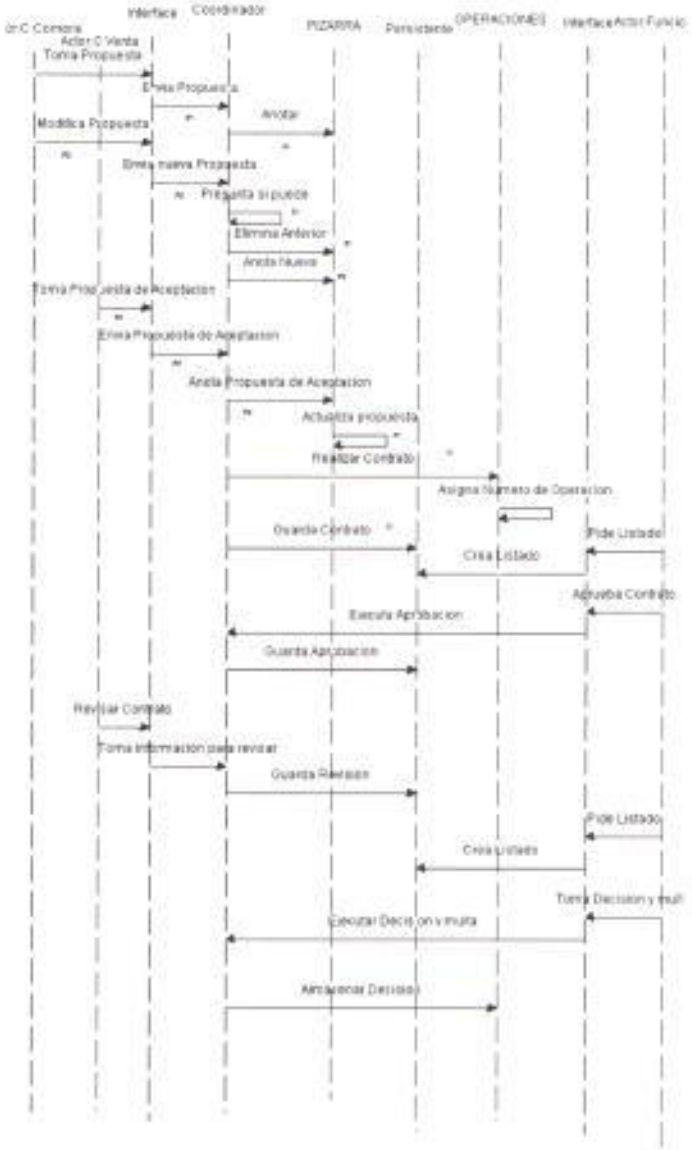


Figura 31: Diagrama de Iteracción de Objetos de Operación directa con modificación y exitosa

4.3.1.3.5 Operación directa sin modificación y exitosa

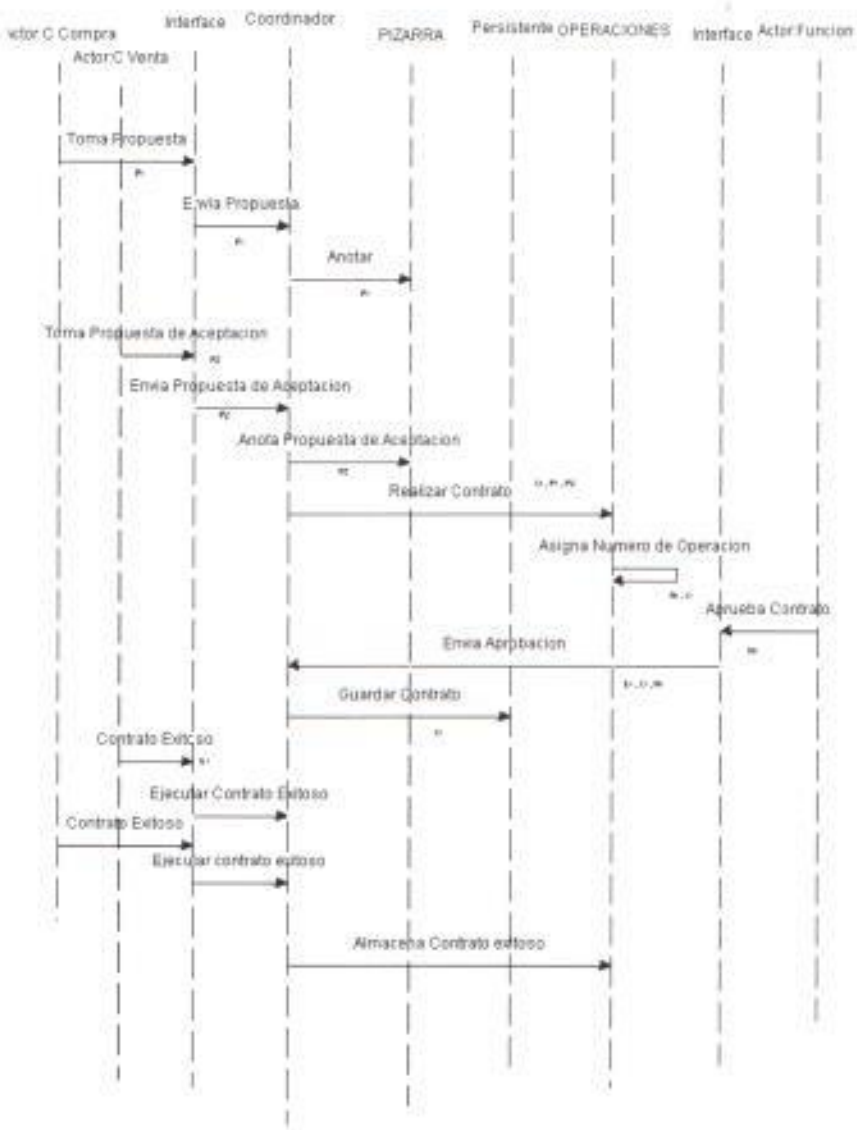


Figura 32: Diagrama de Iteracción de Objetos de Operación directa sin modificación y exitosa

4.3.1.3.6 Operación cruzada con modificación y exitosa

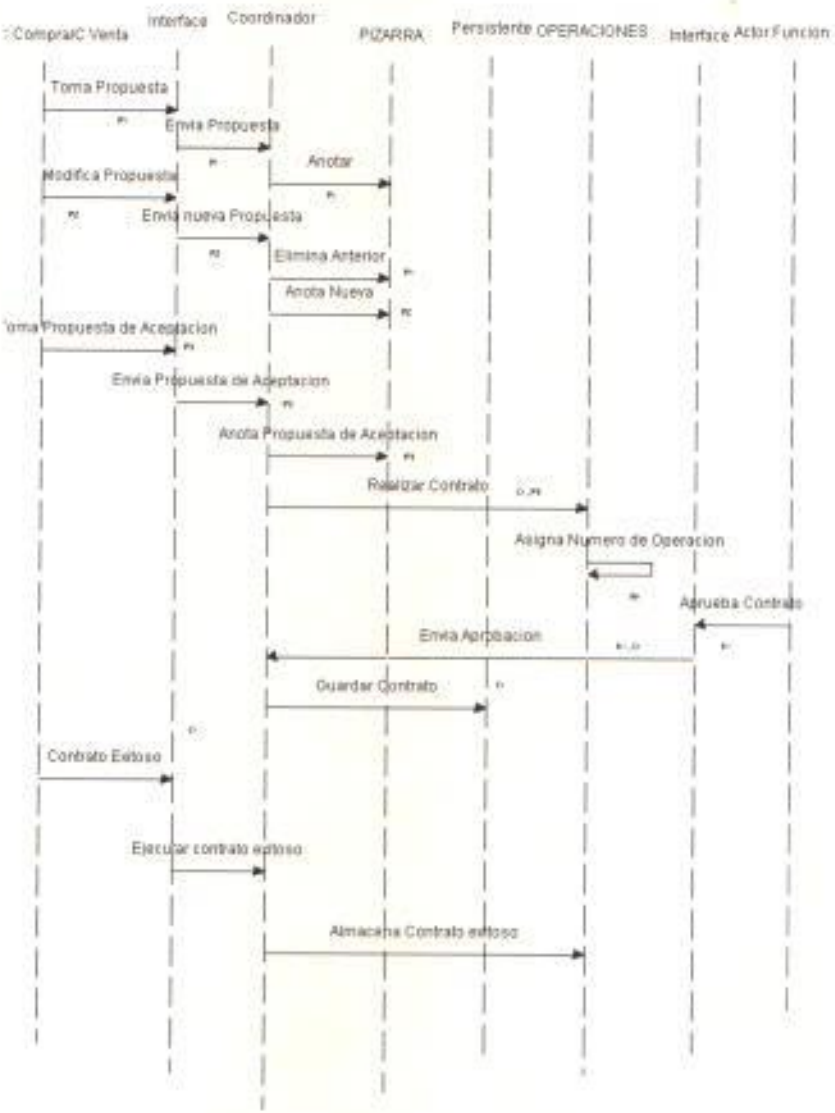


Figura 33: Diagrama de Interacción de Objetos de Operación cruzada con modificación y exitosa

4.3.1.3.7 Operación cruzada sin modificación y exitosa

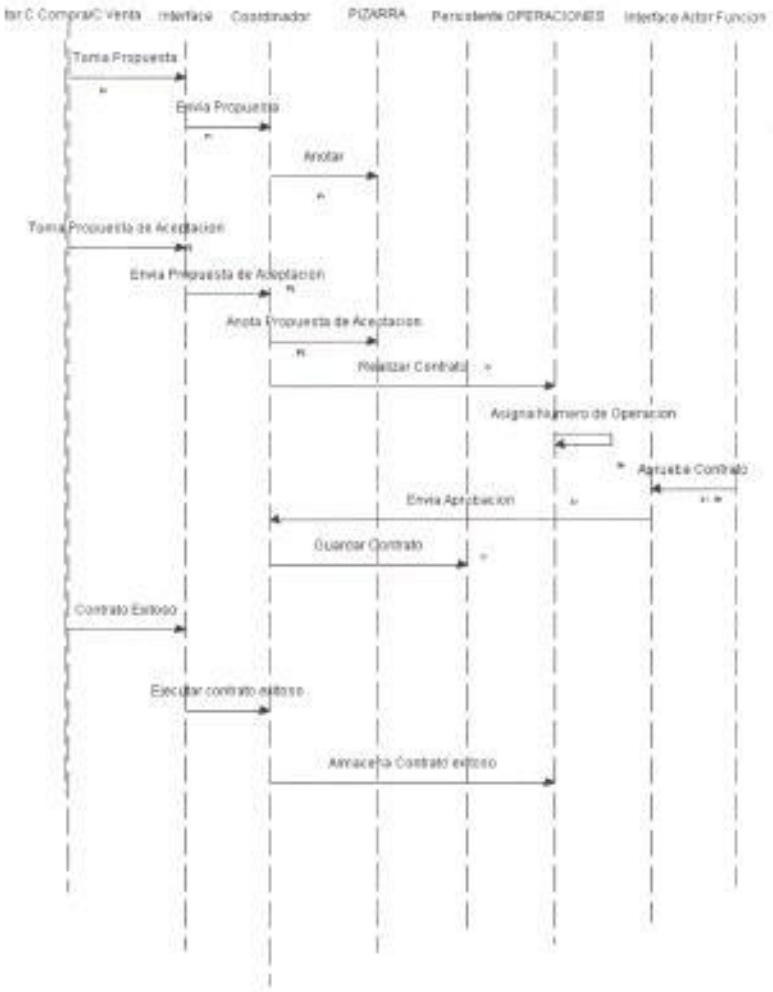


Figura 34: Diagrama de Iteracción de Objetos de Operación cruzada sin modificación y exitosa

4.3.1.3.8 Operación cancelada

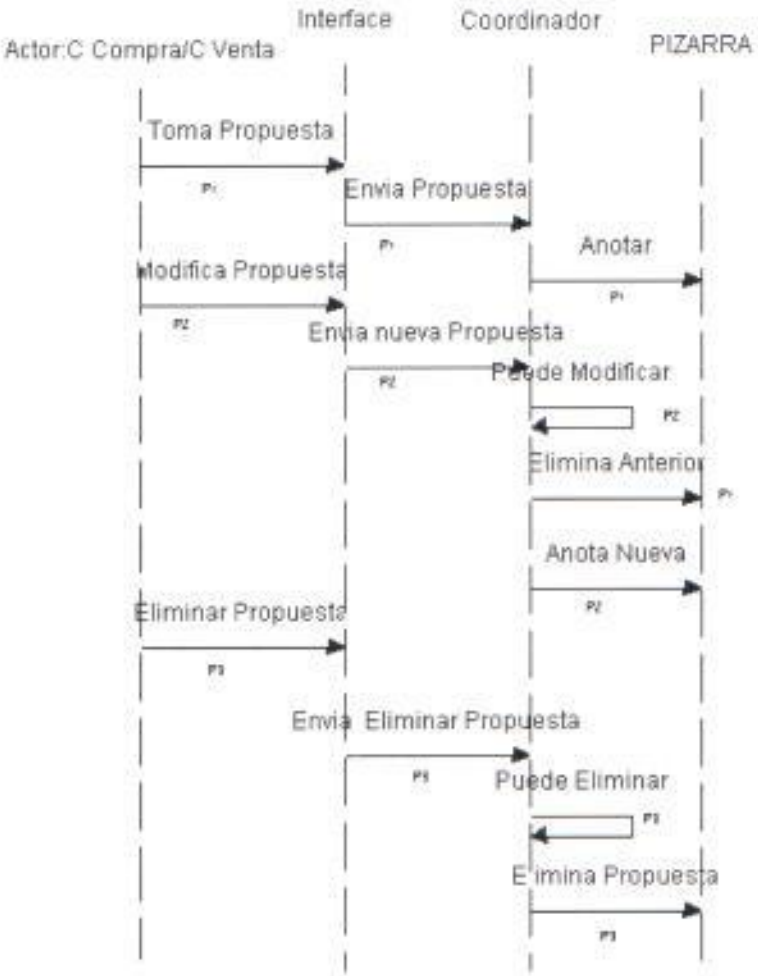


Figura 35: Diagrama de Iteracción de Objetos de Operación cancelada

4.3.2 Modelo Estático

4.3.2.1 Modelo de Análisis de Objetos

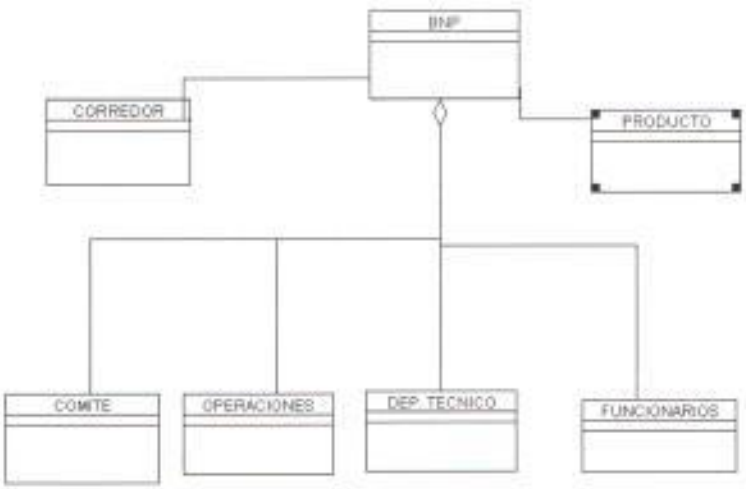


Figura 36. Modelo de Análisis de Objetos.

4.3.2.2 Modelo de Diseño de Objetos

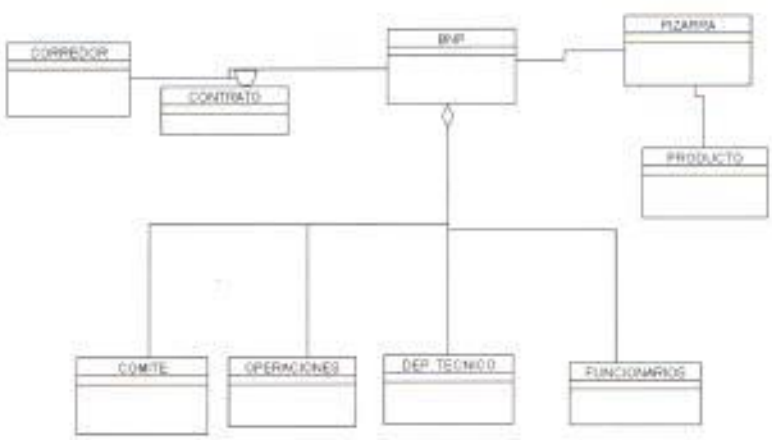


Figura 37. Modelo de Diseño de Objetos.

4.4 Modelo de Implementación

Es la definición, en un lenguaje de programación particular, de las clases y relaciones desarrolladas en el diseño de objetos.

Se escogió Java por ser la herramienta de mayor aceptación en cualquier plataforma, siendo la más eficaz en la implementación de objetos en Internet.

4.4.1 Diseño Descripción de Clases

Clase: Producto

Descripción : Este objeto realiza la función de inscribir al producto y lista los productos de la base de datos

Atributos :

Estado P (propuesto), N, Cantidad Mínima, Número de inscripción, Unidad, Clase, Requisito, Fecha de Ingreso, Fecha de Salida, Descripción, Nombre

Métodos:

Cambia estado, Aceptar, Listar, Buscar

Clase: Propuesta

Descripción: Es el objeto encargado por medio del cual, el actor Corredor puede ingresar una propuesta, ya sea de tipo compra ó venta; una vez ingresada puede participar en negociación.

Atributos :

Estado P (propuesto), Fecha, No Contrato, Lugar, Tipo Operación, Cantidad, No. Corredor

Métodos:

Ingresar Propuesta, Negociación.

Clase : Pizarra

Descripción: Este objeto es el encargado de mostrar todas las propuestas a negociarse. La pizarra está dividida en 2 partes: las propuestas de los demás corredores (parte superior del Applet) y las propuestas del corredor activo (parte inferior); para efectuar la negociación basta seleccionar la propuesta del otro corredor y la del corredor activo.

Super Clase:

Propuesta

Atributos :

Fecha, No. Rueda, Estado P (propuesto), Fecha, No Contrato, Lugar, Tipo Operación, Cantidad, No. Corredor

Métodos:

Inscripción, Propuesta, Aprueba

Clase : Corredor

Descripción: Este objeto se encarga de recoger toda la información necesaria de una persona para poder ser inscrita como corredor. La información ingresada será objeto de revisión por otro cliente quien aprueba o no el ingreso del corredor.

Atributos :

No. Licencia, Tipo, Estado, Fecha de Inscripción

Métodos

CambioMembresía , AutorizaFuncionario

Clase: Persistente

Descripción: Se encarga de recoger toda la información necesaria y la transmite al repositorio de base de datos.

Atributos :

Estado

Métodos

ActualizaBase, IngresaODBC

4.5 Diseño del Servidor

4.5.1 Funcionalidad

Para el desarrollo de la funcionalidad fue necesario tomar en consideración la existencia de 3 objetos denominados: Coordinador, Persistente, y Servidor para cada cliente.

Objeto Coordinador: Es el encargado de registrar a cada cliente que se conecta y guardar en una base temporal para luego ser considerado dentro de los requerimientos de sistema, considerando a demás que se asigna un *server class pool* para que éste pueda accesar, en todo caso en el objeto que se encarga de interactuar con la interfaz del usuario, en este caso el usuario seria el coordinador de la bolsa nacional de productos agropecuarios.

Objeto Persistente: Es el que se encarga de transformar las peticiones de los clientes en sentencias SQLs. para poder interactuar con la base de datos y el objeto coordinador

Objeto Servidor para cada cliente: Es el que se encarga de tomar los requerimientos que los clientes, similar a las funciones que realiza un TP monitor que procesa transacciones para el sistema operativo.

La descripción de estos objetos es necesarios para especificar las diferentes actividades que se desarrollan en el sistema, mencionaremos varias de las funciones principales:

1. Recepción de inscripciones para los corredores
2. Aprobación de las diferentes solicitudes para los corredores
3. Eliminación de productos
4. Envío de propuestas para la negociación
5. Modificación de propuestas en la negociación
6. Aceptación de propuestas
7. Almacenamiento de contratos
8. Generación de listados de los nuevos corredores
9. Clasificación de productos nuevos para su aprobación
10. Aprobación de contratos
11. Permitir que se realicen solicitudes para la inscripción de nuevos productos
12. Cambios de membresía

4.5.2 Tipo de Servidor y Justificación

El servidor a implementar es un servidor de base de datos orientado a procesamiento de transacciones.

Gracias a la tecnología Cliente Servidor se obtienen buenos tiempos de respuestas y además permite que muchos clientes accesen simultáneamente, ofreciéndonos de esta manera la ventaja de tener todo tipo de estructura de redes como lo son la LAN, WAN, MAN, TAN, esto implica que la entrega de paquetes es confiable logrando reducir el esfuerzo de programación, y dándonos la oportunidad de aplicar programación orientada a objetos, de tal manera que se puede ensamblar y desamblar el sistema sin perder la integridad referencial del mismo.

En el sistema se utiliza un objeto coordinador para procesamiento de transacciones, el mismo que se encarga de levantar los servicios para el objeto persistente el cual a su vez se encarga de comunicarse con la base de datos.

4.5.3 Diseño de los Datos manejados en el servidor

Para el diseño del servidor, se utiliza la herramienta SQL server, la cual maneja archivos de extensión "DAT", para almacenar en ellos las bases de datos.

La arquitectura CORBA nos permite manejar archivos de extensión class para crear los objetos servidores que van a realizar un tipo de transacción definido, estos objetos levantan servicios del lado del servidor.

4.5.3.1 DICCIONARIO DE LA BASE DE DATOS

bp_clientes

cliente: Código del Cliente (Todo cliente es una persona).

cl_f_p_propuesta: Fecha de la primera propuesta del cliente.

cl_f_u_propuesta: Fecha de la última propuesta del cliente.

cl_numpropuesta_v: Numero de propuestas de venta.

cl_numpropuesta_c: Número de propuestas de compra.

bp_comite

comité: Código del comité. 4 caracteres.

cm_tipo: Tipo de comité: Arbitraje, Aprob_corredor, Aprob_producto.

cm_n_integrantes: Número de integrantes del comité.

bp_corredor

corredor: Código del Corredor (Todo corredor es una persona).

co_licencia: Número de licencia del corredor

co_estado: Estado del corredor: Propuesto, Activo, Suspendido, Eliminado, Rechazado.

co_nombretarjeta: Nombre de la tarjeta para Negociaciones.

co_numerotarjeta: Número de la tarjeta para Negociaciones.

co_f_activo: Fecha en que inició operaciones el corredor

bp_funcionario

funcionario: Código del Funcionario (Todo funcionario es una persona).

fu_cargo: Cargo que ocupa en la bolsa.

fu_f_ini_trabajo: Fecha en la que inició sus labores. DD-MM-AAAA.

fu_departamento: Departamento en el que labora.

bp_funcionario_comite

comité: Código del comité.

funcionario: Código del funcionario.

bp_negociacion

negociacion: Código de la Negociación. Es de tipo numérico y es secuencial. Al momento de Insertar se calcula el máximo almacenado y se le suma 1.

ne_p_funcionario

ne_p_compra

ne_p_venta

ne_f_negociacion

ne_lugarentrega

ne_f_entrega

bp_personas

persona: Código de la Persona. Es de tipo numérico y es secuencial. Al momento de Insertar se calcula el máximo almacenado y se le suma 1.

pe_nombres: Nombres de la Persona

pe_p_apellido: Primer apellido de la Persona

pe_s_apellido: Segundo apellido de la Persona

pe_direccion: Domicilio de la Persona

pe_telefono: Número telefónico de la Persona

pe_sexo: Sexo de la Persona. 'S' ó 'F'

pe_cedula: Número de cédula o pasaporte de la Persona

pe_f_nacimiento: Fecha de nacimiento de la persona. DD-MM-AAAA

bp_productos

producto: Código del Producto. Es de tipo numérico y es secuencial. Al momento de Insertar se calcula el máximo almacenado y se le suma 1.

pr_n_producto: Nombre del Producto

pr_cant_minima: Cantidad mínima para poder negociar el producto.

pr_und_c_minima: Unidad en la que está dada la cantidad mínima para poder negociar el producto

pr_clase: Clase del Producto. Es un calificativo

pr_descripcion: Descripción adicional del producto

pr_inen: Norma INEN que califica al producto

pr_f_ingreso: Fecha de registro del producto

pr_f_salida: Fecha en que se dejó de negociar el producto

pr_estado: Estado del Producto: Propuesto, Vigente, Eliminado.

pr_corredor: Código del corredor que propuso el producto para negociaciones.

bp_propuesta

propuesta: Codigó de la Propuesta. Es de tipo numérico y es secuencial. Al momento de Insertar se calcula el máximo almacenado y se le suma 1.

pt_producto: Código del producto a negociar.

pt_corredor: Código del corredor que negocia.

pt_cantidad: Cantidad a negociar.

pt_precio_compra: Precio de Compra a negociar. (Si es que el corredor es de compra).

pt_precio_venta: Precio de Venta a negociar.(Si es que el corredor es de venta).

pt_tipo_grado: Grado del producto (pureza, humedad, mezcla, etc.)

pt_empaque: Unidad en la que la cantidad esta propuesta.

pt_comité: Comité responsable de la propuesta.

rueda: Rueda en la que se hizo la propuesta.

pt_f_vencimiento: Fecha de vencimiento de la propuesta.

pt_f_propuesta: Fecha en la que se realizó la propuesta.

pt_forma_pago: Forma de pago deseada o propuesta. 'T' o 'E'.

pt_estado: Estado de la propuesta. 'A' activa o 'N' negociando.

bp_rueda

rueda: Código de la Rueda. Es de tipo numérico y es secuencial. Al momento de Insertar se calcula el máximo almacenado y se le suma 1.

ru_f_inicio: Fecha de inicio de la rueda.

ru_f_fin: Fecha de Fin de la rueda.

bp_tmpcorredor

corredor

tm_nombres

tm_p_apellido

tm_s_apellido

tm_direccion

tm_telefono

tm_sexo

tm_cedula

tm_f_nacimiento

tm_cambio

tm_licencia

4.5.4 Diseño de la aplicación servidora

La aplicación Servidora fue diseñada bajo la estructura de dar al cliente las respuestas de negociar datos de la BNPA. Los objetos Servidores proporcionan el medio de comunicación, entre los datos que recepta el cliente con la información que se guarda en la base de datos.

4.5.4.1 Diagrama de Estructuras

La estructura que utilizamos para la BNPA es Three Tier:

En **Cambio de Membresía** el primer Tier se lo realiza en el Cliente, mediante la página Web que carga un applet, la estructura que se obtiene se lo envía del objeto InsProductos hacia el objeto dispensador. En el segundo Tier el dispensador encuentra un objeto servidor libre y se lo asigna al Cliente. El tercer Tier es formado por el objeto persistente en cual reside la estructura transmitida por el objeto servidor, ejecutando sentencias SQL para manipular la base de datos.

La **Inscripción del Producto** tiene como primer Tier, el applet corriendo en una página HTML, donde su forma carga los campos con los datos que el corredor ingresa, en una estructura que es transmitida hacia el servidor después que el objeto dispensador le asigne el servidor correspondiente. En segundo Tier envía del objeto servidor los datos

del producto en una estructura y recibe un vector del objeto persistente. El tercer Tier modifica los datos en la base de datos, mediante sentencias SQL.

En **Inscripción de la Propuestas** la estructura cargada por el cliente en el primer Tier es enviada al objeto servidor, y en el segundo Tier el objeto servidor busca al persistente para entregarle esta estructura. El tercer Tier ejecuta la sentencia SQL. En el camino de retorno al cliente se obtiene una bandera de éxito o fracaso.

Los datos que se manejan en **Pizarra**, son iguales que en la Inscripción de la Propuesta, con el único cambio que al retornar la respuesta se obtiene un vector del persistente al servidor, y una estructura del servidor al cliente que cargará su applet con esta información.

4.5.5 Algoritmos de Solución.

4.5.5.1 CLASES DEL SERVIDOR

```
// Server.idl
module Server
{
 exception BnpaException
  { string reason;
};
```

```
Struct Struct_Server
{
<< estructura general>>
};
typedef sequence<Struct_pizarra> pizarraSeq;
interface BnpaServer
 {
  pizarraSeq  captura_1(in string codigo);
  pizarraSeq  captura_2(in string codigo);
  string captura_vendedor(in string codigo);
  long actualiza_propuesta(in string codigo);
  long ingresa_negociacion(in string codigo1,in string codigo2);
};

interface BnpaServerDispenser
{
   BnpaPizarra reserveBnpaObject() raises (BnpaException);
   void   releaseBnpaObject(in BnpaPizarra BnpaPizarraObject)
                    raises (BnpaException);
  };
```

};

4.5.5.2 Compromisos de Diseño

En Seguridades: Este proyecto se le debe incluir un sistema de identificación por medio del logon, para tener un nivel de seguridad. Tanto para el corredor como para el funcionario, que son los elementos más críticos en el manejo de datos.

En Aplicaciones: Se necesita un continuo monitoreo del estado de integridad de los datos. Debemos de proveer de un seguimiento a un probable error de conexión con los clientes.

En Sistema: Para el diseño de integración de los sistemas con bases distribuidas, debemos proporcionar confiabilidad en el método de respaldo, y en los medios robustos de comunicación con las bases de datos.

4.6 Diseño del Cliente

Análisis Previo

Previo al diseño del cliente, se elaboró un análisis profundo sobre la base de:

- Realizar una interpretación exhaustiva de que cada usuario (directo o indirecto), y que función cumple en la Bolsa Nacional de Productos.
- Observar que acciones y operaciones realizan los usuarios en la BNPA.
- Diferenciar cual es el escenario donde se llevan las negociaciones, en la BNPA (La Rueda).

El software del cliente fue diseñado para proporcionar las siguientes funciones específicas.

4.6.1 Funcionalidad

El diseño de la parte del cliente se hizo con la finalidad que el respectivo software cumpla con las siguientes funciones:

Interpretación:

- La interpretación de la parte Cliente está basada en los requerimientos de información sobre la Bolsa Nacional de Productos, y acciones que un usuario (corredor y funcionario) puede realizar en ella, cuyas opciones se visualizan en los Applets correspondientes. Para poder llevar a cabo todos los objetivos funcionales del sistema, está implementado usando Applets; que en la parte cliente permitirá visualizar las diferentes opciones de una manera gráfica y amigable.
- Las operaciones se realizan por medio de solicitudes a los objetos servidores, por parte de los objetos clientes.
- Debe interpretar las respuestas emitidas por los objetos servidores y generar las acciones adecuadas. Si sucede algún error debe presentar el mensaje correspondiente, por ejemplo, si los datos no son correctos sobre el corredor a consultar.

Errores:

- Debe detectar errores en el medio de comunicación, entre los clientes y los servidores por ejemplo cuando el servidor no ha sido levantado.
- Detectar los errores cuando el Gatekeeper no ha sido levantado, ya que este habilita los Applets Visibroker para comunicarse con los objetos servidores a través de las

redes, mientras todavía se conforman las restricciones de seguridad impuestas por el Web Browser.

- La desventaja del Gatekeeper es que; si la solicitud del cliente debe pasar por un Servidor Proxy, el paquete IIOP no llega al servidor al que está haciendo la solicitud.
- Netscape 4.04, necesita de un Plug-Ins que es el responsable de actualizar las clases ORB que necesita Netscape y de está manera el Cliente puede abrir clases que incluyen las ORB. Si no se le instala visualiza errores de comunicación.
- Realizar las acciones que permitan ya sea continuar o completar la información, si fuesen posibles.
- Detectar errores que se dan en los ingresos de datos, ya sea datos no válidos (formatos de fecha, nombre con números, etc.) o incompletos (campos que no pueden ser nulos).

Validación:

- Validar los datos ingresados por los usuarios, que sean correctos y completos.
- Validar la consecutividad de los números generados ya sea de contratos, ruedas, licencias de corredores y de productos.

Interfaces:

- Presentar mensajes en línea que informen los diferentes estados de la información. (Las propuestas hechas en cada momento, los datos de los corredores, de los productos, etc.).
- Proveer una barra de estado con información sobre las opciones que se vaya a ejecutar.
- La interfaz gráfica del WEB es lo suficientemente amigable, que permite a ususarios de diferentes niveles poder interactuar desde cualquier parte del mundo.

Requerimientos al Servidor:

- Solicita al servidor los datos de los diferentes entes que actúan en el sistema, tales como: datos del corredor, de los productos, de las propuestas, confirmación de contrapropuestas.
- Solicita el registro de los datos correspondientes a los nuevos corredores, productos, propuestas y negociaciones.

Membresía: Operaciones.

- **Inscripción de Membresía:** La inscripción de membresía es realizada por parte de los aspirantes a corredores de la BNPA. Aquí el aspirante deberá ingresar sus datos

personales, el encargado de otorgar la membresía es el funcionario, el mismo que deberá chequear los requisitos del aspirante, proporcionándole su membresía en caso de haber cumplido con todos; caso contrario se le enviará un mensaje de negado.

- **Modificación de Membresía:** El sistema le permite al corredor de la BNPA modificar ciertos datos personales (como dirección, teléfono, estado civil, etc.), con el objetivo de que él pueda actualizar sus datos.
- **Ceder Membresía:** Un corredor, en el momento que desee dejar de pertenecer a la bolsa podrá ceder o vender su membresía a otro aspirante; para esto sólo basta llenar el Applet correspondiente y enviar su requerimiento para que sea aceptado.

Producto: Operaciones.

- **Inscripción del Producto:** Siempre y cuando el producto cumpla con las normas de inscripción (cantidad, grado, norma INEN, pureza, etc.) de la BNPA, puede ser inscrito. Esta acción es realizada sólo por los corredores.
- **Consulta del Producto:** La consulta es sobre la base del código del producto, código del corredor, nombre del producto, norma INEN y su descripción, esta consulta es publicada para los personajes que negocian en la BNPA.
- **Bloqueo del Producto:** Lo realizan los funcionarios, sobre la base de la no-venta de un producto en un tiempo tal que pueda afectar su estado de composición.

Negociación: Operaciones.

- **Propuesta:** Es realizada por los corredores para la compra o venta de un producto. Se ingresa los datos del corredor, del producto, precio de compra/venta, cantidad, fecha de entrega, forma de pago; todos estos datos se los ingresa en una forma que es cargada por un Applet. En el momento de grabar la propuesta, automáticamente será publicada en la pizarra.
- **Pizarra:** Es un Applet donde los corredores podrán observar todos los productos de la rueda. En la parte inferior se muestran las propuestas del corredor activo (corredor que se encuentra observando la pizarra), y en la parte superior todas las demás propuestas que no sean las suyas.

Cierre de Negociación.: El cierre de negociación se refiere, cuando el corredor oferente se pone de acuerdo con el corredor demandante para firmar ya el contrato, la presentación de esto es una forma igualmente cargada por una Applet que muestra los datos o términos en los cuales se hace el contrato o negociación.

La negociación es establecida en el instante en que el corredor activo seleccione su propuesta y la propuesta de otro corredor en la parte superior, y envíe el aceptar de la propuesta, quedando cerrada la operación.

Respuestas:

Interpretar las respuestas provenientes del servidor y visualizar en las pantallas correspondientes.

4.6.2 Tipo de Cliente y Justificación

La arquitectura utilizada es CORBA ORB. El Cliente esta implementado basándose en objetos; estos invocan a los objetos servidores, utilizando Applets Java bajo la herramienta de Visual Café.

Tipos de Cliente:

- **InsProductos :** Este objeto realiza la función de inscribir al producto e inicialmente lo registra con el estado **P (propuesto)**, invoca al **objeto persistente db,** quien se encarga de actualizar las tablas en la base de datos. Retorna un código al applet, que representa el número de inscripción.

  Este objeto es utilizado por el actor corredor, para que desde el Web pueda registrar su producto a la BNPA.
- **LisProductos:** Lista los productos de la base de datos que tengan estado **P (propuesto)**, y además método Aceptar, cambia el estado de **P (propuesto)** a **V (vigente)** de los productos seleccionados**.** En el método Buscar, retorna la lista de productos de estado P.

Este objeto es usado por un actor Funcionario, quien tiene la autorización de modificar los estados del producto.

- **ModProducto**: Modifica los datos del producto seleccionando, estos datos pueden ser Norma INEN, nombre del producto.

  El actor Corredor modifica los datos del producto que inscribe. Esto lo efectúa en el applet de Modificación del Producto que invoca a este objeto ModProducto.

- **PropuestaCliente :** Es el objeto encargado por medio del cual, el actor Corredor puede ingresar una propuesta, ya sea de tipo compra ó venta; una vez ingresada puede participar en negociación.

  Por medio de este objeto, pueden ingresar una propuesta; la cual inmediatamente puede ser vista por otros corredores. De esta manera se establece la negociación.

- **PizarraCliente:** Este objeto es el encargado de mostrar todas las propuestas a negociarse. La pizarra está dividida en 2 partes: las propuestas de los demás corredores (parte superior del Applet) y las propuestas del corredor activo (parte inferior); para efectuar la negociación basta seleccionar la propuesta del otro corredor y la del corredor activo.

- **CorredorCliente:** Este objeto se encarga de recoger toda la información necesaria de una persona para poder ser inscrita como corredor. La información ingresada será objeto de revisión por otro cliente quien aprueba o no el ingreso del corredor.

- **CambioMembresía:** El objeto recoge la información que ingresa un corredor inscrito y mediante la cual dicho corredor expresa su deseo de ceder su membresía a otra persona no inscrita aún. Nuevamente la información ingresada deberá ser revisada por otro cliente quien aprueba o no el cambio de membresía propuesto.
- **AutorizaFuncionario:** Este objeto es el encargado de aprobar ingresos de nuevos corredores y los cambios de membresías propuestos. AutorizaFuncionario está compuesta por dos pestañas en las que se realizan dichas aprobaciones.

La justificación del diseño bajo esta Arquitectura, es debido a que:

- Esta arquitectura proporciona objetos seguros, portables y persistentes.
- Maneja eficazmente los objetos distribuidos; el cliente no necesita conocer donde reside el objeto distribuido ó que operaciones del sistema se están ejecutando; tampoco necesita conocer como está implementado el objeto servidor.
- Utiliza el modelo cliente servidor Three Tier.
- Respecto a los Applets, son para que tenga una interfaz gráfica en cualquier browser habilitado para Java, con lo que el usuario en estos días esta y debe estar familiarizado.

4.6.3. Diseño de la aplicación Cliente

La aplicación del Cliente fue diseñada bajo el esquema de funcionamiento de la BNPA. Los objetos Clientes van a invocar un solo objeto Servidor, cabe recalcar que esto es transparente para el Cliente.

4.6.3.1 Algoritmos de Solución

Los algoritmos solución de por parte del cliente son principalmente:

- Inscripcion_Corredor
- Modificacion_Corredor
- Ceder_Membresia
- Inscripcion_Producto
- Consulta_Producto
- Bloqueo_Producto
- Inscripcion_Propuesta
- Aprobacion_Propuesta
- Pizarra

La forma para el desarrollo de Algoritmos se hizo de la siguiente manera:

Las palabras reservadas para nuestro lenguaje de algoritmos son:

Begin, End, Input, Print, If - Then - Else - End If, Repeat - Until, While - Wend.

La función de cada palabra reservada es lo que significa en español.

Input: Ingresar por teclado datos.

Formato: Input "Mensaje": <Lista_de_variables>;

Print: Presentar por pantalla datos, mensajes o informacion en general.

Formato: Print "Mensaje", <Lista_de_variables>;

Para algunas de estas que sirven como estructuras de control son de la siguiente manera:

Estructura usada para los algoritmos.

```
Algoritmo Nombre_del_Algoritmo;
Begin
        Instruccion 1;
        Instruccion 2;
        .....
        .....
          Repeat
                Instruccion m;
                Instruccion m+1;
                .....
                .....
          Until (Condicion);
                Instruccion n;
                Instruccion n+1;
                .....
                .....
      While (Condicion)
                Instruccion nn;
```

```
                    Instruccion nn+1;
                    .....
                    .....
                Wend;
End.
```

Las operaciones aritmeticas y logicas son de la forma convencional.

Los procedimientos:

Los procedimientos usados son sobretodo para denotar los Querys a la base de datos: SQL_Insert(<Nombre_base_de_datos>, <Nombre_de_la_tabla>, <Lista_de_campos>), permite insertar un registro en la tabla especificada de la base de datos; SQL_Replace(<Nombre_base_de_datos>, <Nombre_de_la_tabla>, <Lista_de_campos>), permite modificar los campos de un registro leído de la base de datos de la tabla especificada; SQL_Read(<Nombre_base_de_datos>, <Nombre_de_la_tabla>, <Lista_de_campos>), permite leer un registro de una tabla perteneciente a la base de datos especificada.

Active (<Lista_de_campos_o_variables>), Activa la lista de campos o variables permitiendo al usuario posicionarse con los cursores en los valores visualizados en la pantalla; Modify( ), este procedimiento me permite modificar los valores de las variables que han sido activadas con Active, Clear( ), permite barrar la información presente en pantalla.

Las funciones:

SQL_EOF(<Nombre_base_de_datos>, <Nombre_de_la_tabla>), retorna TRUE si se encuentra en el fin de archivo,caso contrario FALSE.

SQL_BOF(<Nombre_base_de_datos>, <Nombre_de_la_tabla>), retorna TRUE si se encuentra en el inicio del archivo, caso contrario FALSE, Date( ) esta función me retorna la fecha dada por el reloj interno del computador.

Con respecto a la definición de variables las vamos a omitir, es decir, que cuando se las usa por primera vez se las define según el tipo de dato asignado.

Desarrollo de Algoritmos

Algoritmo Inscripcion_Corredor;

```
Begin
Repeat
        Bandera = 0;
        Input "el nombre": pe_nombres;
        Input "el primer pellido": pe_p_apellido;
        Input "el segundo apellid": pe_s_apllido;
```

```
Input "Direccion":pe_direccion

Input "Telefono": p_telefono;

Input "Sexo": pe_sexo;

Input "numero de cedula": pe_cedula;

Input "fecha de nacimiento": pe_f_nacimiento;

persona = persona +1;

If (ningun campo es nulo) Then

    Repeat

        Print  "Digite Aceptar para Insertar en la Base de datos";

        Print "Digite Cancelar para no hacer la Inserción en la base de datos";

        Input Respuesta;

                If (Respuesta = "Aceptar") Then

            SQL_Insert(BNPA, persona, bp_personas, pe_nombres, pe_p_apellido,

            pe_s_apellido,  pe_direccion,  pe_telefono,  pe_sexo,  pe_cedula,

            pe_f_nacimiento);

            Bandera=1;

        Else

            Clear( );

            Bandera = 1;

                                End If
```

```
        Until (Bandera = 1);
Until (Bandera = 1);
End.
```

Algoritmo Modificacion_Corredor;

```
Begin
    Bandera =0;
                Input "El número de su Licencia ": Licencia;
        While (.NOT. SQL_EOF(bnpa, bp_corredor))
            SQL_Read (BNPA, bp_personas, persona, pe_nombres, pe_p_apellido,
                pe_s_apellido, pe_direccion, pe_telefono, pe_sexo, pe_cedula,
                pe_f_nacimiento);
            SQL_Read (BNPA, corredor, co_corredor, co_licencia, co_estado,
                co_nombretarjeta, co_numerotarjeta, co_f_activo);
                Print "Datos";
            If (Licencia = co_licencia) Then
                Print "No. de Licencia: ", co_licencia;
                Print "Nombre: ", pe_nombres;
                Print "Primer pellido: ", pe_p_apellido;
                Print "Segundo apellido: ", pe_s_apllido;
```

```
            Print "Direccion: ", pe_direccion
            Print "Telefono: ", p_telefono;
            Print "Sexo: ", pe_sexo;
            Print "Numero de cedula: ", pe_cedula;
            Print "Fecha de nacimiento: ", pe_f_nacimiento;
            Print "Estado del corredor: ", co_estado;
            Print "Tarjeta de crédito: ", co_nombretarjeta;
            Print "Número Tarjeta: ", co_numerotarjeta;
            Print "Fecha de activación: ", co_f_activo;
            Bandera=1;
                                End If
Wend
If (Bandera = 0) Then
        Print "No existe corredor registrado con ese número de licencia";
Else
    Input "Desea modificar algunos datos (S/N) ":Respuesta;
    If (Respuesta = "S") Then
        Active(pe_direccion, pe_telefono, co_nombretarjeta, co_numerotarjeta,
        co_estado, co_f_activo);
        Modify( );
```

```
            SQL_Replace (BNPA, bp_personas, persona, pe_nombres, pe_p_apellido,
            pe_s_apellido, pe_direccion, pe_telefono, pe_sexo, pe_cedula,
            pe_f_nacimiento);
            SQL_ Replace (BNPA, bp_corredor, corredor, co_licencia, co_estado,
            co_nombretarjeta, co_numerotarjeta, co_f_activo);
        End If;
    End If
End.
```

Algoritmo Ceder_Corredor;

```
Begin
                    Bandera =0;
        Input "El número de su Licencia ": Licencia;
    While (.NOT. SQL_EOF(bnpa, bp_corredor))
        SQL_Read (BNPA, bp_personas, persona, pe_nombres, pe_p_apellido,
            pe_s_apellido, pe_direccion, pe_telefono, pe_sexo, pe_cedula,
            pe_f_nacimiento);
        SQL_Read (BNPA, bp_corredor, corredor, co_licencia, co_estado,
            co_nombretarjeta, co_numerotarjeta, co_f_activo);
        If (Licencia = co_licencia) Then
```

```
Print "Datos";
Print "No. de Licencia: ", co_licencia;
Print "Nombre: ", pe_nombres;
Print "Primer pellido: ", pe_p_apellido;
Print "Segundo apellido: ", pe_s_apllido;
Print "Direccion: ",pe_direccion
Print "Telefono: ", p_telefono;
Print "Sexo: ", pe_sexo;
Print "Numero de cedula: ", pe_cedula;
Print "Fecha de nacimiento: ", pe_f_nacimiento;
Print "Estado del corredor: ", co_estado;
Print "Tarjeta de crédito: ", co_nombretarjeta;
Print "Número Tarjeta: ", co_numerotarjeta;
Print "Fecha de activación: ", co_f_activo;
Bandera=1;
                End If
Wend
If (Bandera = 0) Then
    Print "No existe corredor registrado con ese número de licencia";
End If;
```

Input "Esta seguro que desea ceder su membresia (S/N) ":Respuesta;

If (Respuesta = "S") Then

Active(bp_personas, pe_nombres, pe_p_apellido, pe_s_apellido, pe_direccion, pe_telefono, pe_sexo, pe_cedula, pe_f_nacimiento, co_estado, co_nombretarjeta, co_numerotarjeta, co_f_activo);

Modify( );

SQL_Replace (BNPA, bp_personas, persona, pe_nombres, pe_p_apellido, pe_s_apellido, pe_direccion, pe_telefono, pe_sexo, pe_cedula, pe_f_nacimiento);

SQL_ Replace (BNPA, bp_corredor, corredor, co_licencia, co_estado, co_nombretarjeta, co_numerotarjeta, co_f_activo);

End If;

End.

Algoritmo Inscripcion_Producto;

Begin

Repeat

Bandera =0;

Input "El nombredel producto: ": pr_n_producto;

Input "La cantidad mínima: ": pr_cant_minima;

```
Input "La cantidad mínima si es de unidades: ": pr_und_c_minima;

Input "La clase del producto: ": pr_clase;

Input "Descripción: ":pr_descripcion;

Input "Norma INEN: ": pr_inen;

Input "Fecha de ingreso: ": pr_f_ingreso;

Input "Fecha de salida: ": pr_f_salida;

Input "Estado": pr_estado;

Input "Corredor responsable: ": pr_corredor;

producto = producto + 1;

If (ningun campo es nulo) Then

    Repeat

        Print  "Digite Aceptar para Insertar en la Base de datos";

        Print "Digite Cancelar para no hacer la Inserción en la base de datos";

        Input Respuesta;

                If (Respuesta = Aceptar") Then

        SQL_Insert(BNPA,  bp_productos,  producto,  pr_n_producto,

        pr_cant_minima, pr_und_c_minima, pr_clase, pr_descripcion, pr_inen,

        pr_f_ingreso, pr_f_salida, pr_estado, pr_corredor);

        Bandera=1;

                        Else
```

```
                Clear( );
                Bandera = 1;
            End If;
        Until (Bandera = 1);
Until (Bandera = 1);
End.
```

Algoritmo Consulta_Producto;

```
Begin
                        Bandera =0;
        Input "El número Identificador del producto ": NP;
    While (.NOT. SQL_EOF(bnpa, bp_corredor))
        SQL_Read(BNPA, bp_productos, producto, pr_n_producto, pr_cant_minima,
            pr_und_c_minima, pr_clase, pr_descripcion, pr_inen, pr_f_ingreso,
            pr_f_salida, pr_estado, pr_corredor);
        If (NP = producto) Then
            Bandera=1;
            Print "Datos";
            Print "El nombredel producto: ": pr_n_producto;
            Print "La cantidad mínima: ": pr_cant_minima;
```

```
            Print "La cantidad mínima si es de unidades: ": pr_und_c_minima;
            Print "La clase del producto: ": pr_clase;
            Print "Descripción: ":pr_descripcion;
            Print "Norma INEN: ": pr_inen;
            Print "Fecha de ingreso: ": pr_f_ingreso;
            Print "Fecha de salida: ": pr_f_salida;
            Print "Estado": pr_estado;
            Print "Corredor responsable: ": pr_corredor;
            Bandera = 1;
                        End If
    Wend
    If (Bandera = 0) Then
        Print "No existe producto alguno registrado con ese número.";
    End If
End.
```

Algoritmo Bloquea_Producto;

```
Begin
    Bandera =0;
        Input "El número Identificador del producto ": NP;
```

```
While (.NOT. SQL_EOF(bnpa, bp_corredor))
    SQL_Read(BNPA, bp_productos, producto, pr_n_producto, pr_cant_minima,
                pr_und_c_minima,    pr_clase,    pr_descripcion,    pr_inen,
                pr_f_ingreso, pr_f_salida, pr_estado, pr_corredor);
    If (NP = producto) Then
        Bandera=1;
        Print "Datos";
        Print "El nombredel producto: ": pr_n_producto;
        Print "La cantidad mínima: ": pr_cant_minima;
        Print "La cantidad mínima si es de unidades: ": pr_und_c_minima;
        Print "La clase del producto: ": pr_clase;
        Print "Descripción: ":pr_descripcion;
        Print "Norma INEN: ": pr_inen;
        Print "Fecha de ingreso: ": pr_f_ingreso;
        Print "Fecha de salida: ": pr_f_salida;
        Print "Estado": pr_estado;
        Print "Corredor responsable: ": pr_corredor;
        Bandera = 1;
        pr_estado = "Bloqueado"
                            End If
```

```
    Wend
    If (Bandera = 0) Then
        Print "No existe producto alguno registrado con ese número.";
    Else
        Input "Está seguro que desea bloquear el producto(S/N): ": Respuesta
        If (Respuesta = "S") Then
            pr_estado = "Bloqueado";
            SQL_Insert(BNPA, bp_productos, producto, pr_n_producto,
                pr_cant_minima, pr_und_c_minima, pr_clase, pr_descripcion,
                pr_inen, pr_f_ingreso, pr_f_salida, pr_estado, pr_corredor);
        End If
End.
```

Algoritmo Inscripcion_Propuesta;

```
Begin
Repeat
    Bandera =0;
    Input "El nombredel producto: ":producto;
    Input "Corredor: ": corredor;
    Input "La cantidad: ": pt_cantidad;
    Input "El precio de Compra: ": pt_precio_compra;
```

```
Input "El precio de Venta: ":pt_precio_venta;
Input "El tipo o grado: ": pt_tipo_grado;
Input "Empaque: ": pt_empaque;
Input "Rueda: ": rueda;
Input "Fecha de vencimiento": pt_f_vencimiento;
pt_f_propuesta = Date( );
Input "Forma de pago: ": pt_forma_pago;
Input "Estado de la propuesta: ": pt_estado;
propuesta = propuesta + 1;
If (ningun campo es nulo) Then
    Repeat
        Print  "Digite Aceptar para Insertar en la Base de datos";
        Print "Digite Cancelar para no hacer la Inserción en la base de datos";
        Input Respuesta;
            If (Respuesta = Aceptar") Then
        SQL_Insert(BNPA, bp_propuesta, propuesta, producto, corredor, pt_
        cantidad, pt_precio_compra, pt_precio_venta, pt_tipo_grado,
        pt_empaque, comite, rueda, pt_f_vencimiento, pt_f_propuesta,
        pt_forma_pago, pt_estado);
        Bandera=1;
```

```
                    Else
                        Clear( );
                        Bandera = 1;
                    End If;
            Until (Bandera = 1);
Until (Bandera = 1);
End.
```

Algoritmo Aprobacion_Propuesta;

```
Begin
Bandera =0;
Input "El número de la propuesta: ": prop;
While (.NOT. EOF(BNPA, bp_propuesta) .AND. bandera = 0)
    SQL_Read(BNPA, bp_propuesta, propuesta, producto, corredor, pt_ cantidad,
            pt_precio_compra, pt_precio_venta, pt_tipo_grado, pt_empaque, comite,
            rueda, pt_f_vencimiento, pt_f_propuesta, pt_forma_pago, pt_estado);
    If (prop = propuesta) Then
        Bandera = 1;
        rueda = rueda +1;
        Input "Ingrese fecha de finalización de la rueda: ":ru_f_fin;
```

```
        ru_f_inicio = Date( );
        SQL_Insert(BNPA, bp_rueda, rueda, ru_f_inicio, ru_f_fin);
    End If;
Wend;
End.
```

Algoritmo Pizarra;

```
Begin
While (.NOT.EOF(BNPA, bp_propuesta))
        Print "Propuesta: ":propuesta
        Print "El nombredel producto: ":producto;
        Print "Corredor: ": corredor;
        Print "La cantidad: ": pt_cantidad;
        Print "El precio de Compra: ": pt_precio_compra;
        Print "El precio de Venta: ":pt_precio_venta;
        Print "El tipo o grado: ": pt_tipo_grado;
        Print "Empaque: ": pt_empaque;
        Print "Rueda: ": rueda;
        Print "Fecha de vencimiento": pt_f_vencimiento;
        Print "Fecha de propuesta"
```

```
        Print "Forma de pago: ": pt_forma_pago;
        Print "Estado de la propuesta: ": pt_estado;
        SQL_Read(BNPA, bp_rueda, rueda, ru_f_inicio, ru_f_fin);
        Print "Rueda: ": rueda;
        Print "Fecha de Inicio de la rueda: ":ru_f_inicio;
        Print "Fecha Final de la rueda: ":ru_f_fin;
Wend;
End.
```

4.6.4 Compromisos de Diseño

La forma que se realizo el diseño, es que los clientes accesan al Site Web, por medio de Applets, y para poderle dar una interfaz bastante amigable y que ningún nivel de usuarios tenga problemas con el acceso.

Nuestro principal compromiso de diseño es que no resida ningún tipo de datos en la parte del cliente, por que puede pasar, que los datos necesarios para la corrida del sistema se

pierdan deliberadamente, y porque en realidad no es necesario que residan en una maquina diferente del servidor.

Este a su vez esta diseñado, de manera que el usuario pueda accesar a la información con solo tener un browser y acceso a Internet desde cualquier parte remotamente.

Al poderse cubrir todas las especificaciones de requerimientos del sistema de la Bolsa Nacional de Productos Agropecuarios, este cubriría en el ámbito nacional, y posteriormente a gran escala.

CONCLUSIONES y RECOMENDACIONES

1. Siendo una herramienta y ambiente de desarrollo nuevos, la curva de aprendizaje se volvió un factor a tomar en consideración.
2. Java esta considerada como una herramienta poderosa en la implementación, ya que tiene un estándar visual lo suficientemente funcional y, esta orientado a objetos.
3. La tecnología vuelve el manejo de versiones mas transparente al Cliente, y más centralizada al Servidor
4. La arquitectura es justificable en un ambiente distribuido, que requiera una cobertura a gran escala, cuyos elementos son heterogéneos y se necesita un tiempo de respuesta real.
5. Los detalles de comunicación son transparentes para el programador con ORB's
6. Su aplicación es conveniente en situaciones en las que las herramientas tradicionales no ofrecen suficiente funcionalidad ni confiabilidad.
7. Un sistema diseñado basándose en esta arquitectura se ve con amplias perspectivas a futuro, debido al auge que esta teniendo Internet como un mecanismo para hacer negocios.
8. Se puede tomar nuestro sistema como un modelo para futuras simulaciones de negocios
9. No existen desarrolladores de aplicaciones en el mercado que le den un soporte, por lo que consideramos que DCOM tendría una ventaja en el mercado.
10. Apunta a una ambiente Cliente/Servidor "intergaláctico".

BIBLIOGRAFÍA

1. ORFALI R. & HARKEY D., Client / Server Programming with Java and Corba (1ra. Edicion; New York: John Wiley & Sons, Inc, 1996)

2. ORFALI R. & HARKEY D., Client / Server Programming with Java and Corba (2da. Edicion; New York: John Wiley & Sons, Inc, 1998)

3. ORFALI R., HARKEY D. & EDWARDS J., Essential Client / Server Survival Guide (2da. Edicion; New York: John Wiley & Sons, Inc, 1996)

4. R. HARKEY D. & EDWARDS J., Instant Corba (1ra. Edicion; New York: John Wiley & Sons, Inc, 1997)

5. ARNOLD K. y GOSLING J., El Lenguaje de Programacion Java (1ra. Edicion; Madrid: Addison-Wesley Iberoamerica, 1997)

6. JOSHI D. y VOROBIEV P., Migrating from Java 1.0 to Java 1.1 (1ra Edicion; New York: Ventana, 1997)

7. Visigenic: Programmers Guide Version 3.0, September 2, Visigenic Software Inc, 1997.

8. VARAS O. y DELGADO F., "La Bolsa de Productos, como medio de Comercializacion de Productos Agropecuarios" (Tesis, Facultad de Ciencias Administrativas, Universidad Laica Vicente Rocafuerte, 1991)

9. Primer Encuentro Continental de Bolsa de Productos Agropecuarios de América, Guayaquil, Julio 28-30, 1993, "Rueda de Enlace y Disposiciones Generales Relacionadas"(Guayaquil, 1993)pp. 40-47

10. GOYES B., "Indicadores Internacionales de Granos", Revista de Bolsa Nacional de Productos Agropecuarios, No. 5, (ago, 1997) pp. 6-7

1. DOUSDEBES T., "El Fondo de Comercializacion Agricola: Una Nueva Forma de Enfrentar el Problema de Mercadeo Agricola", Revista de la Bolsa de Productos, Volumen 2 No. 5, (abr-may, 1995) pp. 9