//

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES
CUAUTITLAN

SISTEMAS DE INFORMACION:

"EL DESARROLLO DE UN MIDDLEWARE PARA OPTIMIZAR TIEMPOS DE RESPUESTA EN LA EJECUCION DE OPERACIONES EN LOS SISTEMAS DE INFORMACION BANCARIOS."

TRABAJO DE SEMINARIO

QUE PARA OBTENER EL TITULO DE:

LICENCIADA EN INFORMATICA

P R E S E N T A :

LIDIA LAURA HUERTA VALENCIA

ASESOR: M.C.C. VALENTIN ROLDAN VAZQUEZ.

CUAUTITLAN IZCALLI, EDO. DE MEXICO. 2001.

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN
UNIDAD DE LA ADMINISTRACION ESCOLAR
DEPARTAMENTO DE EXAMENES PROFESIONALES

VNIVERSIDAD NACIONAL AVTENOMA DE MEXICO

DEPARTAMENTO DE EXAMENES PROFESIONALES

DR. JUAN ANTONIO MONTARAZ CRESPO
DIRECTOR DE LA FES CUAUTITLAN
P R E S E N T E

ATN. Q. Ma. del Carmen García Mijares
Jefe del Departamento de Exámenes
Profesionales de la FES Cuautitlán

Con base en el art. 51 del Reglamento de Exámenes Profesionales de la FES-Cuautitlán, nos permitimos comunicar a usted que revisamos el Trabajo de Seminario

Sistemas de Información: "El desarrollo de un Middleware para optimizar tiempos de respuesta en la ejecución de operaciones en los Sistemas de Información Bancarios."

que presenta la pasante: Lidia Laura Huerta Valencia

con número de cuenta: 9307723-9 para obtener el título de

Lincenciada en Informatica

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXÁMEN PROFESIONAL correspondiente, otorgamos nuestro VISTO BUENO

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
Cuautitlán Izcalli, Méx. a 6 de Septiembre de 2001

MODULO	PROFESOR	FIRMA
I	M.C.C. Araceli Nivón Zaghi	
II	M.C.C. Valentín Roldan Vazquez	
III	I N G. Miguel Alvarez Pasaye	

A Dios

Quien siempre ha estado conmigo en todo momento y me ha dado los elementos necesarios para vencer los obstáculos que se me han presentado y poder realizar hoy este sueño.
Gracias

A mi Familia

Quienes con su confianza, cariño, apoyo y dedicación siempre me motivaron a seguir adelante, a sobreponerme a las dificultades para lograr mis objetivos y hoy a alcanzar una meta más, por compartir tristezas y alegrías, éxitos y fracasos, por todo esto y mas. Gracias los amo.

A Juan Antonio Orozco Riveros

Quien siempre estuvo conmigo a lo largo de la carrera brindándome cariño y apoyo incondicional.
Gracias

INDICE

INDICE DE FIGURAS

INDICE DE TABLAS

OBJETIVO GENERAL:

- Permitir una comunicación directa entre el terminal financiero y la maquina central, evitando los encolamientos y reduciendo los tiempos de respuesta, utilizando los recursos de la maquina central.

OBJETIVOS PARTICULARES:

- Facilitar la interconectividad de los diferentes sistemas de un organismo.
- Administrar los canales de comunicación Cliente-Servidor.
- Evitar los encolamientos de peticiones de transacciones y reducir tiempos de respuesta.
- Proporcionar altos niveles de seguridad para una buena integridad y disponibilidad de la información.
- Contar con un canal directo entre el terminal financiero y la maquina central.

INTRODUCCIÓN

En los últimos años, las tendencias globalizadoras en el mundo han desarrollado en todas las empresas un creciente compromiso por la integración de los recursos informáticos en sus procesos productivos, los que deben dar solución a necesidades básicas como la respuesta de peticiones en tiempo real, la integridad de los datos y la disponibilidad inmediata de la información de manera eficiente.

Muchas de las empresas dedicadas al giro de banca comercial, han optado por centralizar la mayor parte de sus operaciones en servidores (back-end's), que atienden la información critica de la organización y explotan ésta a través de un software que ocupa la parte intermedia de la arquitectura cliente/servidor ó bien es el modulo intermedio que actúa como conductor entre dos módulos de software es denominado middleware, la comunicación comienza en el API del cliente y termina en el API del servidor, los cuales son robustos y se encargan de resolver los aspectos generales de comunicación de datos y de operaciones complejas que permiten el crecimiento de los aplicativos con poca cohesión, además de facilitar la creación de los clientes (frontend's) sencillos y amigables al usuario, los cuales pueden hacer uso de los distintos programas ubicados en la parte media del esquema general de operación. Estos Clientes (front-end's) generalmente residen en terminales.

Sin embargo, cuando el volumen de información crece, se hacen visibles las salvedades de la arquitectura mencionada, principalmente en lo que se refiere a la capacidad del sistema para atender la enorme cantidad de peticiones que recibe. Cuando la parte en la que el sistema pierde la capacidad de respuesta a las peticiones de terminales se localiza en el servidor central, difícilmente este esquema seguirá operando, pues la solución más conveniente será la de eliminar la carga repartiéndola a otros equipos capaces de garantizar la integridad de los procesos que efectúan, lo que eventualmente, se convertirá en un modelo de computo

distribuido. Cuando, por el contrario, este 'cuello de botella' es ocasionado por el middleware, la opción a seguir será de la optimizarlo para aprovechar al máximo las capacidades del equipo central.

En este caso se encuentran los Sistemas de Sucursales de Banco Santander su esquema depende completamente de Tuxedo, un middleware que controla el flujo de datos desde el terminal financiero (Cliente) hasta el sistema central (Servidor).

Este middleware, si bien era útil para controlar la integridad de los datos via two-face-commit, lo que es decir valida la trama cuando lo recibe el middleware y cuando lo entrega. Analizando estadísticamente el tipo de datos que viajaban a través del middleware resulto que más del 20% de la información total correspondía a operaciones cuyo control por parte de los monitores transaccionales del middleware no era necesario (particularmente consultas y la mayoría de las operaciones de captación), esto debido a que no reciben ningún tratamiento especial.

La transmisión de mensajes por medio de monitores transaccionales es ventajosa cuando estos, además de actuar como intermediarios entre el equipo central y las terminales, realizan operaciones de optimización que no pueden realizarse en los "clientes" y que serian costosas en tiempo y recursos para el equipo central. Es obvio que se vuelve una desventaja cuando un middleware solo se encarga de recibir el mensaje para retransmitirlo hacia otro punto, dado que los mensajes tienen que pasar por varios caminos antes de llegar a su destino final.

Por lo cual sé penso en un middleware que lograra enviar estas transacciones del terminal financiero por un canal directo hacia el equipo central los congestionamientos provocados por estos cuellos de botellas desaparecerían. Así mismo se tendría una mayor eficiencia en los flujos de comunicación disponibles para distintas aplicaciones que corren en la terminal. El manejo de la información del equipo central seria transparente.

En el capitulo I se da un panorama general del esquema cliente/servidor y sus conceptos básicos, en el capitulo II se abordara el tema del middleware su concepto, clasificación y sus tipos. Y en el capitulo III se hablara de las redes. Todo esto para poder conformar el marco teórico y poder así adentrarnos en el capitulo IV el caso practico.

Capitulo I. Arquitectura Cliente/Servidor

Debido a la gran cantidad de información que debe ser procesada y compartida entre empresas ubicadas en diferentes zonas geográficas, fue necesario diseñar sistemas que permitieran acceder a bases de datos remotas con mayor rapidez y eficacia. Lo anterior propició el desarrollo de métodos y tecnologías que permitieron llevar a cabo esta función. Una de estas tecnologías es la arquitectura Cliente/Servidor.

En la arquitectura Cliente/Servidor las aplicaciones de usuario corren en uno o más clientes (Computadoras Personales y/o Estaciones de Trabajo (WS)) mientras que el manejo de la información es controlado por el Administrador de la Base de Datos (Data Base Manager System (DBMS)) a si mismo la integridad de la información sigue siendo controlada por el servidor.

1.1 Base de Datos

Actualmente se reconoce que la información es uno de los recursos más importantes y significativos de cualquier empresa justificándose asi el uso de base de datos.

El uso de una base de datos permite obtener información necesaria para el manejo, control y administración de una empresa o institución

Las bases de datos proporcionan la infraestructura requerida por los sistemas de apoyo a la toma de decisiones y para los sistemas de información estratégicos. Por este motivo es importante conocer la forma que están estructuradas las bases de datos y su manejo.

1.1.1 Definición

Según Francisco Javier Rocandio Pablo.[1]

- "Una base de datos es el conjunto de datos relacionados entre sí, almacenados, estructurados, no redundantes y de fácil acceso"

- "Una base de datos es una serie datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular".

1.1.2 Ventajas en el uso de Bases de Datos

El enfoque de Base de Datos provee una solución sólida y simple al procesamiento de información, evitando muchos problemas en el manejo de la información, además permite:

- Globalización de la información.
- Controlar la redundancia
- Independencia de los datos. Es la capacidad que se tiene de modificar la definición de esquema en un nivel del sistema de base de datos sin que afecte la definición del esquema del nivel inmediato superior. En la independencia de datos se encuentran dos tipos que son:

- La independencia física de datos, es poder modificar el esquema interno (físico) sin tener que alterar el esquema conceptual, esto implica que al hacer alguna modificación al esquema interno no se tenga que volver a escribir los programas de aplicación que usan la base de datos. En ocasiones es necesario modificar el esquema interno para reorganizar ciertos archivos físicos o para mejorar el funcionamiento.

[1] Medios Informáticos, Explotación de Sistemas, México, McGraw Hill

- La independencia lógica de datos, es la capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos y no tener que realizar modificación alguna a los programas de aplicación que hacen uso de la base de datos. Las modificaciones que se hacen en el esquema conceptual , se da regularmente para ampliar o reducir la base de datos, es decir, cuando se altera la estructura lógica de la base de datos.

- Facilitar el uso de la información para el desarrollo de aplicaciones.
- Disponibilidad de los datos. Los datos almacenados en una estructura no posee un dueño único, cada usuario deja de ser propietario de los datos puesto que estos se comparten entre el conjunto de aplicaciones, de tal manera que se facilita la disponibilidad de los datos para todos aquellos que tienen necesidad de ellos.
- Proporcionar un control centralizado de los datos de operación de la empresa.
- Mayor coherencia de los resultados. El uso de las bases de datos implica que los datos que la constituyen sean recolectados y almacenados una sola vez, por esto cuando los usuarios manipulan los datos de la base utilizan los mismo datos, ocasionando así que los resultados de todos sean coherentes y comparables.
- Mejor y más normalizada documentación de la información.

Existe software que hace uso de ciertos modelos, metodologías y técnicas que ayudan a estructurar los datos además de ciertas herramientas que permiten manejar estas estructuras de datos. La herramienta que permite trabajar con los datos, en el Sistema Manejador de Base de Datos (DBMS).

Para acceder a la base de datos, el DBMS cuenta con un lenguaje Manipulador de Datos (DML); este lenguaje proporciona las instrucciones mínimas para poder acceder los datos, entre ellas se encuentran aquellas que nos permiten:

- Abrir y cerrar la Base de datos,
- Consultar y actualizar los datos,
- Iniciar y terminar transacciones,
- Controlar concurrencia, etc.

En el ambiente de Base de Datos podemos identificar dos importantes entes. los cuales están íntimamente relacionados y son conceptualmente distintos, ellos son: la base de datos y el Sistema Manejador de Base de datos (DBMS)

La Base de Datos es un modelo representado por un conjunto de datos, los cuales tiene las siguientes propiedades:

- Están interrelacionados
- Son capaces de evolucionar,
- Son accesibles a múltiples aplicaciones, y
- Su redundancia es mínima y controlada.
- La actualización y recuperación de datos debe asegurar Integridad, Seguridad y Confidencialidad.

El que estén interrelacionados significa que se puede tomar cualquier pieza de información y cruzarla con cualquier otra pieza de información de la base de datos.

El ser capaces de evolucionar significa que en cualquier momento pueden modificar las características de los datos, su estructura o bien agregar nuevas piezas de información, siendo este cambio transparente para los usuarios.

Al ser accesibles por múltiples aplicaciones tendremos en la misma base de datos la información de toda la empresa y esta información será accedida por todas las aplicaciones que se desarrollan.

La redundancia de datos se refiere a la duplicidad de información Se trata que cada pieza de la información solo sea ingresado en la base de datos solo una vez.

1.1.3 Sistema Manejador de Base de Datos (DBMS)

El **Sistema Manejador de Base de Datos (DBMS).** Es un conjunto de rutinas que nos permite definir, crear, acceder, respaldar, recuperar y administrar la base de datos, garantizando la seguridad, integridad y protección de lo datos, así como sincronizando el acceso de múltiples aplicaciones. Esto también es lo que impone cierta disciplina, tanto para moldear la base de datos como para accesarla.

Por seguridad se entiende que solo podrán acceder a la base de datos los usuarios autorizados y solo verán la parte que puedan accesar.

La integridad nos dice que los valores almacenados son correctos y que la base de datos es consistente.

La protección asegura que en caso de corrupción de la base de datos, es posible recuperar la los datos correctos.

- La sincronización controla que varias aplicaciones accedan en forma concurrente a la base de datos.

1.1.4 Objetivos del DBMS

El objetivo primordial de un DBMS es crear un ambiente en que sea posible guardar y recuperar información de la base de datos en forma conveniente y eficiente.

- Minimizar la redundancia de los datos. No tener datos repetidos.
- Garantizar la consistencia de los datos. Obtener la misma información por peticiones similares en un momento dado.
- Integridad de los datos. Reglas dictadas por política o normas de la empresa y que los datos deben de cumplir.
- Seguridad de los datos. Protección de los datos contra accesos, modificaciones o pérdidas, ya sea en forma intencional o no intencional.
- Controlar la concurrencia. Múltiples usuarios pueden accesar a la misma información al mismo tiempo, sin que con ello existan problemas con los datos.
- Proteger los datos los datos contra fallas del sistema. Es la capacidad de restaurar la integridad y consistencia después de una falla del sistema.
- El diccionario de datos. Es la capacidad que da el manejador de la base de poder tener la descripción de los datos que están almacenados en la base de datos.
- Interfaz de alto nivel con los programadores. Manejo de lenguajes de cuarta generación, como es el SQL (Structured Query Lenguage)

1.1.5 Modelos de Bases de Datos

Los bases de datos que existen son:

- Modelo de Red
- Modelo Jerárquico
- Modelo Relacional

Modelo de Red

Representa al mundo real como registros lógicos que representan a una entidad y que se relacionan entre si por medio de flechas. Una base de datos en red se construye a partir de registros que se conectan entre si por medio de ligas. Cada registro es un conjunto de campos (atributos). Un campo almacena el valor de un solo dato y una liga puede asociar exclusivamente dos registros.

Este modelo tiene un lenguaje de registro por registro asociado que se debe incorporar en un lenguaje de programación anfitrión.

En el modelo de red hay dos estructuras de datos fundamentales que son los tipos de registros y los conjuntos. Los tipos de registros son una colección de elementos de datos lógicamente relacionados y un conjunto expresa una relación uno a muchos entre dos tipos de registros.

Desventajas del modelo de red son:

- Resulta difícil definir nuevas relaciones.
- Es complejo darle mantenimiento ya que cualquier cambio en la estructura requiere una descarga de los datos.
- Representa desperdicio de recursos.

- Anomalías de inserción y borrado.

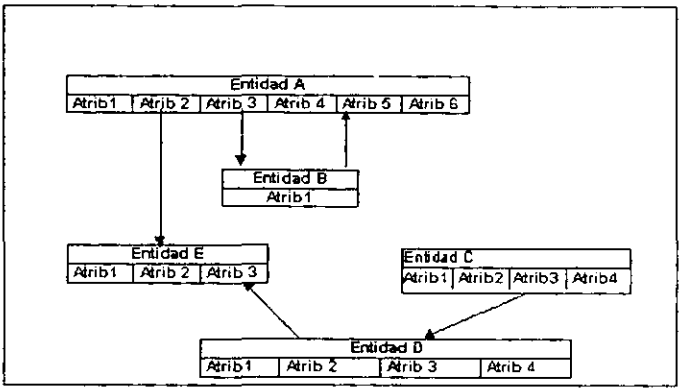


Figura # 1. Modelo de Red

Modelo Jerárquico

El modelo de datos jerárquico es muy similar al modelo de datos de red. Una base de datos jerárquica se compone de un conjunto de registros conectados entre si por medio de ligas. Un registro igual que en modelo de red, es un conjunto de datos o atributos que almacenan cada uno un valor. Una liga es una asociación entre dos registros. Los registros en el modelo de datos jerárquico se organizan formando conjuntos de árboles a diferencia del modelo de red en donde se forman arbitrarias.

Este modelo representa los datos como estructuras jerárquicas de árbol y cada jerarquía representa varios registros relacionados entre si, tiene forma de árbol invertido, en donde un padre puede tener varios hijos pero cada hijo solo tener un padre. En este modelo a las entidades se les denomina segmento el cual es un conjunto de datos homogéneos y se encuentra dividido en segmento raíz y segmento dependiente, atendiendo a su posición en la estructura jerárquica, y entre segmento tipo el cual define a una entidad y el segmento ocurrencia que define a un conjunto de datos de dicha Entidad.

Para el modelo jerárquico no existe un lenguaje estándar, aunque la mayoría de los sistemas gestores de base de datos jerárquicos cuentan con lenguajes de registro a registro.

Un diagrama de estructura de árbol es el esquema de una base de datos jerárquica, utiliza cuadros para representar tipos de registros y líneas que representan ligas. Las ligas conectan dos tipos de registros. Cuando la línea termina en flecha se indica que es una relación de uno a muchos, si la liga tiene flechas en ambos extremos de la línea se indica una relación uno a uno.

Desventajas del modelo jerárquico son:

- No se permite más de una relación entre dos segmentos.
- No se admiten relaciones reflexivas
- No se permite un segmento hijo con más de un segmento padre.
- El árbol se recorre en un cierto orden (Preorden).
- Es obligatorio entrar por el segmento

Modelo Relacional

La visión relacional corresponde al almacenamiento en forma de tablas, aquí los datos se representan en forma de tablas llamadas relaciones. A las columnas de una tabla se les conoce como atributos y a los renglones o filas de la relación se les conoce como tuplas. El modelo representa al mundo real mediante tablas relacionadas entre si por columnas comunes y se ocupa de tres aspectos de los datos: su estructura, su integridad y su manipulación.

Una relación debe cumplir con ciertas características. Las celdas de la tabla deben tener un solo valor, no se permiten arreglos como valores. Todos los valores de cualquier columna deben ser del mismo tipo. Cada columna tiene un nombre único, es decir en una misma tabla no pueden existir dos columnas con el mismo nombre, además el orden de las columnas es insignificante. Dos tuplas de una relación no pueden ser idénticos y el orden de las tuplas es insignificante.

Casi todas la bases de datos relacionales tiene lenguajes de alto nivel.

1.1.6 Normalización

El proceso de normalización se encarga de seguir una serie de pasos o normas que tras aplicar todas ellas, se obtienen los datos agrupados en diferentes tablas, de tal forma que es la estructura optima para su implementación, gestión y explotación desde diferentes futuras aplicaciones. Una tabla se dice que esta en una forma normal cuando satisface un conjunto de restricciones impuestas por dicha norma.

Definición

Los conceptos básicos de normalización permiten que los usuarios puedan contribuir a diseñar una aplicación que van a usar posteriormente o para comprender una aplicación que ya ha sido construida. Además la normalización también es usada en la información de un negocios, y en como están relacionados entre si los diferentes elementos de información.

La normalización se basa en que los datos son independientes de las aplicaciones que las gestionan, y su objetivo es obtener el mayor número de tablas posibles, dejando en cada una de ellas los atributos imprescindibles para representar a la entidad (objeto), o a la relación entre entidades a la que hace referencia la tabla mediante la conexión de sus claves.

Por medio de la normalización, un conjunto de datos en un registro se reemplaza por varios registros que son más simples y predecibles y, por lo tanto, más manejables. La normalización se lleva a cabo por cuatro razones:

- Estructurar los datos de forma que se puedan representar las relaciones pertinentes entre los datos.
- Permitir la recuperación más sencilla de los datos en respuesta a las solicitudes de consultas y reportes.
- Simplificar el mantenimiento de los datos actualizados, insertándolos o borrándolos.
- Reducir la necesidad de reestructurar o reorganizar los datos cuando surjan nuevas necesidades.

Primera Forma Normal

Una de las mejoras básicas que el analista puede hacer es diseñar la estructura de un registro de manera que todos los registros de un archivo tenga la misma longitud. Los registros de longitud variable crean problemas especiales, ya que el sistema debe verificar siempre en donde se encuentran los extremos de un registro. Al fijar la longitud del registro se elimina este problema.

La primera forma normal se alcanza cuando se quitan todos los grupos de repetición, de la forma que un registro tenga longitud fija. Un grupo de repetición, es decir, la aparición repetida de un dato o grupo de datos dentro de un registro, es en realidad otra relación. Por lo tanto, se quita el registro y se le considera como una parte del mismo o como una relación adicional. La parte del registro de los datos que se repite se denomina grupo de repetición.

La parte del registro de los datos que se repite se denomina grupo de repetición

La primera forma normal se alcanza cuando un registro se diseña de longitud fija. Esto se lleva a cabo quitando el grupo de repetición y creando un archivo o relación aparte que contenga el grupo de repetición. El registro original y el nuevo se interrelaciona mediante un punto común de los datos.

Segunda Forma Normal

La segunda forma normal se alcanza cuando un registro esta en la primera forma normal y cada campo depende totalmente de la llave del registro (en al almacenamiento y recuperación). En otras palabras, el analista busca la dependencia funcional: un campo es funcionalmente dependiente si su valor esta asociado de manera única con un campo especifico.

Tercera Forma Normal

La tercera forma normal se alcanza cuando se quitan las dependencias transitivas de un diseño de registro.

El caso general es el siguiente

- A,B, y C son tres datos en un registro
- Si C es funcionalmente dependiente de B y
- B es funcionalmente dependiente de A
- Entonces C es funcionalmente dependiente de A.
- Por lo tanto, existe una dependencia transitiva.

En el manejo de datos, la dependencia transitiva es una preocupación, ya que los datos pueden perderse de manera inadvertida cuando la relación esta oculta

1.2 Concepto Cliente/Servidor

La arquitectura Cliente/Servidor se creó básicamente para un gran número de estaciones de trabajo, computadores personales, servidores de archivos, impresoras y otros equipos que están interconectados a través de una red. Este tipo de arquitectura tiene como idea principal definir servidores especializados con funciones especificas en donde los recursos que proveen dichos servidores estén a disposición de muchos clientes.

El termino Cliente/Servidor describe un sistema en el que una maquina cliente solicita aun segunda maquina llamada servidor que ejecute una tarea especifica. En donde todo el sistema de administración de bases de datos (DBMS) es un servidor, exceptuando las interfaces de consulta que interactuan con el usuario. El cliente suele ser una computadora personal conectada a una LAN, y el servidor es por lo general, una maquina anfitriona, como un servidor de archivos PC, un servidor de archivos UNIX o una macrocomputadora o computadora de rango medio.

En la estructura Cliente/Servidor, los servidores ponen a disposición de sus clientes recursos, servicios y aplicaciones. Dependiendo de qué recursos ofrece el servidor y cuáles se mantienen en los clientes, se pueden hacer distinciones entre distintas estructuras Cliente/Servidor. En estas estructuras se diferencia:

Dónde se encuentran los datos.

Dónde se encuentran los programas de aplicación y

Dónde se presentan los datos

La arquitectura Cliente/Servidor es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que solicita el dialogo o los recursos y servidor al proceso que responde a las solicitudes. Los principales componentes del esquema Cliente/Servidor son entonces el cliente, los servidores y la infraestructura de comunicaciones.

En este modelo, las aplicaciones se dividen de forma que el servidor contiene la parte debe ser compartida por varios usuarios y en el cliente permanece solo lo particular de cada usuario.

La interacción que existe entre el cliente y el servidor durante el procesamiento de una consulta se puede llevar a cabo como lo muestra la figura.

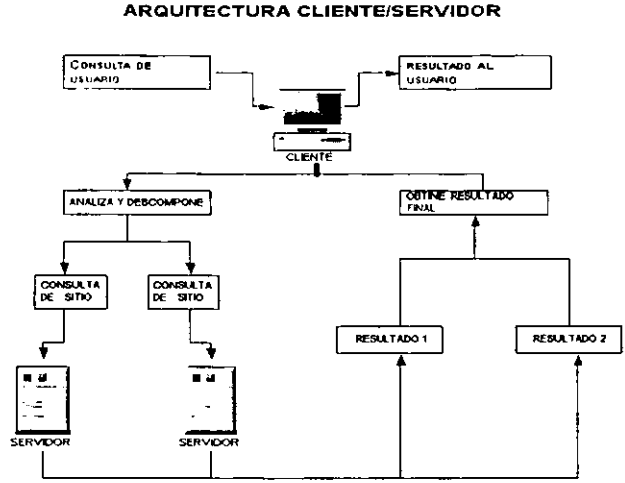


Figura # 2 Arquitectura Cliente/Servidor

En donde:

- El cliente analiza sintácticamente la consulta del usuario y la descompone en varias consultas de sitio independientes.
- Cada consulta de sitio se envía al sitio del servidor apropiado.
- Cada servidor procesa la consulta local y envía el resultado al sitio cliente.
- El sitio cliente combina los resultados de las subconcultas para producir el resultado de la consulta original y mostrarla al usuario.

Clientes y Servidores son entidades lógicas independientes que operan en conjunto a través de una red para realizar una tarea.

Existen arquitecturas cliente/servidor en diferentes planos, normalmente se utilizan a 2 planos y en 3 planos aunque la idea básica es la misma. Todo se reduce al modo en el que la aplicación cliente/servidor se divide en unidades funcionales que más tarde puedan asignarse ya sea al cliente o a uno o más servidores. Las

unidades funcionales más comunes son la interfaz del usuario, la lógica de negocios y los datos compartidos.

Existen muchas variantes posibles de arquitecturas en planos múltiples, dependiendo de la forma en que se divida la aplicación y del middleware que se emplee para la comunicación entre los planos.

En el caso de los sistemas de cliente/servidor en 2 planos, la lógica de la aplicación esta integrada ya sea a la interfaz del usuario en el cliente o a la base de datos en el servidor (o ambas). En el caso de los sistemas de cliente/servidor en 3 planos, la lógica (o proceso) de la aplicación ocupa el plano intermedio; está separada tanto de los datos como de la interfaz del usuario. En teoria, los sistemas de cliente/servidor en 3 planos son de más fácil ampliación y más robustos y flexibles. Además, pueden integrar datos de múltiples fuentes.

1.2.1 Características De Cliente/Servidor

Todos los sistemas de clientes de clientes/servidores poseen las siguientes características distintivas:

- Servicio: El modelo cliente/servidor es fundamentalmente una relación entre procesos ejecutados en aparatos distintos. El proceso del servidor hace de éste un proveedor de servicios. El cliente es un consumidor de servicios. En esencia, cliente/servidor aporta una clara distinción de funciones con base en la idea del servicio.
- Recursos compartidos: un servidor puede atender a muchos al mismo tiempo y regula su acceso a recursos compartidos.
- Protocolo asimétricos: entre clientes y servidor se establece una relación de "muchos a uno". Son siempre los clientes los que inician el diálogo al solicitar un servicio. Los servidores aguardan pasivamente las solicitudes de los clientes.

- Transparencia de ubicación: el servidor es un proceso que puede residir en el mismo aparato que el cliente o en un aparato distinto a lo largo de una red. El software de cliente/servidor suele ocultarles a los clientes la ubicación del servidor mediante el redireccionamiento de las llamadas de servicio en caso necesario. Un programa puede ser un cliente, un servidor o ambos.
- Mezcla e igualdad: el software ideal de cliente/servidor es independiente del hardware o de las plataformas de software del sistema operativo. Usted debe estar en condiciones de mezclar e igualar plataformas de cliente y de servidor.
- Intercambios basados en mensajes: clientes y servidores son sistemas holgadamente acoplados que interactuan a través de un mecanismo de transmisión de mensajes. El mensaje es el mecanismo de entrega para las solicitudes y respuestas de servicio.
- Encapsulamiento de servicio: el servidor es un "especialista". Un mensaje le indica a un servidor que servicio se solicita; éste se le envía luego al servidor para determinar el cumplimiento de la tarea. Los servidores pueden ser sustituidos sin afectar a los clientes, siempre y cuando la interfaz para la publicación del mensaje no cambie.
- Facilidad de escalabilidad: los sistemas de cliente/servidor pueden escalarse horizontal o verticalmente. La escalabilidad horizontal significa la adición o eliminación de estaciones de trabajo del cliente con apenas un ligero impacto en el desempeño. La escalabilidad vertical significa migrar a un aparato servidor más grande y más veloz o a servidores múltiples.
- Integridad: el código del servidor y los datos del servidor se conservan centralmente, lo que resulta en un mantenimiento de menor costo y en la proyección de la integridad de los datos compartidos. Al mismo tiempo, los clientes mantienen su individualidad e independencia.

En conclusión la arquitectura Cliente/Servidor es una tecnología utilizada para el procesamiento de datos: el cliente es la máquina solicitante y el servidor es la máquina proveedora, y debe existir un *software* especializado para controlar la comunicación.

1.2.2 Elementos de la arquitectura Cliente/Servidor

En la arquitectura Cliente/Servidor existen tres elementos: el Cliente (Front End), el Servidor (Back End) y el Middleware (Conectividad) que los enlaza:

El elemento del cliente ejecuta la parte del cliente de la aplicación, este interactua con el usuario. Corre en un sistema operativo (OS: operating system) que ofrece una interfaz gráfica de usuario (GUI: graphical user interface) o una interfaz de usuario orientada a objetos (OOUI: object oriented user interface) y que puede acceder a servicios distribuidos, dondequiera que se encuentren. Lo más común es que el sistema operativo descargue en el elemento de middleware le responsabilidad de manejar los servicios no locales. El cliente también ejecuta un componente del factor de administración de sistemas distribuidos (DSM: distributed system management). Este podría ser desde un agente simple en una PC administrada hasta la terminal de plano frontal completa de la aplicación de DSM en una estación de administración.

El programa cliente cumple dos funciones distintas:

- Por un lado gestiona la comunicación con el servidor, solicita un servicio y recibe los datos enviados por aquel.
- Por otro maneja la interfaz con el usuario, presenta los datos en el formato adecuado y brinda las herramientas y comandos necesarios para que el usuario pueda utilizar las prestaciones del servidor de forma sencilla.

Los clientes realizan generalmente funciones como:

- Manejo de la interface con el usuario
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos.

El servidor sólo tiene que encargarse de transmitir la información de forma eficiente. No tiene que atender al usuario. De esta forma un mismo servidor puede atender a varios clientes al mismo tiempo.

El elemento del servidor ejecuta la parte del servidor de la aplicación. La aplicación del servidor suele correr sobre un paquete comercial de software del servidor. Las cinco plataformas del servidor que compiten por la creación de la próxima generación de aplicaciones cliente/servidor son los servidores de bases de datos de SQL, los monitores de TP, los servidores de groupware, los servidores de objetos y el Web. La parte del servidor depende del sistema operativo para la interfaz con el elemento de middleware que conduce las solicitudes de servicio. También el servidor ejecuta un componente de DSM. Este podria ser desde un agente libre en una PC administrada hasta la terminal trasera completa de la aplicación de DSM (podría proporcionar, por ejemplo, una base de datos de objetos compartidos para el almacenamiento de información de administración del sistema).

Los servidores realizan, entre otras, las siguientes funciones:

- Gestión de periféricos compartidos .
- Control de accesos concurrentes a base de datos compartidas.
- Enlaces de comunicaciones con otras redes de área local o extensa.
- Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y este le responde proporcionándoselo.

La conectividad Cliente/Servidor debe de proporcionar al *front end* un esquema sencillo que permita la conexión transparente a los diferente DBMS en las diferentes plataformas que soporten.

La aplicación de middleware corre tanto en la parte del cliente como en la del servidor de la aplicación. Dividimos este elemento en tres categorías: pilas de transporte, sistemas operativos de redes (NOS: network operating system) y middleware de servicios específicos. El middleware es el sistema nervioso de la infraestructura de cliente/servidor. Al igual que los otros dos elementos, también el middleware cuenta con un componente del software de DSM.

Cualquiera de los componentes debe de cumplir cuando menos algunas de las características básicas como procesamiento local, independencia del back end, independencia de protocolos de comunicación, independencia del servidor.

1.3 Niveles de la arquitectura Cliente/Servidor

La arquitectura se puede clasificar en cinco niveles, según las funciones que asumen el cliente y el servidor:

- Primer nivel: aquí el cliente asume parte de las funciones de representación de la aplicación, ya que siguen existiendo programas en el servidor, dedicados a esta tarea.
- Segundo nivel: en este la aplicación está soportada directamente por el servidor, excepto la presentación que es totalmente remota y reside en el cliente.
- Tercer nivel: la lógica de los procesos se divide entre los distintos componentes del cliente y del servidor.
- Cuarto nivel: el cliente realiza tanto las funciones de presentación como los procesos. Por su parte el servidor almacena y gestiona los datos que permanecen en una base de datos centralizada.
- Quinto nivel: aquí el reparto de tareas es como en el anterior y además el gestor de base de datos divide sus componentes entre el cliente y el servidor. Las interfaces entre ambos, están dentro de las funciones del gestor de datos y, por lo tanto, no tiene impacto en el desarrollo de las aplicaciones.

Capitulo II. Middleware

2.1 Concepto

Middleware es el software que ocupa la parte intermedia de la arquitectura cliente/servidor, es el modulo intermedio que actúa como conductor entre dos módulos de software, para compartir datos, los módulos de software no necesitan saber cómo comunicarse entre ellos, sino cómo comunicarse con el módulo de middleware, Además es el enlace que permite que un cliente obtenga un servicio de un servidor. Empieza en el módulo de API de la parte del cliente que se emplea para invocar un servicio y comprende la transmisión de la solicitud por la red y la respuesta resultante.

Figura # 3 Middleware

Pero no incluye al software que presta el servicio real; esto pertenece a los dominios del servidor. Tampoco a la interfaz del usuario ni a la lógica de la aplicación, en los dominios del cliente.

Es un termino que abarca a todo el software distribuido necesario para el soporte de interacciones entre Clientes y Servidores.

El middleware debe ser capaz de traducir la información de una aplicación y pasársela a otra. Además de permitir también independizar los procesos cliente y servidor.

El concepto de middleware permite también independizar los servicios proporcionados por diferentes objetos que se encuentran en una red proporcionando una red de objetos independientes e interconectados entre sí.

La utilización de middleware permite desarrollar aplicaciones en arquitectura cliente servidor independizando los servidores y clientes, facilitando la interrelación entre ellos y evitando dependencias de tecnologías propietarias.

El middleware proveerá los niveles de seguridad que se necesitan para mantener unos altos estándares de integridad de la información y una completa seguridad de que la información está siendo utilizada por la persona adecuada en la tarea adecuada. También garantizará que los planes de contingencia que se tengan sean viables y que se cuente con la infraestructura necesaria para colocarlos en práctica oportunamente.

El middleware debe estar en capacidad de correr en diferentes plataformas, crecer según las necesidades de la empresa y permitir la completa integración entre los diferentes niveles de computación y las herramientas que sean utilizadas; del mismo modo cumplir con las funciones de un monitor de transacciones.

El middleware produce una apariencia común para todos los servicios de la red, conocida como "imagen de sistema único".

Las soluciones que requieren de un middleware son aplicaciones que corren en forma distribuida, en múltiples y heterogéneos nodos, que accesan múltiples y heterogéneas bases de datos.

2.2 Características del Middleware

Las principales características del Middleware son las siguientes

- Balancear las cargas de trabajo entre los elementos de computación disponibles.
- Manejo de mensajes, que le permite entrar en el modo convencional de un esquema Cliente/Servidor y en general de cualquier forma de paso de mensajes entre procesos.
- Manejo de la consistencia entre los diferentes manejos de bases de datos principalmente en los procesos de OLTP.
- El middleware permite independizar los procesos cliente y servidor.
- Simplifica el proceso de desarrollo de aplicaciones.
- Es el encargado del acceso a los datos: acepta las consultas y datos recuperados directamente de la aplicación y los transmite por la red. También es responsable de enviar de vuelta a la aplicación, los datos de interés y de la generación de códigos de error.
- El middleware produce una apariencia común para todos los servicios de la red, conocida como "imagen de sistema único".
- El middleware es una herramienta adecuada de solución, ya que no sólo es flexible y segura, además permite manejar diferentes ambientes de computación.
- Administración Global, como una sola unidad computacional lógica.
- Administración de la alta disponibilidad de la solución.

2.2.1 Ventajas del Middleware

Entre las principales ventajas del Middleware se encuentran las siguientes:

- Simplifica el proceso de desarrollo de aplicaciones al independizar los entornos propietarios.
- Permite la interconectividad de los diferentes Sistemas de Información.
- Proporciona mayor control del negocio al poder contar con información procedente de distintas plataformas sobre el mismo soporte.
- Facilita el desarrollo de sistemas complejos con diferentes tecnologías y arquitecturas.

Dentro de los inconvenientes más importantes destacan la mayor carga de máquina necesaria para que puedan funcionar.

2.3 Clasificación del Middleware

El middleware es la capa de software entre la lógica de la aplicación del usuario final y los niveles más bajos de tecnología. Existen diferentes modalidades de middleware según su función y la forma de operar.

2.3.1 Middleware General

Es el sustrato de la mayoría de las interacciones de cliente/servidor. Incluye las pilas de comunicación, directorios distribuidos, servicios de autenticación, horario de la red, llamadas a procedimientos remoto y servicios en cola, esta categoría incluye también los sistema operativos de redes NOS, como los servicios distribuidos de archivos e impresiones.

Entre los productos que pertenecen a la categoría del middleware general están: NOS, DCE, NetWare, Named Pipes, LAN Server, LAN Manager, Vines, TCP/IP; APPC y NetBIOS. También debemos incluir a los productos de middleware orientados a mensajes conocido como MOM (menssage- oriented middleware).

2.3.2 Middleware De Servicios Específicos.

Es necesario para cumplir un tipo particular de servicio de cliente/servidor. Pertenece a esta categoría:

- El middleware para bases de datos, como PDBC, DRDA, EDA/SQL, SAG/CLI y Oracle Glue.
- El middleware para OLTP, como ATMI y WS de Tuxedo Transacional, RPC de Encina y TxRPC y XATMI de X/Open.
- El middleware para groupware, como MAPI, VIM, VIC, SMTP y las llamadas de Lotus Notes.
- El middleware para objetos, como CORBA de OMG y Nework OLE (o DCOM) de Microsoft.
- El middleware para Internet, como HTTP, S-HTTP y SSL.
- El middleware para la administración de sistemas, como SNMP, CMIP y ORB.

Estos generalmente trabajan orientados a mensajes

2.4 Tipos Middleware General

2.4.1 Sistemas Operativo para Redes (NOS)

El middleware de un sistema operativo para redes (NOS network operating system) debe ofrecer para crear una "imagen de sistema único" de todos los servicios de la red.

Los NOS se han convertido en entornos reales de computación distribuida por intermedio de los cuales la red de vuelve transparente para los usuarios.

Se enlistan algunos de los tipos de transparencia que se esperan del NOS como parte de su "acto de desaparición de la red".:

- Transparencia de ubicación. Debe de conocer la ubicación de un recurso. Los usuarios no deben verse obligados a incluir la información de ubicación en el nombre del recurso.
- Transparencia de espacio por nombres. Debe de estar en posibilidad de emplear las mismas convenciones de nombramiento (y espacio para nombres) para localizar cualquier recurso en la red.
- Transparencia de acceso. Debe disponer de la facilidad de dar una sola contraseña (o autenticación) que funcione para todos los servidores y todos los servicios de la red.
- Transparencia de duplicación. No debe interesar cuántas copias existan de un recurso.
- Transparencia de acceso local/remoto. Debe de estar en condiciones de trabajar con cualquier recurso de la red como si éste se encontrara en la máquina local. El NOS debe manejar los controles de acceso y proporcionar servicios de directorio.
- Transparencia de tiempo definido. No debe de percibir diferencias de horarios entre servidores. El NOS debe sincronizar los relojes de todos los servidores.

- Transparencia de fallas. Usted debe de estar protegido contra fallas de la red. El NOS debe de manejar los reprocesamientos y reconexiones de sesión. También debe ofrecer ciertos niveles de redundancia de servicios para la tolerancia de fallas.
- Transparencia de administración. Solo debe vérselas con una interfaz de administración de sistema único.

Una de las funciones de un NOS es volver transparente para una aplicación la ubicación física de los recursos (a lo largo de una red). Los NOS permiten que clientes que operan con OS diferentes (como DOS, Mac y Unix) compartan archivos y otros dispositivos.

Servicios De Directorio Global.

El estado de un sistema de cliente/servidores de flujo permanente. Los usuarios entran y salen de la red. Pueden añadirse servicios, y moverse a voluntad. La creación y circulación de datos es incesante. La imagen de sistema único radica en el servicio de directorios del NOS.

En un NOS, el servicio de directorio se implementa como una base de datos de objetos distribuida y aplicada. Es distribuida para permitir que diferente dominios de administración controlen su entorno. Es duplicada para brindar alta disponibilidad y desempeño donde se necesite. Recuerde que si el directorio esta fuera de servicio, toda la actividad de la red se detendrá. Es una base de datos de objetos en el sentido de que todo lo que se registra es una instancia de una clase de objeto. Usted puede emplear la herencia para nuevos tipos de objetos.

Los directorios de NOS cuentan con API e interfaces del usuario que permiten que los programas (o personas) localicen entidades en la red consultando su nombre o atributos.

El NOS usa parte de la tecnología de bases de datos y objetos distribuidos más avanzada para el rastreo de los recursos del sistema distribuido. La tecnología es tan flexible que maneja y registra las toscas entidades actuales de red, como impresoras, usuarios programas y servidores y las entidades de grano más fino que comienzan a aparecer- como los objetos distribuidos.

Servicios de tiempo distribuido.

Mantener una sola noción del tiempo es importante para ordenar eventos que ocurren en clientes y servidores distribuidos, los servios de tiempo distribuidos del NOS, mantiene sincronizados un sistema de cliente/servidor.

- Sincronizando periódicamente los relojes de cada máquina de la red. El NOS suele contar con un agente en cada máquina - llamado relojero en DCE-, el cual consulta a los servidores de tiempo para conocer la hora correcta y ajustar la hora local. Los agentes pueden consultar a más de un servidor de tiempo, y calcular después la hora correcta probable y su imprecisión con base en la respuestas que reciben. El agente puede ajustarla hora local gradual o abruptamente.
- Introduciendo un componente de imprecisión para compensar las desigualdades derivadas de reloj que ocurren entre sincronizadores. Los agentes de hora local están configurados para reconocer los límites de sus reloj de hardware local. Llevan la cuenta del factor de imprecisión y la reporta a través de una llamada de API para consultar la hora. El agente de tiempo solicita una sincronización después de que las derivas del reloj local rebasan cierto umbral de imprecisión.

Manejo De Mensajes De Igual A Igual (KPC)

El propósito de NOS es volver transparente la computación distribuida. Esto significa que debe de crear un entono que oculte el fastidio de tratar con protocolos, redes y pilas de comunicaciones.

Los NOS ofrecen interfaces de igual a igual que permiten que las aplicaciones se comuniquen mediante una semántica de emisión/recepción semejante a la telegráfica. Todos ellos cuentan con alguna modalidad de middleware de llamada a procedimiento remoto (RCP: remote procedure call), la cual oculta el "alambre" para dar la apariencia de que todos los servidores de la red son una función por llamar.

Un modelo alternativo son las colas de mensajes, o middleware orientado a mensajes (MOM: message-oriented middelware). Los NOS no cuentan con un MOM entre sus caracteristicas.

2.4.2 PILA

Como todos los demás elementos de la red, las API de RPC, MOM igual a igual se instalan sobre pilas de comunicaciones. Se ha atacado el problema de la complejidad de las redes dividiendo en capas los protocolos complejos. Cada capa se tiende sobre los servicios provistos por las capas debajo de ella. Ello produce al final una pila de capas semejante a un pastel. En teoría, cada capa de pila cuenta con un conjunto definido de API y protocolos.

La capa inferior del software de comunicación pertenece a los controladores de dispositivos, los cuales provén una interfaz con varios tipos de adaptadores de hardware de comunicaciones. Las pilas se asientan sobre los controladores de dispositivos. Su parte superior queda inmediatamente debajo del NOS. Pilas y NOS son sencillamente divisiones que nos ayudan a crear un mapa.

Las pilas en las capas inferiores, actúan como interfaces con el hardware mediante los protocolos de control de acceso a medios (MAC; media access control) físicos, definido por el IEEE. El control de vinculación lógica (LLC; logical link control) ofrece una interfaz común y un servicio de vinculación confiable para la transmisión de paquetes de comunicación entre dos nodos.

2.4.3 Sóckets

Los sóckets aparecieron en 1981 como la interfaz genérica de Unix BSD 4.2 que ofrecería comunicaciones de Unix a Unix en redes. En 1985, de OS de Sun introdujo NFS y RPC sobre sóckets. En 1986, AT&T lanzó la interfaz de capa de transporte (TLI; transport layer interface), con funcionalidad similar a la de los sóckets pero con mayor independencia de la red. Unix integra tanto sóckets como TLI. Los sóckets TLI son muy parecidos, TLI es simplemente una versión más sencilla de los sóckets . en teoría , una aplicación generada para TLI es independiente de las pilas. Debería correr en IPX/SPX o TCP/IP con muy pocas modificaciones.

Prácticamente todos los sistemas operativos soportan sóckets. La API de sóckets de Windows, conocida coloquialmente como WinSock, es una especificación de proveedor múltiple que estandariza el uso de TCP/IP en Windows. La API WinSock se basa en la interfaz de sóckets de Berkeley.

Los tres tipos más comunes de sócket son los de flujo, de datagramas y en bruto. Los sóckets de flujo y de datagramas sirven de interfaz con los protocolos TCP y UDP, mientras que los sóckets en bruto son interfaz con los protocolos IP.

El tipo de sockets se especifica al momento de sus creación. En teoría, la interfaz del sócket puede ampliarse, y es posible definir nuevos tipos de sóckets para disponer de servicios adicionales. Una dirección de sócket en la Internet TCP/IP se compone de dos partes: una dirección Internet (IP address) y un número de puerto.

2.4.4 Datagramas Contra Sesiones

Los protocolos orientados a conexiones, también conocidos como protocolos basados en sesiones, circuitos virtuales o intercambios secuenciales de paquetes, ofrecen un servicio de conexión bidireccional confiable en una sesión. A cada paquete de información intercambiando en una sesión se le asigna un número de secuencia único mediante el cual se le rastrea y reconoce individualmente. Los paquetes duplicados son detectados y descartados por los servicios de sesión.

Los datagramas también conocidos como protocolos sin conexión o protocolo de transmisión constituyen una modalidad de intercambio simple pero desconfiable.

Los protocolos de datagramas más potentes, como NetBIOS, cuenta con capacidades de difusión. NetBIOS permite enviar datagramas a una entidad nombrada, un selecto grupo de entidades (envío múltiple) o todas las entidades de una red (difusión).

Los datagramas no son confiables en el sentido de que no se les reconoce ni rastrea por medio de un número de secuencia. Los datagramas son muy útiles en situaciones en las que es preciso enviar un mensaje rápidamente sin que el hecho de que no sea recibido represente un problema.

2.4.5 NetBIOS y NetBEUI

NetBIOS es el protocolo por excelencia para comunicaciones programa a programa basada en LAN. Introducido por IBM y Systek en 1984 para la PC Network en IBM, corre ahora, prácticamente sin cambios, en todas las LAN. Se le usa como interfaz con varias pilas, como las de LAN de IBM/Microsoft (NetBEUI), TCP/IP, XNS, Vines, OSI e IPX/SPX.

NetBEUI es la pila de protocolos de los productos para LAN de IBM y Microsoft, como Windows para trabajo en grupo; NT, LAN Manager, Windows 95, PS/2 Warp Connect y OS/2 Warp Server. Apareció originalmente como el transporte de comandos de NetBIOS.

NetBEUI ofrece potentes servicios de datagrama y orientados a conexiones, lo mismo que un dinámico servicio de nombramiento basado en protocolos de descubrimientos. Su principal defecto es que carece de capa de red. Otro, su carencia de seguridad.

Los servicios de NetBIOS son provistos mediante un conjunto de comandos, especificados en una estructura llamada bloque de control de red (NCB: network control black). Esta estructura también contiene los parámetros asociados con el comando y los campos en los que NetBIOS devolverá información al programa. Un comando puede ser emitido ya sea en modo de espera o no espera. En el modo de espera, el hilo de solicitud es bloqueado hasta que el comando se completa. En el modo de no espera, el control es devuelto al hilo de llamada lo más pronto posible, por lo general antes de que se complete el comando. Una vez que esto ocurre, la DLL de NetBIOS coloca un código de retorno en el NCB.

2.4.6 Named Pipes

Named Pipes ofrece comunicaciones bidireccionales altamente confiables entre clientes y un servidor. Brinda una API de programación semejante a archivos que abstrae un intercambio de datos bidireccional basado en sesión. Con Named Pipes, los procesos pueden intercambiar datos como si generaran o leyeran un archivo secuencial.

Uno de los más importantes beneficios de Named Pipes es que forma parte del servicio base de comunicaciones entre procesos. La interfaz de Named Pipes es idéntica independientemente de que los procesos sean ejecutados en una máquina

individual o se distribuyan en una red. Named Pipes corre en pilas NetBIOS, IPX/SPX y TCP/IP.

2.4.7 Llamada A Procedimiento Remoto (RPC)

Las RPC emplean el mecanismo ordinario de llamada a procedimiento. Un proceso en un servidor remoto y se detiene hasta que recibe los resultados. Los parámetros se transmiten como en cualquier procedimiento ordinario. RPC es el protocolo más común para comunicar clientes y servidores.

La RPC, al igual que un procedimiento ordinario, en un sincrónica. El proceso (o hilo) que emite la llamada aguarda hasta que obtiene los resultados. En forma oculta, el software de RPC de tiempo de ejecución reúne valores para los parámetros, forma un mensaje y lo envía al servidor remoto. El servidor recibe la solicitud desempaqueta los parámetros, llama al procedimiento y envía la respuesta al cliente.

Los mejores NOS disponen de un lenguaje de definición de interfaces (IDL: interface definition language) para la descripción de las funciones y parámetros que un servidor exporta a sus clientes. Un compilador de IDL toma estas descripciones y reproduce talones de códigos fuente (y archivos de encabezados) tanto para el cliente como para el servidor. Estos talones pueden vincularse entonces con los códigos del cliente y el servidor. El talón del cliente empaqueta los parámetros en un paquete de RPC de tiempo de ejecución y espera la respuesta del servidor. Por lo que toca a este, el talón del servidor desempaqueta los parámetros, llama al procedimiento remoto, empaqueta los resultados y envía la respuesta al cliente.

A la asociación de un cliente con un servidor se le reconoce como acoplamiento. La información de acoplamiento puede ser de codificación dura en el cliente (algunos servicios, por ejemplo, son realizados por servidores con direcciones conocidas). O bien, un cliente puede encontrar a su servidor consultando un archivo

de configuración o un parámetro del entorno. Un cliente también puede hallar a su servidor en tiempo de ejecución a través de los servicios de directorio de la red. Desde luego que los servidores deben anunciar sus servicios en el directorio. Al proceso de empleo del directorio para encontrar a un servidor en tiempo de ejecución se le llama acoplamiento dinámico.

La manera más fácil de hallar un servidor es permitir que la RPC lo haga por usted. Esto se llama acoplamiento automático, lo que significa que el talón del cliente de RPC localizará a un servidor en una lista de servidores que soporte la interfaz.

Funciones básicas de un sistema RPC

- Encapsular los parámetros de una llamada dentro de un mensaje.
- Enlazar al cliente con el servidor que va a manejar la llamada.
- Enviar el mensaje de llamada
- Encaminar el mensaje de llamada a la función que debe manejarlo.
- Envío del mensaje de respuesta.
- Desencapsular los parámetros de una llamada desde el mensaje de invocación

2.4.8 Manejo De Mensajes Y Colas (MOM)

MOM es una pieza clave de middleware, absolutamente esencial para una clase de productos de cliente/servidor. MOM presenta la vía más fácil para la creación de sistema de cliente/servidor empresariales e interempresariales. Contribuye también a la creación de sistema de cliente/servidor nómadas capaces de acumular en colas de espera de transacciones en curso y ejecutar una carga en masa cuando es posible establecer una conexión con un servidor de oficina.

MOM permite que mensajes de propósito general sean intercambiados en un sistema cliente/servidor mediante colas de mensajes. Las aplicaciones se comunican a través de redes insertando simplemente mensajes en colas y obteniendo mensajes de colas. MOM oculta todas las molestas comunicaciones de las aplicaciones y ofrece por lo general una muy simple API de alto nivel con sus servicios.

El MOM Consortium se formó a mediados de 1993 con el propósito de crear estándares para el middleware de manejo de mensajes. Sus miembros son proveedores de productos como IBM (MQSeries), Covia (Communications Integrator), Peerlogic (PIPES), Horizon Strategies (Menssage Express) y System Strategies (ezBridge).

El manejo de mensajes y la formación de colas de MOM permiten la comunicación en red entre cliente y servidores sin que estén vinculados por una conexión lógica privada especial. Clientes y servidores pueden operar a diferentes tiempos. Todos se comunican insertando mensajes en colas y retirando mensajes de colas. Obsérvese que el servidor envía la respuesta a través de una colas de mensajes. El manejo de mensajes no impone ninguna restricción a la estructura de una aplicación: si no se requiere de respuesta, sencillamente no se le envía.

La mayoría de los productos MOM de manejo de mensajes permiten que el emisor especifique el nombre de la cola de respuesta. Incluyen también algún tipo de campo de formatos que le indica al receptor cómo interpreta los datos del mensaje.

Las colas de mensajes son muy versátiles. Puede usarse para crear relaciones uno a muchos o muchos a uno. Muchos clientes envian solicitudes a la cola de un servidor. Los mensajes son extraídos de la cola por múltiples instancia del programa del servidor, los cuales atienden concurrentemente a los clientes.

Las instancias del servidor pueden extraer mensajes de la cola de acuerdo con su orden de recepción o con algún esquema de prioridad o equilibrio de cargas.

En todos los casos es posible acceder concurrentemente a una cola de mensajes. Los servidores también pueden hacer uso de filtros de mensajes para eliminar aquellos que no desean procesar o transmitirselos a otros servidores.

La mayoría de los productos de manejo de mensajes MOM ofrecen un conjunto de API simple que corre en múltiples plataformas de sistemas operativos. Casi todos ellos brindan también colas de mensajes persistentes (registrados en el disco) y no persistentes (en memoria). Los mensajes persistentes son más lentos, pero pueden recuperarse en caso de fallas de energía eléctrica tras la reiniciación de un sistema. En ambos casos, los mensajes pueden ser ya sea copiados o eliminados de una cola. Una cola de mensajes pueden ser local para la máquina o remota.

Los administradores de un sistema pueden especificar por lo general el número de mensajes que una cola ésta en condiciones de alojar y el tamaño máximo de los mensajes.

Los productos de manejo de mensajes proporcionan en su mayoría un nivel mínimo de tolerancia de fallas bajo la forma de colas persistentes. Alguno de ellos cuenta con algunas modalidad de protección para transacciones, lo que hace posible que la cola participe en un protocolo de sincronización de grabación de dos faces. Asimismo, algunos pueden incluso redireccionar mensajes a las colas alternas en caso de una falla en la red.

Cliente/Servidor En 3 Planos, Estilo MOM

El primer plano de MQ3T se compone de objetos de presentación que se insertan en la cola de eventos del cliente. Usted define los mensajes que un cliente emite o recibe, y deja el resto en manos de MOM. El plano intermedio lo ocupan objetos de negocios, dispuestos entre los clientes y los objetos lógicos de datos que integran el tercer plano. Los objetos de los tres planos se comunican a través de

MOM. MQ3T hará llegar el mensaje a la instancia del objeto correcta, e incluso puede iniciar el objeto por usted.

MQ3T en ruta esencialmente mensajes de MOM entre instancias de objetos. También permite definir visualmente la lógica y reglas de procesamientos de estos mensajes en todos los planos. Usted define las reglas de enrutamiento, difusión y filtración de mensajes. Pueden enviar mensajes a réplicas de objetos para una mayor tolerancia tolerancia de fallas.

Finalmente, puede definir reglas para el modo de manejo de múltiples respuestas, interrupciones, mensajes de arribo tardío y una amplia variedad de errores de manejo de mensajes. MQ3T soporta múltiples plataformas de cliente y servidor, como Window, NT, OS/2, AIX y macrocomputadoras.

2.4.9 Entorno de Computación Distribuida (DCE)

El Entorno de Computación Distribuida (DCE: Distributed Computing Enviroment) de Open Software Foundation (OSF) y ahora de X/Open, crea un entorno abierto de NOS que comprende múltiples arquitecturas, protocolos y sistemas operativos.

DCE ofrece tecnologías distribuidas claves, como una llamada a procedimiento remoto, un servicio de nombramiento distribuido, un servicio de sincronización de tiempo, un sistema de archivos distribuidos, un servicio de seguridad de redes y un paquete de hilos.

DCE permite que un cliente interopere con uno o más procesos del servidor en otras plataformas de computación, aun cuando sean de proveedores diferentes con distintos sistemas operativos. Además, cuenta con un método integrado de

seguridad, nombramiento y comunicaciones entre procesos. Todas estas piezas son útiles para crear un coherente entorno heterogéneo de cliente/servidor.

El DCE es importante porque representa la solución de NOS más completa en el mercado para la integración de servidores múltiples en un entorno heterogéneo de cliente/servidor.

RPC De DCE

DCE cuenta con un lenguaje de definición de interfaces (IDL: interface definition language) y un compilador que facilitan la creación de RPC. El compilador de IDL crea talones de código C portátiles para las partes tanto cliente como de servidor de una aplicación. Los talones se compilan y vinculan con la biblioteca de tiempo de ejecución de RPC, responsable de la búsqueda de servidores en un sistema distribuido, la realización de intercambios de mensajes, el empaquetamiento y desempaquetamiento de parámetros de mensajes y el procesamiento de todos los errores que puedan ocurrir. El mecanismo de RPC ofrece independencia de protocolos y redes.

DCE divide el entorno distribuido en unidades (o dominios) administrativas llamadas *celdas*. Una celda de DCE es una combinación des estaciones de trabajo del cliente y el servidor. El dominio de la celda es definido por el cliente.

Una celda suele consistir en el conjunto de máquinas utilizado por uno o más grupos que trabajan en tareas relacionadas. El tamaño de la celda está determinado sólo por el grupo de facilidad de su administración. Como mínimo, una celda de DCE debe disponer de un servidor de directorio de celda y un servidor de seguridad.

El servicio de directorio de DCE se compone de dos elementos: el servicio de directorio de celdas (CDS: cell directory service) y el servicio de directorio global (GDS: global directory service). Esta doble jerarquía ofrece autonomía de

nombramiento local (al nivel de celdas) e interoperabilidad global (al nivel entre celdas).

DCE: Servicios De Nombramiento Distribuido

DCE divide el entorno distribuido en unidades o dominios administrativas llamadas celdas. Una celda DCE es una combinación de estaciones de trabajo del cliente y el servidor. El dominio de la celda es definido por el cliente. Una celda suele consistir en el conjunto de maquinas utilizado por uno o más grupos que trabajan en tareas relacionadas. El tamaño de la celda está determinado sólo por el grado de facilidad de su administración. Como mínimo, una celda de DCE debe disponer de un servidor de directorio de celda y un servidor de seguridad.

El servicio de directorio de DCE se compone de dos elementos. El servicio de celdas (CDS: Cell Directory Service) y el servidor de directorio global (GDS: Global Directory service). Esta doble jerarquía ofrece autonomía de nombramiento local al nivel de celdas e interoperabilidad global al nivel entre celdas.

DCE: Servicios De Tiempo Distribuido

OFS adoptó el servicio de tiempo distribuido digital. Esta tecnología brinda un mecanismo para sincronizar cada computadora de la red con un estándar reconocido de tiempo. El servicio de Tiempo DCE ofrece API para la manipulación de horas y la obtención horario universal fe fuentes públicas .

DCE: Servicios De Seguridad Distribuida.

OSF adoptó el sistema de autenticación Kerberos del MIT y lo complementó con algunas características de seguridad de HP.

El sistema de autenticación de Kerberos posee en efecto tres cabezas, las cuales residen en el mismo servidor de seguridad el servidor de autenticación, la base de datos de seguridad y el servidor de privilegios.

Los servicios de seguridad de redes de DCE proporcionan autenticación, autorización y administración de cuentas del usuario. DCE soporta la autorización mediante listas de control (ACL: access control list).

DCE y Lista De Control De Acceso.

Los NOS como DCE de OSF cuentan con una serie de API que les permiten a los servidores caer y administrar sus ACL. El NOS de DCE también ofrece ganchos para que los clientes presenten a las aplicaciones servidor sus credenciales de autorización. En DCE éstas se llaman certificados de atributos de privilegios (PAC: privilege attribrute certificate), boletos emitidos por el servidor de seguridad que el cliente debe presentar al servidor.

Los PAC contienen información de autorización específica del cliente, como su grupo designado. DCE dispone de una serie de API del servidor capaces de leer la información contenida en los PAC y de compararla con la información en las ACL.

2.4.10 Sistema Distribuido De Archivos (DFS)

Para su servidor de archivos, OSF eligió el sistema de archivos Andrew (AFS: Andrew file system) de Transarc (y al Carnegie Mellon University) y el cliente sin disco de HP (también basado en protocolos de AFS) el sistema distribuido de archivos (DFS: distributed file system) de DCE ofrece un espacio para nombres uniformes, transparencia de ubicación de archivos y alta disponibilidad. Se basa en operaciones de registro, y ofrece por lo tanto la ventaja de reiniciación y recuperación rápidas tras la descompostura de un servidor. Archivos y directorios pueden duplicarse en múltiples servidores para una mayor disponibilidad.

Las API del sistema de archivos DFS se basan en POSIX 1003.1ª . DFS es interoperable con NFS de Sun, cuyos sitios conceden una vía de fácil migración al servidor de archivos de DFS, más rico en funciones.

DFS brinda un sistema de archivos de imagen única que puede distribuirse entre un grupo de servidores de archivos. El esquema de nombramiento de archivos DFS es independiente de la ubicación. El sistema de archivos está integrado a los mecanismos de seguridad y RPC de DCE. DFS explota al máximo los mecanismo de DCE se le puede administrar desde cualquier nodo de DCE.

2.4.11 Hilos.

Para su paquete de hilos, OFS seleccionó la arquitectura multihilos en concierto 8CMA: concert multithread architecture) de Digital. Este paquete de hilos portátil corre en el espacio del usuario e incluye pequeñas rutinas de envoltura para traducir llamadas a un paquete de hilos nativo basado en el núcleo . los hilos son un componente esencial de las aplicaciones cliente/servidor y son usados por los demás componentes de DCE.

El paquete de hilos DCE provee niveles granulares de multitarea en sistemas operativos carentes de hilos soportados por el núcleo. Las API de hilos de DCE se basan en POSIX 1003.4ª (estándar de hilos P (Pthreads). Los hilos de DCE también soportan entornos de multiprocesadores mediante el empleo de memoria compartida. DCE ofrece un servicio de semáforos para que los hilos sincronicen su acceso a la memoria.

Las Pilas De Middleware

Las pilas de middleware componen la elaborada infraestructura de comunicación indispensable para las comunicaciones entre una estación de trabajo de administración y sus aplicaciones, por una parte, y agentes distribuidos, otras estaciones de administración y operadores por la otra.

2.5 Tipos Middleware De Servicios Específicos

2.5.1 SQL

Se necesita un middleware para igualar los diferentes dialectos y extensiones de SQL, protocolos de manejo de mensajes en red y API "nativas" específicas del proveedor.

Para crear la "ilusión de base de datos única", el middleware debe satisfacer a dos grupos de clientes: 1) los desarrolladores de aplicaciones y herramientas frontales.

Para esto tenemos varias opciones

La Interfaz de SQL común

La idea de crear una API de SQL común que pueda ser empleada por todas las aplicaciones, y dejar después que las diferencias del servidor sean tratadas por los diferentes controladores de base de datos. Claro que decirlo es más fácil que hacerlo.

La CI de Microsoft.

El estándar de API de Windows conectividad de bases de datos abierta (ODBC: open database connectivity) de Microsoft. ODBC define tres niveles de conformidad.

La mayoría de los proveedores de servidores de bases de datos como Microsoft, IBM, Oracle, Sysbase, Tandem, CA/Ingres e Informix. La familia DB2 de IBM soporta PDBC (con extensiones) y la CLI de X/O (con extensiones), pero su protocolo nativo es ESQL/DRDA. Sybase soporta ODBC como API opcional, pero su CLI nativa es el cliente abierto de Sybase.

El Gateway de SQL Abierto.

El gateway abierto es un middleware en boga. La idea es estandarizar con la base en un (o más probablemente dos) FAP industrial abierto, dotarlo de un controlador del cliente común y desarrollar un receptor de gateway recibirá los mensajes de entrada FAP y los traducirá a la interfaz de SQL.

Gateways de SQL abiertos, traducen las llamadas de SQL aun formato y protocolo (FAP: format and protocol) común estandarizado en la industria. El FAP proporciona el protocolo común entre cliente y el servidor. El gateway actúa como el agente corredor que traduce llamadas de API del cliente al formato FAP, las transporta y después las hace corresponder con las llamadas del servidor apropiadas y viceversa.

El gateway abierto debe proveer o soportar una interfaz de SQL estándar. Asimismo, debe ser capaz de localizar servidores remotos y de ofrecer servicios de catálogos sin necesitar un servidor de bases de datos intermediario. Finalmente debe brindar herramientas para la creación del gateway por parte del servidor.

2.5.2 RDA y DRDA

Acceso a datos remotos (RDA: remote data access), y acceso a datos relacionales distribuidos (DRDA: distributed relational data access), son más que meros protocolos de gateways. Ambos cuentan con arquitecturas de extremo a extremo. La mayoria de los gateways simplemente transmite una instrucción SQL a un sistema de bases de datos remoto, tratando por lo general a cada uno de ellos como una transacción independiente. DRDA y RDA persiguen el soporte de transacciones de sitios múltiples. Algunos gateways manejan la conversión de caracteres, pero carecen de algunas de las peculiaridades sofisticadas de DRDA y RDA para la creación de representaciones de datos comunes. Los gateways suelen enlazar dos ubicaciones; DRDA y RDA están hechos para soportar redes principales de datos (con múltiples puntos de entrada y salida).

2.5.3 Monitores de TP

Los monitores de TP se especializan en la administración de transacciones desde su punto de origen por lo general en el cliente y a través de uno o más servidores, para luego volver al cliente originario. Cuando una transacción llega a su fin, el monitor de TP se cerciora de que todos los sistemas involucrados en la transacción se hallen en estado consistente.

Uno de los mayores atractivos de los monitores de TP es que supervisan todos los aspectos de una transacción distribuida, sin importar los sistemas ni administradores de recursos empleados. Un monitor de TP es capaz de administrar recursos de un solo servidor o de servidores múltiples.

La definición de Jeri Edwards, para quien un monitor de TP es "un sistema operativo para el procedimiento de transacciones". El monitor de TP es excelente para dos cosas:

- La administración de procesos. Lo cual incluye poner en marcha los procesos del servidor, canalizar trabajo en dirección a ellos, vigila su ejecución y equilibrar sus cargas de trabajo.
- La administración de transacciones . Lo que significa que garantiza las propiedades ACID para todos los programas que operen bajo su protección.
- Los monitores de TP responde a la necesidad de correr clases de aplicaciones capaces de atender a cientos y en ocasiones miles de cliente. Los monitores de TP ofrecen un sistema operativo que conecta en tiempo real a estas miles de personas impacientes.

Se realiza el acto de canalización cuando un cliente envia una solicitud de servicio, el monitor de TP la destina a un proceso disponible en el fondo de la clase. El proceso servidor se enlaza dinámicamente con la función de DLL llamada por el cliente, la invoca, supervisa su ejecución y regresa los resultados al cliente. Una vez concluido este ciclo, el proceso servidor puede ser reutilizado por otro cliente. El sistema operativo conserva en la memoria las DLL ya cargadas, donde pueden ser compartidas por otros procesos.

El monitor de TP elimina el requerimiento de proceso por cliente canalizado las solicitudes de clientes recibidas a procesos del servidor compartido. Si el número de solicitudes de clientes recibidas excede al número de procesos en una clase de servidor, el monitor de TP puede iniciar dinámicamente otros nuevos; esto es lo que se conoce como equilibrio de cargas.

En su capacidad de equilibrio de cargas, los monitores de TP desempeñan el papel de agentes de tráfico de cliente/servidor.

Además los monitores de TP deben estar preparados para comunicarse con todos los administradores de recursos en los que se ejecuta la transacción, ya sea que se encuentren en la misma máquina o a lo largo de una red.

Cuando los administradores de recursos se hallan en redes, el monitor de TP sincroniza la transacción con los monitores de TP remotos por medio de una grabación en dos faces.

Las versiones para transacciones complementan los ya conocidos intercambios de NOS con los siguientes extensiones de valor agregado:

- Aportan delimitadores de transacciones que permiten a un cliente especificar los límites de inicio y fin de transacción. La mecánica de grabación actual es delgada por lo general a uno de los monitores de TP del servidor, porque asume que el cliente no es confiable para ello.
- Introducen un intercambio en tres vías entre un cliente, el servidor y el monitor de TP (el administrador de transacción). Una nueva transacción le es asignada a un identificador (ID) especifico.
- Implantan información de estado de transacción en cada uno de los mensajes intercambiados. Esta información contribuye a que el monitor de TP identifique el estado de la transacción distribuida y deduzca lo que debe hacer a continuación.
- Permiten que un monitor de TP imponga la semántica de exactamente una vez, lo que significa que el mensaje sólo se ejecuta una vez.
- Garantizan que un proceso servidor se halle en el extremo receptor del mensaje. Las RPC y MOM tradicionales no ponen la menor atención en asuntos de este tipo; dan por supuesto que un programa aparecerá "automáticamente" en el extremo receptor.
- Ofrecen enrutamiento al servidor con base en las clases de servidor, cargas de servidor, recuperaciones automáticas y otros factores.

Esto implica mucho más que un simple intercambio de RPC o MOM. La bibliografía especializada designa a estos servicios complementados con RPC tradicional (TRPC: transactional RPC), colas para transacciones y conversaciones para transacciones. El factor distintivo es que todos los administradores de recursos y procesos invocados por estas llamadas pasan a formar parte de la transacción.

El monitor de TP es informado de todas las llamadas de servicios; emplea esta información para orquestar las acciones de todos los participantes, y hacerlos actuar como parte de una transacción.

TP Ligero.

TP ligero es simplemente la integración de las funciones del monitor de TP a los mecánicos de bases de datos. sólo unas cuantas funciones del monitor de TP han sido objeto de esta integración, entre ellas la función de embarque, cierto nivel de canalización, administración de transacciones de función única y llamadas semejantes a RPC.

TP Pesado.

TP son los monitores de TP tal. La nueva generación de productos de TP pesado para LAN de cliente/servidor incluye a CICS. Encina, Tuxedo, Pathway de Tandem, Top End y ACMS de Digital. Todos estos monitores de TP soportan la arquitectura cliente/servidor y permiten que las PC inciden desde el escritorio algunas transacciones de servidores múltiples muy complejas. Todos estos productos son soportados por herramientas de construcción visual abierta que permiten crear el primer plano con independencia del segundo plano. TP pesado incluye como administración de procesos, equilibrio de cargas, sincronización de transacciones globales, interfaces con múltiples administradores de recursos y recuperación de errores.

Tendencias del Monitor TP.

- Los monitores de TP se convierten en entornos de servidor de aplicaciones exportables. Los monitores de TP pueden correr ya entre casi todos los principales sistemas operativos para servidores. En consecuencia, constituyen entornos de aplicaciones servidor exportables. Usted genera una vez su aplicación en 3 planos y la exporta a los entornos de servidor de su gusto.
- Los monitores de TP se convierten en agentes de tráfico universales. Además de soportar a sus clientes tradicionales y enrutan llamadas de otros tipos de clientes.
- Los monitores de TP se convierten en corredores de recursos. Aparte de bases de datos de SQL, los monitores de TP ahora soportan todo tipo de administradores de recursos de segundo plano, como multiplicidad en los sistemas de archivos, bases de datos jerárquicas, colas persistentes, almacenes de imágenes, depósitos de HTML y bases de datos de documentos de Lotus Notes.
- Los monitores de TP descubren las herramientas de clientes/servidor.

2.5.4 CICS DE IBM.

CICS, IBM ofrece ahora una consistente solución de monitor de TP de cliente/servidor a lo largo de una amplia variedad de plataformas de cliente/servidor. Los cliente y servidores de CICS se comunican mediante una RPC para transacciones llamada interfaz de llamada externa (ECI: external call interface). ECI brinda un entorno de cliente delgado, o pequeño que le permite invocar directamente programas de CICS sobre una amplia serie de transportes, como NetBIOS, TCP/IP e IPX/SPX, algunas herramientas.

Como opción adicional, los clientes de CICS pueden emular con la interfaz de presentación externa. Las terminales han sobrevivido en muchas modalidades comerciales del monitor de TP. CICS soporta la Transactional RPC de Encina basada en DCE en algunas de sus plataformas de Unix. Cabe señalar que la nueva versión de CICS reduce algunas de las complejidades de DCE al no requerir de la instalación de servidores de seguridad y nombramiento en servidores de dominio único.

2.5.5 TUXEDO

Tuxedo, originalmente de Unix Systems Lab, ofrece un entorno de monitor de TP que corre más de 36 plataformas de Unix, como HP-UX, AT&T GIS, AIX; Solaris y OSF/1. Cuenta con soporte de cliente en DOS, Unix, OS/2 y OS de Mac.

Tuxedo es una tecnología clave de middleware para el enlace de Windows NT, Unix, OS/2 y sistemas de macrocomputadoras a redes con NetWare. Tuxedo corre en Unix, NetWare y NT.

El sistema BEA TUXEDO es uno de los más poderosos middlewares para transacciones, es utilizado en plataformas que manejan grandes volúmenes de información, además es una aplicación confiable para distribuir transacciones.

Es una arquitectura Cliente/Servidor que ofrece una comunicación conversasional entre diferentes interfaces, el proceso que comienza la comunicación generalmente es un cliente y el proceso que recibe la comunicación es un servidor. Además administra y controla la comunicación entre el cliente y el servidor. También tiene la capacidad de correr en diferentes ambientes, independientemente de las comunicaciones, del hardware y de la base de datos.

TUXEDO además de ser un middleware muy poderoso, también actúa como un monitor de transacciones, es frecuentemente utilizado por Instituciones bancarias, Cajeros Automáticos y en empresas donde se maneja grandes volúmenes de información.

TUXEDO proporciona una manipulación trasparente de los datos de la aplicación sobre diferentes plataformas. Además permite a los administradores organizar los servidores lógicos, también tiene la habilidad de asignarles prioridades a las peticiones de los clientes.

BEA TUXEDO son aplicaciones modulares proporciona el manejo de las transacciones y permite que se relizen copias de los servicios para poder atender las peticiones de varios clientes a la vez. La rapidez con que se ejecuten dichas peticiones va a depender en gran medida de las características de la maquina en donde se encuentre la parte del monitor de transacciones de TUXEDO, además de la configuración y administración.

Tuxedo tiene las siguientes características:

- Administrador de aplicaciones gráficas. Incluye una utilería basada en Motif, llamada Application Manager, que permite administrar y manejar servidores de Tuxedo remotos. Esta utilería puede vigilar aplicaciones y suministrar estadísticas de desempeño en una base de información de administración (MIB: managemem information base) de SMNP. Si se da un cuello de botella, el software puede reconfigurar parámetros "sobre la marcha" y recupera el equilibrio de carga y la planeación de prioridades.
- Soporte de SMNP. Tuxedo proporciona un agente de SMNP y MIB. Esto significa que famosas estructuras de administración empresarial puede acceder remotamente a la información reunida por el Application Manager de Tuxedo.
- Corretaje de eventos de publicación y suscripción. Tuxedo cuenta ahora con un Event Broker que media entre publicadores de eventos y suscriptores.: por

ejemplo, cuando el precio de ciertas acciones excede un limite. Con Tuxedo usted puede registrar las acciones que éste debería emprender cuando ocurre un evento. Puede activar una serie de pasos de procesamientos entre aplicaciones múltiples en respuesta a eventos especificados con anterioridad. Tuxedo también le permite transferir datos a suscriptores como parte del evento. En el extremo receptor, Tuxedo permite a los suscriptores filtrar eventos con base en "comodines". La ventaja de permitir que un monitor de TP funja como administrador de eventos es que combina confiabilidad de transacciones con administración de eventos.

Tuxedo soporta enrrutamiento dinámico de datos, lo que direcciona transparentemente consultas a los depósitos de datos adecuados y permite que los administradores cambien "sobre marcha" reglas de enrutamiento. Tuxedo también ofrece nuevas características de seguridad con base en listas de control de acceso a consultas (ACL: access control list).le permite controlar el acceso a consultas, aplicaciones y RPC de Tuxedo.

2.5.6 Encina De Transare/IBM.

Encina de Transare fue diseñado desde el principio como monitor de TP basado en DCE de OSF. Los clientes de Encina se comunicarán con sus servidores mediante un cliente de DCE "ligero" que ofrezca una RPC para transacciones por encima de TCP/IP y pilas de APPC.

Encina siempre ha usado a DCE como conducto subyacente. DCE ofrece seguridad y nombramiento entre redes, así como hilos, tiempo distribuido y un sistema de archivos distribuidos (DFS: distributed file system). Encina soporta transacciones entre administradores de recursos múltiples, tales como bases de datos de SQL.

Encina 2.0 incluye bibliotecas C++ para la creación de aplicaciones de procesamiento de transacciones con la estructura de desarrollo Encina++, esta enmascara algunas de las complejidades de DCE para el desarrollador.

Encina Console, interfaz de administración basada en GUI que permite configurar y vigilar el entorno de aplicación distribuido.

Capitulo III. Redes y Comunicaciones

Las redes conectan computadoras, unas con otras, de manera que puedan compartir información. La interconexión de computadoras supone una serie de elementos relacionados para lograr dicha comunicación.

3.1 Definición

Una red es un sistema de comunicaciones que permite que un número de dispositivos independientes se comuniquen entre sí. Es un recurso compartido empleado para intercambiar información entre usuarios.

3.2 Objetivos de las redes de computadoras

- Eliminar el desplazamiento de los individuos en la búsqueda de información.
- Ofrecer transparencia al usuario por medio de compatibilidades técnicas en las terminales.
- Aumentar la capacidad de procesamiento y almacenamiento disponibles por cada uno de los usuarios en un momento determinado.
- Proponer alternativas de enrutamiento para el transporte de la información en caso de fallas en los medios de transmisión.
- Ofrecer acceso a servicios que permitan la manipulación de datos en diferentes áreas geográficas.

Encina 2.0 incluye bibliotecas C++ para la creación de aplicaciones de procesamiento de transacciones con la estructura de desarrollo Encina++, esta enmascara algunas de las complejidades de DCE para el desarrollador.

Encina Console, interfaz de administración basada en GUI que permite configurar y vigilar el entorno de aplicación distribuido.

Capitulo III. Redes y Comunicaciones

Las redes conectan computadoras, unas con otras, de manera que puedan compartir información. La interconexión de computadoras supone una serie de elementos relacionados para lograr dicha comunicación.

3.1 Definición

Una red es un sistema de comunicaciones que permite que un número de dispositivos independientes se comuniquen entre sí. Es un recurso compartido empleado para intercambiar información entre usuarios.

3.2 Objetivos de las redes de computadoras

- Eliminar el desplazamiento de los individuos en la búsqueda de información.
- Ofrecer transparencia al usuario por medio de compatibilidades técnicas en las terminales.
- Aumentar la capacidad de procesamiento y almacenamiento disponibles por cada uno de los usuarios en un momento determinado.
- Proponer alternativas de enrutamiento para el transporte de la información en caso de fallas en los medios de transmisión.
- Ofrecer acceso a servicios que permitan la manipulación de datos en diferentes áreas geográficas.

- Compartir recursos.

3.2.1 Ventajas de las redes de computadora

Es importante mencionar que las redes de comunicaciones, no son simples conexiones que permiten a un usuario accesar a recursos que se encuentran residentes en otras computadoras.

Las ventajas de una red, van más allá de los beneficios mencionados, lo cual se refleja en la tendencia actual hacia la conectividad de datos. No sólo en el envío de información de una computadora a otra sino, sobre todo, en la distribución del procesamiento a lo largo de grandes redes en todas las empresas.

Principales ventajas aportadas por el uso de la red:

- Mantener bases de datos actualizadas instantáneamente y accesibles desde distintos puntos.
- Facilitar la transferencia de archivos entre miembros de un grupo de trabajo.
- Compartir periféricos caros (impresora láser, plotters, discos ópticos, etc,).
- Bajar el costo del software, comprando licencias de uso múltiples en vez de muchas individuales.
- Mantener versiones actualizadas y coherentes del software.
- Facilitar la copia de respaldo de los datos.
- Correo electrónico.
- Comunicarse con otras redes (puentes-bridges).
- Conectarse con minis y mainframes (por passarelas(gateways)).
- Mantener usuarios remotos vía módem.

3.3 Extensión Geográfica

La topografía de una red esta definida por la forma en que se tienden los cables (en un sentido general, se debe entenderse la topografía de una red como la disposición geográfica de los diferentes medios de transmisión, con independencia de que estos estén representados por sistemas físicamente continuos- como un cable- o sistemas compuestos por elementos de comunicación aislados- las antenas de un sistema de radio.) que conectan las distintas estaciones. A la hora de planificar la disposición de los cables, la topografía es más importante que la topología.

Las redes por su cobertura geográfica, se clasifican en:

- Redes de Area Local (LANs)
- Redes de Area Metropolitana (MANs)
- Redes de Area Amplia o Extendida (WANs)

3.3.1 Redes de Area Local (LANs)

El concepto de redes locales o LAN (Local Area Networks) se refiere a la estructuración de redes cuyos componentes o nodos se encuentran en distancias relativamente cortas. Los elementos que se interconectan pueden ser terminales, estaciones de trabajo, microcomputadoras, minicomputadoras entre sí o un mainframe.

Las redes locales pueden estructurarse de dos formas: conectando todas las computadoras entre sí o teniendo una computadora central a la cual estén conectadas las demás computadoras. En el primer esquema es necesario dedicar parte de los recursos a recibir y pasar información a otras computadoras, sin embargo, es accesible en el aspecto económico para redes pequeñas de hasta unos cuantos kilómetros de extensión. Por ejemplo una oficina o un centro educativo.

A esta forma de operar y de compartición de recursos también se le conoce como redes punto a punto. Cabe mencionar que es el sistema operativo de red el que determina la forma de operar de los nodos de la red.

El segundo esquema utiliza la filosofía cliente/servidor, en la cual una computadora es la servidora y está pendiente de las solicitudes que le hagan las computadoras clientes para dar respuesta a ellas.

Bajo esta configuración los clientes dependen del servidor, por lo que también se le conoce como red de servidor dedicado. Los sistemas operativos líderes en este tipo de redes son: Intranetware, de Novell, y segundo, Windows NT Server de Microsoft.

Para poder establecer comunicación entre computadoras es necesario que ambas sigan el mismo conjunto de reglas y de procedimientos para controlar el flujo de datos. Tanto el emisor como el receptor deben de seguir los mismos procedimientos. A esto se le llama protocolo de comunicación.

Suelen emplear tecnología de difusión mediante un cable sencillo al que están conectadas todas las máquinas, operan a velocidades entre 10 y 100 Mbps, tienen bajo retardo y experimentan pocos errores.

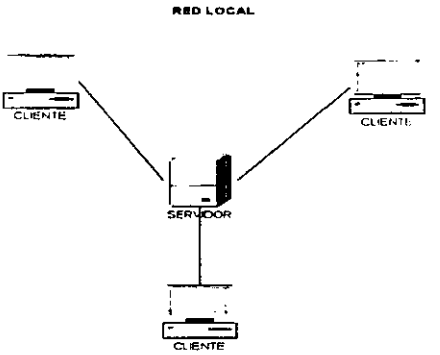


FIGURA # 4 Red Local

3.3.2 Redes de Area Metropolitana (MANs)

Una red metropolitana es esencialmente una red local muy grande que cubre una ciudad entera, administrando el transporte de gran cantidad de datos. Son una versión mayor de la LAN y utilizan una tecnología muy similar. Actualmente esta clasificación ha caído en desuso, y normalmente sólo se distingue entre redes LAN y WAN.

3.3.3 Redes de Área Amplia (WAN)

Son redes que se extienden sobre un área geográfica extensa. Contiene una colección de máquinas dedicadas a ejecutar los programas de usuarios (hosts). Estos están conectados por la red que lleva los mensajes de un host a otro. Estas LAN de host acceden a la subred de la WAN por un router. Suelen ser por tanto redes punto a punto.

Una WAN contiene numerosos cables conectados a un par de ruteadores. Si dos ruteadores que no comparten cable desean comunicarse, deben de hacerlo a través de ruteadores intermedios. El paquete se recibe completo en cada uno de los intermedios y se almacena allí hasta que la línea de salida requerida esté libre.

Se pueden establecer WAN en sistemas de satélite o de radio en tierra en los que cada ruteador tiene una antena con la cual poder enviar y recibir la información.

El desempeño de una WAN está determinado principalmente por dos factores: 1) los conmutadores que enrutan los datos en redes y 2) el tipo (y calidad) de la infraestructura de cableado de área amplia .

Los conmutadores de paquetes dividen los flujos de datos en paquetes, que luego lanzan a la red. Las cabeceras de dirección sirven para conducir a cada paquete a su destino. Protocolos como TCP/IP o IPX/SPX corren en estas redes de conmutación de paquetes.

A continuación se mencionan las tres principales tecnologías de conmutación de paquetes: frame relay, servicios de datos conmutados multimegabit (SMDS: switched multimegabit data services) y modo de transferencia asíncrona (ATM: asynchronous transfer mode).

Tecnología de WAN	Velocidad Máxima	Aplicaciones	Tamaño del paquete	Despliegue
Frame Relay	1-2 Mbit/s (T1/E1)	Datos	Dimensión variable, 4096 bytes máx.	Amplio
SMDS	45 Mbit/s (T3)	Datos	Dimensión variable, 9188 bytes máx (puede dividirse en células de 53 bytes)	Limitado
ATM	2.4 Gbit/s	Datos, voz y video	células de 53 bytes	Limitado pero creciente

Tabla #1 Red WAN: Alternativas de conmutación de paquetes

Frame Relay

Es la tecnología de conmutación de paquetes para WAN más común en la actualidad. Hizo su aparición en 1992, como una implementación modernizada de la antigua tecnología de comunicación de paquetes X.25. El Frame Relay puede enrutar paquetes de dimenciones variables a velocidades de TI (o EI). Solo se puede dar salida a un máximo de 64 Kbits/s en la misma infraestructura. El Frame Relay logra esta hazaña eludiendo la verificación de errores en cada segmento de la red (los puntos intermedios). En cambio, depende de los puntos terminales para la verificación de errores de extremo a extremo, su demanda es muy fuerte.

SMDS.

SMDS fue diseñada originalmente para llenar el vacío de los servicios de WAN de alta velocidad. Soporta paquetes de dimensión variable que pueden dividirse en células fijas de tamaño de ATM para facilitar la migración. Hoy puede correr a velocidades de T3 (45 Mbits); alcanza este alto rendimiento por no ofrecer soporte a circuitos virtuales; cada paquete se las ve por si solo.

ATM.

ATM, se trata de un protocolo de comunicación de paquetes que alcanza velocidades muy altas mediante el empleo de células de datos de dimensión fija, o paquetes para circuitos virtuales. Los circuitos virtuales permanentes (PVC: permanent virtual circuit) son asignados estáticamente; los circuitos virtuales conmutados (SVC: switched virtual circuit) son dinámicos. En ambos casos, un circuito virtual garantiza la calidad del servicio, incluyendo ancho de banda y prioridad.

ATM fue diseñada para la combinación de diferentes tipos de tráfico, como datos, voz y video. Su sistema de transmisión de pequeñas células de información (paquetes) de 53 bits posee la suficiente flexibilidad para trabajar a capacidades que van de megabits a gigabits. La pequeña célula de tamaño fijo permite implementar en el hardware conmutadores de muy alta velocidad. El ancho de la banda de ATM se basa en la demanda y es escalable, lo que significa que cada nodo puede acceder a la red a la velocidad requerida por una aplicación. ATM prevé ofrecer enlace inconsútil en red y eliminar las actuales distinciones entre LAN y WAN. Por lo pronto, sin embargo, las implementaciones de ATM para LAN y WAN siguen constituyendo mercados diferentes.

B-ISDN.

La red digital de servicios integrados de banda ancha (B-ISDN: broadband-integrated services digital network. El CCITT la define como una WAN de comunicación de células capaz de soportar velocidades superiores a los 1.54 Mbits. El estándar de B-ISDN define una capa de adaptación a ATM (AAL: ATM adaptation layer), responsable de la correspondencia de la información de datos, voz y vídeo con y desde formatos de célula definidos por ATM. Así, B-ISDN es en realidad la carretera de la información, integrada en este caso por enlaces de fibra óptica y conmutadores de ATM. SMDN y Frame Relay son los precursores de esta tecnología.

3.4 Topología

Se entiende por topología la estructura física de la red, la forma en que se conectan sus dispositivos y el método que utilizan para el acceso al medio de transmisión. También la forma lógica en que se conectan los nodos mediante canales para constituir una red define la topología de la misma

Hay tres formas posibles de conexión:

- Punto a punto: en la que solo se unen dos estaciones adyacentes, sin pasar a través de una estación intermedia.

- Multipunto: en la que dos o más estaciones comparten un solo cable.

- Lógica: en la cual las estaciones se pueden comunicar entre sí, haya o no conexión física entre ellas.

Todas las topologías, excepto la de estrella, envían generalmente las señales a todas las estaciones conectadas a la red; la estación receptora, o su equipo de comunicaciones, debe seleccionar solo las señales que estén dirigidas a esa estación desde todas las transmisiones de la red.

Se emplean varias topologías en las redes de comunicación de datos entre otras.

- *Redes de estrella* que disponen de canales dedicados entre cada estación y elemento central. Todas las comunicaciones entre las estaciones deben pasar a través del elemento central.
- *Redes de anillo* en la que cada nodo esta conectado a otros dos y los mensajes circulan alrededor del anillo cerrado. Una red en bucle es una red en anillo en la que una estación maestra controla la transmisión.
- *Redes en anillo modificadas* que utilizan un anillo central de elementos de control a los que se conectan las estaciones en una topología de estrella en torno a dichos elementos.
- *Redes de bus* que tienen topología lineal y estaciones conectadas mediante adaptadores.
- *Redes de árbol* que son complejas redes en bus, consistentes en una serie de bifurcaciones que convergen indirectamente en un punto central y tiene un único camino de comunicaciones entre dos estaciones cualesquiera.

3.4.1 Topología de Estrella

La topología en estrella son las más empleadas para conectar terminales locales y remotos a los ordenadores y en las centralistas privadas (PABX). La capacidad de una red en estrella basada en ordenadores depende de la capacidad del elemento central para aceptar mensajes y retransmitirlos cuando sea necesario.

La distancia que cubre una red en estrella esta limitada por los medios de comunicación utilizados; cada uno de ellos debe tener la longitud total necesaria para comunicar cada estación con el elemento central, frente a lo que ocurre en una red de intercomunicación total en la que pueden utilizarse cables de longitud total menor.

Él número de estaciones puede extenderse solo hasta los limites impuestos por la capacidad total del elemento central. El elemento central es el único punto de la red que puede provocar el fallo total de esta; los fallos en los cables de comunicación solo afecta a una de las estaciones. Los retrasos en los mensajes pueden ser elevados, debido a las limitaciones de capacidad del nodo central. El costo inicial de una red en estrella es alto, dado que hay que instalar el elemento central. Los costos increméntales de nuevas estaciones son reducidos hasta los limites de expansión del elemento central.

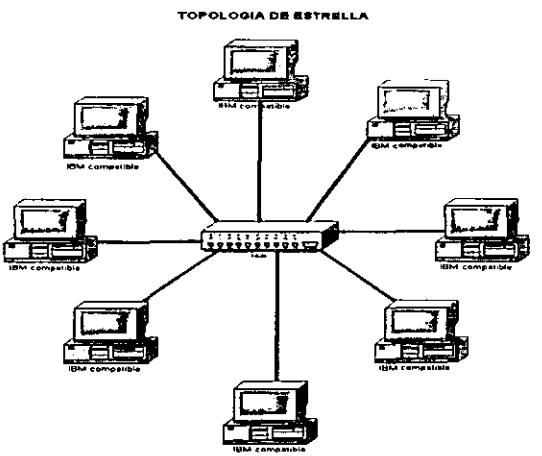


Figura # 5 Topología de estrella.

3.4.2 Topología de Anillo

La topología en anillo se emplea casi exclusivamente en las redes de área local basada en paso de testigo (*token pasing)* o en paso en control de acceso al anillo mediante *slots*. La capacidad de la red esta determinada por el medio y por la capacidad del repetidor que se necesita en cada nodo. La longitud total del anillo y la máxima distancia entre nodos es limitada, pero el alcance total de la red es generalmente mayor que el de un sistema lineal.

El numero máximo de nodos esta limitado por el diseño del sistema. Cada nodo adicional supone la parada del sistema y la reducción de las presentaciones. El anillo vulnerable al fallo en un enlace o repetidor.

Existen sistemas de doble anillo capaces de soportar dos roturas. El retraso en los mensajes aumenta a medida que se añaden más estaciones al anillo, y es mayor que el que se experimenta en un sistema de bus ligeramente cargado. El costo por nodo es generalmente menor que el de otras topologías que ofrecen presentaciones similares y la cantidad de cable requerido es generalmente menor que en las topologías en estrella.

TOPOLOGIA DE ANILLO

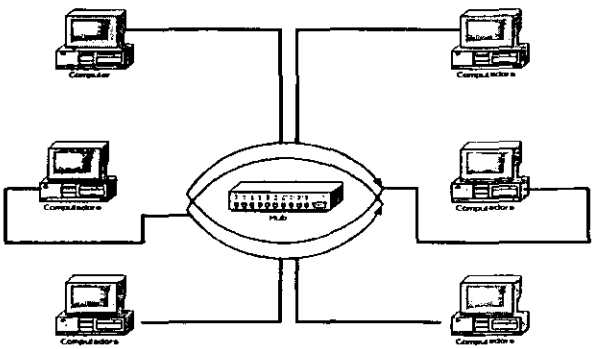


FIGURA # 6 Topologia De Anillo

3.4.3 Topología de Anillo Modificadas

Las topologías en anillo modificadas se emplean en las redes de área local más recientes, como la IEEE 802.5 y la FDDI, para simplificar la expansión de la red y permitir el uso de interfaces menos complejas en las estaciones. La capacidad, distancia máxima, máximo numero de terminales, vulnerabilidad y retraso en los mensajes es la misma que en el caso de sistemas de anillo convencionales, pero la facilidad de expansión es mayor si los elementos de control disponen de conexiones libres. Se necesita menos cable que en caso de una topología den estrella, y el costo puede ser menor que el de un anillo convencional.

3.4.4 Topología de BUS

Las topologías en *bus* se usan mucho para las redes de área local en banda base, *clusters* multipunto para terminales y redes militares de comunicación de datos. La capacidad de red esta limitada por el medio empleado y por el mecanismo de control de acceso. La capacidad total de la red y por el mecanismo decrece a medida que aumenta el numero de estaciones.

La longitud máxima del cable suele ser reducida, dado que se necesita un gran ancho de banda para soportar muchos canales virtuales. Pueden añadirse nuevas estaciones sin necesidad de reconfigurar la red hasta alcanzar el máximo valor permitido por la capacidad o hasta alcanzar valores inaceptables para los retrasos. Los *buses* sondeados tienen un controlador que es el único punto del sistema que lo pueda dejar inutilizado. Para evitar que el cable sea un punto débil del sistema en los sistemas militares suelen emplearse cables duales.

El retraso de los mensajes en los sistemas de paso de testigo se incrementa con el numero de estaciones, y en los sistemas de contienda, los retrasos aumentan con el trafico. Los sistemas de sondeo experimenta unos retrasos definidos por la secuencia de sondeo. El costo por cada estación es generalmente más bajo que en las redes en estrella, pero más alto que en las redes de anillo, y los buses no suponen una inversión inicial tan alta.

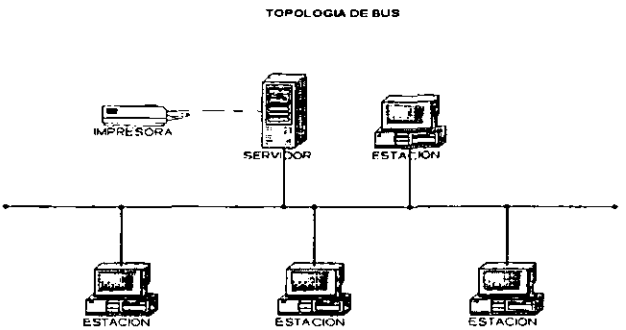


Figura # 7 Topología de BUS.

3.4.5 Topología de Arbol

La topología en árbol pueden estar formadas por un conjunto de *buses* lineales encadenados, pero suelen encontrarse más comúnmente en las redes de área local de banda ancha. La capacidad de la topología en árbol es alta, y limitada únicamente por el ancho de banda del cable.

La distancia máxima que pueden cubrir es mayor que la de buses lineales, porque pueden encadenarse muchos buses utilizando repetidores. Los sistemas de banda ancha se pueden extender a varios kilómetros y se les puede añadir muchisimas estaciones sin necesidad de reconfigurar la red.

El único punto vulnerable de una red de árbol es el nodo raiz, que generalmente se encuentra duplicado. Un fallo en un repetidor o en un cable en cualquier otro punto de la red solo deja fuera de servicio a las estaciones que se encuentran a partir de ese punto. Los retrasos en los sistemas de banda ancha son bajos cuando se emplean canales independientes haciendo uso de las técnicas de multiplexacion por división de frecuencia. El costo es similar al de los sistemas de buses lineales.

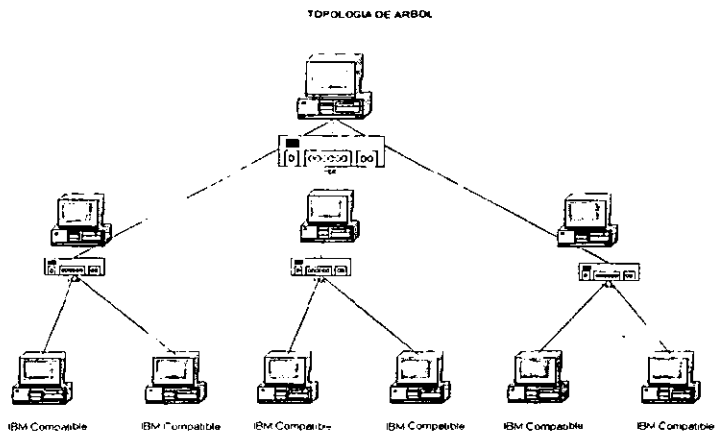


Figura # 8 Topología de Arbol

3.4.6 Factores de evaluación de la topología

La elección de la topología tiene un fuerte impacto en el comportamiento de la red, y requiere de una serie de estudios de factibilidad y velocidad que vayan de acuerdo con los requerimientos establecidos.

Cada topologia tiene sus ventajas y desventajas, que se pueden avaluar en función de:

- capacidad de respuesta (*throughput*)
- distancia máxima obtenible
- máximo número de estaciones
- vulnerabilidad a fallos de los enlaces o estaciones
- retraso en los mensajes
- costo

3.5 Sistemas Operativos

Los sistemas operativos de red pueden operar y procesar solicitudes de aplicación que se ejecutan en clientes, mediante el procesamiento de las solicitudes mismas.

Algunos de los sistemas operativos de Red (NOS) pueden ser Windows NT, NetWare de Novell, UNIX, NT Server entre otros. Se dará un breve resumen de los principales sistemas operativos de Red y se explicará más ampliamente los usuarios en la res de la Subdirección de Crédito y Cobranza.

3.5.1 Windows NT Workstation.

Windows NT es un verdadero sistema operativo de 32 bits muy poderoso, que está disponible en versiones cliente y servidor. Entre las características clave de NT está la multitarea prioritaria, procesos de multilectura o hebras, portabilidad y soporte par multiprocesamiento simétrico. Soporta multitareas preferente, multihilos. Protección de memoria y un sistema de archivos para transacciones. Es apto para red; soporta TCP/IP, NetBEUI, IPX/SPX, PPP y Apple Talk.

Es NT y no los programas especificos quien determina cuando deberá interrumpirse un programa y empezar a ejecutar otro. Procesos de lectura múltiple o hebras, es un termino que en NT, se refiere a los hilos que funcionan como agentes de ejecución. Tener hebras de ejecución múltiples dentro de un mismo proceso, significa que un proceso ejecuta, de manera simultanea, diferentes partes de un programa en diferentes procesadores. El multiprocesamiento simétrico permite entre todos los procesadores disponibles, haciendo que todo funcione mucho más rápido.

NT incluye software de red de pinto a punto para que los usuarios de NT puedan compartir archivos y aplicaciones con otros usuarios que ejecuten NT o Windows para Trabajo en Grupo.

Resumen de Windows NT

- Windows NT es un sistema operativo de 32 bits, que está disponible en versiones cliente y servidor.
- NT ofrece procesamiento multitareas, procesos de lectura múltiple e interrupciones prioritarias.
- Ofrece la capacidad de realizar procesamientos simétricos.
- NT califica para la certificación gubernamental C-2 para ambientes seguros.
- Incluye soporte integrado para IPOX/SPX, TCP/IP, NetBEUI y otros transportes.
- El directorio de servicios de NT 4.0 (NTDS) soporta a 25,000 usuarios por dominio y cientos o miles por empresa.
- NT 4.0 incluye un programa de diagnósticos que proporciona información acerca de los drivers y del uso de la red.

3.5.2 Novell Netware.

Novell NetWare es un servidor de archivos sumamente veloz, efectivo y bien cimentado que soporta a clientes OS/2, Mac y Windows.

El sistema operativo de red Novell NetWare, puede funcionar en diferentes topologías, dependiendo del hardware que se seleccione, este puede ejecutarse en una red configurada como estrella, agrupamiento de estrella, Token Ring e incluso en un bus.

Resumen de Novell Netware

- Puede funcionar en diferentes topologías.
- Está diseñado para ofrecer un verdadero soporte de servidor de archivos de red.
- Puede manejar hasta 1000 usuarios en un solo servidor (versión 4.x).

3.5.3 UNIX.

UNIX es un sistema operativo maduro y rico en funciones que puede ampliarse desde el escritorio hasta la supercomputadora. Además es una opción adecuada de servidores de bases de datos, especialmente por su facilidad de ampliación.

Las características fundamentales del UNIX son: memoria virtual, multitarea y multiusuario. La filosofía original de diseño de UNIX fue la de distribuir la funcionalidad en pequeñas partes.

Al sistema operativo UNIX lo forman varios componentes principales. Entre estos componentes están el núcleo, el shell, el sistema de archivos y las órdenes (o programas de usuarios).

- **El sistema de archivos.** la unidad básica utilizada para organizar la información en el sistema UNIX se denomina archivo. El sistema de archivos del sistema UNIX proporciona un método lógico para organizar, almacenar, recuperar, manipular y gestionar la información. Los archivos están organizados en un sistema en un sistema de archivo jerárquico, agrupados en directorios. Una característica de simplificación importante del sistema UNIX es la forma general de tratamiento de los archivos.
- **El shell.** El shell lee las órdenes y las interpreta como peticiones de ejecución de un programa o programas, lo que realiza posteriormente. Debido a este papel, el shell se denomina intérprete de órdenes. Además de ser un intérprete de órdenes, el shell también es un lenguaje de programación. Como lenguaje de programación, permite controlar cómo y cuándo se llevan a cabo las órdenes.
- **El núcleo.** El núcleo es la parte del sistema operativo que interactúa directamente en el hardware de una computadora, a través de los controladores de dispositivos que están incorporados en el núcleo. Proporciona conjuntos de servicios que pueden ser utilizados por programas, aislando a estos programas del hardware subyacente. Las funciones principales del núcleo son la gestión de la memoria, el control de acceso a la computadora, el mantenimiento del sistema de archivos, el manejo de las interrupciones (señales que finalizan la ejecución), el manejo de errores, la realización de los servicios de entrada y salida (que permiten a las computadoras interactuar con terminales, dispositivos de almacenamiento e impresoras) y la asignación de recursos de la computadora (tales como la UCP o dispositivo de entrada/salida) entre los usuarios.

Los programas interactúan con el núcleo a través de 100 llamadas al sistema, aproximadamente. Las llamadas al sistema dicen al núcleo que lleve a cabo diferente tareas para el programa, tales como abrir un archivo, escribir en un archivo, obtener información sobre un archivo, ejecutar un programa, terminar un proceso, cambiar la prioridad de un proceso y obtener la hora y la fecha.

3.5.4 Windows 95.

Windows 95 es un sistema operativo de 32 bits con multitareas y multilecturas. Cuenta con un sistema de red integrado de 32 bits para permitirle funcionar directamente con la mayoría de las principales redes, incluyendo a NetWare, Windows NT y otras máquinas de punto a punto. Las primeras presentaciones de la estrategia de red de Windows 95 caracterizaban la meta de Microsoft como proveedor del mejor sistema operativo de escritorio para computadoras personales conectadas en red. Con este fin, Windows incorpora plenas capacidades de red punto a punto, permitiendo que se configuren redes autocontenidas de Windows 95 con cada máquina actuando como servidor de red. Además Windows 95 tiene por objeto proporcionar conectividad a las principales arquitecturas de red a través de una interfaz de usuario única. Windows 95 pone énfasis en las redes incorporando soporte punto a punto, conectividad de red de área local y conectividad remota.

3.6 Protocolo TCP/IP

El protocolo es un conjunto formal de reglas que gobiernan los formatos de datos, las temporizaciones, el control de secuencias, el control de acceso y el sistema de detección de errores necesario para indicar y mantener una comunicación, entre dos entidades. El emisor y el receptor utilizan el mismo protocolo o a través del interfaz de comunicaciones o extremo a extremo de la red.

3.6.1 Definición

Las siglas TCP/IP se refieren a dos protocolos de red, que son Transmisión Control Protocol (Protocolo de control de transmisión) e Internet Protocol (Protocolo de Internet) respectivamente. Estos protocolos permanecen a un conjunto mayor de protocolos.

Estos protocolos se encargan de controlar los mecanismos de transferencia de datos. Normalmente son invisibles para el usuario y operan por debajo de la superficie del sistema. Dentro de estos protocolos tenemos:

- TCP. Controla la división de la información en unidades individuales de datos (llamadas paquetes) para que estos paquetes sean encaminados de la forma más eficientes hacia su punto de destino. En dicho punto, TCP se encargará de reensamblar dichos paquetes para reconstruir el fichero o mensaje que se envió. Por ejemplo, cuando se nos envía un fichero HTML desde un servidor Web, el protocolo de control de transmisión en ese servidor divide el fichero en uno o más paquetes, numera dichos paquetes y se los pasa al protocolo IP. Aunque cada paquete tenga la misma dirección IP de destino, puede seguir una ruta diferente a través de la red. Del otro lado TCP reconstruye los paquetes individuales y espera hasta que hayan llegado todos para presentárnoslos como un sólo fichero.

- IP. Se encarga de repartir los paquetes de información enviados entre el ordenador local y los ordenadores remotos. Esto lo hace etiquetando los paquetes con una serie de información, entre la que cabe destacar la direcciones IP de los ordenadores. Basándose en esta información, IP garantiza que los datos se encaminaran al destino correcto. Los paquetes recorrerán la red hasta su destino (que puede estar en el otro extremo del planeta) por el camino más corto posible gracias a unos dispositivos denominados encaminadores o routers.

La siguiente tabla muestra una lista de plataformas que soportan TCP/IP:

Plataforma	Soporte de TCP/IP
UNIX	Nativo
DOS	Piper/IP por Ipswitch
Windows	TCPMAN por Trumpet Software
Windows 95	Nativo
Windows NT	Nativo
Macintosh	MacTCP u OpenTrasnport (Sys 7.5+)
OS/2	Nativo
AS/200 OS/400	Nativo

Tabla # 2 Plataformas que soportan TCP/IP

3.6.2 Como Trabaja TCP/IP.

TCP/IP opera a través del uso de una pila. Dicha pila es la suma total de todos los protocolos necesarios para completar una transferencia de datos entre dos máquinas (así como el camino que siguen los datos para dejar una máquina o entrar en la otra). La pila está dividida en capas como la muestra la tabla siguiente:

EQUIPO SERVIDOR O CLIENTE

Capas de Aplicación	Cuando un usuario inicia una transferencia de datos, esta capa pasa la solicitud a la Capa de Transporte.
▼	
Capa de Transporte	La Capa de Transporte añade una cabecera y pasa los datos a la Capa de Red
▼	
Capa de Red	La Capa de Red, se añaden las direcciones IP de origen y destino para el enrutamiento de datos.
▼	
Capa de Enlace de Datos	Ejecutan un control de errores sobre el flujo de datos entre los protocolos anteriores y la Capa Física.
▼	
Capa Física	Ingresa o egresa los datos a través del medio físico, que puede ser Ethernet vía coaxial, PPP vía módem, etc.

Tabla # 3 Funcionamiento de TCP/IP a través de la PILA

Después de que los datos han pasado a través del proceso ilustrado en la figura anterior, viajan a su destino en otra máquina de la red. Allí, el proceso se ejecuta al revés (los datos entren por la capa física y recorren la pila hacia arriba). Cada capa de la pila puede enviar y recibir datos desde la capa adyacente. Cada capa esta también asociada con múltiples protocolos que trabajan sobre los datos.

3.7 Comunicación

La comunicación de datos consiste en la transmisión y recepción por medios electrónicos, en donde los datos son representados por medio de bits (representación mínima de los datos en una computadora). Existen dos maneras de trasmitir bits por medios electrónicos: en paralelo (transmisión sincrónica) y en serie (transmisión asincrónica).

3.7.1 Modos De Transmisión

- Asincrónico. El modo de transmisión asincrónico o en forma de carácter, trasmite lentamente la información, carácter por carácter, de tal forma que el receptor se prepara para recibir el siguiente carácter, después de haber recibido el anterior.

- Sincrónico. El modelo de transmisión sincrónico permite el envío simultaneo de varios caracteres en bloque, los cuales constituyen las unidades de envío, de tal suerte que se logra enviar una mayor cantidad de información en un menor tiempo. Normalmente se inserta caracteres de control al inicio y al final de cada bloque, con el fin de confirmar que la información no sufrió trastornos durante la transmisión.

3.7.2 Canales De Comunicación

Un canal de comunicación es el medio a través del cual viaja la información computacional entre dos puntos, generalmente distantes. La velocidad, capacidad y costo de transmisión varían según los elementos o medios de conexión.

Un aspecto importante de cada uno de estos elementos es la velocidad de transmisión, la cual denota la cantidad de bits por segundo que el elemento o medio puede trasmitir. Existen diferentes unidades de medida.

- pbs = bits por segundo (bits per second).
- Kbps = kilo bits por segundo (kilo bits per second).
- Mbps = mega bits por segundo (mega bits per second).
- Gbps = giga bits por segundo (giga bits per second).

3.8 Elementos de conexión

Se entiende por elementos o medios de conexión a los cables, tarjetas de red y otros equipos necesarios para conectar entre si los ordenadores. Dentro de los cables de conexión utilizados se encuentran:

3.8.1 Par Trenzado

Consiste en dos hilos trenzados de forma independiente y luego trenzados entre si, y recubierto de una capa aislante externa. Es de fácil instalación y ofrece cierta protección contra las interferencias externas.

Este medio de comunicación está relacionado con las líneas telefónicas y telegráficas. Para utilizarse este medio se requiere de un módem tanto en el lugar de donde se envía datos como en el lugar en donde se reciben.

Lo anterior se debe a que las señales que viajan a través de las líneas telefónicas o telegráficas son análogas y las de las computadoras son digitales. Esto hace necesario convertir las señales digitales y análogas y viceversa para poder transmitir la información.

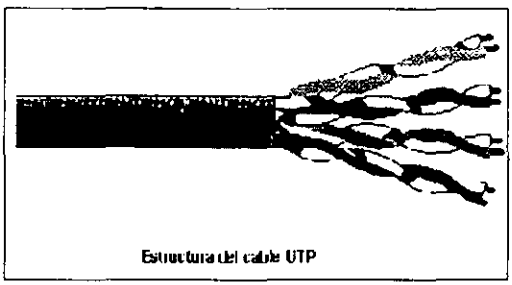


Figura # 9 Par Trenzado

3.8.2 Cable Coaxial

Consiste en un hilo de cobre envuelto en una malla trenzada. Entre ambos se encuentra una capa de material aislante. Hay dos tipos de función del grosor. Soporta comunicaciones en banda base y en banda ancha y ofrece mayor protección que el par trenzado apantallado frente a las interferencias externas.

Se utiliza para comunicaciones de datos en distancias cortas, menores de 15 kilómetros. El cable coaxial es útil en redes locales (LAN), los cuales se encuentran en un área geográfica pequeña como pueden ser las instalaciones de un edificio. Cuando se requiere conectar más computadoras a la red, no causa interrupción de las que ya se encuentran conectadas.

El cable coaxial permite transmitir datos a gran velocidad, es inmune al ruido y a la distorsión de Las señales enviadas, y es uno de los medios menos costosos cuando se trata de comunicación de corta distancia.

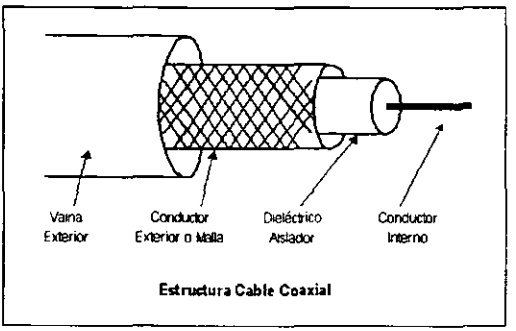


Figura # 10 Cable Coaxial

3.8.3 Fibra Optica

Esta formada por un núcleo de material transparente muy fino, rodeado de otro material con distinto índice de refracción. De esta forma, las señales luminosas que viajan por el núcleo son reflejadas por capas externas, llagando al extremo del cable. Permite mayor velocidad de transmisión de los datos, aunque resulta muy cara su instalación.

Este medio es utilizado por las compañías telefónicas con el objetivo de sustituir los cables que se usan para las comunicaciones para larga distancia. También se utiliza para instalar redes locales privadas.

Se realizan enviando pulsos de luz de la computadora fuente a la computadora destino. Dificulta agregar computadoras a la red cuando aquellas están funcionado. Es muy costoso por lo cual no se recomienda para distancias cortas.

Cuando se trata de transmitir información a larga distancia y además se requiere una lata velocidad, este medio resulta conveniente.

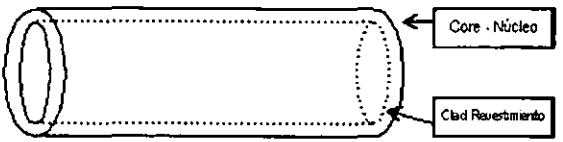


Figura # 11 Fibra Óptica

3.8.4 Ondas de Radio

Este medio de comunicación, además de usar las frecuencias normales de estaciones de AM y FM, utiliza onda corta o radiofrecuencia a distancias cortas. Las principales aplicaciones de este medio son en telefonía celular y en redes locales sin cableado. Es susceptible de sufrir interferencia cuando se utilizan otros medio que involucren frecuencias.

3.8.5 Microondas

Este medio se utiliza para comunicar datos a larga distancia. Sus principales características son que proporcionan velocidad y bajo costo. Es fácil de establecer, pero su uso presenta algunas desventajas debido a las condiciones del medio ambiente, sobre todo la interferencia que pueden provocar otras ondas de radio y los cambios atmosféricos que influyen en la transmisión de datos al modificar la señal que se envía.

3.8.6 Satélite

Este medio de comunicación es parecido alas microondas con la diferencia de que estas utilizan sólo estaciones terrestres y los satélites, además de estos también cuentan con estaciones de órbitas. Las comunicaciones via satélite permiten expandir las redes de comunicación de datos en forma sencilla, simplemente agregando mas estaciones. El uso de satélites puede presentar problemas de seguridad si la comunicación es interceptada por alguien que tenga el equipo receptor adecuado.

Capitulo IV. Caso Practico.
Desarrollo de un Middleware para optimizar tiempos de respuesta en la ejecución de operaciones en los Sistemas de Información Bancarios.

4.1 Esquema General del Banco

En la actualidad, la Red de Sucursales de Banco está administrada de dos formas distintas. La primera, de tipo técnico (por Dominios) y la otra de tipo territorial (Regiones).

Por Dominios. Cada Dominio tiene como máximo 45 sucursales conectadas. Actualmente se tienen 10 Dominios operantes dentro de los cuales se tienen organizadas a las Sucursales.

Por Regiones. Actualmente existen 5 regiones a nivel Nacional. Las regiones son: Noroeste, Norte, Centro, Metropolitana y Sur.

En este momento existen 366 Sucursales a Nivel Nacional de las cuales 142 están en la región Metropolitana, 82 en la región Sur, 55 en la región Centro, 48 en la región Norte y 39 en la región Noroeste. Además de las Sucursales existen 8 Módulos de atención a clientes.

El promedio de personas por Sucursal es de 7, de las cuales 4 son ejecutivos y 3 son cajeros. Existen cerca de 15 Sucursales que están fuera de este promedio con hasta 12 personas. Los Módulos de Atención (localizados dentro de empresas o escuelas) tienen 2 o 3 personas atendiendo a clientes. Cada una de estas personas tiene una terminal con la aplicación de Terminal Financiero. Este esta compuesto por las plataformas de Mostrador[2] y Negocios[3].

4.1.1 Enlaces y Disponibilidad

El ancho de banda disponible para las Sucursales es de 64Kb a nivel Nacional (32Kb destinados a voz y 32Kb destinados a datos), no existen excepciones. Este ancho de banda es suficiente para enlazar a 15 terminales. Dentro de las sucursales se tiene una red Ethernet para interconectar a las terminales con el servidor local. En el caso de la red externa (WAN) se trata de una topología de estrella de estrellas.[4]

4.1.2 Características Generales

Cada Sucursal cuenta con un servidor SUN ultra 5 el cual contiene la Base de Datos y los binarios de los diferentes servidores lógicos Tuxedo (Back End) los cuales se encargan de realizar las transacciones u operaciones del terminal financiero, asimismo se encuentran os programas ejecutables de Terminal Financiero y de los diferentes aplicativos (Front End).

[2] En el anexo # 1 se encuentra esquematizada la Plataforma de Mostrador
[3] En el anexo # 2 se encuentra esquematizada la Plataforma de Negocios
[4] En el anexo # 3 se encuentra el esquema de lo antes mencionado.

Es mediante el servidor que se gestiona y administra la comunicación de la Sucursal con el equipo central.

Asimismo cabe mencionar que la seguridad en las sucursales se maneja por medio de perfiles los cuales son asignados dependiendo al puesto que ocupe el usuario dentro de la sucursal. Esto se maneja localmente, cuando hay un cambio en la seguridad este es reflejado a nivel central y cada sucursal tendrá que replicar este cambio localmente esto es realizado por medio de servicios Tuxedo que lo que hacen es llevar la información necesaria de una lado al otro.

En cada sucursal el servidor SUN está dedicado a atender las Transacciones solicitadas por las de las terminales pertenecientes a la misma sucursal. Es en el servidor en donde se alojan los datos de identificación de la sucursal así como las características de todos los custodios. Aquí también se especifica, por medio de un archivo de configuración, cuando una petición será atendida por un servicio Tuxedo local o por uno remoto esto es por dominios.

El servidor cuenta con un Sistema Operativo Solaris v 5.2.1 (Unix).

Cuenta además con un manejador de Base de Datos de Oracle v7.3.3, en esta Base de Datos se almacenan los datos necesarios para la Seguridad del Sistema, las tablas de control, las bitácoras, el Diario electrónico y las afectaciones a Totales.

El servidor tiene instalado el middleware TUXEDO que administra las operaciones de comunicación con las terminales y con los sistemas remotos. TUXEDO. Además cuenta con un conjunto de servidores lógicos que fueron programados en C++ y C ansi que a su vez tiene implementados servicios que se encargan de realizar las transacciones que solicita el terminal financiero.

Los servidores hacen operaciones con las Base de Datos local y central (ambas en Oracle); así como también, tenemos servidores centrales que gestionan, en caso que la naturaleza del movimiento lo requiera, operaciones que realizan accesos a tablas de DB2.

Los servicios de estos servidores lógicos son invocados desde el terminal financiero (Front End) el cual se ejecuta desde las terminales que se encuentran en la sucursal las cuales están conectadas al Servidor SUN através del WSNADDR (WSL de cada servidor).

La llamada al Middleware TUXEDO comienza en el módulo de API de la parte del cliente el cual fue diseñado en Visual Basic 4.0, 5.0 y 6.0 que se emplea para invocar un servicio TUXEDO y comprende la transmisión de la solicitud por la red y la respuesta resultante.

TUXEDO además de ser un middleware muy poderoso, es ocupado en Banco como un monitor de transacciones, es decir permite administrar el manejo de las transacciones y permite que sé realicen copias de los servicios para poder atender las peticiones de varios clientes a la vez. La rapidez con que se ejecuten dichas peticiones depende en gran medida de las caracteristicas de la maquina en donde se encuentre la parte del monitor de transacciones de TUXEDO, además de la configuración y administración.

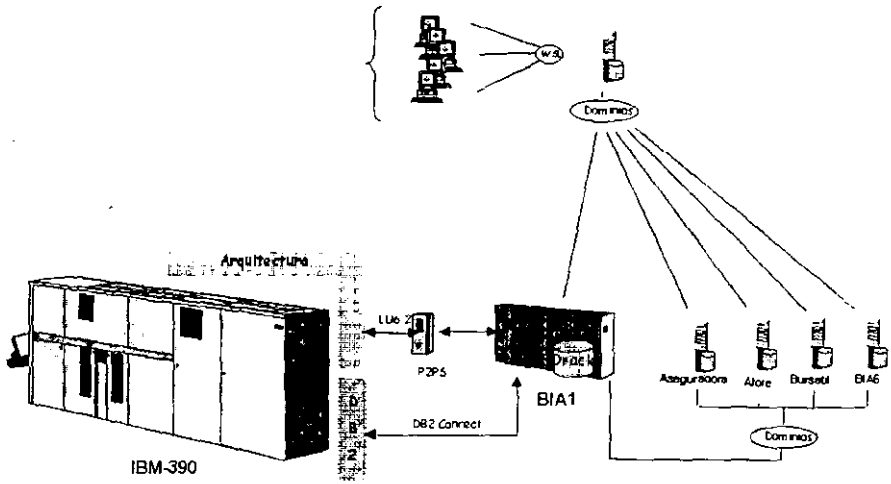


Figura # 12 Diagrama de Operación de TUXEDO

4.2 Problemática

En el esquema del banco la transmisión de mensajes es por medio de un tubo que empieza en el módulo del API de la parte del cliente que se emplea para invocar un servicio y comprende la transmisión de la solicitud por la red y la respuesta resultante a este se le denomina middleware. La administración de dichos mensajes es por medio de monitores transaccionales esto es ventajosa cuando estos, además de actuar como intermediarios entre el equipo central y las terminales, realizan operaciones de optimización que no pueden realizarse en los "clientes" y que serian costosas en tiempo y recursos para el equipo central. Pero esto se vuelve en una desventaja cuando TUXEDO solo se encarga de recibir el mensaje para retransmitirlo hacia otro punto, dado que los mensajes tienen que pasar por varios caminos antes de llegar a su destino final, formándose así los cuellos de botella. El problema que se presenta actualmente es que ha crecido tanto la red de sucursales

que ahora la maquina central ya no cumple con las características que necesita el monitor de transacciones para poder responder a las peticiones que realizan los clientes, además esto es producido en gran medida por la configuración de TUXEDO en el esquema del banco.

Por lo cual sé penso en un middleware que lograra enviar estas transacciones del terminal financiero por un canal directo hacia el equipo central los congestionamientos provocados por estos cuellos de botellas desaparecerían. Así mismo se tendría una mayor eficiencia en los flujos de comunicación disponibles para distintas aplicaciones que corren en la terminal. El manejo de la información del equipo central seria transparente.

4.3 TRADAL (Traductor de Mapas ALTAMIRA)

TRADAL, se desarrolló con el fin de proveer a las terminales un canal para la ejecución de transacciones CICS, las cuales se procesan en un HOST central IBM 390. Este componente se utiliza en aplicaciones orientadas al usuario final para un ambiente Windows NT.

4.3.1 Objetivo

La finalidad de este medio de comunicación directo entre el terminal financiero y el equipo central es la optimización del tiempo de respuesta en la ejecución de operaciones en línea.

Sus principales ventajas son:

- Permite una comunicación directa entre el terminal financiero y el HOST central.
- Proporciona una interfaz de programación sencilla para los programas de aplicación.
- Administra los canales de comunicación disponibles para las distintas aplicaciones que corren en la terminal.

- Su ejecución permite la disponibilidad del mismo canal de comunicación a distintas aplicaciones ejecutándose concurrentemente.
- Opera el manejo de datos del equipo central en forma transparente para el programador.

4.3.2 Esquema de operación. ActiveX Exe

Antes de abordar el tema del funcionamiento del componente TRADAL, es conveniente realizar una breve introducción a los conceptos básicos de la tecnología ActiveX, los cuales le hacen ser una herramienta fácil de programar.

Los inicios del COM se remontan a los tiempos en los que apareció OLE 1.0 con la idea de manipular texto y gráficos desde una misma aplicación, lo que exigía un intercambio de datos entre aplicaciones. Este requisito fue cubierto en Windows 3.0 por medio de la técnica DDE(Dinamyc Data Exchange – Intercambio dinámico de datos). Esta técnica consistía en que una aplicación contenedora mantuviera vínculos con ficheros creados por otras aplicaciones, de forma que los cambios producidos en esto ficheros por sus respectivas aplicaciones fueran reflejados automáticamente en la aplicación contenedora. Los problemas que esto implicaba fueron la dependencia de la estructura de directorios, pérdidas de vínculos y un elevado número de archivos concernientes a un único documento. Buscando mejores soluciones, Microsoft introduce con Windows 3.1 OLE (Object Linking and Embedding – Vinculación e incrustación de objetos). Esta técnica permite incrustar o vincular un objeto sin que la aplicación contenedora tenga que conocer el tipo de datos de dicho objeto. En este caso, para modificar el objeto incrustado, la aplicación contenedora invoca a la aplicación que fue utilizada para crear el objeto.

El problema ahora es que cada vez que se manipula el objeto habría que abrir la ventana correspondiente a la aplicación utilizada para crearlo, debido a la ausencia de un protocolo de comunicación eficiente entre las dos aplicaciones. Entonces nación COM.

COM(Component Object Model) es el modelo básico de componentes que permite a los mismos comunicarse independientemente del lenguaje y de la plataforma. No se trata de un lenguaje orientado a objetos, sino un estándar para el manejo de componentes. El lenguaje, la estructura y los detalles de implementación quedan para el programador de la aplicación.

COM especifica un modelo de componentes de software y los requisitos de programación que permite a los objetos COM interactuar con otros objetos. Este objeto puede estar dentro de un único proceso, en otros procesos, incluso en máquinas remotas. Asimismo, puede estar escrito en otro lenguaje y estructuralmente puede ser bastante diferente. Ésta es la razón por la que COM se define como un estándar binario; es un estándar aplicable después de que un programa haya sido traducido al código ejecutable.

ActiveX es básicamente una extensión de OLE en un intento por incorporar ese concepto al desarrollo de aplicaciones para Internet.

Los componentes ActiveX interactúan con la aplicación y entre sí mediante una relación cliente-servidor. Dependiendo de cómo estén implementados, pueden ejecutarse en el mismo proceso que la aplicación cliente o en un proceso diferente.

TRADAL es una aplicación habilitada para ActiveX que se ejecuta en un proceso diferente a la aplicación cliente, esto permite que el mismo componente este disponible para varias aplicaciones mediante la concentración de sus operaciones para optimizar los medios de comunicación y proporcionando a los programas, una interfaz de comunicación sencilla.

4.3.3 Diagrama de operación de TRADAL

El funcionamiento de TRADAL se esquematiza y resume en esta figura:

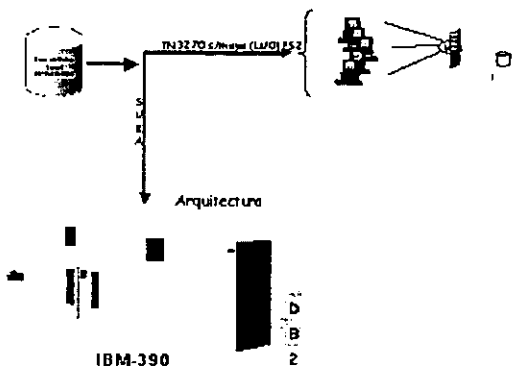


Figura # 13 Operación del SUBA-TRADAL

1. El programa de Visual Basic crea una instancia de la clase "Conexion390" y ejecuta el método conectar para abrir la sesión en 390. Si el método devuelve *false*, significa que no ha sido posible establecer la conexión.
2. Con la sesión obtenida, el programa envía datos a 390 en formato de tramas separadas por "|" (pipes) a través del método "EnviaRecibe", en donde se indica además de otros datos, la transacción CICS que se desea ejecutar.
3. Con esta petición, el TRADAL busca dentro de una base de datos local en Access, los datos correspondientes a la ejecución de esta transacción, para codificarla en formato 390.
4. Se envía entonces la información codificada hacia el HOST y se espera la respuesta. En caso de errores en la comunicación, la ejecución del método se termina y devuelve como valor un entero diferente de 0.
5. Se recibe la respuesta de 390, se consulta nuevamente la base de datos para decodificar los datos y devolver la información al programa de Visual Basic.
6. El programa recibe los datos.

Los detalles de su operación, así como la integración del componente a una aplicación programada en Visual Basic se relatarán en el punto **Implementación de la clase Conexión390 en un proyecto de Visual Basic**.

4.3.4 CICS.

Dado que TRADAL trabaja con transacciones de un ambiente CICS, es necesario aclarar algunas cuestiones básicas sobre el tema.

CICS (Customer Information Control System) es un software de propósito general para el procesamiento de transacciones en línea desarrollado por IBM.

Este subsistema controla aplicaciones en línea y de base de datos, proporcionando además un conjunto de funciones estándar requeridas por los programas de aplicación para la comunicación con terminales locales, remotas u otros sistemas.

CICS actúa como una interface entre los programas de aplicación y el sistema operativo que puede ser configurada para dar soporte a redes con una amplia variedad de terminales y subsistemas.

De acuerdo al procesamiento que pueden recibir las llamadas a los programas de aplicación desde una terminal, podemos encontrar estas dos estructuras soportadas por CICS:

1. **Transacción no conversacional.** El requerimiento es inmediato y rara vez requiere de seguimiento. La ventaja de este tipo de transacciones es que hacen muy pocas demandas sobre los recursos de cómputo. Son consultas de una sola vez y usualmente rápidas de ejecutar.

2. **Transacción pseudoconversacional.** En este tipo de transacciones se hace una serie de requerimientos. Esto demanda una mayor atención de los recursos de computo, pues generalmente las consultas se realizan en varias ocasiones y las peticiones siguientes están fuertemente asociadas con las consultas previas.
   Se emplea el término 'pseudoconversacional' porque mientras el cliente esta preparando los datos para la siguiente llamada al programa de CICS, el sistema operativo tiene el control de los recursos de cómputo. El éxito de esta transacción está en que los recursos del mainframe se ceden al programa sólo hasta que éste tiene los datos suficientes para continuar su ejecución.

Para interactuar con el CICS es necesario trabajar desde una terminal que cuente con los permisos necesarios en el ambiente de trabajo del equipo central.

Primero debe firmarse desde la terminal para poder accesa o teclas AID (Attention Identificaction) en CICS. Las teclas AID son un conjunto de teclas que disparan una interrupción que el mainframe puede identificar y comprenden desde la tecla PF1 a PF24, las teclas PA1 y PA2, la tecla CLEAR y la tecla ENTER.

CICS es una herramienta poderosa en el equipo central, pero para interactuar con él, la terminal requiere de un mecanismo de comunicación de datos a través de la red. Este medio se describe en una serie de estándares indicados en los RFC(Request for comments) de Internet, que se resumen principalmente en el TELNET 3270 y sus variantes.

El TELNET 3270 permite a la terminal 'emular' una extensión del ambiente en el HOST central de acuerdo a características particulares del equipo, como caracteres por pantalla, líneas de impresión, capacidad para la paginación de memoria, etc.

La gran desventaja de este tipo de emulación se encuentra en que la interfaz que se muestra al usuario es la misma que ofrecen los programas de CICS, los cuales se programan en COBOL, y obviamente, su manera de operar es complicada para la mayoría de los usuarios.

Para operar los datos emitidos por la transacción desde un front-end diferente al proporcionado por las herramientas de emulación 3270, TRADAL utiliza un tipo de emulación 3267 que permite manejar la información principal ignorando los detalles de la presentación, de los cuales se encargará la aplicación que haga uso de los datos que TRADAL devuelva.

4.3.5 La base de datos TRADAL.mdb

En la ejecución de transacciones por medio del TRADAL, es necesario incluir cierta información en la base de datos, la cual se utiliza para dar formato a los datos de envío y recepción, de acuerdo a la arquitectura de la transacción en ALTAMIRA. El archivo de la base de datos es TRADAL.mdb, creado en Microsoft Access.

Dado que la información en la base de datos es la que determina las características de cada transacción, el mantenimiento que requiere TRADAL cuando se suscitan cambios en la arquitectura de transacciones, debe reflejarse principalmente en éste archivo.

La ubicación del archivo mdb para TRADAL se define en el archivo Suba.ini, bajo la llave [MDB]. Más adelante se explicará con detenimiento.

Para ingresar nuevos registros a la base de datos se cuenta con una aplicación para la captura y modificación de datos, el Editor TRADAL (editorTRADAL.exe).

Estructura de la base de datos

La base de datos del TRADAL se compone de 6 tablas que son:

- **Campo.** Descripción de campos para transacciones 390.

  Llave: *id_campo*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
Id_campo	Numérico	Entero largo	Identificador del campo
Id_transaccion	Numérico	Entero largo	Identificador de la transacción a la que pertenece el campo
Nombre	Texto	16 caracteres	Nombre del campo
Descripcion	Texto	50 caracteres	Breve descripción del campo
DelimitadorASCII	Numérico	Entero largo	Valor decimal del delimitador del campo en código ASCII
Longitud_maxima	Numérico	Entero largo	Longitud máxima del campo
direccion	Texto	1 caracter	Indica si el campo es de Entrada o salida
Constante	Texto	32 caracteres	Permite la entrada de un valor constante al campo
PosInicio	Numérico	Entero largo	Posición en la que se encuentra el inicio del campo. Válido únicamente para campos TS de ALTAMIRA

error	Numérico	Entero largo	Si el valor es 1, indica que el campo se muestra solo al ocurrir un error
LoopBack	Numérico	Entero largo	Si el valor es 1, indica que el campo es de entrada y se desea devolver en la trama de salida.

- **Campo_servicio.** Relación de campos asociados a un servicio.

  Llave: *id_campo_servicio*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
Id_campo_servicio	Numérico	Entero largo	Identificador del campo
Id_servicio	Numérico	Entero largo	Identificador del servicio al que pertenece el campo
nombre	Texto	16 caracteres	Nombre del campo en el servicio
direccion	Texto	1 carácter	Indica si el campo es de entrada (0) o de salida (1)
descripcion	Texto	50 caracteres	Descripción del campo.
DelimTrama	Texto	1 carácter	Delimitador del campo en la trama

- **CampoServicio_Campo.** Asociación de los campos de un servicio con los campos de una o más transacciones.

  Llave: Compuesta por *id_campo_servicio* y *id_campo*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
Id_campo_servicio	Numérico	Entero largo	Identificador del campo en el servicio
Id_campo	Numérico	Entero largo	Identificador del campo en la transacción

- **Flujo.** Asocia un servicio con una o varias transacciones.

  Llave: Compuesta por *id_servicio* y *id_transaccion*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
Id_servicio	Numérico	Entero largo	Identificador del servicio
Id_transaccion	Numérico	Entero largo	Identificador de la transacción
Orden	Numérico	Entero largo	Indica el orden de ejecución de las transacciones para este servicio

- **Servicio.** Tabla de servicios disponibles.

  Llave: *id_servicio*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
Id_servicio	Numérico	Entero largo	Identificador del servicio
Nombre	Texto	10 caracteres	Nombre del servicio
Descripción	Texto	255 caracteres	Descripción del servicio

- **Transaccion.** Transacciones disponibles.

  Llave: *id_servicio*

  Campos:

Nombre	Tipo de datos	Longitud	Descripción
id_transaccion	Numérico	Entero largo	Identificador de la transacción
Nombre	Texto	10 caracteres	Nombre de la transacción
Estado	Texto	1 caracter	Indica si la operación es transaccional (I) o conversacional (C).
Teclaf	Texto	1 caracter	Simulación de la tecla que debe 'oprimirse' en 390
Caso	Texto	1 caracter	Informa a la transacción del tipo de emulador que la esta solicitando. En el caso de

			TRADAL es 1.
Descripcion	Texto	40 caracteres	Descripción de la transacción
Online			Indicado si la transacción es Online

4.3.6 Configuración del TRADAL

Muchos de los parámetros que utiliza el TRADAL y sus aplicaciones asociadas para la conexión con ALTAMIRA son especificas de la terminal en la que se ejecutan, además, es común la necesidad de modificarlos con relativa frecuencia. Por esta razón, los valores referentes a la dirección IP del Host al que se realizará la conexión, el nombre del ambiente CICS, el nombre de la terminal en las tablas de arquitectura de ALTAMIRA y la dirección de la base de datos de TRADAL se localizan dentro de un archivo ini y no se inicializan como parámetros dentro de código. El TRADAL se encarga de obtener estos valores del fichero Suba.ini e interpretarlos, sin embargo, la ubicación de este archivo también es variable, así que para localizarlo requiere de algunas indicaciones en el archivo Win.ini de la terminal.

WIN.INI

Este archivo se encuentra generalmente en C:\WINNT, contiene parámetros de configuración para programas instalados en la computadora, así como información para perfiles de usuario, y de configuración regional. Para que TRADAL pueda localizar el archivo SUBA.INI, el WIN.INI debe contener una sección llamada [SUBA] con la siguiente información:

```
[SUBA]
ServerName=g:
SubaDirectory=\platafor
```

En donde:

- **ServerName** indica el nombre de la unidad de disco en la que se localiza el archivo SUBA.INI. Dicha unidad puede ser local o remota. En este ejemplo el drive donde se encuentra el archivo es **g:**; en esta unidad, las máquinas de las sucursales se conectan a un servidor local que utilizan como *File Server*(Servidor de archivos).
- **SubaDirectory** contiene el path del directorio en el que se ubica el archivo SUBA.INI.
- La combinación de ambos valores genera la ruta completa del archivo SUBA.INI ('g:\platafor' en este ejemplo).

SUBA.INI

Con los datos del WIN.INI, TRADAL consigue la ubicación del archivo SUBA.INI. El siguiente es un formato general de este archivo.

```
[Channel<Canal>]
CICS=<CICS>
<IP TERM> = <IP HOST>,<TERM ID>

[Version]
Numero = 3

[MDB]
ArchivoMDB=c:\TRADAL.mdb
```

[Channel<Canal>]. Bajo esta llave se indican los valores de configuración para la comunicación con el HOST central, y puede aparecer en varias ocasiones dependiendo del número de opciones posibles de conexión para la terminal. Dependiendo del canal informado a TRADAL, se elegirá el apartado CHANNEL y sus valores asociados para realizar la conexión.

CICS: Contiene el nombre del ambiente CICS a ejecutar.

<IP>: El nombre de esta variable no es constante, pues se refiere a la dirección IP de una terminal y su valor esta compuesto por la dirección IP del HOST de 390 al que se conecta esta terminal y el identificador de la terminal en las tablas de arquitectura separados por un signo ',' (coma). Por ejemplo:

100.3.9.18 = 120.20.1.168,E$03

Significa que la computadora con IP 100.3.9.18 realizará la conexión al Host con dirección IP 120.20.1.168 y utilizará E$03 como nombre de terminal. Esto permite que un mismo archivo SUBA.INI pueda especificados para cada terminal en un archivo SUBA.INI compartido.

[MDB]

ArchivoMDB: Esta variable contiene la ruta donde se encuentra la base de datos TRADAL.MDB

4.3.7 Editor TRADAL

Esta aplicación se utiliza para editar el contenido de la base de datos del TRADAL, permitiendo un fácil acceso a las estructuras de control utilizadas por TRADAL para el manejo de servicios y la ejecución de transacciones.

El fichero mdb que utiliza este editor será el indicado en el Suba.ini bajo la llave [MDB]. Para accesar a un mdb diferente, cambie el valor de esta llave con el valor del path del archivo mdb que desea modificar.

Menú principal

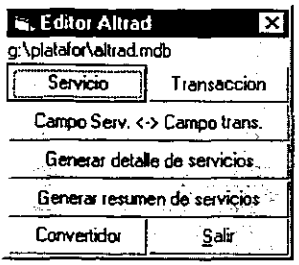


Figura # 14 Menú principal

El menú principal del Editor TRADAL consta de las siguientes partes.

Dirección de la base de la base de datos. Indica la ruta del archivo mdb que se esta editando.

- **Servicio.** La base de datos agrupa las transacciones en 'Servicios', los cuales definen las tramas a recibir por el TRADAL para ser traducidas posteriormente al formato de datos que el CICS espera. Un servicio se compone de una o muchas transacciones.
- **Transacción.** Las transacciones son propiamente los programas que se ejecutan en el CICS y definen las características de ejecución para cada servicio de TRADAL mediante el flujo de llamadas a 390.
- **Campo servicio – Campo transacción.** Aquí se especifican las relaciones entre los campos definidos para el servicio y para la transacción. Más adelante se explicará como funciona esta relación.
- **Detalle de servicios.** Esta opción genera un listado detallado de los servicios existentes, las transacciones que lo componen, la relación entre sus campos, y los detalles de cada transacción. Esto se guarda un archivo de texto.
- **Resumen de servicios.** Esta opción genera un listado general de los servicios existentes y las transacciones que lo componen. También se guarda un archivo de texto.

- **Convertidor.** Los campos de una transacción en 390 son identificados mediante un delimitador especial. Este delimitador tiene la longitud de un byte y se define en forma hexadecimal para la arquitectura de aplicaciones de ALTAMIRA. Sin embargo, el TRADAL requiere su equivalente decimal para poder operarlo. Además, el equipo IBM 390 utiliza el código EBCDIC para la impresión de caracteres, así como para su lectura desde una comunicación directa, por lo que es necesaria la conversión de los delimitadores indicados en la arquitectura de ALTAMIRA (que están en EBCDIC hexadecimal) a un código ASCII decimal. El convertidor realiza estas conversiones.

- **Salir.** Termina la ejecución del Editor TRADAL.

Servicio

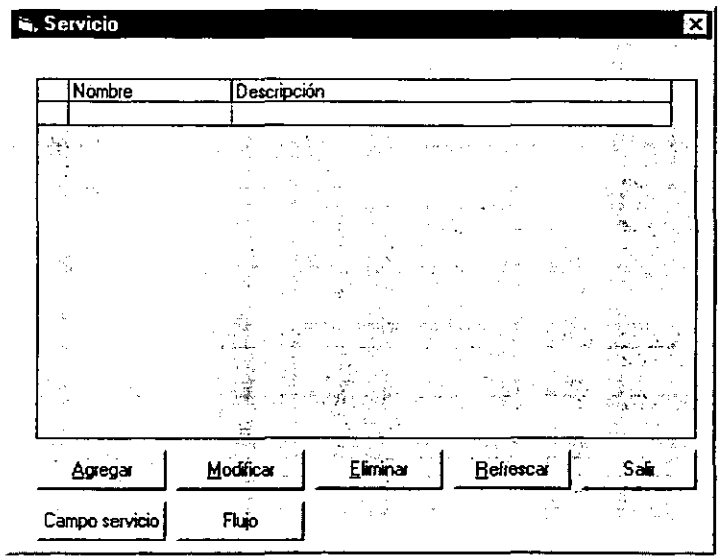


Figura # 15 Servicio

La pantalla muestra el listado de los servicios que actualmente están disponibles en la base de datos. Las opciones disponibles para el menú de Servicios son:

- **Agregar.** Muestra la pantalla de captura para ingresar un nuevo servicio.
- **Modificar.** Permite modificar los datos de un servicio ya existente seleccionado en el grid.
- **Eliminar.** Elimina de la base de datos el servicio seleccionado en el grid. Para poder borrar un servicio, es necesario eliminar antes todas sus relaciones.
- **Refrescar.** Actualiza los datos desplegados en la pantalla.
- **Salir.** Cierra la opción 'Servicios'.
- **Campo servicio.** Muestra la pantalla de captura para agregar, eliminar o actualizar los campos del servicio.
- **Flujo.** Muestra la ventana para actualizar o modificar las relaciones entre los servicios y las transacciones.

Agregar y modificar servicios

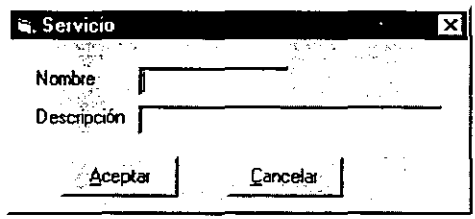


Figura # 16 Agregar y modificar servicios

La opción 'Agregar' del menú de servicios muestra la pantalla de captura para el nuevo servicio. El nombre es un identificador de hasta 10 caracteres y la descripción es una breve explicación del uso o el significado de este campo, su valor es opcional y su longitud máxima es de 255.

En la modificación de servicios se habilita esta pantalla con los datos actuales del servicio a modificar.

Actualizando los campos del servicio

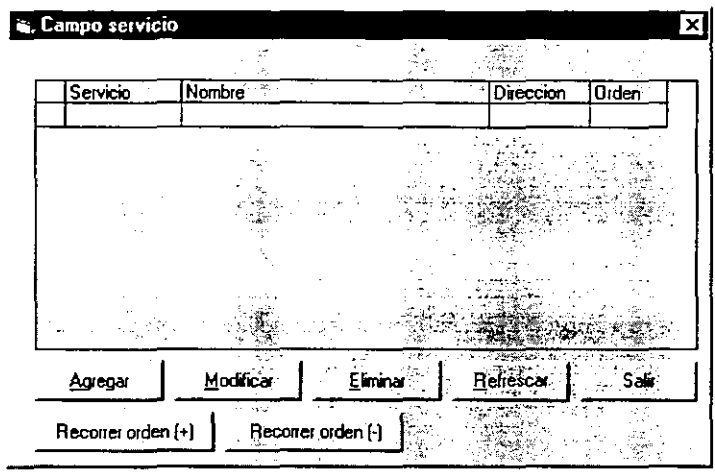


Figura # 17 Actualizar campos

La opción 'Campo servicio' del menú de servicios muestra los campos que integran el servicio seleccionado como se ve en la siguiente pantalla.

Las operaciones de agregar nuevos registros, modificar los campos ya existentes, eliminar campos, refrescar el contenido de la pantalla y la de cerrar la pantalla son similares a las que existen en el menú 'Servicio'. Las opciones 'Recorrer orden(+)' y 'Recorrer orden(-)' se utilizan para aumentar o disminuir en 1 el valor que tienen los campos del servicio en 'Orden'. Como las tramas de los servicios son posicionales, la propiedad 'Orden' de los campos indica su ubicación dentro de la

trama, así es que, si se desea agregar o eliminar un campo intermedio deberá usar alguna de estas opciones.

Agregar campo

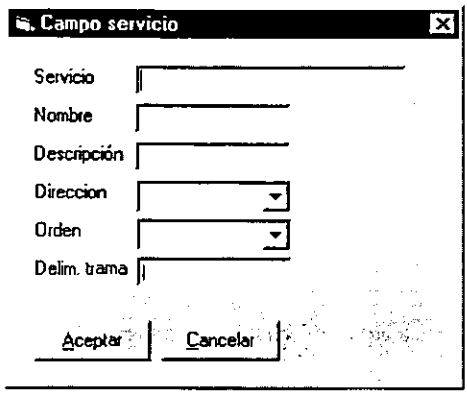


Figura # 18 Agregar Campo

La pantalla de captura para campos del servicio que se muestra después de seleccionar la opción 'Alta' del menú 'Servicio' requiere de la siguiente información:

- **Servicio.** Este dato es informado automáticamente por el Editor TRADAL y se refiere al nombre del servicio al que pertenecerá el campo.
- **Nombre.** Nombre del nuevo campo. Puede tener un máximo de 16 caracteres.
- **Descripción.** Breve explicación del uso o significado del campo.
- **Dirección.** Indica si el campo es de entrada (información a enviar) o de salida (información de respuesta).
- **Orden.** Especifica el orden del campo dentro de la trama, ya sea de entrada o de salida.

- **Delimin. Trama.** Se refiere al carácter que se utilizará para indicar el fin del campo dentro de la trama.

Flujo del servicio

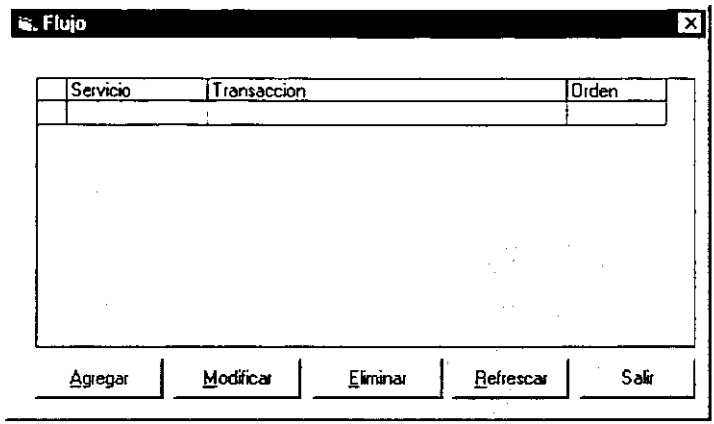


Figura # 19 Flujo del servicio

El flujo del servicio se compone de las transacciones que el TRADAL debe ejecutar cuando se realice la llamada al servicio. El servicio no es un concepto de ALTAMIRA, sino de TRADAL y es una manera de asociar transacciones, además de ser la interfaz entre los datos enviados por la aplicación cliente y los datos que el HOST central procesa.

El orden muestra el lugar de ejecución para cada transacción dentro del servicio.

La pantalla de captura para el flujo del servicio se muestra a continuación.

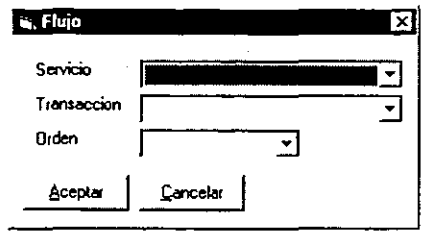


Figura # 20 Captura para el Flujo del Servicio

El servicio corresponde al nombre del servicio cuyo flujo se desea crear, la transacción debe ser aquella que deseamos incluir en el flujo, y el orden indica en que lugar de ejecución se llamará a este servicio, o sea, si la transacción será la primera, la segunda o la n-ava en ejecutarse.

Imaginemos que un servicio ABCDE tiene asociadas las transacciones AA01, AA04 y BB10, cuyo valor en la propiedad orden es 1, 2 y 3 respectivamente. Esto significa que una llamada al servicio ABCDE desde el TRADAL hará que este ejecute primero la transacción AA01, después la AA04 y por último la transacción BB10.

Transacción

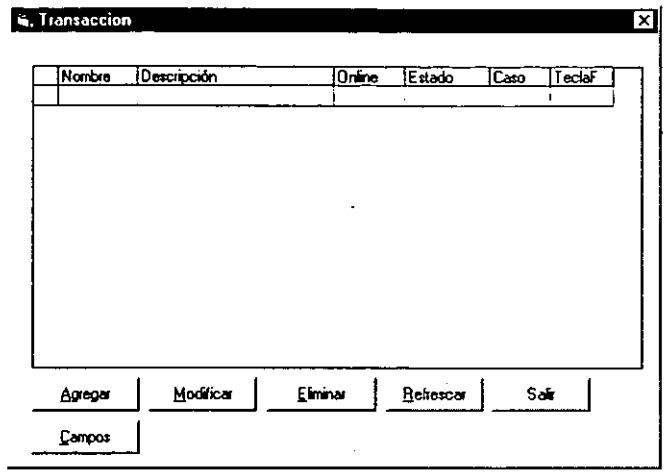


Figura # 21 Transacción

En esta pantalla se manipulan las características concernientes a las transacciones que serán utilizadas a través de TRADAL.

Las transacciones son nuestro punto básico de comunicación con CICS. Las características tanto de la transacción como de sus campos, se encuentran especificadas en la arquitectura de ALTAMIRA.

La siguiente figura muestra la pantalla para el alta y la modificación de transacciones.

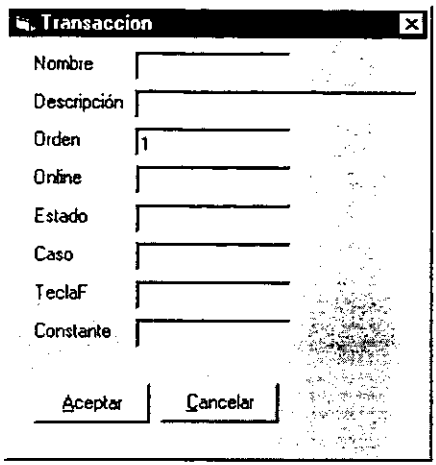


Figura # 22 Alta y Modificación de una Transacción

- **Nombre.** Es el identificador de la transacción indicado en la PCT (Program Control Table) de ALTAMIRA.
- **Descripción.** Comentario breve del propósito o uso de la transacción.
- **Online.** Indica si la operación se realiza en línea (N) o no (F). Generalmente su valor es (N).
- **Estado.** Se utiliza para informar que la transacción es conversacional (C) o no (I).
- **Caso.** Informa al CICS el tipo de llamada que se esta utilizando para ejecutar la transacción. Para TRADAL es 1.
- **TeclaF.** Corresponde a la tecla AID (Attention Identificaction) que la transacción espera para iniciar el procesamiento de los datos enviados. En algunas ocasiones se requerirá mas de una tecla AID; para estos casos se recomienda indicar solo la tecla más representativa.

Si se tiene la opción de utilizar una emulación 3270 para accesar a la arquitectura de aplicaciones del CICS estos datos pueden obtenerse del formato para la transacción.

La arquitectura de transacciones se accesa desde una aplicación de CICS, cuyo nombre puede variar de equipo a equipo dependiendo de la organización.

El siguiente es un ejemplo del formato en CICS para una transacción.

```
      0001 D$8F              ARQUITECTURA DE APLICACIONES          DA01
05/01/01
      5216 CIDFDT01                MANTENIMIENTO DE TRANSACCIONES
XXXX 13:51:38
       Transaccion ..... = XXXX : <DESCRIPCION>                    NEW COPY

       Aplicacion ..... = AH AHORRO CAPTACION
_______ESTADO_________
       Programa ..... = PAHU821                (A/D) A Activada
       Plan DB2 ..... = RCUP0211
_________ALTAMIRA___________
       Formato/Mapa ent. = CAHW0211 / PAHM821      Tipo Altamira.... = M
(N/E/M)
       Codigo de ayuda.. = CU21              Entrada..... = CONS-CUE
       Longitud Commarea = 3000
       Camb.Ses/Recuper. = N / N (S/N) : (S/N)
_____INFORM.ADICIONAL_____
       Tipo (Trans/Conv) = C (T/C)          Transac. local .. = XXXX
       Contable / Cajero = N : (S/N) : (A/B/ ) Documentos ..... =
       Tipos operacion.. = ABMC O (A-B-M-C-E-P-O) Diario Elec. .... = N (S/N)
       Inicio desde TERM = S (S/N)          APB 4700 ..... = N (S/N)
       Inicio por Arquit.= N                                    (S/N)
_____________STAMPS_____________
       PFs standard = S (S/N) Cambio estado = 22/04/98 19:40 ISOTZP0
       Actualizar tecleo = S (S/N) Alta ..... = 25/02/94 CICSUSER
       Pintar fast-path = N (S/N) Ult.mod. .... = 05/12/00 10:44 EJCG000
       Pfs por Arquitec.= S (S/N) Primer Uso .. = 25/02/94
       Tiene ayuda activa= N (S/N) Ultimo Uso .. = 05/01/01

     F2 Modif. F3 Alta F4 LIMPIA F5 Frmto. F6 Baja F8 Pfs. 10 ALTAM. CL
Borra
```

Tabla # 4 Formato en CICS para una Transacción

Solo se explican los valores de interés para TRADAL.

- **Transacción:** Muestra el identificador de la transacción y su descripción. Corresponden a los valores de l mismo nombre en la pantalla de captura de transacciones en el Editor TRADAL
- **Tipo (Trans/Conv).** Se refiere al tipo de transacción y corresponde al campo 'Estado' del editor TRADAL. El valor **'C'** corresponde a las conversacionales y **'T'** a las transaccionales.
- **Inicio desde TERM.** Un valor **'S'** indica que la transacción puede ser llamada 'directamente' desde la terminal, o sea, sin necesidad de activar un menú previo. Las transacciones marcadas con **'N'** no pueden ser llamadas por TRADAL.

Captura de campos para una transacción

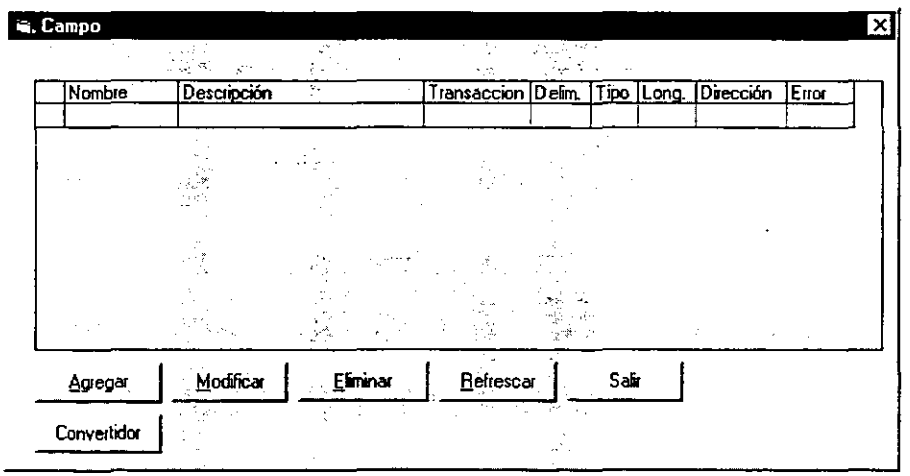


Figura # 23 Captura De Campos para una Transacción

Para agregar campos a una transacción seleccione la opción 'campos' de la pantalla 'Transacción'. Esta es la forma para la manipulación de campos para transacciones.

De las funciones que ofrece solo se explican la de **Agregar** y **Convertidor** puesto que las otras son similares a aquellas del mismo nombre que se han explicado en otras pantallas.

Agregar. Muestra la pantalla para la captura de un nuevo campo.
Convertidor. Presenta el convertidor de código ASCII – EBCDEC.

Los datos a indicar en la pantalla para el alta de campos son:

- **Transacción.** Lo escribe el editor en forma automática y corresponde al nombre de la transacción que incluirá este campo.
- **Nombre.** Es el nombre del campo. Se sugiere utilizar el mismo que en la arquitectura de 390 aunque no es indispensable.

- **Descripción.** Breve explicación de lo que el campo representa o utiliza.
- **Delimitador ASCII.** Código ASCII del delimitador para este campo, el cual debe informarse con un número decimal. A la derecha de éste campo se muestra un botón que activa el convertidor ASCII – EBCDEC.
- **Tipo.** Este campo ya no es operable.
- **Longitud máxima.** Longitud máxima del campo activo en caracteres.
- **Dirección.** Muestra si el campo es de entrada(1), de salida(2) u indefinido(3). El valor 'indefinido' indica que el campo puede ser opcional de entrada u opcional de salida.
- **Constante.** Si se informa un valor para este campo, cualquier ejecución de esta transacción utilizará siempre ese valor para este campo.

- **Pos. Inicio.** Cuando la transacción no opera con delimitadores, debe indicarse entonces la posición en la que inicia este campo. Por lo general es la práctica la que hace identificar estos casos especiales.
- **Error.** Cuando el valor de esta propiedad es 1 (Si), el campo sólo se incluirá en la trama si ocurrió un error. Solo funciona en campos de salida.
- **LoopBack.** Si su valor es 1 (Si), el campo estará disponible para mostrarse en la trama de salida. Esto opera solo en los campos de entrada y en algunos casos sirve para corroborar la información que recibió la transacción y guardarla en un Log local.

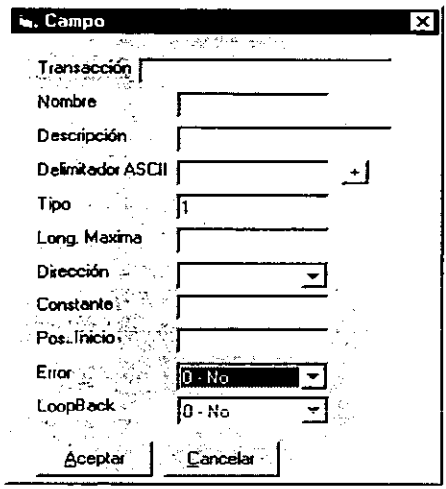


Figura # 24 Alta de Campos

Casi todas las propiedades que deben ingresarse en esta pantalla se obtiene del formato de la transacción en la arquitectura de aplicaciones del CICS. Para obtenerla es recomendable hablar con el personal encargado de controlar el ambiente.

Este es un ejemplo del listado de campos para una transacción.

```
0001 DS8F          ARQUITECTURA DE APLICACIONES          DA01
05/01/01
5216 CIDFDT01          LISTADO DE CAMPOS          XXXX 17:58:35

P SALTO  FORMATO: CAHW0211 CAHW0211                  L  1: 39
CAMPO____LITERAL_____________DELIM_ATT__LNG__TIP_RUTINA__VRUT_
LIT470
    01 CLAVCTA  NUMERO DE CUENTA    03  A  11  R QR2CCNV  S
    02 NOMBRE   NOMBRE DEL TITULAR  28  A  50  S        N
    03 PERCALC  PERCALC             06  X   3  S        N
    04 INDFECA  INDFECAL            27  X   1  S        N
    05 CAPITAL  CAPITAL             08  A  20  S        N
    06 PEAPLIC  PEAPLIC             09  X   3  S        N
    07 METLIQU  METLIQUI            07  X   2  S        N
    08 LIBRESA  LIBRESA             0B  X  20  S        N
    09 DISPONI  DISPONI             0C  X   3  S        N
    10 INDCONF  INDCONF             0A  X   1  S        N
    11 MESES    MESES               0D  X  20  S        N
    12 BLOQUEO  BLOQUEO             0E  X  20  S        N
    13 OTBLOQ   OTBLOQUEO           0F  X  20  S        N
    14 DISPON1  DISPON1             10  X  20  S        N
    15 INTER    INTER               11  X   1  S        N
    16 CORREO   CORREO              12  X   2  S        N

F2 Mantt F3 Alta  F4 Ay.Ac F6 Relac 10 Mover F7 Re.Pa F8 Av.Pa Cl Salir
```

Tabla # 5 Ejemplo Campos de una transacción

- **Campo.** Se sugiere que este sea el nombre que se indique al campo en el editor TRADAL.
- **Literal.** Es una breve descripción del campo.
- **Delimitador.** Corresponde al delimitador para esta transacción. Aquí, los delimitadores están informados en Código EBCDIC con un número hexadecimal. Es necesario convertirlos a su valor en ASCII.
- **ATT.** Indica el tipo de campo. No se informa dentro del editor.
- **LNG.** Muestra la longitud del campo. Corresponde a la propiedad 'Longitud máxima' en el editor.
- **TIP.** Informa si el campo es de entrada (R), de salida (S) u opcional de entrada o salida (O).

Relación entre campos de un servicio y campos de sus transacciones asociadas

La existencia de servicios y transacciones requiere un punto de convergencia para ambos conceptos. Además de la relación de flujo entre el servicio y las transacciones que lo integran, se debe indicar la correspondencia entre los campos de uno y otro. Esto deja abiertas las siguientes posibilidades:

1. La existencia de campos en el servicio que no están asociados a ningún campo de alguna transacción. Esto sucede cuando dentro de la trama de entrada se desean registrar datos, por un motivo independiente a la ejecución de transacciones y que por tanto TRADAL debe ignorar.

Como ejemplo suponga que se tiene un servicio llamado CNCTA para la consulta de saldos con la trama de entrada

<Folio de operación>|<Numero de cuenta>|

y una transacción CC01 para realizar la consulta que espera recibir como información únicamente el número de cuenta. El primer dato se informa, pero para que sea utilizado debe asociarse al campo de cualquier transacción, si no se hace, simplemente será ignorado.

La otra posibilidad es la de informar campos ficticios en la salida del servicio. Utilizando el ejemplo anterior diremos que el servicio CNCTA tiene una trama de salida con la forma

<Código de error>|<Vacio>|<Saldo>|

y que el campo <Vacio> no se trata de un dato devuelto por ninguna transacción sino que solo se desea en la salida para propósitos de estandarización.

2. La existencia de campos en la transacción que no se asocian a ningún campo del servicio. Esto significa que podemos crear un servicio que nunca informe un determinado campo de la transacción, en el caso de las entradas, o que muestre sólo algunos de los campos que la transacción devuelve, tratándose de las salidas.

Para ilustrar el caso referente a la información que envía el servicio, imagine una transacción llamada RT02 que realiza un retiro a una cuenta de cheques. Requiere de tres datos: el número de cuenta, el importe y opcionalmente un número de referencia. Podemos crear un servicio con el mismo nombre (el nombre de un servicio es un dato arbitrario) con sólo dos campos de entrada:

CUENTA e IMPORTE, para el número de cuenta y el importe de la operación respectivamente. Ninguna de las llamadas a este servicio informará el número de referencia. Cabe señalar que el dato no informado es opcional, por lo que esto no impedirá la ejecución de la transacción, pero los campos requeridos (obligatorios) deben informarse siempre.

Para asociar los campos de un servicio y transacciones se requiere primero que exista un flujo entre el servicio y las transacciones deseadas.

Cuando en la pantalla de 'Relación Campo servicio Campo transacción' dentro la lista 'Servicio' seleccione el servicio, sus campos aparecerán en el grid 'Campos del servicio', y la lista 'Transacción' estarán disponibles las transacciones relacionadas con el servicio especificado.

Elija la transacción cuyos campos desea relacionar para verlos en el grid marcado como 'Campos de la transacción', y la lista 'Transacción' estarán disponibles las transacciones relacionadas con el servicio especificado.

Elija la transacción cuyos campos desea relacionar para verlos en el grid marcado como 'Campos de la transacción'.

Seleccione del grid correspondiente el campo del servicio al que desea asociar un campo de la transacción, el cual seleccionará del grid 'Campos de la transacción'. Oprima el botón con una flecha hacia la derecha (->), y automáticamente la nueva relación se mostrará en el grid 'Relación entre campos del servicio y campos de la transacción'.

Los campos del servicio ya asociados no se muestran en el grid de 'Campos del servicio', puesto que no pueden asociarse a más de un campo de transacción. En cambio, todos los campos de la transacción estarán disponibles para asociarse a más de un campo del servicio.

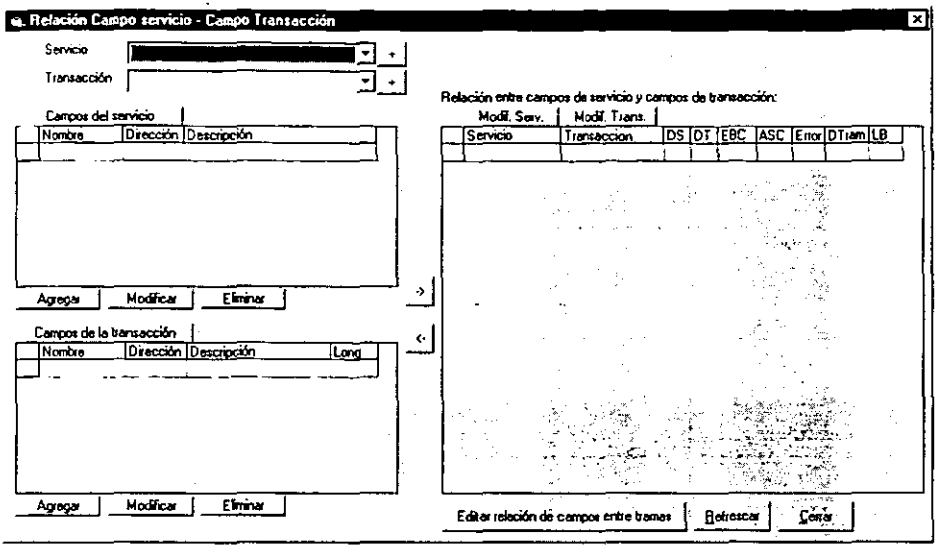


Figura # 25 Relación entre campos de un servicio y campos de transacciones asociadas

Para eliminar una relación presione el botón (<-).

Además de la asociación de campos, esta pantalla permite accesar a otras opciones disponibles en el menú principal del editor.

- **Campos de la transacción.** Despliega la pantalla para manipulación de campos de la transacción activa.
- **Campos del servicio.** Muestra la pantalla para manipulación de campos del servicio activo.
- **Modif. Serv.** Accesa directamente a la pantalla de modificaciones para el campo del servicio seleccionado en el grid 'Relación entre campos del servicio y campos de la transacción'.
- **Modif. Trans.** Accesa directamente a la pantalla de modificaciones para el campo de la transacción seleccionada en el grid 'Relación entre campos del servicio y campos de la transacción'.

- **Refrescar.** Actualiza en pantalla el contenido de la base de datos.
- **Cerrar.** Cierra la pantalla actual.

Los botones **Agregar**, **Modificar** y **Eliminar** ubicados debajo de los grid's 'Campos del servicio' y 'Campos de la transacción' son accesos directos a las opciones correspondientes en las pantallas "Servicio" y "Transacción".

4.3.8 El Probador TRADAL

Con la configuración de la terminal lista para la conexión por TRADAL, tanto en el equipo local como en las tablas centrales de arquitectura y del ambiente CICS, y con la definición de servicios en la base de datos local, se tienen las condiciones necesarias para ejecutar transacciones en 390. Pero antes de incorporar esta comunicación a un programa de Visual Basic, es recomendable probar la ejecución de servicios.

El probador TRADAL es una herramienta sencilla que permite probar los servicios disponibles en la base de datos local desde una terminal configurada para utilizar TRADAL. El probador recibe la información con el formato especificado para el servicio y devuelve información relacionada a la ejecución de la transacción desde su procesamiento de entrada hasta los datos obtenidos en la salida. Esto permite inferir con mayor precisión la causa de los errores generados por algunos servicios durante su procesamiento así como comprobar la conectividad del equipo con el equipo central.

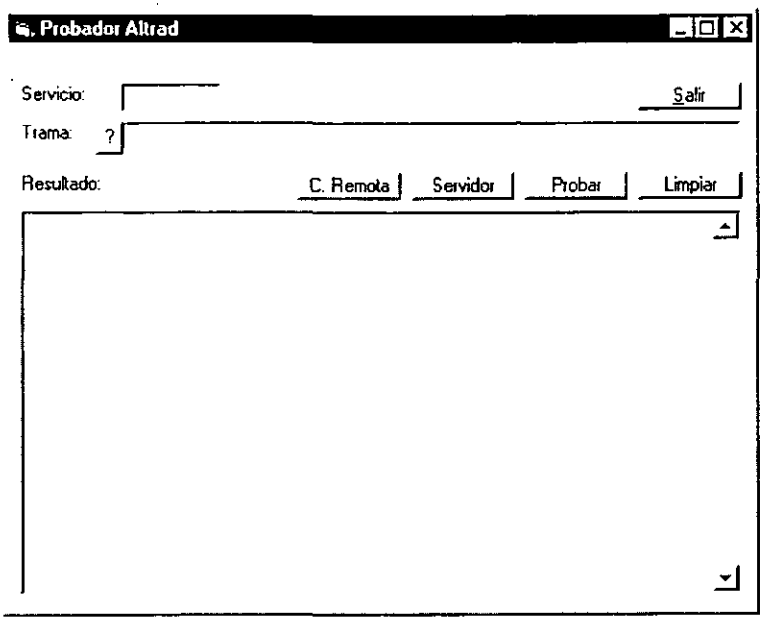


Figura # 26 Probador TRADAL

Para ejecutar un servicio desde el Probador TRADAL siga estos pasos:

1. En el cuadro de texto 'Servicio' escriba el nombre del servicio que desea ejecutar.
2. Ingrese la trama de entrada esperada por el servicio en el cuadro de texto 'Trama'. Si no conoce la trama oprima el botón marcado con el signo de interrogación (?) para obtener el orden de los datos de entrada.
3. Oprima aceptar. Aparecerá en la pantalla los sucesos durante la ejecución de esta transacción.

Al concluir oprima 'Salir'.

4.3.9 Rastreo de operaciones por TRADAL en la terminal

Cuando se ejecuta un servicio, TRADAL genera un archivo histórico de la información que ha sido trasmitida y recibida desde la terminal. Este archivo se crea por día y su nombre tiene la forma BC*aammdd*.log. Los caracteres *aammdd* corresponden a los dos últimos dígitos del año, el mes y el día de la fecha a la cual corresponde el archivo.

Un ejemplo del archivo LOG generado es el siguiente:

```
12:53:53 FsCommReceive  = <FF><FD>(
12:53:53FsCommSend : <FF><FC>(
12:53:53 FsCommReceive  = <FF><FD><18>
12:53:53FsCommSend : <FF><FC><18>
12:53:53 FsCommReceive  = <FF><FB><3>
12:53:53FsCommSend : <FF><FD><3>
12:53:53 FsCommReceive =
<FF><FD><3><FF><FD><0><FF><FB><0>Application Required. No Installation
Default <D><A>Enter Application Name:
<D><A>
12:53:53 FsCommSend :
cicssit1<D><A><FF><FB><3><FF><FB><0><FF><FD><0>
12:53:53 FsCommReceive  = <FF><FD><19>
12:53:53 FsCommSend : <FF><FB><19>
12:53:53 FsCommReceive =
<FF><FB><19><D>@<C9><D5><C9><C3><C9><D6>@<C4><C5>@<E2><C5><E
2><C9><D6><D5>@<D7><C1><D9><C1>@<C3><C9><C3><E2><E2><C9><E3><
F1>@<F0><F4>`<F0><F1>`<F2><F0><F0><F1>@<F1><F2>z<F5><F8>z<F0><F3
><15><FF><EF>
12:53:53 FsCommSend : <FF><FD><19>
```

```
12:54:39                FsCommSend                                    =
QG30DP2C00000NI1QG30I<8B>ISOTZP5<80>OCUTBRE<A7><FF><D1>
12:54:43 FsCommReceive  =       DP2C00000OC1SQG30<1> QGA0024
LOGON ACEPTADO<2> <F7><A7><FF><D1>
12:54:58                FsCommSend                                    =
CU21DP2C00000NI1CU21I<3>57000000049<A7><FF><D1>
```

Tabla # 6 Log de las operaciones por TRADAL

Cada línea se refiere a un conjunto de bytes enviados o recibidos en una llamada a un socket que establece la comunicación de datos vía TCP/IP. La primera parte de la línea corresponde a la hora en la que el mensaje fue registrado en el LOG. Los datos enviados se informan con *FsCommSend* y los datos recibidos con *FsCommReceive*. La información restante corresponde a los datos de la operación. Estos datos se escriben como caracteres cuando son imprimibles dentro del código ASCII, y en su equivalente hexadecimal dentro signos '< >' cuando no lo son.

Es necesario recordar que la información trasmitida entre la terminal y el HOST central, se encuentra en código EBCDIC (Éste es un estándar en la mayoría de los servidores de IBM), así que antes de ser grabada en el archivo LOG, esta información es convertida a código ASCII para leerla con facilidad.

4.3.10 Implementación de la clase Conexión390 en un proyecto de Visual Basic

Para ejecutar transacciones en 390, TRADAL proporciona la clase **Conexion390**, que contiene los métodos y propiedades necesarias para gestionar la comunicación con ALTAMIRA. Una vez que se ha registrado la librería TRADAL en la terminal se pueden crear instancias de la clase de la siguiente manera:

Dim <*instancia*> As New TRADAL.Conexion390

Donde *instancia* será cualquier nombre válido para el objeto.

Ejemplo:

Dim c As New TRADAL.Conexion390

Básicamente, una 'conversación' con ALTAMIRA implica los siguientes pasos:

1. Conexión al equipo central.
2. Ejecución de las operaciones, las cuales pueden ser:
3. Transaccionales. En este caso el CICS espera recibir una sola trama de entrada y devolver una sola trama de salida
4. Conversacionales. Para estas, el cliente puede enviar una serie de tramas que, de acuerdo a sus características y a las de la operación, van orientadas a distintas tareas, como realizar el alta de varios registros, la petición de más registros obtenidos de una consulta anterior o el fin de la operación conversacional.
5. Rastreo de errores en la comunicación de datos, si los hubo.
6. Rastreo de errores en la operación, si los hubo.
7. Desconexión con el equipo central al finalizar la sesión.

¿Cómo Conectarse A 390?. El método conecta y la propiedad bConectado

Para conectarse al equipo central haga uso de la función **Conecta**. Este método devuelve un valor booleano. Si la conexión se realiza con éxito, su valor será igual a *True*, en caso contrario, el valor devuelto será *False*. La sintaxis es:

instancia.Conecta(<*canal*>)

Donde <*canal*> es un número entero que indica los parámetros que se utilizarán para realizar la comunicación TELNET hacia el CICS. Estos parámetros se toman del SUBA.INI bajo la llave *Channel*<*canal*> cuya estructura se explicó anteriormente.

Ejemplo:

```
If Not c.Conecta(1) Then
    MsgBox "No se puede conectar"
End If
```

Generalmente se elige el canal 1 para la conexión de los aplicativos en producción, sin embargo, esto es solo una sugerencia.

Una vez conectados, en ocasiones es necesario verificar si la conexión sigue establecida, o si otra instancia de la clase se encargó de establecerla. Para saber si el componente está o sigue conectado se utiliza la propiedad **bConectado**, esta es de solo lectura y su valor es **True** para indicar que la comunicación se encuentra establecida.

Aunque la conexión se realiza como un método particular de una instancia específica de la clase, esta conexión queda abierta para cualquier otra instancia de la clase Conexion390, incluso si ha sido creada en una aplicación diferente. La razón es que el TRADAL proporciona esta clase únicamente como una interfaz para realizar operaciones, pero la ejecución de estas se realiza en un módulo interno independiente de la clase. Este módulo es único, por lo que sin importar cuantas instancias ni cuantos aplicativos estén operando, todos ellos utilizarán el mismo medio proporcionado por el TRADAL para el envío de datos a 390.

Esto se debe a que para la ejecución de transacciones, la arquitectura de ALTAMIRA asigna una LU0 a la terminal. Si requiriéramos de diferentes canales para comunicarnos con CICS, necesitaríamos tener esa cantidad de LU0's asignadas a nuestro equipo, lo que implicaría un desperdicio de recursos en 390.

Con la conexión abierta por una instancia de la clase, no es necesario que las demás instancias tengan que recurrir al método conecta nuevamente, de hecho, esta conexión será utilizada por cualquier aplicación que llame a TRADAL. Considere el siguiente ejemplo:

```
Dim c2 As New TRADAL.Conexion390
c1.Conecta 1
If Not c2.bConectado Then
        Msgbox "No se puede conectar"
End If
```

Para este ejemplo, si el método *Conecta* de la instancia *c1* se ejecuta sin problemas, la comunicación con 390 quedará abierta, por lo que el valor de *c2.bConectado* será *True*, sin necesidad de haber ejecutado su método *Conecta*. Esto funciona también con todas las propiedades y métodos relacionados con el envío y recepción de datos del CICS como se verá más adelante.

Además de la apertura del canal de comunicaciones, es necesario 'firmar' la terminal. Esta firma es una transacción especial que mediante el nombre de usuario y su password, permite la ejecución de diversas transacciones desde la terminal a las que se tiene acceso dependiendo del nivel de seguridad del usuario.

Para firmar la terminal, la clase proporcional el método **LogOn**.

instancia.LogOn <canal>, <Usuario>, <Password>

<canal> equivale a la variable indicada en el método **Conecta**.

<Usuario> Es un nombre de usuario válido en el ambiente CICS.

<Password> Es la contraseña asignada al usuario.

Ejemplo:

c.LogOn 1, "ALGUIEN", "CERO"

Este método devuelve un valor *True* cuando la firma se realiza exitosamente.

En algunos ambientes de CICS, existen transacciones que pueden ejecutarse desde cualquier terminal sin necesidad de haberla firmado, por eso es posible transaccionar con el equipo central aunque el proceso de firma falle.

Cuando el usuario no tiene acceso a una transacción determinada dentro del CICS, TRADAL no puede capturar este error, el cual aparece ante el usuario como un error en la recepción de datos. En estos casos es conveniente asegurarse primero si el usuario tiene efectivamente permiso para ejecutar la transacción que genera éste problema.

¿Cómo Ejecutar Una Transacción?

Una vez que se ha establecido la comunicación, es posible solicitar la ejecución de transacciones al CICS. Para correr una servicio se utiliza el método **EnviaRecibe**, el cual realiza las operaciones necesarias para codificar, enviar, recibir y decodificar datos de ALTAMIRA, liberando al programador de los detalles asociados a la ejecución de las transacciones en el equipo central. Su sintaxis es:

instancia.EnviaRecibe <canal>, <Servicio>, <TramaEnvio>, <TipoOperacion>, <Folio>, <TramaRecepcion>

En donde:

<canal> es un entero que indica el canal que se utilizará para la comunicación de datos. Al igual que en el método conecta, este canal hace referencia a los parámetros que se utilizan en la conexión con el equipo central.

<Servicio> es una cadena que contiene el nombre del servicio que va a ejecutarse. TRADAL corre las transacciones asociadas a este servicio de acuerdo al orden que se ha indicado en la base de datos.

<TramaEnvio> contiene la trama con los campos de entrada esperados por el servicio. Cada campo debe separarse con el delimitador especificado en la base de datos.

<TipoOperacion> es una cadena que contiene el tipo de operación.

<Folio> es un entero que representa el folio del movimiento.

<TramaRecepcion> es una cadena de caracteres que se pasa por referencia y en la que se almacenará la respuesta obtenida.

Nota: Para los parámetros <TipoOperacion> y <Folio> se sugieren los valores "" y 0 respectivamente. Estos parámetros no son utilizados por el componente y cualquier valor será ignorado.

El fragmento de código siguiente ejecuta un servicio llamado "CU21" que recibe un solo campo de entrada y muestra el resultado obtenido en un cuadro de diálogo.

```
Servicio = "CU21"

Trama ="57000000049|"

c.EnviaRecibe 1, Servicio, Trama, "", 0, TramaRecibe

Msgbox TramaRecibe
```

La anterior es una llamada típica a un servicio que ejecuta operaciones transaccionales.

Si durante la ejecución del servicio se presenta algún error, este puede detectarse de dos formas:

1. Si el método **EnviaRecibe** devuelve un valor distinto de cero.
2. Cuando el valor de la propiedad **bError** es *True*.

Ambas formas son válidas y pueden utilizarse indistintamente, y en los dos casos, el valor de la trama de recepción estará vacío.

Para obtener una descripción más específica del error ocurrido, la clase **Conexión390** proporciona la propiedad **ErrorMsg.** Esta propiedad es una cadena de caracteres que despliega una descripción más detallada del error.

A pesar de las propiedades mencionadas es probable que la información proporcionada por la propiedad **ErrorMsg** sea insuficiente para determinar cuál es la causa que produce tal error al ejecutar el servicio.

La propiedad **strLOG** permite observar con mayor detalle el comportamiento del TRADAL cuando ejecuta una transacción específica. Sin embargo, para que TRADAL escriba algún resultado dentro de esta propiedad, debe asignarse a *True* el valor de la propiedad **DebugOn** antes de utilizar el método **EnviaRecibe**.

Ejemplo:

```
Resultado = ""
c.DebugOn = True            'Activa bandera para debuguear
c.EnviaRecibe 1, txtServicio, txtTrama, "", 0, TramaRecibe
If c.bError Then            'si hubo error...
  MsgBox d.ErrorMsg         '...mostrar la descripción del error
Else                        'si no hubo error...
  Resultado = c.strLOG & Chr(13) & Chr(10)   'datos adicionales
  Resultado = Resultado & "Trama recepcion: " & Chr(13) _
  & Chr(10) & TramaRecibe                    'respuesta
  MsgBox Resultado          '...mostrar resultado
End If
```

Tabla # 7 Ejemplo de la propiedad DebugOn

¿Cómo Desconectarse De 390?

El último paso en la comunicación con ALTAMIRA es la desconexión, lo que implica el fin de la sesión en el ambiente CICS y el cierre del canal TCP/IP que se utiliza para enviar y recibir datos. La función **Desconecta** de la clase Conexión390 realiza estas operaciones. Su sintaxis es la siguiente:

Desconecta <*canal*>

Donde <*canal*> es un número entero. Éste número debe ser el mismo que usó para activar la conexión en la función Conecta.

Ejemplo:

d.Desconecta 1

Al igual que en la conexión de un canal desde cualquier instancia de la clase, la desconexión cierra el canal para TODAS las instancias, lo cual significa que TRADAL ha cerrado el medio de comunicación con el equipo central. Sin embargo, la desconexión no libera de la memoria el proceso TRADAL de la terminal. Si se recuerda, TRADAL corre en un proceso diferente al cliente que lo ejecuta, y como todo proceso independiente, necesita recibir un mensaje para terminar su ejecución dentro de la memoria principal.

Para terminar la ejecución de TRADAL, debe notificarse con la propiedad **bTermina** de cualquier instancia de Conexión390. Cuando el valor de esta propiedad sea *True* TRADAL entenderá que se desea remover su proceso una vez que se destruya la instancia que solicitó la desconexión.

El siguiente código libera la memoria ocupada por TRADAL:

```
d.bTermina = true
d.Desconecta 1
Set d = Nothing
```

Debe tenerse cuidado al terminar el proceso de TRADAL, pues si se solicita cuando existen otras instancias de Conexión390, se generara un error de automatización ActiveX para los ejecutables que intenten hacer uso de estas instancias. En las aplicaciones que utilizan clases de ActiveX con extensión 'exe', las clases sólo se encuentran disponibles si el proceso está en memoria. Para realizar la primera llamada, una aplicación ejecuta el ActiveX si este no se encuentra en la RAM, y en caso contrario, hace uso de la copia existente en memoria, pero si este ActiveX se retira, no hay forma de realizar una nueva llamada, generando el error mencionado.

4.3.11 Errores comunes en la ejecución de TRADAL

Cuando se utiliza una aplicación que hace uso de la librería TRADAL, es posible la aparición de algunos problemas que resultan de una configuración errónea del equipo o de los componentes instalados en el mismo. Se listan a continuación algunos de los errores más comunes.

Error en la conexión al HOST central.

Este error se determina por el mensaje "No se puede conectar" que ocurre después de intentar la conexión de la terminal con el HOST central. Este problema tiene diversas causas.

1. **La terminal no cuenta con permisos para la conexión.** Para acceder al ambiente CICS es necesario que la terminal cuente con los permisos de seguridad necesarios para realizar una conexión TELNET al equipo central.
   Para comprobar que la terminal puede conectarse siga estos pasos:

a) En el menú 'Inicio' de la barra de tareas, elija la opción 'Programas' | 'Accesorios' | 'Telnet'. Debe aparecer una pantalla como esta:

b) Del menú principal, en el submenú 'Conectar' seleccione la opción

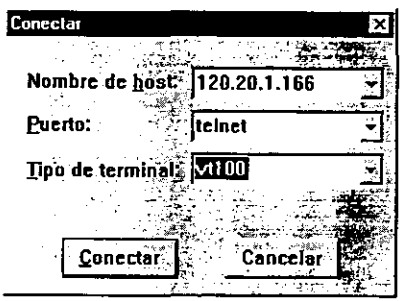


Figura # 27 Conectar

c) 'Sistema remoto...' para ver el siguiente cuadro de diálogo:

d) Dentro del valor correspondiente al nombre de Host escriba la dirección IP del Host al que se conectará su terminal. Esta dirección debe ser la misma que la indicada en el archivo SUBA.INI. Oprima entonces el botón 'Conectar'.

e) Si aparece el mensaje "Application Required. No Installation Default Enter Application Name:" significa que la terminal puede realizar una comunicación TELNET con el HOST. Cuando aparece el mensaje "se ha perdido la conexión al Host" es necesario corroborar con las personas que controlan el ambiente en el equipo IBM 390 para solicitar el acceso al mismo. Ahora es necesario comprobar que la terminal puede accesar al CICS.

f) Teclee el nombre del ambiente CICS sobre el que va a trabajar. Al igual que la dirección IP del Host, este debe ser el indicado en el fichero SUBA.INI. La información que introduzca no aparecerá en la pantalla, así que debe tener cuidado de no cometer errores al escribir el nombre del ambiente. Al terminar presione <ENTER>.

g) En el caso de que la terminal cuente con la configuración correcta en el equipo central para el acceso al CICS, se mostrará un mensaje parecido a este:

INICIO DE SESION PARA CICS 08-01-2000 10:51:49

pero si en lugar de ello, aparece el mensaje "se ha perdido la conexión al Host" habrá que recurrir a los responsables del CICS para obtener el permiso de acceso al ambiente deseado.

En algunas ocasiones este mismo error se genera cuando la terminal esta 'fuera de servicio', lo que significa que la terminal cuenta con permisos para transaccionar con CICS, y que de hecho lo estaba haciendo sin problemas, pero por diversas cuestiones temporales se le ha negado la posibilidad de conectarse al equipo central. En este caso también es necesario recurrir al personal que controla el ambiente CICS.

1. **El archivo SUBA.INI no esta configurado correctamente.** Si realizó la prueba anterior con éxito y a pesar de ello, las aplicaciones no realizan la conexión de TRADAL, es recomendable comprobar los parámetros indicados en el fichero SUBA.INI, para asegurarse de que corresponden a los valores correctos de la terminal, del Host Central y del ambiente CICS.

   En algunas ocasiones existe un archivo SUBA.INI configurado correctamente, pero no es el mismo archivo que TRADAL utiliza para configurar la conexión. Recuerde que TRADAL busca este fichero SUBA.INI en la ruta indicada dentro de WIN.INI (Véase el tema correspondiente).

2. **No corresponden las versiones de los componentes.** Sucede cuando los componentes que TRADAL utiliza se encuentran disponibles en la computadora, pero corresponden a versiones distintas Este error no es propio de la conexión, sino de la aplicación TRADAL en general, pero se detecta en este punto porque es la primera operación que hacen los programas que lo invocan. Los mensajes característicos de este error es "Unexpected Error", "Error inesperado", "Automation Error" y "ActiveX can't create object".

   Para resolverlo debe correrse el instalador del TRADAL. Este programa se llama InstSuba y se recomienda ponerse en contacto con el proveedor de la aplicación a fin de obtener una versión actualizada del mismo.

Error en recepción de datos FSCOMMRECEIVE = -1

Este error se presenta en la ejecución de servicios, una vez que se ha logrado la conexión con 390. Significa que no se han recibido los datos esperados del equipo central, lo cual puede ser causado porque:

1. **El usuario no tiene permiso para ejecutar la transacción.** Verifique las características de seguridad del usuario en el ambiente CICS.
2. **La transacción concluyó anormalmente en el CICS.** En este caso no hay nada que hacer desde la terminal.
3. **Se excedió el límite de tiempo para esperar la respuesta.** Esta relacionado con errores en la ejecución de la transacción en el equipo central. Tampoco existe forma de corregirlo desde la terminal.

Cuando el problema persiste lo más recomendable es monitorear el comportamiento de la terminal en el CICS para establecer si el problema esta en el cliente o en la transacción.

CONCLUSIONES

El esquema de Banco Santander esta basado en un modelo Cliente/Servidor a tres planos, el uso del middleware que es el medio que el cliente emplea para invocar un servicio y que comprende la transmisión de la solicitud por la red y la respuesta resultante, juega un papel muy importante en este modelo.

El middleware que se utiliza en el banco es TUXEDO que además funje como monitor de transacciones lo cual resulta ser muy ventajoso, cuando este realiza operaciones de optimización que no pueden realizarse en los 'clientes' y que serían costosas en tiempo y recursos para el equipo central, pero que se vuelve una desventaja cuando el middleware solo se encarga de recibir el mensaje para retransmitirlo hacia otro punto, dado que los mensajes deben pasar por varios caminos antes de llegar a su destino final formando cuellos de botella.

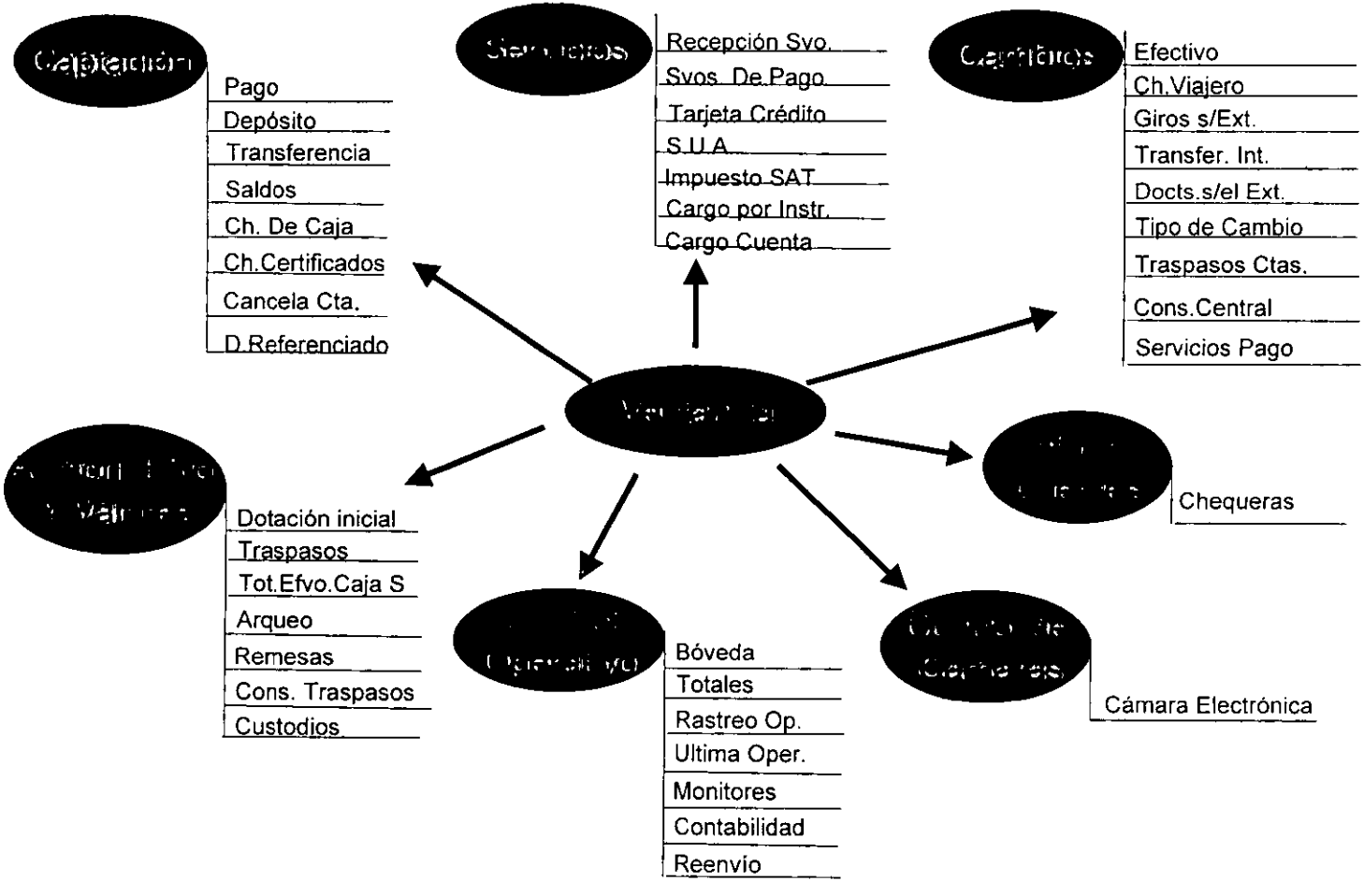
Por lo cual se vio la necesidad de desarrollar un nuevo middleware llamado TRADAL que enviara las transacciones por un canal directo del cliente hacia el equipo central, provocando así que los congestionamientos provocados por estos cuellos de botella desaparecieran.

El TRADAL es un componente para administrar comunicaciones entre el terminal financiero (cliente), se desarrolló con el fin de proveer a las terminales un canal para la ejecución de transacciones CICS, las cuales se procesan en una maquina central IBM 390. Este componente se utiliza en aplicaciones orientadas al usuario final para un ambiente Windows NT.

El TRADAL fue probado con éxito con la transacción de saldos en toda la red de sucursales de banco Santander.

En conclusión el TRADAL resulto ser un middleware efectivo para el esquema del banco, pero definitivamente no podria reemplazar a TUXEDO en su totalidad, ya que para que el TRADAL pueda funcionar se necesita que exista una transacción equivalente a cada servicio de TUXEDO en el equipo central, las cuales no existen y desarrollarlas llevaría mucho tiempo. Además hay aplicativos que no podrían operar directamente con la maquina central por lo cual se tomo la decisión que los dos middleware se implementarían en el esquema del banco, solo que se optimizaría al máximo la configuración de TUXEDO en cuanto a los servicios que no tuvieran , una transacción en 390.

Anexo # 1. Esquema de Plataforma de Mostrador



Anexo # 2. Esquema de Plataforma de Negocios

PLAZO

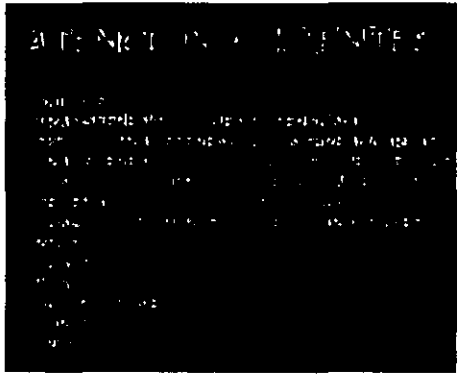
ALTA CONTRATOS DE PRODUCTOS PLAZO
ALTAINVERSION
CAMBIO INSTRUCCIONES A INVERSIONES
CONSULTA DE MOVIMIENTOS A PLAZO
ANULACION DE INVERSIONES
CONSULTA Y REIMPRESION DE OPERACIONES

AFORE

CONSULTA E IMPRESIÓN DE RECIBOS

BANCA ESPECIALIZADA

CONTRATACION
CONSULTA DE SALDOS
CAPTURA DE ORDENES
MOVIMIENTOS
PORTAFOLIO
LIQUIDACIONES
FONDOS DE INVERSION



SEGUROS

CONTRATACION
AUTOS
VIDA
ACCIDENTES PERSONALES
HOGAR

VISTA

MANTENIMIENTO SISTEMADE FIRMAS
SOLICITUD DE CHEQUERAS
CONTRATACION PRODUCTOS VISTA
CONSULTA LOG

BIA

CONTRATACION
ACTUALIZACION DE ENTORNO
REPOSICION DE PLASTICOS
CONTRATACION FAX/BEEP

SERVICIOS

PRE-NEGOCIACION NEGOCIO ADQUIRIENTE
UNIVERSIDADES
PAGO AUTOMATICO
CONSOLIDADO DE SERVICIOS
TESOFE (DEVOLUCION IMPUESTOS)
RECEPCION DE IMPUESTOS
RECAUDACION

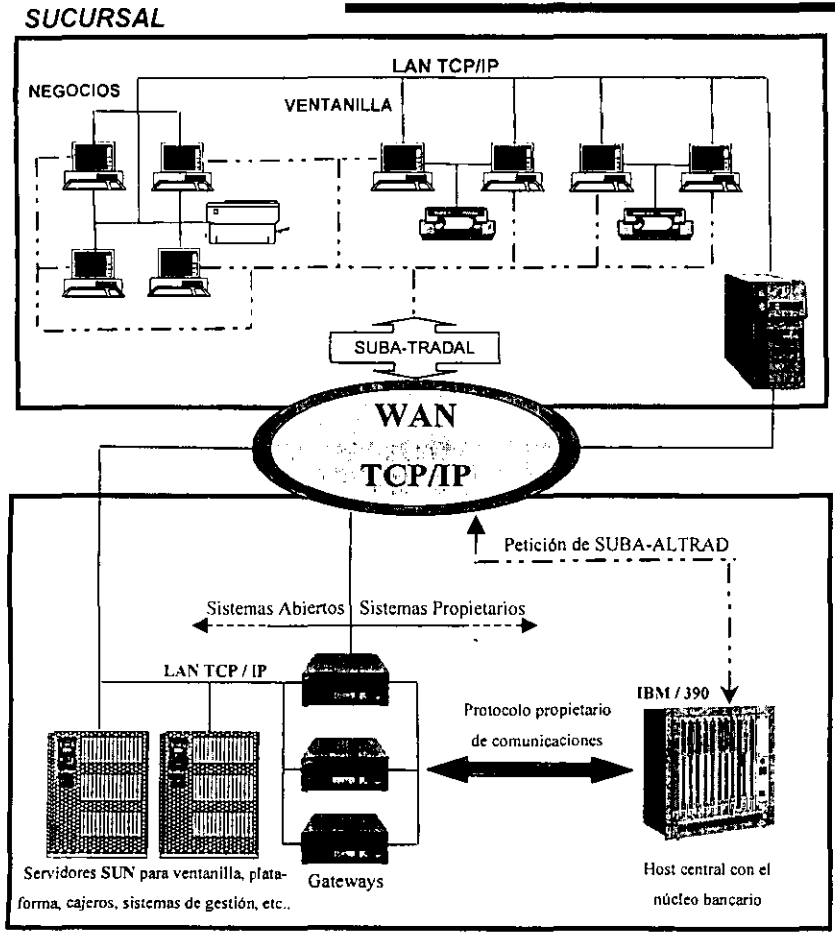
SUPERBASE

ALTA PERS. FISICAS-MORALES
LOCALIZACION DE PERS.FIS-MORALES
ALTA DATOSCOMPLEM.PERS.FIS.-MORALES
ACTUALIZACION DE PERS.FIS.-MORALES
ACTUALIZACION DE DOMICILIOS
LOCALIZACION DE CLIENTES POR NUM. CTA.

Crédito Efectivo

CONSULTA DE CLIENTES
SOLICITUD Y CONTRATACION DE CREDITO
SOLICITUD Y EMISION DE CUESTIONARIO DE SALUD
EMISION DE TABLA DE AMORTIZACIONES
CONSULTA DE TASAS DE CREDITO
SOLICITUD Y CONTRATACION DE CREDITO DE COBERTURA
SOLICITUD DE TARJETA DE CREDITO

Anexo # 3 Enlaces



PC IBM 300GL (32 MB RAM)
Windows NT WorkStation
Correo / Browser Communicator 4.5 (Netscape)
Office Microsoft 97
TUXEDO Runtime
TCLTK Software para visualizar firmas

LexMark Optra S 1650
PostScript

Olivetti PR2

Servidor SUN ULTRA 5 (SUCURSAL)
Solaris 2.5.1
DBMS Oracle v7.3.3
TUXEDO
Aplicativos de Sucursal en VB 5 y VB 4
DCE

Sun E10000 (HOST CENTRAL)
Solaris 2.5.1
DBMS Oracle v7.3.3
TUXEDO
DCE
SunLink 9.0
DB2 Connect

BIBLIOGRAFIA

Introducciòn a los Sistemas de Base de Datos
Jeffrey D. Ullman - Jennifer Widom
Edit. Prentice-Hall
2ª Edición
México 1999

Cliente/Servidor
Guía de Supervivencia
Robert Orfali, Dan Harkey,
Edward Jery
Edit. Mc Graw Hill
2ª Edición
Estados Unidos 1996

Todo Sobre Windows Nt 4.0
B.Kretschmer-C. Schneirder
Edit. Marcombo S.A. De C.V.
1ª Edición
Barcelona España 1996

Net Ware, Intranet Ware
Instalación, Configuración y Administración
José Luis Lara- Cristina Raya
Edit. Rama
1ª Edición
México 1997

Introducción a la Técnica y Diseño de Sistemas de Comunicaciones y Redes de Ordenadores
John Freer
Edit. Anaya Multimedia
2ª Edición
México 1990

Oracle 8
Michael Abbey- Michael J. Corey
Edit. Mc Graw Hill
1ª Edición
España 1999

UNIX Sistema V Versión 4
Kenneth H. Rosen
Edit. Osborne - Mc Graw Hill
2ª Edición
Madrid 1997