CAPÍTULO

2

MARCO TEÓRICO

2.1 Características, ventajas y desventajas de las bases de datos relacionales.

El uso constante de las tecnologías, requieren de gestión automatizada en el uso de sus sistemas de ficheros.
La tecnología informática existente, debe disponer de técnicas de bases de datos que se organicen y se gestionen por grandes volúmenes, mismas que son utilizadas en diversas disciplinas.
Uno de los pilares de cualquier organización es la información que necesita para su funcionamiento; así mismo, una de sus actividades principales es el tratamiento de dicha información el cual, ya se haga de forma manual o automática, tienen como objetivo proporcionar a las personas autorizadas la información que necesitan en el momento y el lugar adecuados. Por ello, uno de los componentes básicos de cualquier organización es su sistema de información.

Base de Datos.

Una base de datos (BD) es una colección estructurada de datos.
En esta colección, los datos deben estar estructurados de forma que reflejen fielmente los objetos, las relaciones y las restricciones existentes en el mundo real representada por la base de datos (propiedades estáticas). Para que esta

representación sea fiable, la base de datos debe reflejar los cambios que los sucesos puedan provocar en el mundo real (propiedades dinámicas).

Los mecanismos de estructuración de datos que se pueden utilizar dependen del sistema informático con el que se vaya a crear y manipular la base de datos, lo que se conoce como sistema de gestión de bases de datos.

Un Data Base Management System, en español, Sistema de Gestión de Base de Datos (SGDB), es una herramienta de software que permite la creación y manipulación de la base de datos.

Proporciona el método de organización necesaria para el almacenamiento y recuperación de grandes cantidades de datos.

Todo sistema de gestión de base de datos está basado en un modelo de datos. Estos proporcionan estructuras de datos predefinidas con sus operadores asociados (modelos clásicos), o bien mecanismos de estructuración de datos o constructores de tipos más generales.

E.F. Codd (1970) propuso el modelo relacional, el cual lo denomina como una "gestión de datos basadas en la lógica de predicados y la teoría de conjuntos utilizado para modelar problemas y administrar dinámicamente datos de forma estructurada", este modelo es elegido para la construcción de casi todo los SGDB comerciales.

En un modelo relacional se emplean tablas para poder organizar los elementos de datos. Cada fila representa una instancia de esa entidad y cada tabla corresponde a una entidad de aplicación.

Álgebra relacional.

El álgebra relacional es un conjunto de operaciones simples sobre tablas relacionales, a partir de las cuales se definen operaciones más complejas.

Dichas operaciones utilizan una o dos relaciones existentes para crear una nueva relación. Esta nueva relación puede entonces usarse como entrada para una

nueva operación. Se pueden aplicar las siguientes operaciones del algebra relacional.

- Selección: Consiste en recuperar un conjunto de registros de una tabla o de una relación indicando las condiciones que deben cumplir los registros recuperados. La selección también es conocida como consulta. En dichas consultas se utilizan diferentes operadores de comparación (=, <, >, <=, >=, <>) además de los operadores lógicos (and, or, xor) y la negación (not).
- Unión: Es utilizada para poder recuperar datos a través de varias tablas conectadas unas con otras mediante cláusulas de UNIÖN, en cualquiera de sus tres variantes, unión por la derecha, por la izquierda y unión completa (der. E izq.).
- Proyección: Es un caso específico de la selección. Es una segunda selección, en la que se seleccionan solo aquellos campos que se desean obtener.
- Asignación: Consiste en asignar un valor a uno o varios campos de una tabla.

Cálculo relacional.

El cálculo relacional es un lenguaje de consulta que describe la respuesta deseada sobre una Base de datos sin especificar cómo obtenerla.
La solución para toda consulta en este tipo de cálculo se define por:

- Una lista de resultados: Son aquellos registros que cumplen las condiciones que se desean.
- Una sentencia de cualificación: Contiene las condiciones que deseamos que cumplan los registros de la lista de resultados.

A diferencia del álgebra relacional que recurre a diferentes tablas y realiza varios pasos para obtener un resultado, el cálculo realiza la operación en un único paso gracias a lenguajes de interrogación de bases de datos.

El cálculo relacional incluye un concepto denominado cuantificador, los cuantificadores averiguan el número de registros afectados por una determinada operación, incluso antes de realizarla.

Podemos dividir a los cuantificadores en:

- Existenciales: Buscan el número de registros que devolverá una consulta.
- Universales: Indican que una condición se aplica a todas las filas de algún tipo.

Las siguientes operaciones del cálculo relacional:

- Reunión: Permite unir datos de varias relaciones.
- Diferencia: Identifica filas que están en una relación y no en otra.
- Intersección: Permite unir campos idénticos pertenecientes a las filas que son comunes en dos relaciones.
- Producto: Consiste en la realización de un producto cartesiano entre dos tablas dando como resultado todas las posibles combinaciones entre los registros de la primera y los registros de la segunda.

Cardinalidad.

La cardinalidad es la forma en la que se relacionan las entidades, o expresa cuantas entidades se relacionan con otras entidades.

- Uno a uno (1:1): Cada instancia o elemento de la entidad A está asociado solamente a un elemento de la entidad B.
- Uno a muchos (1:M): Cada instancia o elemento de la entidad A está asociado a varios elementos de la entidad B, entonces la clave que forma el vínculo entre ambas entidades, pasa hacia la entidad que tiene mayor grado de cardinalidad, es decir, el que posee la denominación “muchos”.
- Muchos a muchos (N:M): Los elementos de la entidad A están asociados a varios elementos de la entidad B, y los elementos de la entidad B están asociados a varios elementos de la entidad A, cuando esto sucede, se genera una nueva entidad denominada “Entidad Asociada”.

Normalización.

La normalización es el proceso de simplificar la relación entre los campos de un registro. En dicho proceso se somete un esquema de relación a una serie de pruebas para “certificar” si pertenece o no a una cierta forma normal. Esta se basa en las dependencias funcionales entre los atributos de una relación.
La normalización de los datos puede considerarse como un proceso de análisis de los esquemas de relación de datos basado en sus claves primarias para alcanzar las propiedades deseables de minimizar la redundancia y minimizar las anomalías de inserción, eliminación y actualización de los datos. Los esquemas de relación insatisfactorios que no cumplan determinadas condiciones, las pruebas de formas normales, se descomponen en esquemas de relación más pequeños que satisfagan dichas pruebas y que de este modo posean las propiedades deseables.

Forma normal.

La forma normal de una relación hace referencia a la condición de forma normal más alta y de este modo, indica el grado al que ha sido normalizada.

- Primera Forma Normal (1FN): Esta forma se definió para prohibir los atributos multivaluados, los atributos compuestos y sus combinaciones. Establece que el dominio de un atributo debe incluir solo valores atómicos. Así la 1FN prohíbe las “relaciones dentro de relaciones”. La 1FN también prohíbe los atributos compuestos que por sí mismos son multivaluados. Estos se denominan relaciones anidadas porque cada tupla puede tener una relación dentro de sí.
- Segunda Forma Normal (2FN): Se basa en el concepto de dependencia funcional total. Consiste en identificar que atributos no primos dependen funcionalmente de la llave primaria. La relación se encuentra en 2FN, cuando cumple con las reglas de la 1FN y todos sus atributos que no son

claves (llaves) dependen por completo de la clave. Cada tabla que tiene un atributo único como clave, está en 2FN.

Dependencia Transitiva: En una tabla (bidimensional) que tiene por lo menos tres atributos (A, B, C) en donde A determina a B y B determina a C pero este no determina a A.

- Tercera Forma Normal (3FN): Una relación estará en 3FN si y sólo si está en 2FN y todos sus atributos no primos dependen no transitivamente de la llave primaria. Se estará en 3FN si no existen dependencias transitivas entre los atributos, es decir, cuando existe más de una forma de llegar a referencias de un atributo en una relación.

Características de las bases de datos.

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios: Debido al carácter integrador que tiene la base de datos, ésta tendrá que estar compartida por distintos grupos de usuarios, lo que significa que éstos podrán acceder simultáneamente a los datos.
- Integridad de los datos: El sistema de bases de datos debe asegurar en todo momento la calidad de la información almacenada, evitando que ésta se deteriorare por un uso correcto (actualizaciones que no son válidas, accesos concurrentes no válidos, etc.).
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoría: Los sistemas de bases de datos deben asegurar que la información almacenada solo la accedan las personas autorizadas y en la forma autorizada.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Ventajas de las bases de datos.

- Control sobre la redundancia de datos: Los sistemas de ficheros, almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos.

  En los sistemas de bases de datos, todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos.

- Consistencia de datos: Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantengan consistentes.
- Compartición de datos: En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados.
- Mantenimiento de estándares: Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

- Mejora en la integridad de datos: La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se debe encargar de mantenerlas.
- Mejora en la seguridad: La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros.
- Mejora en la accesibilidad a los datos: Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.
- Mejora en la productividad: El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación.

  El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel.

- Mejora en el mantenimiento: En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan.

  Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena

en disco, requiere cambios importantes en los programas cuyos datos se ven afectados.

Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.

- Aumento de la concurrencia: En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.
- Mejora en los servicios de copias de seguridad: Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos.

  En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se tiene que volver a realizar. Sin embargo, los SGBD actuales funcionan de modo que se minimiza la cantidad de trabajo pérdido cuando se produce un fallo.

Desventajas de las bases de datos.

- Complejidad: Los SGBD son conjuntos de programas que pueden llegar a ser complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder realizar un buen uso de ellos.
- Costo del equipamiento adicional: Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de

almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.

- Vulnerable a los fallos: El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse. Es por ello que deben tenerse copias de seguridad (Backup).

2.2. Metodologías para el desarrollo de sistemas orientados a objetos

En cuanto a las metodologías orientadas a objetos, existen un gran número de métodos actualmente. Muchos de los métodos pueden ser clasificados como orientados a objetos porque soportan de manera central los conceptos de la orientación a objetos.

Se puede decir que:

- El análisis orientado a objetos es un método de análisis que examina los requisitos desde la perspectiva de las clases y objetos que se encuentran en el vocabulario del dominio del problema.
- El diseño orientado a objetos es un método de diseño que abarca el proceso de descomposición orientado a objetos y una notación para describir los modelos lógico y físico, así como los modelos estático y dinámico del sistema que se diseña.

Algunas de las metodologías más conocidas y estudiadas se relacionan en la figura 2.2.1.

Año	Metodología
1968	Aparecen los primeros conceptos de la programación estructurada de DIJKSTRA
1974	Técnicas de programación estructurada WAGNER Y JACKSON. Jackson System Development (JSD).
1975	Primeros conceptos acerca del diseño estructurado MYERS Y JURIDOS
1977	Conceptos sobre el análisis estructurado GANE Y SARSON
1978	Conceptos sobre el análisis estructurado DEMARCO Y WEINBERG
1985	Análisis y diseño estructurado para sistemas en tiempo real de WARD Y MELLOR
1987	Análisis y Diseño Estructurado para sistemas en tiempo real de HATLEY y PIRHBAY
1991	Aparece OMT
1993	Aparece Objetory/ OOSE (Object Oriented Software Engineering) JACOBSON
1994	Aparece BOOCH Object Oriented Design (OOD)
1999	Concepto de proceso unificado de JACOBSON.
1999	Concepto de racional unified proces (RUP)

Figura 2.2.1 Metodologías orientadas a objetos

A continuación se detallan las principales metodologías de las antes mencionadas.

Metodología Jackson System Development.

La metodología JSD o desarrollo estructurado de Jackson fue desarrollada por Michael Jackson y fue especialmente popular en Europa. Sus modelos describen el mundo real en términos de entidades, acciones y secuencias de acciones. Las

entidades suelen aparecer como sustantivos en los documentos de especificaciones de requerimientos y las acciones aparecen como verbos.

JSD comienza ciertamente con unas consideraciones acerca del mundo real y en ese sentido, se podría decir que está orientada a objetos. Sin embargo, Jackson identifica pocas entidades (objetos) y muestra solo un poco de su estructura.

La aproximación JSD es compleja y difícil de comprender en su totalidad. Una razón de dicha complejidad es el fuerte uso que hace del pseudocódigo; los modelos gráficos son más fáciles de entender.

JSD es una metodóloga útil para las siguientes aplicaciones:

- Software concurrente en el cual los procesos deben sincronizarse entre sí.
- Software de tiempo real. El modelo JSD es extremadamente detallado y se centra en el tiempo.
- Programación de computadoras paralelas. El paradigma de múltiples procesos de JSD puede servir de ayuda. JSD no está bien adaptado para otras aplicaciones.
- Análisis de alto nivel. JSD no facilita una alta comprensión del problema. No es eficiente a efectos de abstracción y simplificación. Maneja meticulosamente los detalles, pero no ayuda a los desarrolladores a captar la esencia del problema.
- Bases de datos. El modelado JSD esta sesgado hacia las acciones, apartándose de las entidades y de los atributos. Por lo tanto, es una técnica poco adecuada para el diseño de bases de datos.
- Software convencional que corre en un sistema operativo. La abstracción de JSD formada por cientos o miles de procesos produce confusión y es innecesaria.

Metodología de Booch Object- Orienteed Design (OOD).

El método de Booch abarca un micro proceso de desarrollo y un macro proceso de desarrollo. El nivel micro define un conjunto de tareas de análisis que se reaplican en cada etapa en el macro proceso.

Por eso se mantiene un enfoque evolutivo. El método Booch está soportado por una variedad de herramientas automatizadas. Booch explica que el desarrollo orientado a objetos es fundamentalmente diferente de las aproximaciones funcionales tradicionales al diseño, como pueden ser las basadas en flujo de datos.

La metodología de Booch incluye una variedad de modelos que abarcan los aspectos de objetos dinámico y funcional de un sistema de software.

Metodología de Coad y Yourdon Object Oriented Analysis (OOA).

El método de Coad y Yourdeon se considera con frecuencia como uno de los métodos más sencillos de aprender. La notación del modelado es relativamente simple y las reglas para desarrollar el modelo son evidentes.

A continuación se muestra una descripción resumida del proceso de análisis que proponen Coad y Yourdon:

- Identificar objetos usando el criterio de qué buscar.
- Definir una estructura de generalización-especificación.
- Definir una estructura de todo-parte.
- Identificar temas (representaciones de componentes de subsistemas)
- Definir atributos.

- Definir servicios. En lo referente al diseño, el método de Coad y Yourdon se desarrolló estudiando como realizan su trabajo de diseño Diseñadores especializados del software orientado a objetos.

El enfoque de diseño se dirige no solamente a la aplicación, sino también a la infraestructura para la aplicación. Una breve descripción del proceso de diseño orientado a objetos de Coad y Yourdon sigue a continuación:
Componente del dominio del problema:

- Agrupar todas las clases específicas al dominio.
- Diseña una jerarquía de clases apropiada para las clases de aplicación.
- Trabaja, cuando sea apropiado, para simplificar la herencia.
- Refina el diseño para mejorar el rendimiento
- Desarrolla una interfaz con el componente de gestión de datos
- Refina y añade objetos a bajo nivel, en caso de ser necesario
- Revisa el diseño y propone adiciones al modelo de análisis.

Componente para la gestión de tareas:

- Identificar tipos de tareas.
- Establecer prioridades
- Identificar la tarea que servirá de coordinadora para otras tareas
- Diseñar objetos apropiados para cada tarea.

La componente para la gestión de datos:

- Diseñar las estructuras de datos y su distribución.
- Diseñar servicios necesarios para manejar las estructuras de datos

- Identificar las herramientas que pueden ayudar en la implementación de la gestión de datos.
- Diseñar clases apropiadas y la jerarquía de clases.

Componente de interacción humana:

- Definir los actores humanos.
- Desarrollar escenarios para las tareas.
- Diseñar una jerarquía de órdenes de usuario.
- Refinar la secuencia de interacción del usuario.
- Diseñar clases relevantes y la jerarquía de clases.
- Integrar adecuadamente las clases de interfaz gráfica de usuario.

Metodología de Wirfs-Brock.

El método de Wirfs-Brock no hace distinción clara entre las tareas de análisis y diseño. En su lugar, propone un proceso continuo que comienza con la valoración de una especificación del cliente y termina con el diseño.
El análisis propuesto en este método se describe a continuación:

- Evaluar la especificación del cliente.
- Usar un análisis gramatical para extraer las clases candidatas de la especificación.
- Agrupar las clases en un intento de determinar superclases.
- Definir responsabilidades para cada clase.
- Asignar responsabilidades a cada clase.
- Identificar relaciones entre clases.

- Definir colaboraciones entre clases basándose en sus responsabilidades.
- Construir representaciones jerárquicas de clases para mostrar relaciones de herencia.
- Construir un diagrama de colaboraciones para el sistema. Wirfs- Brock define una secuencia de tareas técnicas en el cual el análisis conduce ineludiblemente al diseño.

Una breve descripción de las tareas relacionadas al diseño de Wirfs-Brock se indica a continuación:

- Construir protocolos para cada clase. Un protocolo es una descripción formal de los mensajes a los cuales la clase responderá.
- Refinar contratos entre objetos en protocolos refinados.
- Diseñar cada operación (responsabilidad).
- Diseñar cada protocolo (diseño de interfaz).
- Crear una especificación de diseño para cada clase.
- Describir, en detalle, cada contrato.
- Definir responsabilidades privadas.
- Especificar algoritmos para cada operación.
- Observar consideraciones y restricciones especiales.
- Crear una especificación de diseño para cada subsistema.
- Identificar todas las clases encapsuladas.
- Describir los contratos en detalle para los cuales el subsistema es un servidor.

- Observar consideraciones y restricciones especiales.

El proceso unificado.

El proceso unificado es un método de desarrollo de software configurable que se adapta a proyectos según su tamaño y complejidad. Se basa en muchos años de experiencia en el uso de la tecnología orientada a objetos, en el desarrollo de software de misión crítica en una variedad de industrias por la compañía Rational: Grady Booch, James Rumbaugh y Jacobson.

El proceso unificado describe los diversos pasos involucrados en la captura de los requerimientos y en el establecimiento de una guía arquitectónica para diseñar y probar el sistema hecho de acuerdo a los requerimientos y a la arquitectura.

El proceso describe que entregables producir, como desarrollarlos y también provee patrones. El proceso unificado es soportado por herramientas que automatizan entre otras cosas, el modelado visual, la administración de cambios y las pruebas.

El proceso unificado ha adoptado un enfoque que se caracteriza por:

- Interacción con el usuario continúa desde el inicio.
- Mitigación de riesgos antes de que ocurran.
- Liberaciones frecuentes.
- Aseguramiento de la calidad.
- Involucramiento del equipo en todas las decisiones del proyecto.
- Anticiparse al cambio de requerimientos.

Las características primordiales del proceso unificado son:

- Iterativo e incremental.
- Centrado en la arquitectura.

- Guiado por casos de uso.
- Confrontación de riesgos.

2.3 Características, ventajas y desventajas de Oracle 10g.

Oracle es un Manejador de Base de Datos Relacional ampliamente utilizado a nivel mundial y es considerado como uno de los manejadores más completos y confiables del mercado, es fabricado por Oracle Corporation cuyo logo aparece en la figura 2.3.1.

Figura 2.3.1 Logo de Oracle Corporation

La tecnología Oracle se encuentra prácticamente en todas las industrias del mundo y en las oficinas de 98 de las 100 empresas “Fortune”, es la primera compañía de software que desarrolla e implementa aplicaciones para empresas con activaciones 100% por internet y su dominio ha sido casi total teniendo entre sus competidores a Microsoft SQL Server, PostgreSQL y MySQL o Firebird antes de la compra de Sun Microsystems.

Oracle 10g salió al mercado en el 2004 en su revisión 1, y en el año 2005 en su revisión 2, no es la versión más reciente dado que la más reciente es la versión 11g, sin embargo, Oracle 10g en su revisión 2 fue la versión más innovadora de Oracle introduciéndola en un nuevo terreno en el clustering, automatización y alta disponibilidad, además del respeto y la confiabilidad ganada en la industria, esta

versión ha tenido el mayor impacto en la industria de las bases de datos inclusive provoco cambios importantes en la forma de operación de los centros de datos. Oracle 10g ofrece una arquitectura escalar única que posibilita a las aplicaciones de todo tipo aprovisionar dinámicamente servidores y recursos de almacenamiento adicionales según se requiera dependiendo de la dinámica y necesidades del negocio.

Las nuevas propiedades de automatización de Oracle 10g, redujeron los costos de su administración significativamente también por el hecho que se evita la necesidad de contar con costosas herramientas y utilidades de administración.

Características.

La figura 2.3.2 muestra los componentes principales del RDBMS Oracle 10g.

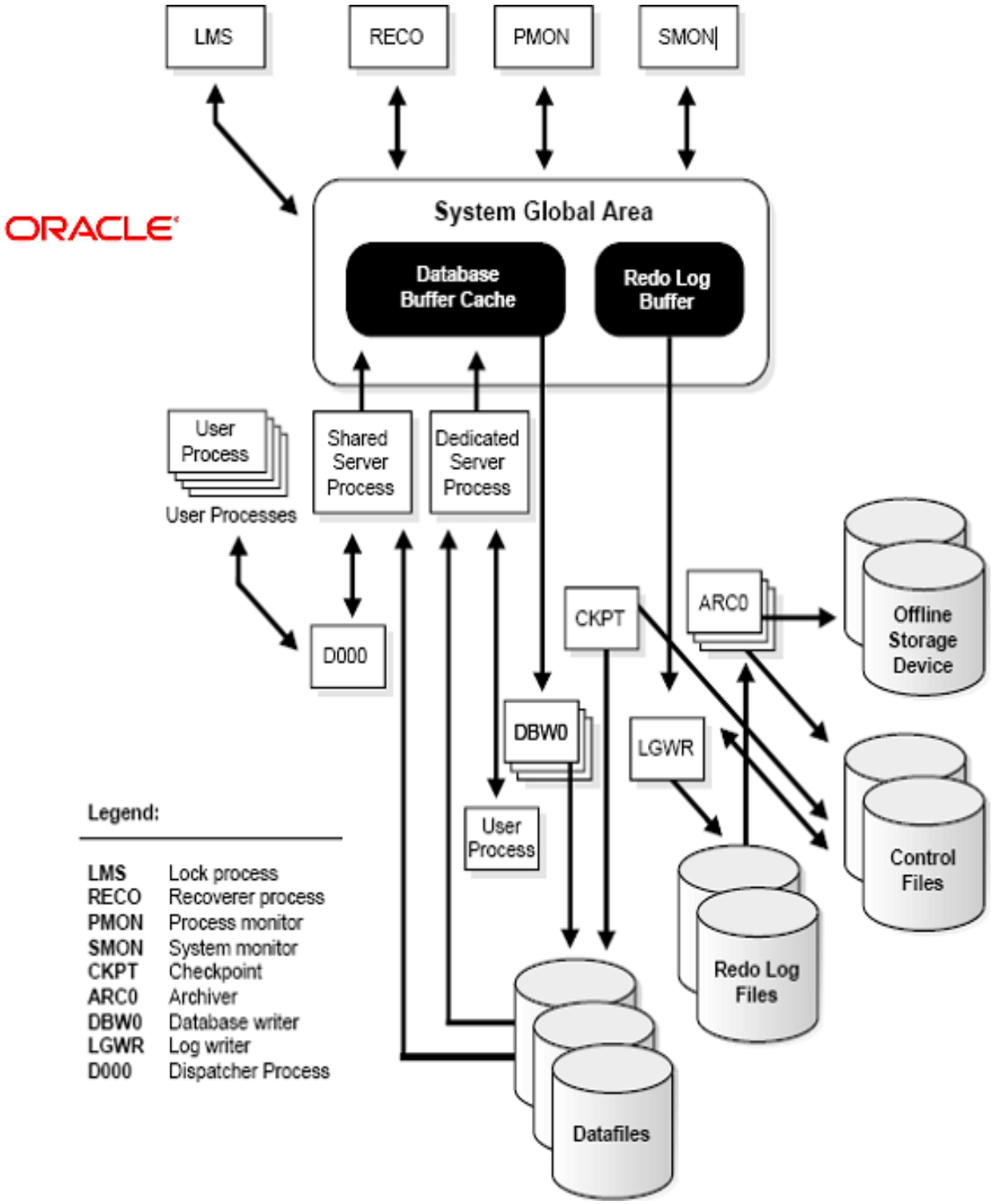

Figura 2.3.2 Componentes básicos de Oracle

- Características de desarrollo:

Característica	Descripción
Tiene múltiples ediciones	Oracle 10g cuenta con 6 ediciones: • Oracle Database Enterprise Edition (EE). • Oracle Database Standard Edition (SE). • Oracle Database Standard Edition One (SE1). • Oracle Database Express Edition (XE). • Oracle Database Personal Edition (PE). • Oracle Database Lite Edition (LE). Permitiendo seleccionar la más adecuada para las necesidades del negocio.
Es multiplataforma	Oracle 10g soporta todos los sistemas operativos productivos incluidos Windows, Unix, Linux.
Soporta cluster	Con la opción RAC toma también ventaja de la solución propuesta para clusterware, eliminando la complejidad de tener que instalar y configurar el cluster. Las capacidades de administración de almacenamiento automático distribuyen eficientemente los datos almacenados entre los discos disponibles asegurando un rendimiento óptimo.
Soporte para alto volumen de transacciones	Soporta el despliegue de un gran número de usuarios, permitiendo a la aplicación un rápido y fácil escalado desde decenas a miles de decenas de usuarios online.
Manejo de todos los tipos de datos	Soporta todos los tipos de datos relacionales estándar además del almacenamiento de XML, texto, documentos, imágenes, audio, video y datos de localización.
Múltiples formas de acceder a los datos	El acceso a los datos se realiza mediante interfaces estándar tales como SQL, JDBC, SQLJ, ODBC, OLE DB y ODP. NET, SQL/XML, XQuery y WebDAV. La lógica de negocio desarrollada en la BD puede ser escrita en Java o

	en PL/SQL .
Soporta consultas distribuidas	Soporta consultas distribuidas y transacciones entre dos o más BD, e incluye soporte para conectar vía ODBC las BDs más comunes (no de Oracle). Además, están disponibles entradas transparentes para BDs más específicas.
Particionamiento	La opción de particionamiento puede ser usada para simplificar las operaciones comunes de administración en grandes entornos de BD, con soporte para realizar estrategias de partición de datos.

- Características de administración:

Característica	Descripción
Mantenimiento sin Interrupción del servicio	Diseñado para proteger las operaciones de negocio criticas del impacto del mantenimiento de rutina. Pueden ser añadidos y usados por Oracle nuevo hardware, memoria y almacenamiento sin la necesidad de reiniciar el manejador. En la BD, las tablas pueden ser reubicadas o su tipo de almacenamiento puede ser cambiado, pueden ser añadidos o rehechos nuevos índices, pueden añadirse columnas y pueden ser eliminadas y cambiadas de nombre sin interrupciones para los usuarios finales que están accediendo a los datos.
Administración de cambios y de configuración	El paquete de diagnósticos proporciona un conjunto de diagnósticos automáticos y capacidades de monitoreo dentro de la BD, mientras que el paquete de puesta a punto (o “tuning”) ofrece una solución fácil que automatiza la compleja tarea que consume tanto tiempo del tuning de aplicaciones.

	El paquete de administración de cambios analiza las complejas dependencias asociadas con el cambio de la aplicación y automáticamente realiza los cambios requeridos en la BD, reduciendo errores, mientras que el paquete de administración de la configuración reduce la labor asociada al manejo de múltiples BD, automatizando la instalación, actualizando y clonando la BD.
Múltiples configuraciones de hardware	Soporta desde pequeñas máquinas de un solo procesador hasta entornos Symmetric Multi-Processing (SMP), los entornos cluster y grid son también soportados con la opción de Oracle Real Application Cluster.
Recuperación de errores	Proporciona un conjunto de capacidades únicas de flashback que ayudan al administrador a diagnosticar y deshacer de forma sencilla el efecto de los errores humanos incluyendo cambios en una sola fila, cambios hechos por una transacción malintencionada o bien todos los cambios hechos a una o más tablas (incluyendo la eliminación de una tabla), y todos los cambios hechos a una BD entera.
Replicación de datos	Proporciona un marco de trabajo para capturar, organizar y procesar eventos en la BD, tales como aquellos que son causados por cambios de datos o creados por las aplicaciones de negocio. Estos eventos, junto con los cambios en los datos asociados o los mensajes de aplicación, pueden ser automáticamente propagados y aplicados a una o más BDs o aplicaciones, proporcionando una solución integrada para la consulta de mensajes y la replicación de datos.

Seguridad avanzada	Proporciona características de seguridad a nivel de fila y de columna, así como características de encriptación de datos, la administración de llaves, la autentificación de proxy, el contexto de la aplicación y los roles seguros de la aplicación. A estas características se suman las características de seguridad comunes, como las revisiones, la comprobación de la complejidad de las contraseñas, los roles de la BD, los procedimientos almacenados y las funciones. Todas las comunicaciones con la BD Oracle pueden ser encriptadas.

- Características de inteligencia de negocio:

Motor de calculo OLAP	Para la inteligencia de negocio, esta versión de Oracle proporciona capacidades analíticas, estadísticas y de modelado que pueden ser accedidas directamente desde cualquier entorno basado en SQL. Estas capacidades pueden ser expandidas con el uso de las opciones de Oracle On-Line Analytical Processing (OLAP) y Data Mining (minería de datos), proporcionando un motor de calculo OLAP de gran rendimiento.
Indexación específica y optimización a consultas	Las aplicaciones de Inteligencia de negocio se beneficiarán particularmente de la indexación por mapeo de bits, de las capacidades de unión, de la re-escritura transparente de consultas y de las operaciones de consulta paralelas. Todo ello da como resultado una mejora significativa en el rendimiento de las consultas.

Ventajas

- Es el RDBMS más utilizado y documentado a nivel mundial.
- Puede ejecutarse en múltiples plataformas, desde una PC hasta arquitecturas de múltiples servidores.
- Puede ejecutarse en múltiples sistemas operativos, desde Windows hasta Unix.
- Soporta todas las funciones de un RDBMS “Formal” y de alto volumen, así como un lenguaje de diseño muy completo (PL/SQL).
- Mediante PL/SQL se pueden programar triggers y procedimientos almacenados, con una integridad referencial declarativa poderosa.
- Es ideal para empresas que necesita soporte de un alto volumen de transacciones y aplicaciones con almacenamiento intensivo de datos.
- Introduce nuevos conceptos como índices de mapas de bits, vistas materializadas, particionamiento, tablas externas y más.

Desventajas

- Su mayor inconveniente es el precio de sus licencias.
- Sus licencias además de caras pueden llegar a ser vendidas incluso a nivel procesador.
- Obliga a sus clientes a estarse actualizando dado que las versiones anteriores pierden soporte.
- Elevado costo en la formación oficial, mediante su división de “Educación” manejando incluso niveles de certificación.

2.4. Características, ventajas y desventajas del JDK 1.5.

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código de máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

Componentes

Bibliotecas de Java, que son el resultado de compilar el código fuente desarrollado por quien implementa la JRE (Java Runtime Environment) y que ofrecen apoyo para el desarrollo en Java. Algunos ejemplos de estas bibliotecas son:

Las bibliotecas centrales, que incluyen:

- Una colección de bibliotecas para implementar estructuras de datos como listas, arrays, árboles y conjuntos.
- Bibliotecas para análisis de XML.
- Seguridad.
- Bibliotecas de internacionalización y localización.

Bibliotecas de integración, que permiten la comunicación con sistemas externos. Estas bibliotecas incluyen:

- La API para acceso a bases de datos JDBC (Java DataBase Conectivity).

- La interfaz JNDI (Java Naming and Directory Interface) para servicios de directorio.
- RMI (Remote Method Invocation) y CORBA para el desarrollo de aplicaciones distribuidas.

Bibliotecas para la interfaz de usuario, que incluyen:

- El conjunto de herramientas nativas AWT (Abstract Windowing Toolkit), que ofrece componentes GUI (Graphical User Interface), mecanismos para usarlos y manejar sus eventos asociados.
- Las Bibliotecas de Swing, construidas sobre AWT pero ofrecen implementaciones no nativas de los componentes de AWT.
- APIs para la captura, procesamiento y reproducción de audio.
- Una implementación dependiente de la plataforma en que se ejecuta de la máquina virtual de Java (JVM), que es la encargada de la ejecución del código de las bibliotecas y las aplicaciones externas.
- Plugins o conectores que permiten ejecutar applets en los navegadores Web.
- Java Web Start, para la distribución de aplicaciones Java a través de Internet.
- Documentación y licencia.

JDK/J2SE.

JDK (Java Development Kit) y J2SE (Java 2 Standard Edition) son nombres para el mismo componente utilizado en ambientes Java, el cual agrupa las diversas funcionalidades necesarias para desarrollar programas Java.

El conjunto de clases base proporcionadas por el JDK/J2SE incluyen clases de uso común tales como: manipulación de string's, fechas y números así como todas las funcionalidades *base* esperadas de un lenguaje computacional como:

ordenamiento y manipulación de arreglos, operaciones matemáticas complejas (trigonométricas y exponenciales) y escritura/lectura de "streams".

Además de estas clases base, el JDK/J2SE posee otra serie de clases especializadas que también ofrecen la base para la creación de otros componentes Java que incluyen: applets, objetos CORBA, fragmentos auditivos, soporte de criptografía (seguridad) y manipulación de XML entre otras funcionalidades.

J2SE 5.0 (30 de septiembre de 2004) — nombre clave: *Tiger*. (Originalmente numerado 1.5, esta notación aún es usada internamente) desarrollado bajo JSR 176, Tiger añadió un número significativo de nuevas características.

- Plantillas (genéricos) — provee conversión de tipos (type safety) en tiempo de compilación para colecciones y elimina la necesidad de la mayoría de conversión de tipos (type casting).
- Metadatos — también llamados anotaciones, permite a estructuras del lenguaje como las clases o los métodos, ser etiquetados con datos adicionales, que puedan ser procesados posteriormente por utilidades.
- Autoboxing/unboxing — Conversiones automáticas entre tipos primitivos (como los int) y clases de envoltura primitivas (como integer).
- Bucle for mejorado — La sintaxis para el bucle for se ha extendido con una sintaxis especial para iterar sobre cada miembro de un array o sobre cualquier clase que implemente “Iterable”, como la clase estándar “Collection”.

Características

- Lenguaje simple. Se lo conoce como lenguaje simple porque viene de la misma estructura de C y C++; ya que C++ fue un referente para la creación

de java por eso utiliza determinadas características de C++ y se han eliminado otras.

- Orientado a objeto. Toda la programación en java en su mayoría está orientada a objeto, ya que al estar agrupados en estructuras encapsuladas es más fácil su manipulación.
- Distribuido. Permite abrir sockets, establecer y aceptar conexiones con los servidores o clientes remotos; facilita la creación de aplicaciones distribuidas ya que proporciona una colección de clases para aplicaciones en red.
- Robusto. Es altamente fiable en comparación con C, se han eliminado muchas características con la aritmética de punteros, proporciona numerosas comprobaciones en compilación y en tiempo de ejecución.
- Seguro. La seguridad es una característica muy importante en java ya que se han implementado barreras de seguridad en el lenguaje y en el sistema de ejecución de tiempo real.
- Indiferente a la arquitectura. Java es compatible con los más variados entornos de red, cualesquiera que sean estos desde Windows 95, Unix a Windows Nt y Mac, para poder trabajar con diferentes sistemas operativos.
- Java es muy versátil ya que utiliza byte-codes que es un formato intermedio que sirve para transportar el código eficientemente o de diferentes plataformas (Hardware - Software).
- Portable. Por ser indiferente a la arquitectura sobre la cual está trabajando, esto hace que su portabilidad sea muy eficiente, sus programas son iguales en cualquiera de las plataformas, ya que Java especifica tamaños básicos, esto se conoce como la máquina virtual de Java.
- Interpretado y compilado a la vez. Java puede ser compilado e interpretado en tiempo real, ya que cuando se construye el código fuente este se transforma en una especie de código de máquina.
- Multihebra o Multihilos. Java tiene una facilidad de cumplir varias funciones al mismo tiempo, gracias a su función de multihilos ya que por cada hilo

que el programa tenga se ejecutarán en tiempo real muchas funciones al mismo tiempo.

- Dinámico. El lenguaje java es muy dinámico en la fase de enlazado, sus clases solamente actuarán en la medida en que sean requeridas o necesitadas con esto permitirá que los enlaces se puedan incluir desde fuentes muy variadas o desde la red.
- Produce applets. En java se pueden crear aplicaciones independientes y applets.
- Alto rendimiento. Java es considerado de alto rendimiento por ser tan veloz en el momento de correr los programas y por ahorrase muchas líneas de código.

Ventajas

- El JDK es una herramienta libre de licencias (sin costo), creada por Sun, está respaldado por un gran número de proveedores.
- Existe soporte dado por Sun.
- Debido a que existen diferentes productos de Java, hay más de un proveedor de servicios.
- Sun saca al mercado cada 6 meses una nueva versión del JDK.
- Es independiente de la plataforma de desarrollo.
- Existen dentro de su librería clases gráficas como awt y swing, las cuales permiten crear objetos gráficos comunes altamente configurables y con una arquitectura independiente de la plataforma.
- Java permite a los desarrolladores aprovechar la flexibilidad de la Programación Orientada a Objetos en el diseño de sus aplicaciones.
- El conocimiento sobre tecnología Java está en alto crecimiento en el mercado.

- Se puede acceder a bases de datos fácilmente con JDBC, independientemente de la plataforma utilizada o el manejo de las bases de datos es uniforme, es decir transparente y simple.
- El sistema de Java tiene ciertas políticas que evitan se puedan codificar virus con este lenguaje. Existen muchas restricciones, especialmente para los applets, que limitan lo que se puede y no hacer con los recursos críticos de una computadora.
- Es una fuente abierta, así que los usuarios no tienen que luchar con los gastos sobre patente y son Independientes de la plataforma.
- Java realiza la colección de basura de las ayudas, así que la gestión de memoria es automática.
- Usando Java podemos desarrollar aplicaciones web dinámicas.
- Permite que se pueda crear programas modulares y códigos reutilizables.

Desventajas

- La velocidad. Los programas hechos en Java no tienden a ser muy rápidos, supuestamente se está trabajando en mejorar esto. Como los programas de Java son interpretados nunca alcanzan la velocidad de un verdadero ejecutable.

- Java es un lenguaje de programación. Esta es otra gran limitante, por más que digan que es orientado a objetos y que es muy fácil de aprender sigue siendo un lenguaje y por lo tanto aprenderlo no es cosa fácil. Especialmente para los no programadores.
- Java es nuevo. En pocas palabras todavía no se conocen bien todas sus capacidades.
- Hay diferentes tipos de soporte técnico para la misma herramienta, por lo que el análisis de la mejor opción se dificulta.
- Para manejo a bajo nivel deben usarse métodos nativos, lo que limita la portabilidad.

- El diseño de interfaces gráficas con awt y swing no es simple o existen herramientas como el JBuilder que permiten generar interfaces gráficas de manera sencilla, pero tienen un costo adicional.
- Puede ser que no haya JDBC para bases de datos poco comerciales.
- Algunas herramientas tienen un costo adicional.
- El código Java puede ser a veces redundante en comparación con otros lenguajes.
- Java no dispone de operadores de sobrecarga definidos por el usuario., esta característica podía complicar la lectura y mantenimiento de los programas.

Pero en general Java posee muchas ventajas y se pueden hacer cosas muy interesantes con esto. Hay que prestar especial atención a lo que está sucediendo en el mundo de la computación, a pesar de que Java es relativamente nuevo, posee mucha fuerza y es tema de moda en cualquier medio computacional. Muchas personas apuestan a futuro y piensan en Java.

2.5. Conceptos básicos de mediación y comunicaciones.

Conceptos de comunicaciones.

- Comunicación de datos: Es el proceso de comunicar información en forma binaria entre dos o más puntos. Requiere cuatro elementos básicos (figura 2.5.1) que son:
    - Emisor: Dispositivo que transmite los datos.
    - Mensaje: Lo conforman los datos a ser transmitidos.
    - Medio: Consiste en el recorrido de los datos desde el origen hasta su destino.
    - Receptor: Dispositivo de destino de los datos.

DIAGRAMA DEL PROCESO DE LA COMUNICACIÓN.

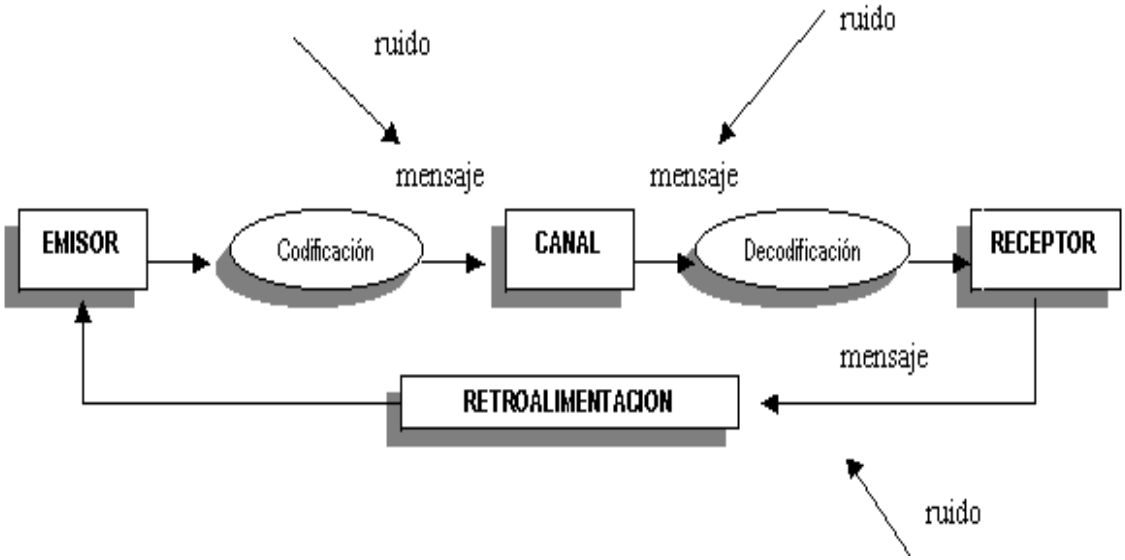

Figura 2.5.1 D Diagrama del proceso de la comunicación

- Codificación: Es la conversión de un sistema de datos a otro sistema llamado destino. De esto se desprende que la información resultante es equivalente a la información de origen.
- Decodificación: Es el proceso del cual el receptor interpreta el mensaje y lo traduce en información significativa. El receptor debe recibir primero el mensaje y luego interpretarlo.
- Retroalimentación: Es el último eslabón del proceso de comunicación, es el paso que cierra el circuito, poniendo el mensaje de respuesta devuelta en el sistema. La forma en que se puede saber si la comunicación se logró efectivamente es a través de la retroalimentación que nos dé el receptor, por medio de su reacción o respuesta.
- Protocolos: Conjunto de reglas que posibilitan la transferencia de datos entre dos o más computadoras.
- FTP: En inglés File Transfer Protocol, protocolo de conexión para extraer archivos entre equipos.
- Trama: Grupo de bits con un formato predefinido usado en protocolos orientados a bits.

- Parser: Proceso de clasificación que genera el CDR.
- Paquetes: Cantidad mínima de datos que se transmite en una red o entre dispositivos. Tiene una estructura y longitud distinta según el protocolo al que pertenezca. También llamado TRAMA.
- Códigos: Acuerdo previo sobre un conjunto de significados que definen una serie de símbolos y caracteres. Toda combinación de bits representa un carácter dentro de la tabla de códigos. Las tablas de códigos más reconocidas son las del código ASCII y las del código EBCDIC (Extended Binary Coded Decimal Interchange Code, Código Ampliado de Intercambio decimal codificado en binario).
- ASCII: Es un código de siete bits, usa cadenas de bits representables con siete dígitos binarios (que van de 0 a 127 en base decimal), para representar información de caracteres.
- Hilos de transmisión: Para describir el circuito que compone un canal se utiliza el término “pares”. Uno de los hilos del par sirve para transmitir o recibir datos y el otro es la línea de retorno electrónico.
- Switch: Equipo que permite realizar la conmutación de llamadas entre dos equipos de comunicación de telefonía celular.
- Interconexión: Sistema que genera las facturas correspondientes para otros operadores de telefonía con los que se tengan firmados contratos de interconexión o intercambio de capacidad.
- Plataforma de Tarificación: Sistema que maneja el saldo al cliente.
- AAA (Authentication, Authorización, Accounting): Sistema utilizado para prestar el servicio de envío y recepción de datos, ya sea para Web o PTT.
- Multimedia: Sistema que proporciona el servicio para extraer los archivos de Ringtones e imágenes de los diferentes proveedores.
- Proceso de facturación: Es la etapa en la que se realiza el conjunto de actividades mediante la cual se generan las facturas correspondientes a los consumos de los usuarios o suscriptores de los servicios públicos de telecomunicaciones.

- Proceso de tarificación: Es la etapa en la que se realiza el conjunto de actividades mediante la cual se le aplica un valor monetario a los consumos medidos en el proceso de tasación.
- Proceso de tasación: Es la etapa en la que se realiza el conjunto de actividades mediante la cual se mide el consumo de los usuarios o suscriptores de los servicios públicos de telecomunicaciones.
- Servidores: Computadoras dedicadas para proporcionar servicios a otras computadoras tales como servicios web, bases de datos, almacenamiento en discos, acceso a las impresoras, unidades para respaldo de archivos, acceso a otras redes o computadoras centrales.

El CDR en sus siglas en inglés es Call Details Record o en español Registro con el detalle de una llamada.

El CDR está compuesto por los siguientes campos:

- El número que hace la llamada (el que llama).
- El número que recibe la llamada (interlocutor).
- Cuando se inició la llamada (fecha y hora).
- La duración de la llamada.
- El cobro al número de teléfono por la llamada.
- El identificador de la central telefónica.
- El número de serie que identifica el registro.
- Dígitos adicionales en el número de llamada.
- El resultado de la llamada (si fue respondida, ocupado, etc.).
- La ruta por la que la llamada entró.
- Tipo de llamada.
- Condiciones de falla encontrados.

Proceso de mediación.

La mediación implica recolectar y dar formato a los registros a utilizar. El sistema de mediación actúa como un buffer valorizado, desacoplando la infraestructura de los switches desde el sistema de soporte operacional y permitiendo a cada uno ser configurado independientemente de los demás.

Los Sistemas de Mediación implementan las siguientes funciones genérales:

- Recepción de datos de CDRs desde las centrales.
- Control en la recepción de archivos, secuencia de generación de archivos, recepción de archivos duplicados, etc.
- Planificación del procesamiento de archivos.
- Conversión de los archivos con CDRs originales, en archivos en el formato requerido por el sistema de facturación. Esto implica validación y filtrado de ticket de acuerdo a los escenarios de registro definidos para facturar.
- Creación de copias de respaldos de los archivos con CDRs recibidos.
- Envío de los archivos con CDRs a sistemas de facturación, de análisis de fraude, de control de tráfico, búsquedas, control de consumo telefónico, etc.

La conexión con las centrales telefónicas y nodos, desde el Sistema de Mediación se realiza por medio de módulos que resuelven las particularidades de comunicación con cada tipo de central o nodo (ver figura 2.5.2).

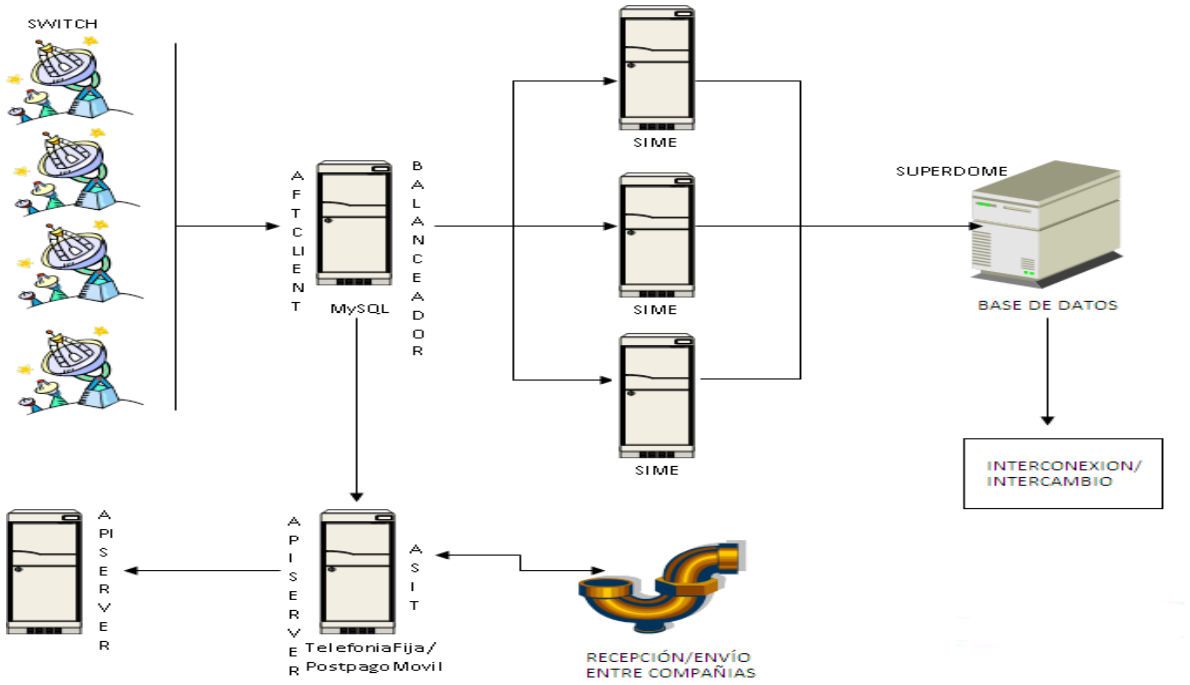

Figura 2.5.2 Proceso de mediación.

Dependiendo del tipo de fuente de datos, varía la forma en que se reciben los archivos. En algunos casos la comunicación es iniciada por la central y en otros la misma se inicia desde el Sistema de Mediación.

El sistema cuenta con un módulo de control que se encarga de procesar los archivos provenientes de las centrales, revisar desfasamiento en la recepción de archivos, controlar y autorizar la recepción de reenvíos así como generar alarmas.

Una vez recibidos los archivos de tasación originales se planifica su procesamiento por los módulos de validación y formateo de CDRs, con el objetivo de enviarlos al sistema de facturación de la empresa.

Por lo que el Sistema de Mediación es un proceso en donde los datos son transformados a datos formateados. Esta transformación precisa una plataforma distinta que conecta los dos componentes de un sistema de telecomunicaciones, la red switcheada y los sistemas de información del negocio, mientras se asegura que estas funcionen independientes unas de otras.

El Sistema de Mediación recolecta una gran cantidad de datos útiles en un tiempo determinado, en forma segura y confiable desde varias tecnologías distintas.