Universidad Central “Marta Abreu” de Las Villas

Facultad de Matemática Física y Computación

Tesis presentada en opción al grado de Master en Computación Aplicada

Módulo de Transformadores de Distribución

Autor: Ing. Magdiel García González

Tutores: Dra. Luisa González González

Msc. Abel Rodríguez Morffi

Santa Clara

2007

"Profundizar en el conocimiento científico es una de las mejores vías para lograr plenitud y libertad."

- Pilar Álvarez Pellicero

-A mi hijo Andy, el mayor suceso de mi vida.

- A Raúl Fernández Alvares, a quien le debo gran parte de mi formación profesional.
- A Luisa Gonzáles y Abel Rodriguez de quienes creo no puedan existir mejores tutores.
- A los integrantes del grupo SIGERE que me ayudaron sobre todo en la conclusión de esta tesis.
- A Mideisy por tener la dedicación y paciencia necesaria durante mi etapa de estudios.
- A mis compañeros del departamento Comercial
- A mi mama y abuela por la formación que me dieron desde niño.

RESUMEN

En nuestro país, durante los últimos años, la distribución ha sido el área menos atendida de la Unión Eléctrica, con altos índices de pérdidas e interrupciones, un creciente índice de transformadores dañados, y el escaso nivel de informatización y automatización. Como respuesta a algunos de los problemas informáticos detectados, en esta investigación se desarrolla un sistema para el control de los transformadores en la OBE de Sancti Spiritus, teniendo en cuenta que la topología de las redes informáticas no es la más adecuada para contar con una Base de Datos Centralizada provincial donde accedan todos los usuarios de la provincia, taller o territorio; se diseña una Base de Datos Distribuida aplicando un conjunto de herramientas computacionales que apoyan los pasos de diseño de estos tipos de Bases de Datos, y se crea un método de replicación de datos entre los servidores de la provincia con los territorios y con los talleres, para poder seguir el ciclo de vida completo de cada transformador. Además de mantener datos históricos de todas las operaciones que se realizan para estudios comparativos del comportamiento de las diferentes equipos y prever posibles fallas futuras en ellos, también se da la posibilidad de visualizar la información mediante reportes y a través de una aplicación del tipo Sistema de Información Geográfico para ver y localizar en un mapa los Bancos de Transformadores.

ABSTRACT

In our country, during the last years, the distribution has been the least attended area of the Electrical Union, with high indexes of losses and interruptions, an increasing rate of damaged transformers, and the scarce level of information and automation. As response to some of the computer detected problems, in this investigation a system develops for the control of the transformers in the OBE of Sancti Spíritus. Bearing in mind that the topology of the computer networks is not most adapted to be provided with a Centralized provincial Database where there should gain access all the users of the province, workshop or territory; it is designed a Distributed Database applying a set of rules that support the steps of design of these types of Databases, and creates a method of replicación of information between the servants of the province with the territories and with the workshops, to be able to continue the finished life cycle of every transformer. In addition to supporting historical information of all the operations that different teams realize for comparative studies of the behavior of and foreseeing possible future flaws in them, also there happens the possibility of visualizing the information by means of reports and across an application of the type Geographical Information System to see and to locate in a map the Transformers Bank.

ÍNDICE

INTRODUCCIÓN.....1
CAPÍTULO 1. AUTOMATIZACIÓN DE LAS EMPRESAS ELÉCTRICAS. 5
1.1. ANTECEDENTES 5
1.2. VISIÓN DEL SIGERE 6
1.3. ALCANCE Y COMPOSICIÓN DEL SIGERE 6
1.4. ARQUITECTURA DEL SIGERE 7
1.4.1. Enlaces con SIGE-UNE..... 7
1.4.2. Subsistema de Instalaciones..... 8
1.4.3. Subsistema de Operación..... 8
1.4.4. Subsistema de Explotación 9
1.4.5. Subsistema de Planificación 10
1.4.6. Subsistema de Gestión y Control..... 10
1.5. SITUACIÓN ACTUAL DEL CONTROL DE LOS TRANSFORMADORES 10
1.6 MÓDULO DE TRANSFORMADORES DE DISTRIBUCIÓN E INTERCAMBIOS 11
1.7. CONCLUSIONES PARCIALES 14
CAPÍTULO 2. ACERCA DE LAS BASES DE DATOS DISTRIBUIDAS 15
2.1. INTRODUCCIÓN 15
2.2. ESTRATEGIAS PARA EL DISEÑO 16
2.3. EL DISEÑO DE LA DISTRIBUCIÓN 18
2.4. EL PROBLEMA DE LA FRAGMENTACIÓN..... 19
2.4.1. Alternativas de fragmentación 20
2.4.2. Grado de fragmentación..... 21
2.4.3. Reglas de la fragmentación..... 22
2.4.4. Diseño de la fragmentación 22
2.4.5. Información requerida..... 23
2.4.6. Diseño de la fragmentación horizontal 24
2.4.7. Diseño de la fragmentación vertical 26
2.4.8. Diseño de la fragmentación mixta 28
2.5. EL PROBLEMA DE LA ASIGNACIÓN 29

2.5.1. Modelo de asignación 30
2.5.2. Información Requerida 33
2.5.3. Modelo de asignación presentado por Özsu y Valduriez 34
2.6. HERRAMIENTAS DE AYUDA AL DISEÑO DE BDD 37
2.7. CONCLUSIONES PARCIALES 39
CAPÍTULO 3. ASPECTOS DEL DISEÑO Y LA IMPLEMENTACIÓN 40
3.1. PROBLEMA DEL CONTROL DE LOS TRANSFORMADORES DE DISTRIBUCIÓN 40
3.2. MODELACIÓN CONCEPTUAL 42
3.2.1. Caracterización de aplicaciones 43
3.2.2. Caracterización de la red y los sitios 47
3.3. MODELACIÓN LÓGICA 49
3.2. MODELACIÓN FÍSICA 50
3.3. IMPLEMENTACIÓN DEL MÓDULO DE TRANSFORMADORES DE DISTRIBUCIÓN 50
CONCLUSIONES 62
RECOMENDACIONES 64
BIBLIOGRAFÍA 65
ANEXO .....

INTRODUCCIÓN

Existen varias razones técnicas para distribuir datos, en particular, los sistemas distribuidos se adaptan mejor a las necesidades de las organizaciones descentralizadas ya que reflejan más adecuadamente su estructura y tienen importantes ventajas con respecto a los centralizados. La distribución involucra el hecho de que los datos no residen necesariamente en el mismo sitio pero poseen propiedades comunes que los vinculan, y se facilita su acceso a través de una interfaz común. Es necesario destacar que el enlace ente los datos se lleva a cabo en una red de comunicación, que por generalización determina que los sitios estén localizados en diferentes áreas geográficas y con capacidad de procesamiento autónomo.

Desde el punto de vista conceptual es necesaria la descentralización pues los sistemas distribuidos sustituyen grandes sistemas centralizados, siendo la economía la fuerza motriz real de la tendencia hacia la descentralización; ésta se justifica desde el punto de vista tecnológico ya que permite autonomía local y promueve la evolución de los sistemas y los cambios en los requerimientos de usuario, proporciona una arquitectura de sistemas simple, flexible y tolerante a fallas, además de ofrecer buenos rendimientos.

Las diferentes técnicas de diseño de distribución de datos generalmente se basan en la semántica de los datos y se rigen por idénticos principios que las bases de datos centralizadas incorporando otros detalles específicos, como la fragmentación de las entidades y su posterior localización en los diferentes sitios de la red. Esta fragmentación es muy útil a fin de mejorar los tiempos de respuesta y garantizar el paralelismo del sistema [3, 8, 20, 21, 25, 37]. Dado que los datos pueden estar replicados, el control de concurrencia y los mecanismos de recuperación son mucho más complejos que en un sistema centralizado. Esto amplía el grado de complejidad de los sistemas con respecto a los sistemas centralizados. En el diseño de la distribución influyen muchos factores relacionados con la BD, las aplicaciones que acceden a la misma, la comunicación de la red, y sobre el sistema de computadoras que la soporta; lo que hace que sea muy complicada la formulación de un problema de distribución. Nuevos retos se crean al considerar el diseño la distribución, el decidir cómo fragmentar y distribuir los datos sobre los diferentes sitios y cuáles de estos datos deben ser replicados. A pesar de la complejidad de este proceso existe un gran déficit de metodologías y herramientas de apoyo al mismo.

Un problema que desde el punto de vista tecnológico necesitaba reflejar la estructura inherentemente distribuida de las diferentes unidades de la Empresa Eléctrica, teniendo

autonomía local y ofreciendo buen rendimiento es el del control de los bancos de transformadores, formados por transformadores, pararrayos y portafusibles ubicados en ellos, así como las operaciones relacionadas en la explotación de los mismos. La Organización Básica Eléctrica (OBE) de Sancti Spíritus se dio a la tarea de implementar un módulo para la gestión del control de los bancos de transformadores instalados de una forma descentralizada, y hacer que este control forme parte de una herramienta existente, el Sistema Integral de Gestión de Redes (SIGERE). La información que se maneja es de vital importancia para la operación de la red electroenergética nacional y se realiza siguiendo procedimientos existentes o creados para ello, previamente aprobados por el área técnica de la Unión Eléctrica.

Actualmente, en muchos países existen sistemas computacionales de control de la distribución de la energía eléctrica y el comercial con grandes posibilidades para el diseño, control y manejo de las redes. En nuestro país, durante los últimos años, la distribución ha sido el área de trabajo menos atendida de la Unión Eléctrica. Los efectos objetivos de esto son los altos índices de pérdidas e interrupciones, el creciente índice de transformadores dañados, y el escaso nivel de informatización y automatización. Entre los efectos subjetivos se encuentran el bajo nivel técnico, la inestabilidad del personal que atiende la distribución, la falta de estructuras uniformes, entre otros. Se opina que uno de los efectos más importantes es la carencia de información adecuada sobre los diferentes elementos y acciones de la distribución, la ausencia de un Sistema Informático Integral que permita el empleo de técnicas modernas en el control y automatización de la información inherente a las redes.

Como respuesta a los problemas informáticos detectados se identifica como necesidad de automatización el desarrollo de un Sistema Integral de Gestión de la Distribución (SIGEDI) que después evoluciona a SIGERE cuyo objetivo es mejorar radicalmente el control de las redes, permitiendo la reducción de los gastos totales de explotación de la distribución, un mejor servicio a nuestros clientes y la superación profesional de nuestros técnicos que permitiría un uso más racional de los recursos humanos.

En el diseño del paquete integrado SIGERE se han concebido cuatro módulos principales y veintiocho subsistemas. Dentro del módulo de explotación se encuentra el Subsistema de Transformadores. El transformador es el equipo que más abunda en las redes eléctricas cubanas con una distribución espacial muy variada. Éste es el más cercano al cliente y es el que más tiende a fallar; de ahí la importancia de minimizar sus averías. Esta es la razón por la que se necesita hacer un sistema capaz de automatizar toda la información relacionada con ellos, poder

seguir sus ciclos de mantenimiento, su estado de carga; así como ser capaz de prevenir las fallas. Para diseñar el subsistema hay que tener en cuenta la estructura geográficamente distribuida de las Empresas Eléctricas y sus propias características de comunicaciones; teniendo en cuenta los cuatro niveles bien definidos: nacional, provincial, territorial y de sucursal, además de un quinto nivel que es el del taller al mismo nivel de la provincia.

Como la base de datos (BD) del SIGERE ya se encuentra en todas las provincias, implementada en Servidores Microsoft® SQL Server™, y la topología de las redes informáticas no es la más adecuada para contar con una BD provincial donde accedan todos los usuarios ya sean de la provincia, taller o territorio, es necesario diseñar una BD fragmentada y crear un método de replicación de datos entre los servidores de la provincia con los territorios y con los talleres, para poder seguir el ciclo de vida completo de cada transformador. Debe, además, mantener datos históricos de todas las operaciones que se realizan, esto le da la facilidad de hacer numerosos estudios comparativos del comportamiento de las diferentes instalaciones, equipos, así como prever posibles fallas futuras en ellos. También debe dar la posibilidad de visualizar la información mediante reportes y a través de una aplicación del tipo Sistema de Información Geográfico, para ver y localizar en un mapa los Bancos de Transformadores por circuitos, por voltaje, transformadores por potencia, entre otros parámetros. En los lugares donde se tenga asociado el cliente al poste, puede verse hasta los clientes que se alimentan de un banco determinado, las rutas de cobro, el circuito perteneciente a cada banco, así como el circuito primario del que se alimenta. Esto es de suma importancia a la hora de agilizar la toma de decisiones pues cuenta con la ubicación espacial del equipo y la instalación.

OBJETIVO GENERAL

Automatizar el control de la gestión de los transformadores de distribución en la empresa eléctrica.

OBJETIVOS ESPECÍFICOS

Hacer un estudio sobre la teoría existente respecto al diseño de Base de Datos Distribuidas como una solución posible al problema planteado.

Diseñar e implementar un sistema para el control de los transformadores, incluyendo las pruebas y mediciones que se realicen sobre estos equipos.

Brindar una arquitectura de software escalable y extensible, aplicable a todas las empresas eléctricas del país, de manera que responda de forma eficiente y fiable a las necesidades de la Unión Eléctrica.

Hacer uso de las herramientas desarrolladas en el laboratorio de Base de Datos de la Universidad Central “Marta Abreu” de las Villas para el diseño de Bases de Datos Distribuidas.

PREGUNTAS DE INVESTIGACIÓN

¿Será posible la realización de una aplicación capaz de gestionar el control de los transformadores posibilitando el ahorro de tiempo y esfuerzo usado actualmente para esta tarea?

¿Constituirá la utilización de las bases distribuidas una solución eficiente que permita que en cada localidad esté la información necesaria en el tiempo necesario adaptándose a las condiciones de comunicaciones existentes?

NOVEDAD CIENTÍFICA

La novedad consiste en la sistematización de resultados teóricos y su empleo en la obtención de un sistema de bases de datos distribuidas que funcione en un ambiente desconectado.

La tesis está estructurada en tres capítulos:

El Capítulo 1 se expone una panorámica sobre la automatización de las empresas eléctricas con el propósito de argumentar la necesidad de implementación del Módulo de Transformadores de Distribución y de Intercambios en un ambiente distribuido.

En el Capítulo 2 ofrece un resumen sobre el diseño de bases de datos en ambientes distribuidos, haciéndose énfasis en el modelo teórico y el enfoque que se le ha dado al problema de diseño de distribución. Se exponen investigaciones que permiten su implementación computacional.

En el Capítulo 3 se exponen los resultados del diseño e implementación del Módulo de Transformadores de Distribución y de Intercambios, se validan los resultados de la investigación en la gestión del control de los bancos de transformadores en la OBE de Sancti Spíritus, siguiendo el ciclo de los transformadores desde que se instala en un banco hasta que son retirados de la línea, almacenando todas las actividades que se realizan sobre un banco de transformadores, conformando la historia de cada banco que consta con la información necesaria para futuros estudios. También se ofrecen detalles de la aplicación de herramientas de ayuda al diseño de bases de datos distribuidas en el proceso de diseño de la arquitectura de la información y se arriba a conclusiones.

Este documento culmina con las conclusiones, recomendaciones, bibliografía y los anexos.

CAPÍTULO 1. AUTOMATIZACIÓN DE LAS EMPRESAS ELÉCTRICAS.

En el presente capítulo se aborda el problema de automatización de las empresas eléctricas y se da a conocer, a grandes rasgos, el Sistema Integral de Gestión de Redes Eléctricas (SIGERE) así como el Módulo de Transformadores de Distribución contenido en el mismo. Aquí se detalla la necesidad del uso de las Base de Datos Distribuidas (BDD) para enfrentar la necesidad de contar con la información de cada equipo en cada momento de su ciclo de vida. Se exponen las características del Módulo de Transformadores, las facilidades que el mismo brinda en lo referente a la hora de programar mantenimientos, evitar averías, detectar sobrecargas y optimizar el uso de los equipos detectando zonas subcargadas; esto ayuda a tomar las decisiones correctas en lo referente al manejo adecuado de los Transformadores.

1.1. ANTECEDENTES

Actualmente, en muchos países existen sistemas computacionales de control de la distribución de la energía eléctrica y comercial con grandes posibilidades para el diseño, control y manejo de las redes. En nuestro país, durante los últimos años, la distribución ha sido el área de trabajo menos atendida de la Unión Eléctrica. Los efectos objetivos de esta situación eran evidentes: altos índices de pérdidas e interrupciones, creciente índice de transformadores dañados, escaso nivel de informatización y automatización, Entre los efectos subjetivos se encontraron: bajo nivel técnico, inestabilidad del personal que atiende la distribución, falta de estructuras uniformes, entre otros. Uno de los efectos más importantes ha sido la carencia de información adecuada sobre los diferentes elementos y acciones de la distribución, la ausencia de un Sistema Informático Integral que permita el empleo de técnicas modernas en el control y automatización de las redes. Esto se refleja en:

- Empleo de una amplia variedad de modelos y procedimientos por parte de cada una de las Organizaciones Básicas Eléctricas (OBE).
- Aplicación de una gran diversidad de codificaciones en instalaciones y equipos.
- Manejo de normas desactualizadas y poca divulgación entre los técnicos de la base.
- Falta de una correcta separación entre la información operativa y estratégica.
- Insuficiente capacitación de técnicos y cuadros, especialmente los de nueva promoción.
- Mala calidad de la información usada en la toma de decisiones.
- Uso de varias aplicaciones por diferentes entidades con un alto grado de dificultad para su generalización.

- Duplicación de esfuerzos de varias entidades en tareas con objetivos similares.
- El software no cumple con las normas de diseño y documentación actuales que permitan su actualización en el marco de un Sistema Integral.
- Dificultades en el desarrollo de las auditorias y los mecanismos de control.
- Despilfarro de recursos humanos en tareas repetitivas que pueden ser automatizadas dentro de un Sistema Integrado.
- Muchos técnicos de redes no tienen acceso a la informática.
- Altos costos en la transmisión de la información, con un deficiente uso del correo electrónico.
- La información de la facturación no se aprovecha en el cálculo de los indicadores técnicos de las redes.
- Ausencia de un interfaz única que facilite la introducción paulatina de técnicas modernas tales como los Sistemas de Información y Posicionamiento Geográfico, SCADA (Supervisory Control And Data Acquisition), telecomandos, entre otros.

1.2. VISIÓN DEL SIGERE

SIGERE forma parte del Sistema de Gestión Empresarial de la Unión Nacional Eléctrica (SIGE) y está integrado por todos los equipos, instalaciones, infraestructura y acciones que forman la red de Transmisión y Distribución de electricidad en Cuba. El sistema recoge datos técnicos, económicos y de gestión que facilita la dirección, operación, explotación y planificación de las redes; el mismo está orientado al cliente permitiendo la reducción de costos operativos y mejora la calidad del suministro que se le brinda a los mismos.

1.3. ALCANCE Y COMPOSICIÓN DEL SIGERE

SIGERE incluye las instalaciones que van desde la salida de las subestaciones de las centrales generadoras hasta las instalaciones de medición del cliente, donde se vincula con los Sistemas Informáticos de Generación y Comercial respectivamente. Este sistema consta de dos modelos de redes que componen el Sistema Electroenergético Nacional (SEN), uno de distribución, que es esencialmente radial y que comprende las instalaciones concebidas dentro de la red de Distribución de Energía; y el otro que comprende todas las instalaciones y accesorios de la red de Transmisión, el cual es básicamente en forma de malla donde pueden existir lazos en la interconexión de los elementos que conforman la red.

1.4. ARQUITECTURA DEL SIGERE

La visión arquitectónica del SIGERE considera la división en subsistemas semejante a los puestos de trabajo que se encuentran en las estructuras reales. La interfaz común a estos es alfanumérica y gráfica, y esta última contiene representaciones esquemáticas monolineales y los datos necesarios para el soporte de un Sistema de Información Geográfico (conocido como GIS por sus siglas en inglés) en los Subsistemas que lo necesiten; así como planos de detalles, croquis y fotos de los elementos e instalaciones que lo ameriten. En la figura 1.1 se muestra un organigrama con la estructura de alto nivel del sistema, y después se describe brevemente el alcance de cada uno de los contenidos del mismo, especificando el alcance de cada módulo con las letras D para Distribución, T para Transmisión y R para Redes.

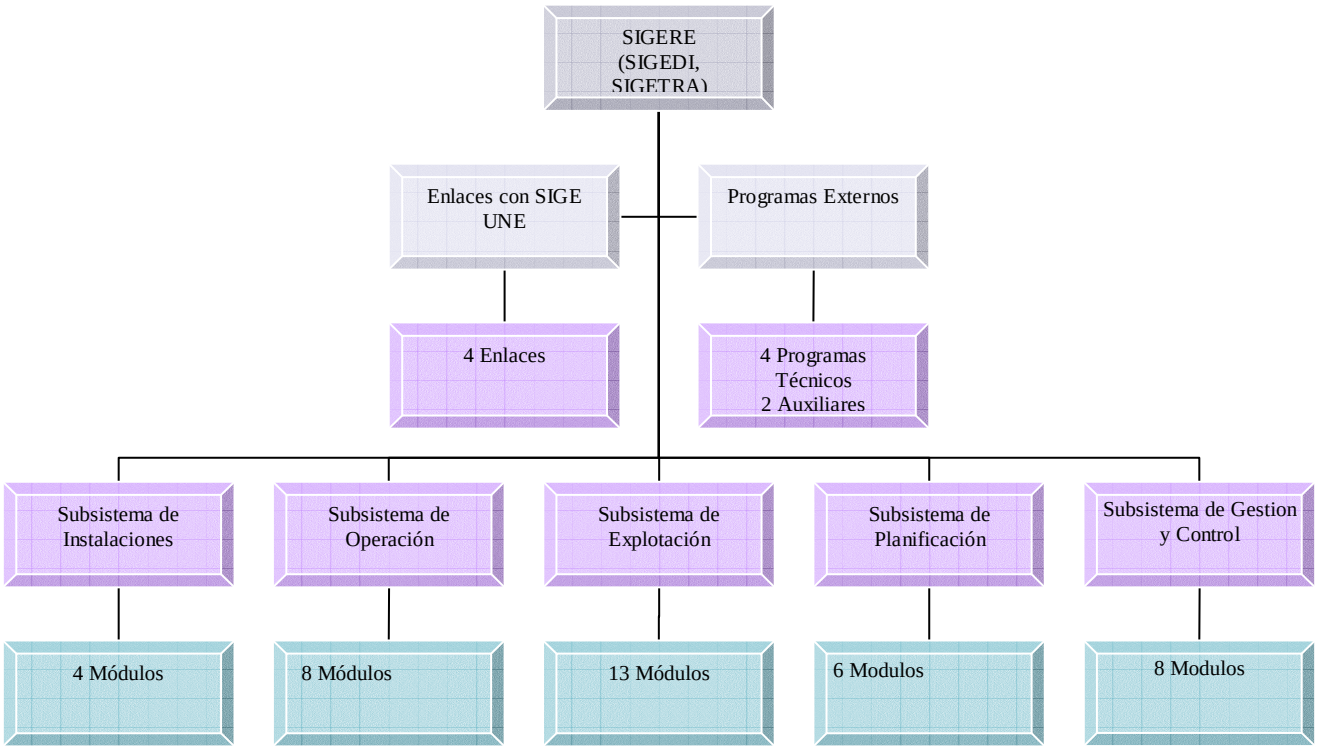


Figura 1.1. Estructura del SIGERE.

1.4.1. Enlaces con SIGE-UNE

En los enlaces con otros sistemas de la UNE se incluyen los módulos que sirven de punto de encuentro entre la información contenida en SIGERE y el resto de los sistemas desarrollados en el marco del SIGE, los cuales se nombran a continuación:

- Sistema de Gestión Comercial (SIGECO)
- Sistema de Contabilidad (SISCONT)

- Sistema de Recursos Humanos (SIGRH)
- Sistema de Mantenimiento y Explotación de Centrales.

1.4.2. Subsistema de Instalaciones

El núcleo del sistema lo constituye el Subsistema de Instalaciones. Una instalación es un conjunto de posiciones con una topología implícita a cada instalación que pueden estar ocupadas por elementos eléctricamente importantes. Estos elementos pueden ser equipos eléctricos simples o instalaciones más sencillas; y además pueden o no tener apoyos que las fijen en el espacio físico a través de un sistema de coordenadas que las referencien en el marco geográfico. Todas las instalaciones del Sistema estarán codificadas con códigos alfanuméricos. Este código es permanente e independiente de su posición en la red. Por el momento se evitan los cambios de codificación de los desconectivos previamente codificados y señalizados con connotación en la operación de la red. Todos los equipos importantes del sistema serán identificados de acuerdo a sus especificaciones (fabricante, país, modelo, número de serie), o alternativamente por un número dado por la empresa. Se recogen todos los datos nominales del equipo de acuerdo a las normas internacionales existentes.

El Subsistema de Instalaciones define los nomencladores y estructuras en cada nivel, así como la ubicación de las instalaciones y su posición eléctrica y geográfica. Se compone de los siguientes módulos:

- Nomencladores básicos.
- Ubicación de instalaciones.
- Esquemas monolineales.
- Sistema de Información Geográfica para la OBE (SIG-OBE)

1.4.3. Subsistema de Operación

El Subsistema de Operación incluye todo el tratamiento de las actividades propias del Despacho Provincial y Despachos de Distribución, Centros de Quejas y Operadores de Subestaciones en la operación de las redes de distribución. Se incluyó en el alcance de este subsistema la actividad que se realiza en los Despachos Territoriales. Internacionalmente, este subsistema está altamente automatizado debido a la presencia de Sistemas SCADA, Sistemas de Aviso Telefónico Integrados. Se divide en los siguientes módulos.

- Gestión de incidencias.
- Atención al cliente.

- Control de defectos.
- Lecturas.
- Control de la red (Switching)
- Control de vías libres.
- Gestión de la calidad.
- Informes y reportes operativos.

1.4.4. Subsistema de Explotación

El Subsistema de Explotación es la interfaz entre el Subsistema de Instalaciones y los técnicos que lo explotan en las OBEs territoriales y los diferentes departamentos de la OBE Provincial. Éste proporciona, de acuerdo al puesto de trabajo, el acceso regulado a la actualización de las diferentes instalaciones a través de las acciones (levantamientos, nuevas instalaciones, mantenimientos, celajes, etc.) que se realizan según los procedimientos y registros establecidos por el Manual de Distribución. Este subsistema adquiere especial importancia al establecerse la Nueva Política de Mantenimiento que establece la necesidad de hacer el mantenimiento por diagnóstico. La explotación de los Grupos Generadores se hará en el Sistema de Gestión de la Generación aunque estén en la distribución.

- Subestaciones de Distribución
- Líneas
- Subestaciones de Transmisión
- Líneas de Transmisión
- Desconectivos
- Protecciones
- Capacitares
- Transformadores de Distribución
- Generadores en Distribución
- Alumbrado Público
- Circuitos
- Apoyos
- Servicios
- Mediciones Tecnológicas

1.4.5. Subsistema de Planificación

El Subsistema de Planificación comprende las funciones centralizadas relacionadas con el estudio, proyecto y control de inversiones que se realizan sobre las instalaciones en las OBEs provinciales y territoriales y las estructuras de la Empresa de Construcción Integral de la Empresa Eléctrica. En este subsistema se han concebido los siguientes módulos que actualmente están en fase de diseño.

- Estudios
- Estudios de Desarrollo
- Proyectos
- Proyectos de la Transmisión
- Control de Inversiones
- Control de Inversiones Mayores

1.4.6. Subsistema de Gestión y Control

Este subsistema agrupa todos los módulos que se relacionan con el intercambio y diseminación de la información, el control de los planes, acciones e indicadores básicos que permiten a los técnicos y gerentes tomar las mejores decisiones sobre la explotación, operación y planificación de las redes.

- Intercambios
- Información Gerencial
- Control de Pérdidas
- Programación y Control
- Auditorias
- Órdenes de Trabajo
- Accesos y Seguridad
- Información del SIGERE

1.5. SITUACIÓN ACTUAL DEL CONTROL DE LOS TRANSFORMADORES

El transformador es el equipo que más abunda en nuestras redes eléctricas y su distribución espacial es la más variada, es el más cercano al cliente y el que más tiende a fallar; de ahí la importancia de minimizar sus averías. Esta es la razón por la que se necesita hacer un sistema

capaz de automatizar toda la información relacionada con ellos, poder seguir sus ciclos de mantenimiento, su estado de carga, así como ser capaz de prevenir las fallas.

En la actualidad, el control de los transformadores, así como todas las tareas y operaciones realizadas a estos equipos se recogen en tarjetas, haciendo muy engorroso el control y la actualización de los datos de los mismos; así como el movimiento de esta información junto al transformador cuando se requiera. Hoy en día, inmersos en el proceso de rehabilitación de las redes del país, se ha dispuesto la tarea de modernizar todas las líneas de distribución, así como lograr que el servicio de energía eléctrica tenga una calidad y fiabilidad comparado con los estándares mundiales. Para ello se ha destinado gran cantidad de recursos, y unido a esto se decidió también priorizar la automatización de la información que aparejado a todas estas mejoras, permita explotar con un máximo de aprovechamiento de los recursos y minimizar los costos.

1.6 Módulo de Transformadores de Distribución e Intercambios

El transformador tiene un ciclo de vida muy difícil de seguir ya que pasa por diferentes estados en diferentes ubicaciones. Debido a esto, es necesario desarrollar un módulo para su explotación junto un esquema de replicación (Módulo de Intercambios) que permita que la información relacionada con este equipo esté disponible en cualquier momento en el lugar que se necesita y con el nivel de actualización necesario manteniendo la sincronización de los datos entre los servidores.

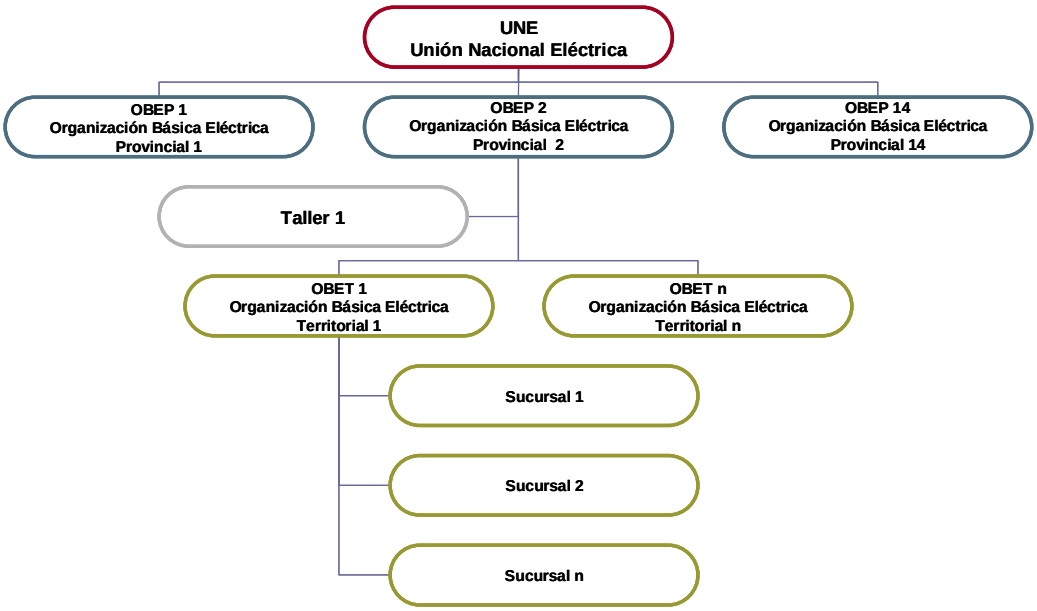


Figura 1.2. Organigrama de la Empresa Eléctrica en Cuba.

En la confección del Módulo de Transformadores se debe tener en cuenta la estructura inherentemente distribuida de las Empresas Eléctricas y sus propias características de comunicaciones. Como se observa en la figura 1.2, existen cuatro niveles bien definidos: nacional, provincial, territorial y sucursal, además de un quinto nivel que es el del taller.

Ciclo de Vida del Transformador

El control de los transformadores se hace con el apoyo de una BD provincial, por lo que sólo intervienen la provincia, el territorio y el taller como nodos del sistema; no participando en este módulo el nodo Unión Eléctrica. La BD de SIGERE se encuentra ya en todas las provincias y la topología de las redes informáticas no es la más adecuada para contar con una BD provincial donde accedan todos los usuarios ya sean de la provincia, taller o territorio.

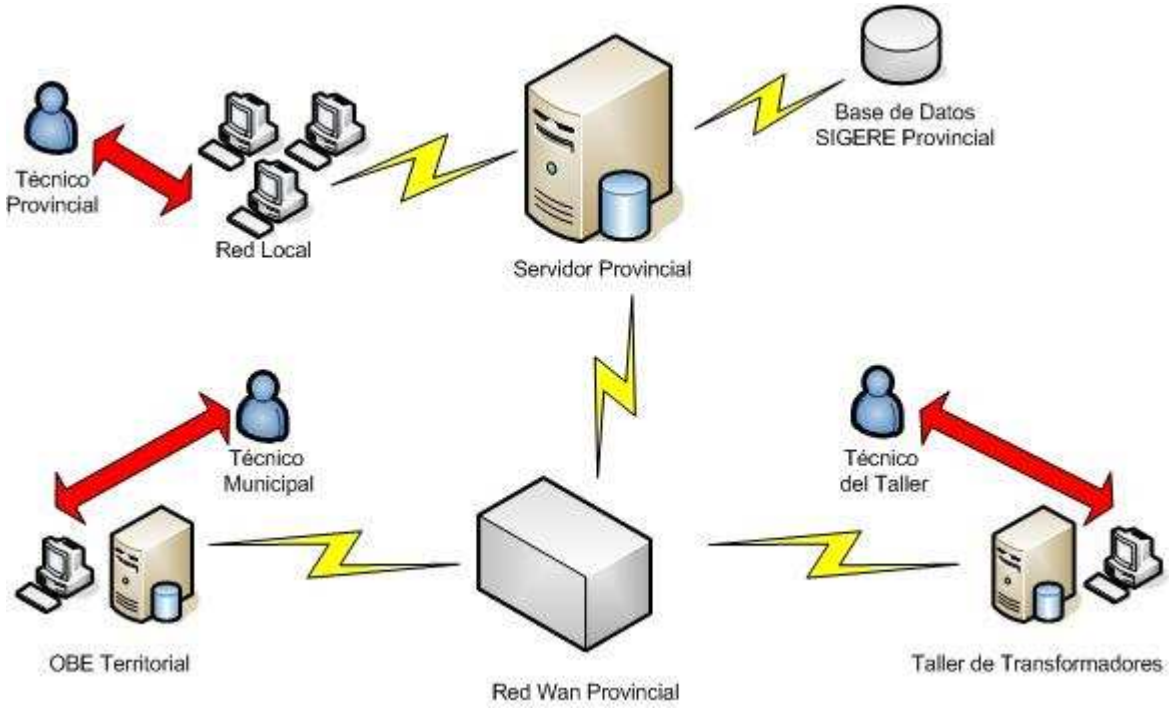


Figura 1.3. Topología actual de las redes de comunicaciones en las empresas eléctricas

La comunicación entre la OBE Provincial y las OBEs territoriales se hace a través de líneas telefónicas de baja velocidad; aunque en los últimos tiempos se ha introducido gradualmente las redes inalámbricas, aún no se cuenta con este sistema en todas las provincias, ni es política de la Unión Eléctrica hacerlo extensivo en poco tiempo, debido a esto es necesario diseñar una BDD y crear un método de replicación de datos entre los servidores de la provincia con los territorios y la provincia con el o los talleres, para poder seguir el ciclo de vida completo del Transformador,

contar con la información actualizada de cada equipo en la provincia que permita realizar estudios estadísticos y cálculos técnicos necesarios en el quehacer diario de las empresas envueltas en un proceso de automatización que hace que cada día se necesite la información real del estado y características de los equipos explotados por los diferentes municipios con una mayor exactitud. En la figura 1.3 se muestra una topología típica de una red de comunicaciones de la Empresa Eléctrica en una provincia.

El ciclo de vida del transformador comienza en el taller mediante la verificación de un transformador nuevo. Después, este equipo pasa a formar parte de los transformadores disponibles en la OBE provincial, donde aparecerán todos los datos de su ficha. Allí, el técnico lo tiene como disponible y dependiendo de sus características y las necesidades de las diferentes OBEs territoriales, se le asigna a una de estas, para donde deberá ir toda la información del transformador junto a él; luego, el técnico del municipio lo mueve hacia el banco que lo necesita, comenzando, de esta manera, la historia del equipo. En el municipio sólo se encuentran los datos pertenecientes a las instalaciones y equipos que atiende este territorio, mientras que en la provincia está centralizada toda la información de las OBEs territoriales.

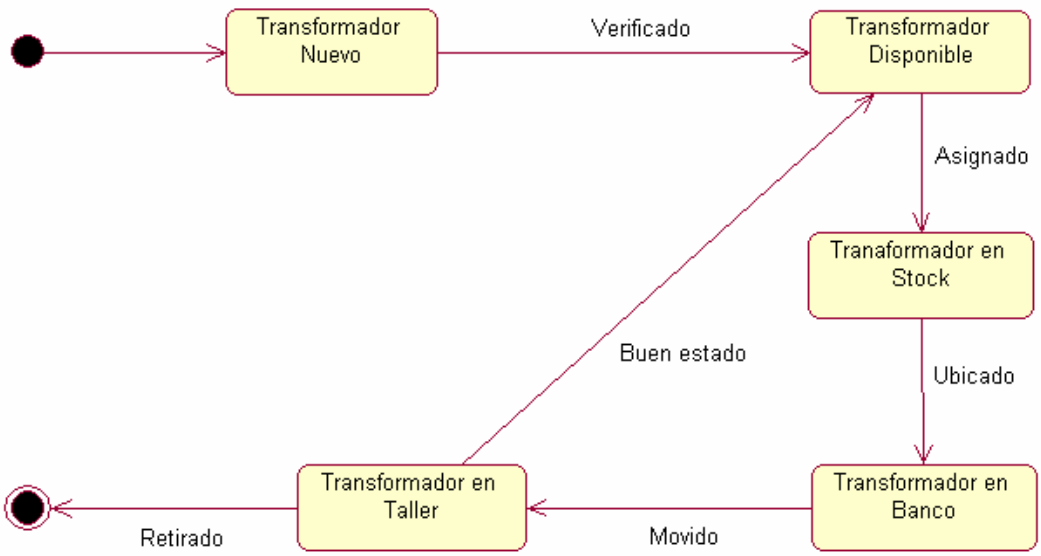


Figura 1.4. Diagrama de estados de un transformador.

En la OBE territorial se realizan todas las operaciones sobre el transformador y van conformando una historia. Existen dos formas de que el transformador vuelva a subir hacia la provincia: moviéndolo directo al taller o mediante un *Reporte de inspección al transformador dañado* que

de ser positivo el daño, lo retira automáticamente del Banco y lo envía al taller, a la vez que se crea una *Necesidad de transformador* para sustituir este averiado.

Luego de llegar los datos y el transformador físico al taller, se realiza la *Defectación o Diagnóstico en el Taller*. Esta operación es la que dice si el transformador se retira definitivamente por su daño o si sólo era un pequeño problema solucionable. En este caso, el transformador no pierde su historia, se da como disponible, pasa a la provincia y vuelve a fluir dependiendo de sus características técnicas y las necesidades de las OBEs territoriales. Nótese que en todo este ir y venir no pierde los datos de todas las pruebas que se le hayan realizado aunque vaya a parar a una OBE territorial diferente. En la figura 1.4 se muestra el diagrama de estados de un transformador para una mayor comprensión de ella.

1.7. CONCLUSIONES PARCIALES

Se puede concluir que SIGERE es un sistema de gran envergadura que requiere de la información necesaria para la operatividad más eficiente de las redes de distribución. Para esto es necesario del Módulo de Transformadores de Distribución y del Modulo de Intercambios, que precisan del diseño de una BDD debido a la información que se genera, su tratamiento y distribución, y a las características propias de comunicación existentes en la Empresa Eléctrica que hace imposible el uso de una BD Centralizada. Existe la necesidad de utilizar un mecanismo de replicación de datos entre los servidores de las OBEs territoriales y taller con el servidor provincial para poder mantener actualizada la información. El Módulo de Transformadores ayudará a optimizar el uso de recursos, pues contando con la información actualizada permitirá tomar decisiones a tiempo, evitando posibles roturas o averías, además de programar el ciclo de mantenimientos; lo que alarga la vida útil de estos equipos.

CAPÍTULO 2. ACERCA DE LAS BASES DE DATOS DISTRIBUIDAS

El uso y generalización de los Sistemas de Bases de Datos Distribuidas (SBDD) está condicionado por dos motivos fundamentales: el continuo descenso del costo del hardware y el incremento del costo del software de aplicación a gran escala. Su objetivo fundamental es integrar la manipulación de datos para presentarlos como una única colección global y coherente. En muchos casos en la práctica se presentan organizaciones geográficamente distribuidas para las cuales las vías centralizadas no representan una alternativa factible, y la migración hacia SBDD resulta natural, ya que esta alude de manera natural la estructura descentralizada de la organización. En este capítulo se hace referencia a algunas cuestiones teóricas sobre el diseño de BDD.

2.1. INTRODUCCIÓN

Desde mediados de la década del 80 hasta la actualidad se ha producido un auge notable de los sistemas distribuidos debido, entre otros factores, a los avances en la tecnología de comunicaciones apareciendo redes locales de alta velocidad, y al desarrollo de poderosos microprocesadores incorporados a computadoras cuyos costos han ido decreciendo paulatinamente. A consecuencia de esto, también ha cambiado la manera de realizar la manipulación de datos, ahora en ambientes distribuidos. Los sistemas distribuidos se adaptan mejor a las necesidades de las organizaciones descentralizadas ya que reflejan más adecuadamente su estructura y tienen importantes ventajas con respecto a los centralizados [37], pero al mismo tiempo añaden nuevas complejidades en su diseño e implementación.

Las BDD pertenecen a este tipo de sistemas y pueden ser consideradas como una colección de datos que lógicamente pertenecen a una BD única y físicamente residen en los diferentes sitios de una red de computadoras. La distribución involucra el hecho de que los datos no residen necesariamente en el mismo sitio pero poseen propiedades comunes que los vinculan, y se facilita su acceso a través de una interfaz común. Es necesario destacar que el enlace ente los datos se lleva a cabo en una red de comunicación, que por generalización determina que los sitios estén localizados en diferentes áreas geográficas y con capacidad de procesamiento autónomo. El diseño de un SBDD implica la toma de decisiones sobre la ubicación de los programas que accederán a la BD y sobre los propios datos que constituyen esta última, a través de los diferentes sitios de una red de computadoras.

El diseño de un SBDD incluye la ubicación de los datos y las aplicaciones en los distintos sitios de una red y un diseño factible de la red. El caso de un Sistema de Gestión de Bases de Datos Distribuidas (SGBDD), la distribución incluye tres elementos: la distribución del SGBDD propiamente dicho, la distribución de los programas de aplicación que se ejecutan sobre éste, y la distribución de los datos. El primer aspecto no constituye un problema en este trabajo, pues se considera que se tiene instalado el SGBDD en cada servidor en los que se encuentran almacenados los datos, el segundo aspecto representa de manera implícita que el sistema distribuido responde a la estructura de la empresa de forma que los programas de aplicación se ubican en los lugares donde son necesarios. Como puede observarse, es un enfoque común asumir que la red ha sido diseñada y por tanto sólo se centra la atención en la distribución de los datos.

2.2. ESTRATEGIAS PARA EL DISEÑO

A la hora de abordar el diseño de una BDD se puede optar principalmente por dos tipos de estrategias: la ascendente y la descendente. La estrategia ascendente podría aplicarse en aquel caso donde haya que proceder a un diseño a partir de BD existentes, con el fin de integrarlas en una sola. En este caso se partiría de los esquemas conceptuales locales (ECL) y se trabajaría para llegar a conseguir el esquema conceptual global (ECG). La estrategia descendente (véase la figura 2.1) debería resultar familiar a la persona que posea conocimientos sobre el diseño de BD centralizadas, exceptuando la fase del diseño de la distribución. Este enfoque es más apropiado para aplicaciones nuevas y para sistemas homogéneos. Ambos tipos de estrategia no son excluyentes, y no resultaría extraño a la hora de abordar un trabajo real de diseño de una BD que se pudiesen emplear en diferentes etapas del proyecto una u otra estrategia. Aunque la estrategia ascendente se pueda presentar con facilidad en la vida real, se prefiere pensar en el caso donde se requiera realizar un diseño inicial, y se avanza en el desarrollo del trabajo siguiendo la estrategia descendente.

En este trabajo es de particular interés la estrategia descendente, donde se comienza con un análisis de los requisitos que definirán el entorno del sistema en aras a obtener tanto los datos como las necesidades de procesamiento de todos los posibles usuarios de la BD. Igualmente, se deberán fijar los requisitos del sistema, los objetivos que debe cumplir respecto a unos grados de rendimiento, seguridad, disponibilidad y flexibilidad, sin olvidar el importante aspecto

económico. Como puede observarse, los resultados de este último paso sirven de entrada para dos actividades que se realizan de forma paralela.

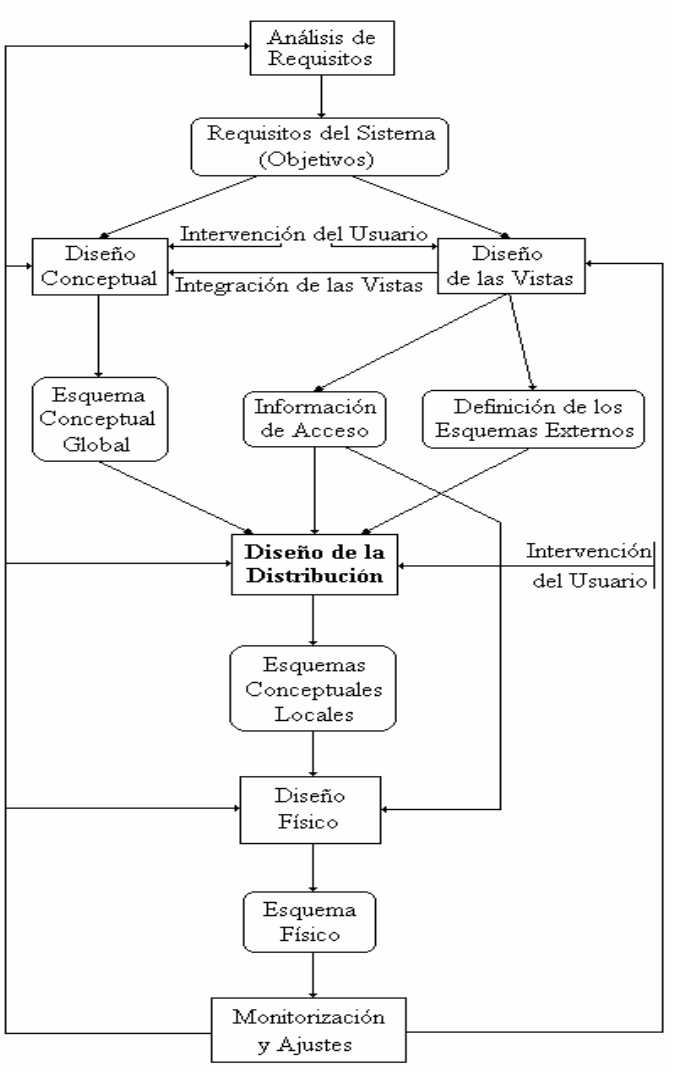


Figura 2.1 Estrategia de diseño descendente. Adaptado de [37].

El diseño de las vistas trata de definir las interfaces para el usuario final y, por otro lado, el diseño conceptual se encarga de examinar el entorno de la aplicación para determinar los tipos de entidades y establecer la relación entre ellas. Existe un vínculo entre el diseño de las vistas y el diseño conceptual; el diseño conceptual puede interpretarse como la integración de las vistas del usuario, lo cual es de vital importancia ya que el modelo conceptual debería soportar no sólo las aplicaciones existentes, sino que debería estar preparado para futuras aplicaciones. En el diseño conceptual y de las vistas del usuario se especificarán las entidades de datos y se determinarán las aplicaciones que funcionarán sobre la BD; así mismo, se recopilarán datos estadísticos o

estimaciones sobre la actividad de estas aplicaciones. Dichas estimaciones deberían girar en torno a la frecuencia de acceso de una aplicación a los distintos esquemas y atributos de esquemas que necesita.

Desarrollado el trabajo hasta aquí, se puede abordar la confección del ECG. Este esquema y la información relativa al acceso a los datos sirven de entrada al paso distintivo: el diseño de la distribución. El objetivo de esta etapa consiste en diseñar los ECL que se distribuirán en los diferentes sitios del sistema distribuido. Sería posible tratar cada tipo de entidad como una unidad de distribución; en el caso del modelo relacional, cada entidad se corresponde con un esquema de relación. Resulta bastante frecuente dividir cada esquema en subesquemas menores denominadas fragmentos que luego se ubican en uno u otro sitio. De ahí, que el proceso del diseño de la distribución conste de dos actividades fundamentales: la fragmentación y la asignación [6, 16, 18, 29, 37, 52, 55]. El último paso del diseño de la distribución es el diseño físico, el cual proyecta los esquemas conceptuales locales sobre los dispositivos de almacenamiento físico disponibles en los distintos sitios. Las entradas para este paso son los esquemas conceptuales locales y la información de acceso a los fragmentos. Por último, la actividad de desarrollo y diseño es un tipo de proceso que necesita de una monitorización y un ajuste periódicos.

2.3. EL DISEÑO DE LA DISTRIBUCIÓN

El Proceso del diseño de la distribución consta de dos actividades fundamentales:

La fragmentación, que determina la forma en que los esquemas globales se subdividen en fragmentos horizontales, verticales o mixtos.

La asignación, que determina la ubicación de los fragmentos en los sitios de procesamiento.

En el diseño de la distribución de los datos, se deben de tomar en cuenta los siguientes objetivos:

Procesamiento local: La distribución de los datos, para maximizar el procesamiento local, corresponde al principio simple de colocar los datos tan cerca como sea posible de las aplicaciones que los utilizan. Se puede realizar el diseño de la distribución de los datos para maximizar el procesamiento local agregando el número de referencias locales y remotas que le corresponden a cada fragmentación candidata y la localización del fragmento, que de esta forma se seleccione la mejor solución de ellas.

Distribución de la carga de trabajo: La distribución de la carga de trabajo sobre los sitios es una característica importante de los sistemas de cómputo distribuidos. Esta se realiza para tomar ventaja de las diferentes características potenciales de cada sitio, y maximizar el grado de

paralelismo de ejecución de las aplicaciones. Sin embargo, la distribución de la carga de trabajo podría afectar negativamente el procesamiento local deseado.

Costo de almacenamiento y disponibilidad: La distribución de la base de datos refleja el costo y disponibilidad del almacenamiento en diferentes sitios; para esto es posible tener sitios especializados en la red para el almacenamiento de datos. Sin embargo, el costo de almacenamiento de datos no es tan relevante si se compara con el de CPU, entrada-salida y costos de transmisión de las aplicaciones.

El diseño de la distribución tiene como principio clave alcanzar máxima localidad de datos y aplicaciones, ubicando los datos tan cerca como sea posible de las aplicaciones que los utilizan lo que permite reducir el tráfico de comunicaciones en la red. En un sistema bien diseñado el 90 % de los datos deben ser ubicados localmente y solo el 10 % debe ser accedido remotamente [6]. La fragmentación refleja el criterio de mantener los datos locales en el sitio donde frecuentemente son accedidos por las aplicaciones, es por ello que constituyen una unidad apropiada de asignación.

En el diseño de la distribución hay que tener en cuenta una serie de factores que contribuyen a obtener un diseño óptimo. La organización lógica de la BD, la localización de las aplicaciones, las características de acceso de las aplicaciones a la BD y las características del sistema en cada sitio, tienen una decisiva influencia sobre la distribución.

La información necesaria para el diseño de la distribución puede dividirse en cuatro categorías:

Información sobre la BD.

Información sobre las aplicaciones.

Información sobre la red de computadoras.

Información sobre los sitios de procesamiento.

Las dos últimas son de carácter cuantitativo y servirán, principalmente, para desarrollar el proceso de asignación.

2.4. EL PROBLEMA DE LA FRAGMENTACIÓN

El diseño de la distribución tiene como principio clave alcanzar máxima localidad de datos y aplicaciones, ubicando los datos tan cerca como sea posible de las aplicaciones que los utiliza, lo que permite reducir el tráfico de comunicaciones en la red. En un sistema bien diseñado el 90 % de los datos deben ser ubicados localmente y solo el 10 % debe ser accedido remotamente [6]. La fragmentación refleja el criterio de mantener los datos locales en el sitio donde frecuentemente

son accedidos por las aplicaciones, es por ello que constituyen una unidad apropiada de asignación. El principal problema de la fragmentación radica en encontrar la unidad apropiada de distribución. Un esquema de relación no es una buena unidad por muchas razones. Primero, las vistas de la aplicación normalmente son subesquemas de relaciones. Además, la localidad de los accesos de las aplicaciones no está definida sobre esquemas completos, pero sí sobre subconjuntos estos. Por ello, sería normal considerar como unidad de distribución a estos subesquemas de relaciones.

Segundo, si las aplicaciones tienen vistas definidas sobre un determinado esquema (considerándolo ahora una unidad de distribución) que reside en varios sitios de la red, se puede optar por dos alternativas. Por un lado, el esquema no estará replicado y se almacena en un único sitio, o existe réplica en todos o algunos de los sitios en los cuales reside la aplicación. Las consecuencias de esta estrategia son la generación de un volumen de accesos remotos innecesario. Además, se pueden realizar réplicas no esenciales que causen problemas en la ejecución de las futuras actualizaciones y puede no ser deseable si el espacio de almacenamiento está limitado.

Tercero, la descomposición de un esquema de relación en fragmentos, tratados cada uno de ellos como una unidad de distribución, permite el proceso concurrente de las transacciones.

Se debe precisar también que la fragmentación trae consigo algunos inconvenientes si las aplicaciones presentan requisitos tales que prevengan la descomposición de esquemas en fragmentos mutuamente excluyentes. Estas aplicaciones cuyas vistas estén definidas sobre más de un fragmento pueden sufrir una degradación en el rendimiento. Por tanto, puede ser necesario recuperar los datos de dos fragmentos y llevar a cabo sobre ellos operaciones costosas de unión y acople. Además, como resultado de la fragmentación los atributos implicados en una dependencia se descomponen en diferentes fragmentos que pueden destinarse a sitios diferentes. En este caso, la tarea de verificar las dependencias puede resultar en una búsqueda de los datos implicados en un gran número de sitios.

2.4.1. Alternativas de fragmentación

Se pueden considerar dos alternativas fundamentales y básicas, la fragmentación vertical y la fragmentación horizontal. Si se toma una relación determinada, esta puede ser dividida en fragmentos menores, mediante un proceso de división horizontal o vertical. La fragmentación horizontal trabaja sobre las tuplas, dividiendo los esquemas de relación en subesquemas que

contendrán un subconjunto de las tuplas que pertenecen al original. La fragmentación vertical, en cambio, se basa en los atributos del esquema para efectuar la división. Existe otra alternativa de fragmentación llamada mixta o híbrida, que hace uso de los dos tipos de partición antes mencionados. La fragmentación mixta puede realizarse de tres formas diferentes (véase la figura 2.2):

1. Desarrollando primero la fragmentación vertical y luego aplicar la fragmentación horizontal sobre los fragmentos verticales (llamada partición VH).
2. Desarrollando primero una fragmentación horizontal y luego aplicar la fragmentación vertical sobre los fragmentos horizontales (llamada partición HV).
3. Aplicando sobre una relación, de forma simultánea y no secuencial la fragmentación horizontal y vertical, generando una rejilla y los fragmentos formaran las celdas de dicha rejilla.

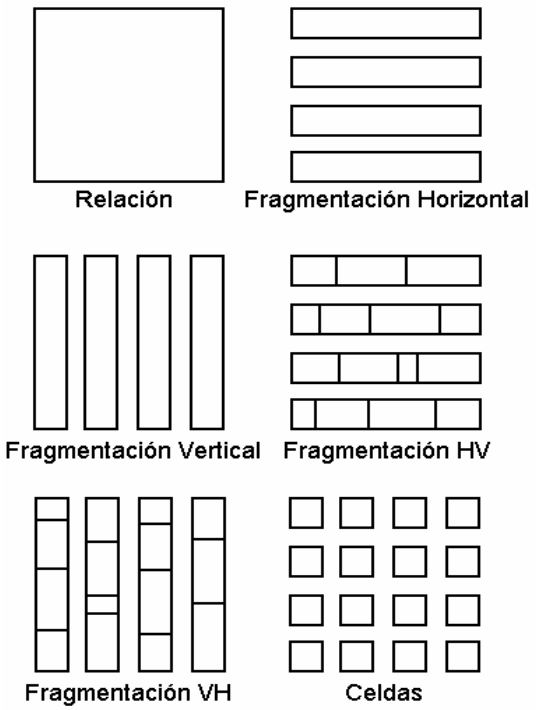


Figura 2.2. Distintos tipos de fragmentación. Adaptado de [55].

2.4.2. Grado de fragmentación

Cuando se va a fragmentar una base de datos, se debería sopesar qué grado de fragmentación va a alcanzar, ya que éste será un factor que influirá notablemente en la ejecución de las consultas. El grado de fragmentación puede variar desde una ausencia de la división, considerando las relaciones como unidades de fragmentación; o bien, fragmentar a un grado en el que cada tupla o

atributo forme un fragmento. Ante estos dos casos extremos, evidentemente se ha de buscar un compromiso intermedio, el cual debe establecerse sobre las características de las aplicaciones que hacen uso de la BD. Dichas características se podrán formalizar en una serie de parámetros. De acuerdo con sus valores, se podrá establecer el grado de fragmentación de la base de datos.

2.4.3. Reglas de la fragmentación

Existen tres reglas que debe cumplir el proceso de fragmentación, las cuales asegurarán la ausencia de cambios semánticos en la BD durante el proceso, logrando así una fragmentación correcta [37].

1. Completitud: Dado un esquema de relación R y su descomposición en una serie de fragmentos (R_1, R_2,…, R_n), se considera completa si y solamente si cada elemento de datos en R se encuentra en uno o varios fragmentos R_i. Esta propiedad asegura una descomposición sin pérdida. En el caso de la fragmentación horizontal el elemento de dato, normalmente, es una tupla, mientras que en el caso vertical es un atributo.

2. Reconstrucción: Si un esquema de relación R se descompone en una serie de fragmentos (R_1, R_2,…, R_n), la relación R debe poderse reconstruir a partir de sus fragmentos, es decir se puede definir un operador relacional ∇ tal que: $R = \nabla R_i$, $\forall R_i \in F_R$. El operador ∇ será diferente dependiendo de las diferentes formas de fragmentación. La reconstrucción a partir de los fragmentos asegura la preservación de las dependencias.

3. Disyunción: Si un esquema de relación R se descompone horizontalmente en una serie de fragmentos (R_1, R_2,..., R_n), y un elemento de datos d_i se encuentra en algún fragmento R_j, entonces no se encuentra en otro fragmento R_k ($k \neq j$). Esta regla asegura que los fragmentos horizontales sean disjuntos. Si R se descompone verticalmente, sus atributos primarios clave normalmente se repiten en todos sus fragmentos.

2.4.4. Diseño de la fragmentación

El propósito del diseño de la fragmentación es determinar los fragmentos que constituyen unidades lógicas de asignación y consiste en agrupar tuplas o atributos con las mismas propiedades. Los fragmentos deben ser colecciones homogéneas de información desde el punto de vista de accesos de las aplicaciones, o sea, que todas las instancias de los fragmentos sean accedidas uniformemente por las aplicaciones que las usan [6].

2.4.5. Información requerida

Antes de comenzar el proceso de fragmentación el diseñador debe poseer información de la BD e información de las aplicaciones [5, 6, 37]. La información de la BD es la concerniente al ECG, en el cual es necesario representar las relaciones, sus atributos y la conexión entre relaciones. La información de las aplicaciones debe representarse en términos cualitativos y cuantitativos. La cualitativa guía la actividad de fragmentación, mientras que la información cuantitativa se incorpora fundamentalmente a los modelos de localización. La principal información cualitativa describe los predicados que satisfacen las aplicaciones de los usuarios. Dichos predicados deben guiar la localidad de las aplicaciones, es por ello que se definen en base a predicados elementales o simples que los usuarios suministran para indicar las agrupaciones físicas de los datos que se presentan para las diferentes aplicaciones, y luego estos predicados simples se combinan adecuadamente para reflejar, por último, la localidad integrada del grupo de aplicaciones.

Los predicados simples p_j son expresiones formadas por las cláusulas:

p_j : < A_i > <operador> <Valor>

donde:

<A_i> es un atributo que pertenece a la relación R y debe estar definido sobre un dominio D_i, <operador> $\in$ {=,≠,<,≤,>,≥} y

<Valor> es seleccionado del dominio de A_i (Valor $\in$ D_i).

Frecuentemente las solicitudes de usuarios requieren predicados más complejos que pueden expresarse como combinaciones boolenas de predicados simples. Los predicados *minterm* son un tipo de combinación que puede definirse como la conjunción de todos los predicados simples tanto en su forma natural como negada [37].

Así, para un conjunto de predicados simples $Pr_i = \{p_{i1}, p_{i2},..., p_{im}\}$, el conjunto de predicados *minterm* $M_i = \{m_{i1}, m_{i2},..., m_{iz}\}$ se define como:

$M_i = \{ m_{ij} \mid m_{ij} = \Lambda\, p_{ik}^{*}\}$, $1 \le k \le m$, $1 \le j \le z$ donde $p_{ik}^{*} = p_{ik}$ o $p_{ik}^{*} = \neg p_{ik}$. $p_{ik} \in Pr_i$

Los predicados de selección o predicados simples, a partir de los cuales se hallan los predicados *minterm*, que determinan el proceso de fragmentación deben poseer dos propiedades fundamentales para garantizar una fragmentación correcta y eficiente: *completitud* y *minimalidad*.

Un conjunto de predicados Pr es completo si garantiza, para cada aplicación del conjunto de aplicaciones, que dos tuplas cualesquiera de un mismo fragmento sean accedidas con igual

probabilidad. Esta propiedad permite que todos los elementos de un fragmento sean referenciados de forma homogénea por cada aplicación. Por su parte la minimalidad es una propiedad mucho más intuitiva y generalmente se apela a la experiencia y sentido común del diseñador más que al empleo de técnicas formales. El conjunto de predicados Pr es minimal si todos los predicados son relevantes en la determinación de la fragmentación, o sea, si para cada predicado p_j, que causa que la relación R se fragmente en R_i y R_j, existe al menos una aplicación que accede de forma diferente a ambos fragmentos. Se puede afirmar que un conjunto minimal es siempre completo, sin embargo lo contrario no siempre es verdadero.

Con respecto a la información cuantitativa acerca de las aplicaciones es necesario conocer:

Selectividad del *minterm*. Número de tuplas de la relación que pueden ser accedidas por una solicitud especificada acorde a un predicado *minterm* dado.

Frecuencia de acceso. Frecuencia con la que las aplicaciones acceden a los datos, o sea, frecuencia de activación de la aplicación en cada sitio.

El tipo de acceso (lectura o escritura) realizado por la aplicación a cada objeto de datos.

2.4.6. Diseño de la fragmentación horizontal

La fragmentación horizontal de una relación R es la subdivisión de sus tuplas en subconjuntos disjuntos llamados fragmentos. Los datos contenidos en cada subconjunto deben poseer “propiedades geográficas” comunes y contener todos los atributos de la relación original. Las propiedades geográficas identifican las agrupaciones de datos que se tratan en cada sitio donde debe residir la BDD. Su principal objetivo es producir fragmentos con máxima localidad respecto a las aplicaciones. La posibilidad de no fragmentar horizontalmente un esquema debe ser considerada en dependencia de las aplicaciones que la accedan. Si el beneficio observado por fragmentar un esquema se ve limitado, el costo debido a la fragmentación puede no ser compensado por la ventaja del procesamiento local [6].

Existen varios criterios que permiten descartar la posibilidad de fragmentar una entidad:

No existan predicados que fragmenten ese esquema, esto puede estar dado porque no existan aplicaciones que lo referencien de forma diferenciada en los sitios que procesen la entidad.

Se generen fragmentos que no sean unidades apropiadas de localización.

No se generen predicados completos y minimales que representen una fragmentación correcta, esto se fundamenta en las características de las aplicaciones.

Se distingan fragmentos de una relación con rasgos muy similares.

Se han realizado numerosos estudios sobre el tema de la fragmentación horizontal, esto ha contribuido al surgimiento de varias metodologías y algoritmos:

En [37] se describe un algoritmo iterativo que realiza la fragmentación horizontal de una relación usando los minterms, formados a partir de predicados simples extraídos de las consultas del usuario. Son consideradas dos versiones de fragmentación horizontal: primaria y derivada. En [36] se describe un algoritmo no iterativo, basado en teoría de grafos. El mismo utiliza una matriz de afinidad de los predicados simples, y los fragmentos finales son definidos procesando la clusterización de los predicados simples. En [12] se describe un conjunto de algoritmos para fragmentar horizontalmente cuatro modelos de clases diferentes en un sistema distribuido basado en objetos. En [58] se describe un algoritmo de clausura transitiva paralela para fragmentar datos en bases de datos deductivas. En [44] se proponen dos estrategias para la fragmentación horizontal de relaciones recursivas, basado en el concepto de conjuntos de orden parcial.

Fragmentación horizontal primaria

La fragmentación horizontal primaria es muy común en la realidad y se aplica para lograr un incremento del paralelismo, posibilita decrementar el tiempo de respuesta de las solicitudes de usuarios y disminuir el costo de procesamiento de las solicitudes locales. Se realiza de acuerdo a un predicado que determina qué tuplas de la relación serán almacenadas en cada sitio y se define por una operación de selección. En el diseño del esquema de fragmentación se sugiere que la cantidad de fragmentos disjuntos sea mayor o igual que la cantidad de sitios en la red. Esta facilidad simplifica los modelos de optimización y es justificable por el hecho de que en las aplicaciones reales el usuario asocia los fragmentos candidatos a los sitios de localización de manera natural.

Para construir el esquema de fragmentación primaria pueden seguirse tres pasos, el primero de ellos consiste en encontrar los predicados simples relevantes a cada aplicación asegurando que los mismos sean completos y minimales. La búsqueda de los predicados debe realizarse de forma exhaustiva concentrándose solamente en las aplicaciones más importantes y sin distinguir los fragmentos cuyas propiedades sean similares.

Existen controversias entre diversos autores con respecto a la determinación de los predicados simples. Para unos [5], basta con incluir en el conjunto de predicados, los predicados p_j en su forma natural capaces de fragmentar una relación en dos porciones. Otros, por su parte [37], consideran en el conjunto de predicados, los predicados simples en su forma natural y también

los que están de forma negada (principalmente para predicados cuyo dominio es numérico). Esta última alternativa añade complejidad a los predicados *minterm* debido a que aumenta considerablemente la cantidad de predicados simples útiles para formar los predicados *minterm* y como resultado se obtendrían muchos *minterm* sin significado. Además, de esta forma los predicados simples pierden relevancia a la vista del diseñador, porque es como si se consideraran dos predicados para fragmentar. Las propiedades de completitud y minimalidad son satisfechas por ambas variantes, pero se debe prestar especial atención cuando se utilice la segunda variante porque el predicado p_i causa que la relación R se fragmente en R_1 y R_2, por tanto $\neg p_i$ debe aplicársele al fragmento resultante que no cumpla la condición de selección impuesta por p_i. El segundo paso consiste en construir el conjunto de predicados *minterm* acorde al conjunto de predicados simples. Los predicados *minterm* determinan los fragmentos candidatos en el proceso de localización.

El tercer y último paso del proceso de diseño es la eliminación de algunos predicados *minterm* que resultan insignificantes o contradictorios para un conjunto de implicaciones I [37], que en cierta medida constituyen restricciones de integridad del sistema.

Fragmentación horizontal derivada

La fragmentación horizontal derivada es el concepto de fragmentar un esquema aplicándole la misma partición que se le aplica a otro esquema y es derivada a través de la integridad referencial acorde a una operación de selección ya especificada en el esquema al que hace referencia la llave foránea. La decisión de seleccionar una relación candidata a fragmentación horizontal derivada se basa en dos criterios [37]: la usada en más aplicaciones o la de mejores características de acople.

2.4.7. Diseño de la fragmentación vertical

La fragmentación vertical no es más que la división de un esquema en fragmentos, cada uno de los cuales contiene un subconjunto de los atributos del original. Su principal objetivo es mejorar el rendimiento de las transacciones; para esto, los fragmentos deben ser diseñados de tal manera que las aplicaciones accedan al menor número de atributos irrelevantes posibles. Se entiende por atributos irrelevantes aquellos que no son necesarios para resolver la consulta.

En cualquier BD, las transacciones usualmente no requieren que todos los atributos usados sean extraídos durante su ejecución. El acceso a atributos almacenados que son irrelevantes incrementan los costos de almacenamiento y procesamiento, especialmente cuando el número de tuplas involucradas es muy grande. En una BDD, cuando los atributos relevantes están en

diferentes fragmentos de datos y localizados en diferentes sitios se genera un costo adicional debido al acceso de datos remotos. Por esto, una de las características deseables de una BDD que se debe favorecer con la fragmentación vertical es la accesibilidad local en cualquier sitio.

Si cada aplicación accediera a un subconjunto diferente y disjunto de atributos, el diseño de la fragmentación vertical sería obvio; sin embargo, en la práctica, lo más común es que las aplicaciones accedan a conjuntos de atributos diferentes y solapados. Un aspecto a decidir es si los fragmentos van a ser solapados o disjuntos. En el primer caso, los fragmentos se adaptan mejor a los requerimientos de las aplicaciones al aumentar las posibilidades de encontrar en un solo fragmento los datos requeridos; mejor aún si se asignan al mismo sitio donde se ejecutará la aplicación, algo que es particularmente beneficioso si la misma es de lectura, pero aumenta considerablemente la complejidad del SGBDD que tendrá que encargarse de mantener la consistencia de todos los datos. Esto puede reducir el rendimiento del sistema si no se realiza adecuadamente. En el segundo caso no hay redundancia, pero una determinada fragmentación puede afectar aquellas aplicaciones que al ejecutarse usen atributos de más de un fragmento, siendo necesaria la realización de un acople, operación que es bastante costosa, en particular cuando los fragmentos se encuentran en distintos sitios de la red.

Por tanto, no basta con maximizar el número de aplicaciones que acceden a un solo fragmento y minimizar las que acceden a más de uno, sino que es necesario evaluar cómo influye una determinada fragmentación en el rendimiento total del sistema. Para esto se debería tener en cuenta toda una serie de aspectos como son: la frecuencia de activación de las transacciones, los métodos de acceso empleados, las estrategias de procesamiento de transacciones, costos de transmisión y posibilidad de replicación, entre otros.

Una solución teórica simple al problema de la fragmentación vertical sería escoger un criterio o función objetivo, evaluarlo para todas las posibles particiones y seleccionar aquella que optimiza el criterio. Esta solución se tropieza con dos grandes dificultades. Primero, la obtención de una función objetivo que convierta todos los aspectos enumerados anteriormente en un modelo matemático razonable. Segundo, el número posible de particiones es muy grande incluso para un número moderado de atributos, de manera que incluso evaluar el criterio más simple para todas las particiones no resulta factible. Esto conduce a la idea, bastante aceptada, de que es inútil buscar soluciones óptimas al problema de la fragmentación vertical y es necesario recurrir a heurísticas que disminuyan el espacio de solución. El concepto de usar la fragmentación vertical con el objetivo de mejorar el rendimiento de los SGBD ha aparecido con frecuencia en la

literatura, a continuación se mencionan algunos de los trabajos que más sobresalen. En [17] definieron un algoritmo que agrupa los atributos de una relación basándose en la afinidad entre ellos. Los atributos accedidos juntos por las aplicaciones tienen mayor afinidad y el algoritmo BEA desarrollado en [30] es usado para agruparlos en clusters. En [34] extienden el trabajo de [17] y definen algoritmos para el agrupamiento de atributos en fragmentos solapados y no solapados con el objetivo de minimizar el número de fragmentos que usa una transacción y refinar los fragmentos usando factores de costo que reflejan el ambiente físico donde se almacena la base de datos. En [7] optimizaron este trabajo desarrollando un algoritmo que obtiene una partición binaria óptima para BD relacionales. Ellos usan información de factores físicos para disminuir el número de accesos a disco. Posteriores refinamientos son logrados aplicando el algoritmo de partición binaria iterativamente. En [35] desarrollaron un algoritmo que usa una técnica gráfica donde se construye una matriz de afinidad de atributos , que es representada por un grafo a partir del cual se genera un árbol linealmente conectado y todos los ciclos del árbol forman fragmentos de relaciones. En [37] discuten este trabajo previo sobre fragmentación vertical en BDD usando información de las frecuencias de acceso y aplican el algoritmo BEA que agrupa los atributos de una relación basándose en los valores de la afinidad entre ellos. Los grupos de atributos son clusterizados y se usan ecuaciones de costo para definir la mejor posición a lo largo de la diagonal de la matriz clusterizada para dividir la relación en fragmentos. En [33] se argumenta que los primeros algoritmos para la fragmentación vertical son *ad hoc*, por eso se propone una función objetivo llamada Evaluador de Particiones para determinar la “calidad” de las particiones generadas por varios algoritmos.

2.4.8. Diseño de la fragmentación mixta

La fragmentación mixta o también denominada fragmentación híbrida o anidada puede ser construida alternando, en cada relación, fragmentaciones horizontales y verticales recursivamente hasta obtener un fragmento apropiado. Esta puede ser convenientemente representada por un árbol de fragmentación, en el cual la raíz corresponda a la relación global, las ramas correspondan a los fragmentos y los nodos intermedios correspondan a resultados intermedios de las expresiones de fragmentación definidas. Los nodos hijos de un nodo determinado representan la descomposición de este nodo por una operación de fragmentación [6, 37]. El número de niveles de anidamiento puede ser grande pero siempre finito. En el caso de la fragmentación horizontal, el punto de parada es cuando cada fragmento esté compuesto por una tupla, mientras

que para la fragmentación vertical la condición para terminar es cuando los fragmentos estén constituidos por atributos. Sin embargo las aplicaciones con más de dos niveles de fragmentación no son de interés práctico debido a que los esquemas globales normalizadas generalmente son de pequeño grado y no se deben ejecutar demasiadas fragmentaciones verticales porque el costo de los acoples sería demasiado alto.

Existen varias propuestas de soluciones para la fragmentación mixta. En [36] se propone un método gráfico para la fragmentación horizontal, y para generar la fragmentación mixta sugieren combinar los algoritmos de fragmentación horizontal y vertical. En [22] también combina los algoritmos de los esquemas de fragmentación vertical y horizontal para lograr un esquema de fragmentación mixta optimizado.

Para diseñar fragmentación mixta es más conveniente comenzar con una fragmentación horizontal lo que se justifica considerando los beneficios que guían la naturaleza de la fragmentación horizontal y dentro de los cuales es necesario resaltar que la mayoría de las solicitudes y transacciones se originan y relacionan actividades locales, con lo que se garantiza menor tiempo de respuesta y menor costo de procesamiento para solicitudes locales; otra ventaja adicional es que el esquema de fragmentación es consistente con la estructura de la organización. No obstante, el diseñador de la BD tiene la posibilidad de decidir cuál de los dos tipos de fragmentación aplicar primero, en base al conocimiento que tenga de las aplicaciones.

2.5. EL PROBLEMA DE LA ASIGNACIÓN

Asumiendo que la BD es adecuadamente fragmentada, entonces se decide sobre la asignación o ubicación de los fragmentos en los diferentes sitios de la red. Cuando los datos son ubicados, éstos pueden estar replicados o mantenidos como una sola copia. Las razones para la replicación son la confiabilidad y la eficiencia de las solicitudes de sólo lectura. Si existen múltiples copias de un elemento de datos, habrá una mayor probabilidad de que alguna copia de los datos esté accesible en algún lugar, incluso si ocurriera una falla en el sistema. También, las solicitudes de sólo lectura que acceden a los mismos artículos de datos pueden ser ejecutadas en paralelo ya que las copias existen en múltiples sitios. Por otra parte, la ejecución de solicitudes de actualización causa problemas ya que el sistema tiene que asegurar que todas las copias de los datos sean propiamente actualizadas. De aquí que la decisión con relación a la replicación es una cuestión que depende de la proporción de la solicitudes de sólo lectura con respecto a las solicitudes de

actualización. Esta decisión afecta a casi todos los algoritmos del SGBDD y a las funciones de control.

	Completa	Parcial	Particionada
Procesamiento de Solicitudes	Fácil	Igual Dificultad	
Administración de Directorios	Fácil o No Existente	Igual Dificultad	
Control de Concurrencia	Moderada	Difícil	Fácil
Confiabilidad	Muy Alta	Alta	Baja

Tabla 2.1. Alternativas de replicación

Una BD no replicada (comúnmente llamada una BD particionada) contiene fragmentos que son ubicados en los sitios, y existe una copia única de cada fragmento en la red. En caso de replicación, o bien la BD existe completa en cada sitio (BD completamente replicada), o los fragmentos son distribuidos en los sitios de tal manera que la copias de un fragmento puedan estar en múltiples sitios (BD parcialmente replicada). En esta última el número de copias de un fragmento puede ser una entrada para el algoritmo de asignación o una variable de decisión cuyo valor lo determina el propio algoritmo. En la tabla 2.1 se puede ver una comparación entre estas tres alternativas de replicación con respecto a varias funciones del SBDD.

2.5.1. Modelo de asignación

La asignación de recursos a través de los nodos de una red de computadoras es un problema que ha sido estudiado extensivamente [1, 2, 4, 9-11, 14-16, 18, 19, 22-24, 26, 28, 29, 31, 38-43, 49, 51, 53-57]. Muchos de estos trabajos no solucionan totalmente el problema del diseño de BDD, en ocasiones resuelven sólo el de ubicar archivos individuales en una red de computadoras. El problema de asignación consiste en lo siguiente.

Existe un conjunto de fragmentos $F = \{F_1, F_2, ..., F_n\}$ y una red constituida por los sitios $S = \{S_1, S_2, ..., S_m\}$ en los cuales se ejecuta un conjunto de aplicaciones $Q = \{q_1, q_2, ..., q_q\}$. El problema de la asignación involucra el encontrar la distribución óptima de F sobre S. Un detalle muy importante que necesita ser discutido es la definición de optimalidad. La optimalidad puede ser definida con respecto a dos medidas:

1. Costo mínimo: La función de costo consiste en el costo de almacenar cada F_i en un sitio S_j, el costo de solicitar F_i en el sitio S_j, el costo de actualizar F_i en todos los sitios donde esté almacenado, y el costo de comunicación. El problema de asignación trata de encontrar un esquema de ubicación que minimice una función de costo combinada.

2. Desempeño: La estrategia de asignación está diseñada para mantener una métrica del desempeño. Dos de las más conocidas son: minimizar el tiempo de respuesta y maximizar el rendimiento del sistema en cada sitio.

La mayoría de los modelos que han sido propuestos hasta el momento hacen esta distinción de optimalidad. Si uno realmente analiza el problema con más profundidad, puede apreciar que la medida de optimalidad debe incluir tanto el rendimiento como el costo. En otras palabras, se debe buscar un esquema de asignación que, por ejemplo, responda a las solicitudes de los usuarios en el tiempo mínimo y al mismo tiempo mantenga el costo de procesamiento al mínimo. Una afirmación similar puede ser hecha para una maximización del rendimiento. Debido a la complejidad del problema, no se ha desarrollado modelos completos con estas características.

Considérese una formulación muy simple del problema. Sean F y S tal como fueron definidos anteriormente. Para facilitar el modelado del problema de asignación se hacen las siguientes suposiciones y definiciones.

Asúmase que Q puede ser modificado de forma tal que sea posible identificar las consultas de actualización de las de sólo lectura, y considérese un solo fragmento F_k:

$$T = \{t_1, t_2, ..., t_m\}$$

donde t_i es el tráfico de sólo lectura generado en el sitio S_i para F_k, y

$$U = \{u_1, u_2, ..., u_m\}$$

donde u_i es el tráfico de actualización generado en el sitio S_i para F_k.

Asúmase que el costo de comunicación entre dos pares de sitios cualesquiera S_i y S_j es fijo para una unidad de transmisión. Además, supóngase que es diferente para las consultas de actualizaciones y de sólo lectura, de forma tal que se pueda definir:

$$C(T) = \{c_{12}, c_{13}, ..., c_{1m}, ..., c_{m-1,m}\}$$

$$C'(T) = \{c'_{12}, c'_{13}, ..., c'_{1m}, ..., c'_{m-1,m}\}$$

donde c_{ij} y c_{ij}' es el costo de comunicación para consultas de actualización y de sólo lectura entre los sitios S_i y S_j respectivamente.

Sea d_i el costo de almacenar el fragmento en el sitio S_i. De esta forma, se puede definir $D = \{d_1, d_2, ..., d_m\}$ para el costo de almacenamiento del fragmento F_k en todos los sitios. Asúmase que no hay restricciones de capacidad ni para los sitios ni para los enlaces de comunicación.

Entonces el problema de asignación puede ser formulado como un problema de minimización de costo donde se desea encontrar el conjunto $I \subseteq S$ que especifique donde serán almacenadas las copias de los fragmentos. En lo adelante, x_j denotará la variable de decisión para la ubicación. $x_j = 1$ si F_k está asignado a S_j; en otro caso es igual a 0. La formulación más precisa es:

$$\min \left[\sum_{i=1}^{m} \left(\sum_{j|S_j \in I} x_i u_j c_{ij}' + t_j \min_{j|S_j \in I} c_{ij} \right) + \sum_{j|S_j \in I} x_j d_j \right]$$

sujeto a: $x_j = 0,1$

El segundo término de la función objetivo calcula el costo total de almacenar todas las copias duplicadas del fragmento. El primer término, por otro lado, corresponde al costo de transmitir las actualizaciones a todos los sitios que mantienen las réplicas del fragmento, y al costo de ejecución de las solicitudes de sólo lectura en el sitio, lo cual debe ser al costo mínimo de transmisión de datos.

Esto es una formulación muy simplista que no es aplicable al diseño de BDD como se explica más adelante, y aunque lo fuera, existe otro problema: su complejidad. Existe un gran número de razones por las cuales formulaciones simplistas como la anterior no son adecuadas para el diseño de BDD:

1. No se puede tratar los fragmentos como ficheros individuales que pueden ser ubicados de forma aislada. La ubicación de un fragmento tiene un impacto en las decisiones sobre la ubicación de los demás que son accedidos junto con él por lo que debe tomarse en cuenta la relación entre los mismos.

2. No se modela la relación entre la ubicación y el procesamiento de solicitudes de forma adecuada. El acceso a los datos por parte de las aplicaciones es modelado de forma muy sencilla. Una solicitud de usuario es emitida desde un sitio, y todos los datos para responder a la misma

serán transferidos a ese sitio. En los SBDD, el acceso a datos es mucho más complicado que el simple acceso a ficheros remotos que la mayoría de los modelos sugieren.

3. No se toma en cuenta el costo del chequeo de las restricciones de integridad; la ubicación de dos fragmentos involucrados en la misma restricción de integridad en dos sitios diferentes puede ser muy costoso.

4. No se considera el costo de los mecanismos de control de concurrencia.

5. No existen modelos heurísticos generales que tengan como punto de partida el conjunto de fragmentos y den como solución una ubicación cercana al óptimo sujeto a los tipos de restricciones discutidos anteriormente.

Los modelos desarrollados hasta el momento asumen un gran número de simplificaciones y son aplicables a ciertas formulaciones muy específicas. Por lo que Özsu y Valduriez [37] presentan un modelo relativamente general como el mostrado más adelante en la sección 2.5.3.

2.5.2. Información Requerida

Información sobre la BD:

- $sel_i(F_j)$: Se define como selectividad de un fragmento F_j con respecto a una solicitud q_i, como el número de tuplas de F_j que tienen que ser accedidas para poder procesar q_i.

- $size(F_j) = card(F_j) * length(F_j)$. El tamaño de un fragmento F_j dado por la cantidad de tuplas del fragmento y su longitud en bytes.

Información sobre las aplicaciones:

La mayoría de la información relacionada con las aplicaciones ya ha sido recopilada durante el proceso de fragmentación, pero para el modelo de asignación se requiere alguna información adicional. Las dos más importantes son:

- RR_{ij}: Número de accesos de lectura que una consulta q_i realiza sobre un fragmento F_j, durante su ejecución.

- UR_{ij}: De forma similar representa el número de accesos de actualización.

También se necesita definir u_{ij} y r_{ij}, como sigue: u_{ij} toma valor 1 si q_i actualiza el fragmento F_j; 0 en otro caso. r_{ij} = 1 si q_i lee el fragmento F_j; 0 en otro caso.

- $o(i)$: Sitio donde se origina la consulta q_i.

- MaxTime(i): Tiempo de respuesta máximo admisible de cada aplicación.

Información sobre los sitios:

Para cada sitio, se necesita conocer su capacidad de almacenamiento y su velocidad de procesamiento. Para captar estos valores se pueden elaborar funciones o simplemente pueden ser estimados.

- USC_k : Costo unitario para almacenar un bloque de datos en el sitio S_k .

- SPC_k : Costo de procesar una unidad de trabajo en el sitio S_k . La unidad de trabajo pudiera ser igual a los valores de *RR* y *UR* .

Información sobre la red:

En este modelo se asume la existencia de una red simple, donde el costo de comunicación pueda ser definido en términos de un frame de datos.

- g_{ij} : Costo de comunicación por frame entre los sitios S_i y S_j .

- *fsize* : El tamaño (*en bytes*) de un frame.

Se pudiera realizar un modelo más elaborado sobre la red, que incluyera otros parámetros como la capacidad de los canales, y la distancia entre los sitios; pero con esto se llegaría a un modelo mucho más complejo.

2.5.3. Modelo de asignación presentado por Özsu y Valduriez

En un ambiente distribuido, los fragmentos de datos son ubicados en sitios diferentes, de manera que en el procesamiento de las consultas se combinan datos de diferentes sitios para obtener la respuesta final. Para hacer esto de una forma económica se requiere de una buena función de costo que justifique la fragmentación de los datos, contemple las frecuencias de uso de todas las consultas y actualizaciones, y los sitios a los que los resultados deben ser enviados. En la mayoría de los sistemas de la vida real, toda la información requerida para la solución del modelo es prácticamente imposible de recopilar ya que el sistema no tiene un comportamiento estático en el tiempo y la cantidad de datos que se requieren para el modelo podría ser abrumadora.

A continuación se plantea un modelo de asignación de fragmentos cuyo objetivo es minimizar el costo total de procesamiento y almacenamiento sujeto a algunas restricciones sobre el tiempo de respuesta [37]. El modelo tiene la siguiente forma:

min(Costo Total)

sujeto a:

restricciones de tiempo de respuesta

restricciones de almacenamiento

restricciones de procesamiento

A continuación se muestra las componentes del modelo.

x_{ij}: Variable de decisión definida como 1 si F_i es almacenado en S_j; 0 en otro caso

Costo Total:

La función de costo total tiene dos componentes: el procesamiento de solicitudes y el almacenamiento. Y lo se puede expresar de la siguiente forma:

$$TOC = \sum_{\forall q_i \in Q} QPC_i + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} STC_{jk}$$

donde QPC_i es el costo de procesamiento de las solicitudes de la aplicación q_i, y STC_{jk} es el costo de almacenamiento del fragmento F_j en el sitio S_k.

El costo de almacenamiento está dado por:

$$STC_{jk} = USC_k \times size(F_j) \times x_{jk}$$

y las dos sumatorias calculan el costo de almacenamiento total en todos los sitios, para todos los fragmentos.

El costo de procesamiento de las solicitudes es un poco más difícil de especificar. La mayoría de los modelos para resolver el clásico problema de asignación de archivos en sistemas distribuidos lo separan en dos componentes: el costo de procesamiento para sólo lectura, y el costo de procesamiento para la actualización. Pero en este caso que se trata de un modelo para el problema de ubicación de BD, especificado como el costo de procesamiento (PC) y el costo de transmisión (TC). Así, el costo de procesamiento de solicitudes (QPC) para la aplicación q_i sería:

$$QPC_i = PC_i + TC_i$$

Como ya se había referido, el costo de procesamiento PC contiene tres factores, el costo de acceso (AC), el costo del chequeo de restricciones de integridad (IE), y el costo del control de concurrencia (CC):

$$PC_i = AC_i + IE_i + CC_i$$

La especificación detallada de cada uno de esos factores de costo, dependen de los algoritmos usados para realizar estas tareas. Una especificación detallada de AC es la siguiente:

$$AC_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} \left(u_{ij} \times UR_{ij} + r_{ij} \times RR_{ij}\right) \times x_{jk} \times SPC_k$$

Los dos primeros términos de la fórmula anterior calculan el número de accesos de solicitudes de usuarios q_i al fragmento F_j. Nótese que $(UR_{ij} + RR_{ij})$ da el número total de accesos de sólo lectura y de actualización. Se asume que el costo total de procesar las solicitudes es el mismo. La sumatoria calcula el número total de accesos a todos los fragmentos referenciados por q_i. La multiplicación por SPC_k da como resultado el costo del acceso al sitio S_k. Y usaremos de nuevo x_{jk} para seleccionar sólo los valores de costo de los sitios donde los fragmentos han sido almacenados.

Hay una cuestión muy importante que destacar aquí. La función de costo de acceso asume que procesar una consulta incluye la descomposición de esta en un conjunto de subconsultas, cada una de las cuales se ejecuta sobre un fragmento almacenado en un sitio, y la transmisión del resultado hacia el sitio donde fue originada la solicitud. Esta es una visión muy reducida del problema la cual no tiene en consideración la complejidad del procesamiento de la BD. Por ejemplo la función de costo no tiene en cuenta el costo de realizar acoples (si fuese necesario), lo cual puede ser ejecutado de varias maneras. En un modelo que sea más real que el modelo genérico, estos aspectos no deben ser omitidos. El costo del chequeo de restricciones de integridad puede ser especificado muchas veces como la componente de procesamiento, excepto en caso de que el costo de la unidad local de procesamiento pudiera cambiar y alterar el verdadero costo de chequeo de restricciones de integridad.

Los costos operacionales de la transmisión de datos para las consultas de actualización y de sólo lectura son bien diferentes. En las consultas de actualización es necesario actualizar tantos sitios como réplicas existan, mientras que en las consultas de sólo lectura, es suficiente con acceder a sólo una de las copias. Además, al final de una solicitud de actualización, no hay transmisión de datos de regreso hacia el sitio que originó la solicitud, sino sólo un mensaje de confirmación, mientras que en las consultas de sólo lectura se puede producir una enorme transmisión de datos. La componente de actualización de la función de transmisión es:

$$TCU_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} u_{ij} \times x_{jk} \times g_{o(i),k} + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} u_{ij} \times x_{jk} \times g_{k,o(i)}$$

El primer término representa el envío del mensaje de actualización desde el sitio $o(i)$ que origina q_i, a todas las réplicas del fragmento que requieren ser actualizadas. El segundo término es para la confirmación. El costo de las consultas de sólo lectura es:

$$TCR_i = \sum_{\forall F_j \in F} \min_{S_k \in S} (r_{ij} \times x_{ij} \times g_{o(i),k} + r_{ij} \times x_{jk} \times \frac{sel_i(F_j)}{fsize} \times length(F_j) \times g_{k,o(i)})$$

El primer término en *TCR* representa el costo de transmitir la consulta de sólo lectura hacia los sitios que tienen copias de fragmentos que necesitan ser accedidos. El segundo término representa el costo para la transmisión de los resultados desde esos sitios hasta el sitio que originó la solicitud. La ecuación establece que entre todos los sitios que contienen copias de un mismo fragmento, sólo el sitio que logre el mínimo costo de transmisión total es el que debe ser elegido para la ejecución de la operación. Resumiendo, la función de costo de transmisión para la solicitud q_i puede plantearse como:

$$TC_i = TCU_i + TCR_i$$

la cual define la función de costo total.

Restricciones:

1. De tiempo de respuesta: $q_i \leq \text{MaxTime(i)}, \forall q_i \in Q$.

2. De almacenamiento: $\sum_{\forall F_j \in F} STC_{jk} \leq$ capacidad de almacenamiento en el sitio $S_k, \forall S_k \in S$

3. De procesamiento: $\sum_{\forall q_i \in Q}$ procesar la ejecución de q_i en el sitio $S_k \leq$ capacidad de procesamiento de $S_k, \forall S_k \in S$

El modelo anteriormente expuesto es muy general, el cual aborda muchos temas desde el punto de vista teórico; para lograr un desarrollo práctico es necesario hacer algunas consideraciones adicionales. Aunque una gran cantidad de investigadores han propuesto modelos y han diseñado algoritmos para ubicar fragmentos en BDD, la mayoría de los modelos son muy complejos y no muy bien comprendidos por lo que es difícil usarlos en un ambiente real. Al implementar la tecnología de BDD se han detectado problemas de desarrollo de metodologías prácticas integradas dentro del proceso general de modelado de los datos y la falta de herramientas que brinden ayuda al diseño de la ubicación de los datos en los diferentes sitios de una red de computadoras. Una propuesta interesante para resolver este problema es la planteada en la sección 2.5.

2.6. HERRAMIENTAS DE AYUDA AL DISEÑO DE BDD

Cualquier ayuda al diseño de BDD debe integrar metodologías prácticas de diseño que sean correctas y completas; y la implementación de herramientas enfatice mejor la interfaz y

cooperación entre las mismas. Se han propuesto soluciones desde el punto de vista teórico o práctico a las problemáticas de diseño identificadas en este capítulo, que han dado lugar a su perfeccionamiento e implementación computacional en un conjunto de herramientas, obteniéndose resultados confiables en un tiempo de respuesta aceptable. En el laboratorio de Bases de Datos de la Universidad Central "Marta Abreu" de Las Villas se han desarrollado herramientas para estos fines [13, 27, 32, 45-50], concebida como seis aplicaciones informáticas de alto valor práctico que mejoran considerablemente el proceso de diseño y tienen valor independiente:

1. El sistema SIADBDD que consiste en una herramienta integrada de software para el diseño de BDD, que se apoya en el trabajo de los productos de software que aparecen más abajo y que colaboran entre sí para materializar diseños de BDD logrando un óptimo diseño de distribución de datos en la red disminuyendo grandemente la complejidad del proceso de diseño.
2. El sistema ERECASE 2.0 que sistematiza la resolución de problemas de modelación de esquemas conceptuales globales como una herramienta CASE que brinda un amplio conjunto de construcciones del modelo Entidad-Relación Extendido y aplica diferentes validaciones estructurales a éste.
3. El sistema NETWIZARD 2.0 que es capaz de captar la información necesaria para la caracterización de la red y los sitios de procesamiento que darán soporte a los SBDD.
4. El sistema APPWIZARD 2.0 que es un asistente que ayuda a realizar, con menor complejidad, el trabajo de capturar información sobre las aplicaciones que forman parte de cualquier SBDD que se quiera diseñar.
5. El sistema FRAGMENTER para realizar el proceso de fragmentación de relaciones globales para obtener un conjunto de fragmentos lógicos que serán objeto de ubicación en los diferentes sitios de procesamiento donde residirá la BDD. Para esto usa información sobre las aplicaciones, los sitios de procesamiento y la comunicación de la red.
6. El sistema ALLOCATOR que implementa dos métodos metaheurísticos, uno basado en Algoritmos Genéticos Generacionales, y otro basado en el método Q-Learning de Aprendizaje Reforzado. Estos métodos pueden ser elegidos libremente por los diseñadores para materializar un diseño de distribución de datos determinado, ubicando los fragmentos obtenidos en pasos anteriores, concluyendo así el proceso de diseño de BDD.

La obtención de la gran cantidad de información requerida para el diseño de BDD es una tarea difícil que requiere de tiempo, experiencia y esfuerzo. Esto ha sido reducido mediante la

integración a nivel de procesos, donde los datos apropiados para cada tarea son obtenidos de un catálogo del diseño y a su vez cada herramienta coloca sus salidas en éste, para que sirvan de entrada a las tareas siguientes. Esta integración es positiva porque ahorran una gran cantidad de tiempo y esfuerzo.

La herramienta SIADBDD coordina la interoperabilidad entre las herramientas a través del catálogo que permite las funciones de búsqueda, obtención, transferencia y evaluación de la información almacenada para lleva a término diseños de BDD. Esta herramienta usa la metáfora de proyecto para realizar diseños de BDD, donde es posible crear nuevos proyectos o abrir proyectos ya guardados. Al crear un proyecto de diseño se genera automáticamente un nuevo catálogo, que es abierto en caso de que se quiera continuar con alguna tarea de diseño comenzada previamente. Una vez creado o abierto un proyecto se hacen accesibles las demás herramientas a través de opciones de menú y barra de herramientas.

Desde el punto de vista metodológico siguen los tres niveles de modelación de la arquitectura de referencia de las BDD: conceptual, lógico y físico. El nivel conceptual es abordado por los asistentes ERECASE, APPWIZARD y NETWIZARD introducidos en la sección 2.1; el nivel lógico es emprendido por FRAGMENTER y ALLOCATOR, mientras que el nivel físico se logra mediante réplicas que se materializan al ejecutar cada uno de los *scripts* generados por ALLOCATOR en cada uno de los SMBDD de cada uno de los sitios registrados por NETWIZARD.

2.7. CONCLUSIONES PARCIALES

Como conclusión de capítulo, se debe llamar la atención en la complejidad computacional de los métodos involucrados en el proceso de diseño de distribución, así como en la aparente ausencia de métodos para llevar los esquemas lógicos locales a esquemas físicos locales, así como la toma de decisiones respecto a cuánto replicar. Las herramientas enumeradas en la sección 2.5 han sido empleadas con éxito no sólo en la solución de varios problemas, por lo que se consideran apropiados para llevar a cabo el diseño de la BDD para el control de los bancos de transformadores de la OBE de Sancti Spiritus.

CAPÍTULO 3. ASPECTOS DEL DISEÑO Y LA IMPLEMENTACIÓN

En este capítulo se validan las herramientas obtenidas como resultado de este trabajo, en la gestión del control de los transformadores como parte del SIGERE en la OBE de Sancti Spiritus, siguiendo el ciclo de los transformadores desde que se instalan en un banco hasta que son retirados de la línea, almacenando todas las actividades que se realizan sobre los mismos.

3.1. PROBLEMA DEL CONTROL DE LOS TRANSFORMADORES DE DISTRIBUCIÓN

La OBE de Sancti Spiritus se dio a la tarea de implementar un módulo para el control de los transformadores instalados y las operaciones relacionadas en la explotación de los mismos, reflejando la estructura inherentemente distribuida de las diferentes unidades de la Empresa Eléctrica, teniendo autonomía local y ofreciendo buen rendimiento. En Cuba, durante los últimos años, la distribución ha sido el área de trabajo menos atendida de la Unión Eléctrica presentándose altos índices de pérdidas e interrupciones, un creciente índice de transformadores dañados, y un escaso nivel de informatización y automatización. Como respuesta a los problemas informáticos detectados se identifica como necesidad el desarrollo de un sistema que permita controlar los transformadores instalados con una reducción de los gastos totales de explotación de la distribución, influyendo en un mejor servicio a los clientes.

El transformador es el equipo más cercano al cliente que más abunda en las redes eléctricas cubanas con una distribución espacial muy variada y el mayor índice de fallas; de ahí la importancia de minimizar sus averías. Esta es otra razón por la que se necesita desarrollar un sistema capaz de automatizar toda la información relacionada con ellos, poder seguir sus ciclos de mantenimiento, su estado de carga; así como ser capaz de prevenir las fallas. Para diseñar el sistema hay que tener en cuenta la estructura geográficamente distribuida de las Empresas Eléctricas y sus propias características de comunicaciones; teniendo en cuenta los cuatro niveles bien definidos: nacional, provincial, territorial y de sucursal, además de un quinto nivel que es el del taller al mismo nivel de la provincia.

La comunicación entre la OBE Provincial y las OBEs territoriales se hace a través de líneas telefónicas de baja velocidad; aunque en los últimos tiempos se han introducido gradualmente las redes inalámbricas, aún no se cuenta con este sistema en todas las provincias, ni es política de la Unión Eléctrica hacerlo extensivo en poco tiempo. Como la topología de las redes informáticas no es la más adecuada para contar con una BDC provincial donde accedan todos los usuarios, ya

sean de la provincia, taller o territorio, es necesario diseñar una BDD, para poder seguir el ciclo de vida completo de cada transformador manteniendo datos históricos de todas las operaciones que se realizan. Esto le da la facilidad de hacer numerosos estudios comparativos del comportamiento de las diferentes instalaciones, equipos, así como prever posibles fallas futuras en ellos. El transformador tiene un ciclo de vida muy difícil de seguir ya que pasa por diferentes estados en diferentes ubicaciones. En el diseño de la BDD para el control de los transformadores se debe tener en cuenta la estructura inherentemente distribuida de las Empresas Eléctricas y sus propias características de comunicaciones.

El control de los transformadores se hace en el ámbito provincial, por lo que sólo intervienen la provincia, el territorio y el taller como sitios del sistema. El ciclo de vida del transformador comienza en el taller mediante la verificación de un transformador nuevo. Después pasa a formar parte de los transformadores disponibles en la OBE provincial donde aparecerán todos los datos de su ficha. Allí, dependiendo de sus características y las necesidades de las diferentes OBEs territoriales, es asignado a una de estas, y de ahí hacia el banco que lo necesita. En el municipio sólo se encuentran los datos pertenecientes a las instalaciones y equipos que atiende este territorio, mientras que en la provincia está centralizada toda la información de las OBEs territoriales.

En la OBE territorial se realizan todas las operaciones sobre el transformador y van conformando una historia. Existen dos formas de que el transformador vuelva a subir hacia la provincia: moviéndolo directo al taller o mediante un *Reporte de inspección al transformador dañado* que de ser positivo el daño, lo retira automáticamente del Banco y lo envía al taller, a la vez que se crea una *Necesidad de transformador* para sustituir este averiado. Luego de llegar el transformador y sus datos al taller, se realiza la *Defectación o Diagnóstico en el Taller*. Esta operación es la que dice si el transformador se retira definitivamente por su daño o si su problema es solucionable. En este caso, el transformador no pierde su historia, se da como disponible, pasa a la provincia y vuelve a fluir dependiendo de sus características técnicas y las necesidades de las OBEs territoriales. Nótese que en todo este ir y venir no pierde los datos de todas las pruebas que se le hayan realizado aunque sea situado en una OBE territorial diferente.

3.2. MODELACIÓN CONCEPTUAL

En la modelación conceptual se tomaron en cuenta todas las necesidades de información para el control de los transformadores, así como datos adicionales solicitados por la OBE de Sancti Spiritus. En la figura 3.1 se puede observar un diagrama ERE que representa una visión arquitectónica de la información del problema a solucionar.

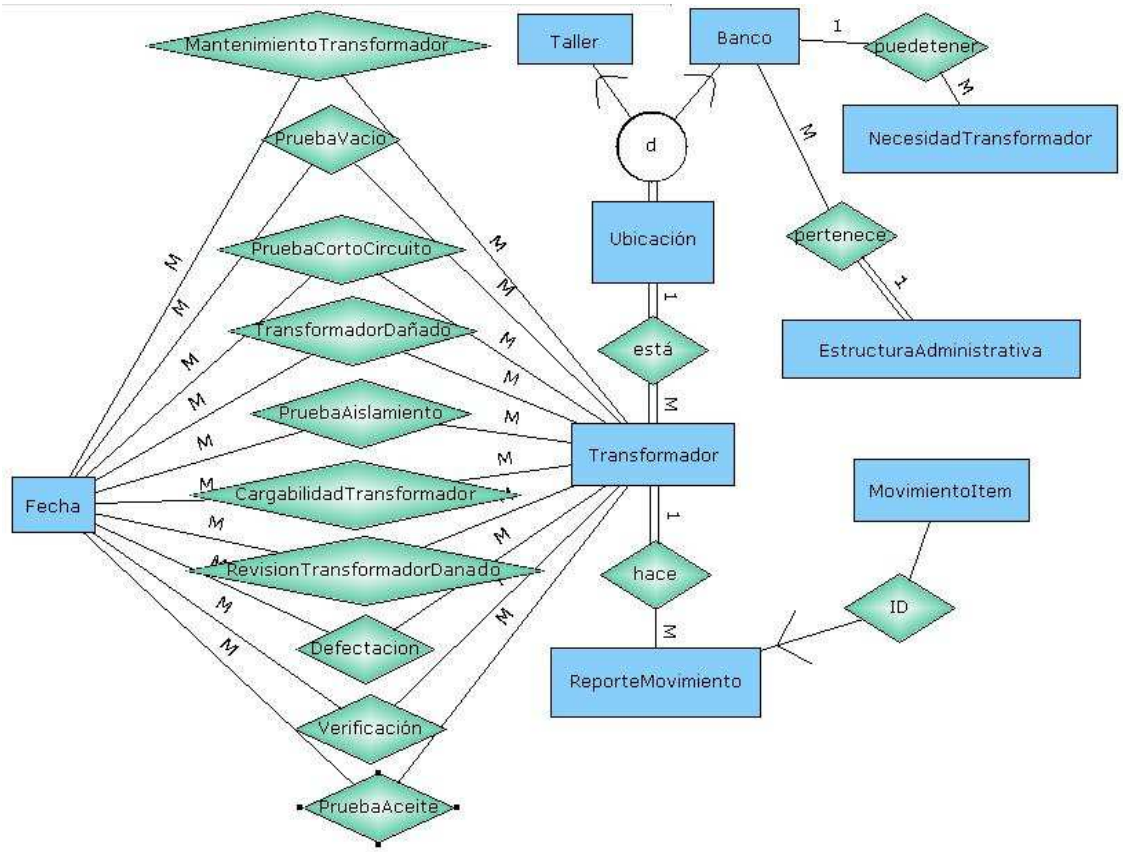


Figura 3.1. Diagrama ERE para el control de transformadores obtenido en ERECASE.

A continuación se presentan los esquemas relacionales obtenidos por ERECASE eliminándose FECHA y UBICACIÓN por ser innecesarios. En el anexo 2 se muesta los atributos que corresponden a cada esquema.

- Transformador
- Defectación
- Transformadordañado
- Pruebaaceite
- Pruebavacío
- Pruebacortocircuito
- Necesidadtransformador
- Banco
- Taller
- Pruebaaislamiento
- Verificación
- Mantenimientotransformador
- Revisióntransformadordañado
- Cargabilidadtransformador
- Movimientoitem
- Reportemovimiento
- Estructuraadministrativa

3.2.1. Caracterización de aplicaciones

A continuación se listan las principales aplicaciones junto a los datos para su caracterización:

APLICACIÓN 1: Verificación del Transformador

Descripción: Esta verificación se realiza en el taller a todo transformador nuevo que entra. Consiste en una serie de pruebas para saber si el equipo está apto para su uso.

Sitio donde se ejecuta: Taller

Frecuencia de activación en cada sitio (Semanalmente): 25

Esquemas que usa:

- TRANSFORMADOR

Cardinalidad: 7200

Selectividad: 65

Atributos de lectura/escritura:

id_EAdministrativa, Id_Transformador, Numemp, Id_Capacidad, Id_VoltajePrim, NoSerie,EstadoOperativo, NumFase, Id_Voltaje_Secun, NSección, Número, Tipoalimentación, Marca, AñoFabricación, Frecuencia, Id_CorrienteN, TipoEnfriamiento, PolaridadGrupo, Impedancia

- VERIFICACIÓN

Cardinalidad: 15000

Selectividad: 125

Atributos de lectura/escritura:

Id_Transformador , Id_EAEquipo, FechaEjecución, TipoTrabajo, CartaTécnica, Hermeticidad, Polaridad, Aceptado, EjecutadaPor, Observaciones, OrdenTrabajo , AltoVoltajeFrecIndustrial, PruebaAltoVoltaje, Folio, Name, Año, Id_FabricanteActual, UltAccionVer, Id_FabricanteAnterior, Impedancia, Peso, PesoAislante, Código, Temperatura

- PRUEBAACEITE

Cardinalidad: 50000

Selectividad: 350

Atributos de lectura/escritura:

Id_Transformador, Id_EAEquipo, Fecha, NivelPromedio, ColoraciónAceite, AcidezAceite, NivelAceite

- PRUEBAVACÍO

Cardinalidad: 50000

Selectividad: 350

Atributos de lectura/escritura:

Id_Transformador, Id_EAEquipo, Fecha, KVacío, KNúcleo, CorrVacío100, CorrVacío110, PerdNúcleo100, PerdNúcleo110

- PRUEBACORTOCIRCUITO

Cardinalidad: 50000

Selectividad: 350

Atributos de lectura/escritura:

Id_Transformador, Id_EAEquipo, Fecha, Voltaje, PérdidaCobre, Impedancia

- PRUEBAAISLAMIENTO

Cardinalidad: 50000

Selectividad: 350

Atributos de lectura/escritura:

Id_Transformador, Id_EAEquipo, Fecha, Temperatura, BajaAltaTierra1, AltaBajaTierra1, BajaAltaTierra2, AltaBajaTierra2, VoltMegger

APLICACIÓN 2: Movimiento de Transformador

Descripción: Es la acción de mover el equipo desde una instalación o taller hacia otra.
Sitios donde se ejecuta: OBE Territorial, OBE Provincial
Frecuencia de activación en cada sitio (Semanalmente):
- OBE Territorial: 17
- OBE Provincial: 46
Esquemas que usa:
- TRANSFORMADOR
Cardinalidad: 7200
Selectividad: 65
Atributos de solo lectura:
id_EAdministrativa, Id_Transformador, Código, Numemp, Fase, NoSerie
Atributos de lectura/escritura:
TapEncontrado, TapDejado, UltAcciónVer
- Reportemovimiento
Cardinalidad: 5000
Selectividad: 58
Atributos de lectura/escritura:
Id_Movimiento, FechaEjecución, Descripción, NoMovMedioBásico, H19, EjecutadaPor, Observaciones, OrdenTrabajo, Folio, Año
- MOVIMIENTOITEM
Cardinalidad: 12500
Selectividad: 100
Atributos de lectura/escritura:
Id_Movimiento, Id_Transformador, Id_EAEquipo, Causamos, Desde, Hacia, FaseConectada

APLICACIÓN 3: Buscar Transformador

Descripción: Es la acción de buscar uno o más equipos teniendo en cuenta una o más características.

Sitios donde se ejecuta: Taller, OBE Territorial, OBE Provincial

Frecuencia de activación en cada sitio (Semanalmente):

- Taller: 18
- OBE Territorial: 45
- OBE Provincial: 90

Esquemas que usa:

- TRANSFORMADOR

Cardinalidad: 7200

Selectividad: 65

Atributos de solo lectura:

id_EAdministrativa, Id_Transformador, Código, Numemp, Id_Capacidad, Id_VoltajePrim, Fase, NoSerie, NecesidadEmitida, CE, PosiciónBanco, EstadoOperativo, TapEncontrado, TapDejado, NumFase, UltAcciónVer, Id_Voltaje_Secun, NSección, Número, Tipoalimentación, NVerificado, Marca, AñoFabricación, Frecuencia, Id_CorrienteN, TipoEnfriamiento, PolaridadGrupo, Impedancia

- BANCO

Cardinalidad: 4500

Selectividad: 45

Atributos de solo lectura:

Código, CódigoAntiguo, Id_EAdirección, Codirección, Seccionalizador, Circuito, Conexión, Tipoalimentación, NSección, Tipoterreno, Id_VoltajeSalida, TipoSalida, Id_VoltajePrimario, Id_EAdministrativa, EstadoOperativo, Id_TipoCarga, NumClientes

- ESTRUCTURAADMINISTRATIVA

Cardinalidad: 8

Selectividad: 3

Atributos de solo lectura:

Id_EAdministrativa, Nombre, N_Consumidores

Con estos datos capturados se procede a capturar los predicados simples que indican localidad; que en este caso se corresponden con las estructuras administrativas de la OBE de Sancti Spiritus, las cuales son:

Id_EAdministrativa = ‘22’ para UEB Subcentro Fomento, Id_EAdministrativa = ‘23’ para UEB OBE Jatibonico, Id_EAdministrativa = ‘24’ para UEB Subcentro La Sierpe, Id_EAdministrativa = ‘25’ para UEB Subcentro Taguasco, Id_EAdministrativa = ‘26’ para UEB OBE Trinidad, Id_EAdministrativa = ‘36’ para UEB OBE Yaguajay, Id_EAdministrativa = ‘63’ para UEB OBE Cabaiguán, Id_EAdministrativa = ‘64’ para UEB OBE Sancti Spíritus.

Todos los datos anteriores fueron captados con APPWIZARD como se observa en el anexo 3.

3.2.2. Caracterización de la red y los sitios

Este proceso tiene un alto nivel de automatización mediante NETWIZARD, sólo se necesita indicar los sitios que alojarán la BDD y los agentes cliente y servidor de esta herramienta se encargan de recopilar la información necesaria. Los sitios identificados son:

1. UEB Subcentro Fomento
2. UEB OBE Jatibonico
3. UEB Subcentro La Sierpe
4. UEB Subcentro Taguasco
5. UEB OBE Trinidad
6. UEB OBE Yaguajay
7. UEB OBE Cabaiguán
8. UEB OBE Sancti Spíritus
9. Taller.

Los resultados obtenidos sobre la caracterización de los sitios aparecen en la tabla 3.1. La velocidad del procesador (*ProcClock*) se mide en gigahertz, la capacidad de disco ocupada y libre de *DiskSpace* y *FreeSpace* respectivamente se mide en gigabytes, mientras que el tiempo de lectura y escritura a disco (*R/AccTime* y *W/AccTime*) se mide en milisegundos.

SiteName	IP	ProcClock	DiskSpace	FreeSpace	R/AccTime	W/AccTime
Transformadores	172.19.231.12	2.33	200	67	0.003435	0.455237
Tecn_Cab	172.19.235.16	2.90	120	45	0.008568	0.003454
Tecn_Jco	172.19.229.19	3.27	250	100	0.004322	0.003456
Tecn_ssp	172.19.229.46	3.00	100	32	0.003012	0.003454
Tecn_Obetdad	172.19.233.23	0.70	60	9	0.006322	0.008434
Tecn_Yag	172.19.234.8	1.50	80	5	0.004284	0.006765
Sigere_Fto	172.19.236.10	2.53	149	54	0.003255	0.003767
Tecn_Sierpe	172.19.237.5	1.65	130	24	0.004222	0.005677
Tecn_Taguasco	172.19.238.10	2.83	80	7	0.003522	0.004545

Tabla 3.1. Parámetros de los sitios

FromStation	ToStation	AverageTime
172.19.236.10	172.19.229.19	2.36
172.19.236.10	172.19.237.5	1.56
172.19.236.10	172.19.238.10	3.56
172.19.236.10	172.19.233.23	2.78
172.19.236.10	172.19.234.8	1.69
172.19.236.10	172.19.231.16	6.20
172.19.236.10	172.19.229.46	4.50
172.19.236.10	172.19.231.12	2.60
172.19.229.19	172.19.237.5	1.90
172.19.229.19	172.19.238.10	4.20
172.19.229.19	172.19.233.23	1.60
172.19.229.19	172.19.234.8	2.80
172.19.229.19	172.19.231.16	4.93
172.19.229.19	172.19.229.46	5.35
172.19.229.19	172.19.231.12	2.82
172.19.237.5	172.19.238.10	3.70
172.19.237.5	172.19.233.23	4.15
172.19.237.5	172.19.234.8	2.37

172.19.237.5	172.19.231.16	2.52
172.19.237.5	172.19.229.46	2.95
172.19.229.5	172.19.231.12	3.60
172.19.238.10	172.19.233.23	5.80
172.19.238.10	172.19.234.8	6.10
172.19.238.10	172.19.231.16	3.01
172.19.238.10	172.19.229.46	7.1
172.19.238.10	172.19.231.12	2.3
172.19.233.23	172.19.234.8	5.4
172.19.233.23	172.19.231.16	3.1
172.19.233.23	172.19.229.46	6.5
172.19.233.23	172.19.231.12	4.5
172.19.234.8	172.19.231.16	3.45
172.19.234.8	172.19.229.46	3.8
172.19.234.8	172.19.231.12	3.7
172.19.231.16	172.19.229.46	1.23
172.19.231.16	172.19.231.12	2.56
172.19.229.46	172.19.231.12	2.78

Tabla 3.2. Parámetros de la red

Los resultados obtenidos sobre la caracterización de la red de comunicación aparecen en la tabla 3.2, registrándose el promedio del costo de envío (*AverageTime*) de un paquete de un sitio a otro (*FromStation* y *ToStation*) medido en segundos.

3.3. MODELACIÓN LÓGICA

Durante el análisis se identificó que BANCO tiene dependencia de existencia de ESTRUCTURAADMINISTRATIVA, lo que provocó una FHD en el esquema BANCO interviniendo el atributo Id_EAdministrativa. Se reconoció que el esquema TRANFORMADOR tiene atributos que sólo se usan en el TALLER motivando a elegir la opción de realizar primeramente una FV en FRAGMENTER, quedando en un fragmento los atributos usados por la

aplicación 1 “Verificación del Transformador” (Id_EAdministrativa, Id_Transformador, Numemp, Id_Capacidad, Id_VoltajePrim, NoSerie, EstadoOperativo, NumFase, Id_Voltaje_Secun, NSección, Número, Tipoalimentación, Marca, AñoFabricación, Frecuencia, Id_CorrienteN, TipoEnfriamiento, PolaridadGrupo, Impedancia); y el resto de los atributos en otro fragmento, con la llave primaria repetida en ambos fragmentos verticales como es lógico. Posteriormente se sometió a FH el fragmento de TRANSFORMADOR cuyos datos no eran usados en el Taller, obteniéndose una FHD de BANCO por el atributo Código; así mismo el esquema NECESIDADTRANSFORMADOR es sometido a FH, obteniéndose a su vez una FHD por el atributo Código de BANCO.

También se identificó que los esquemas de relacionales de las pruebas, obtenidos a partir de las asociaciones entre FECHA y TRANSFORMADOR en el ECG, tienen dependencia de existencia de TRANSFORMADOR, motivando a realizar FHD para cada prueba por el atributo Id_Transformador.

3.2. MODELACIÓN FÍSICA

El nodo principal (OBE Provincial) se configuró como publicador, realizando dos publicaciones. Una del Tipo Transaccional con actualización en un solo sentido para los codificadores y una del tipo Mezcla con filtros dinámicos que permiten realizar las fragmentaciones necesarias.

Las Suscripciones en los nodos de las OBEs territoriales se configuraron del tipo PULL atendiendo a las propias características de las redes de comunicaciones de la OBE con una programación de una conexión diaria en horarios preferentemente de madrugada para aprovechar el poco tráfico existente por las líneas.

3.3. IMPLEMENTACIÓN DEL MÓDULO DE TRANSFORMADORES DE DISTRIBUCIÓN

La codificación del Módulo de Transformadores se hizo utilizando la herramienta de programación orientada a objetos Borland Delphi, versiones desde la 6 hasta la 2006. Está hecho para trabajar en ambiente Cliente-Servidor, cuenta con un sistema de autenticación integrado el cual es único para todo el SIGERE lo que permite restringir el acceso a las opciones dependiendo de los privilegios de cada usuario, además de convertirlo en auditable, guardando en cada acción el usuario que la realiza, manteniendo una historia de operaciones realizadas sobre el sistema.

Se reconocieron tres actores del sistema y los casos de uso que se muestran en la figura 3.1: Técnico del Taller, Técnico de la OBE Territorial y Técnico de la OBE Provincial.

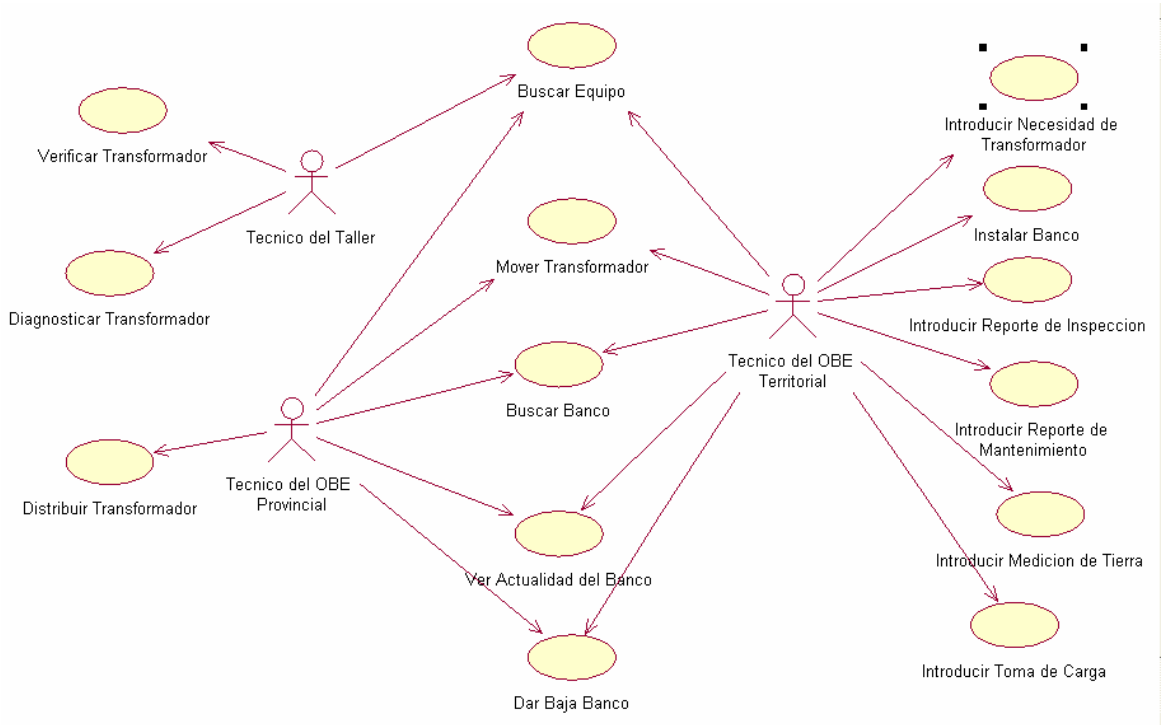


Figura 3.2. Diagrama de Casos de Uso del sistema.

Los casos de usos están divididos en los realizados sobre las Instalaciones, entiéndase Banco de Transformadores, y las realizadas sobre los Equipos que comprenden a los Transformadores, Pararrayos y Portafusibles.

Casos de Uso Realizados sobre los Bancos de Transformadores

Mantenimiento: Su objetivo es reflejar el estado de los principales equipos del Banco de Transformadores durante el mantenimiento, así como los principales parámetros, tales como el aterramiento y el estado de carga del banco.

Medición de tierra: Sirve para recoger los datos de la medición de tierra del Banco.

Instalar Banco: El registro de Centros de Transformación tiene como objetivo reflejar levantamientos y nuevos servicios, así como modificaciones realizadas a los levantamientos. Se incluyen además datos muy específicos de las cámaras de transformadores y centros de repartos los cuales han sido elaborados en base a la experiencia de la OBE soterrada en Ciudad Habana.

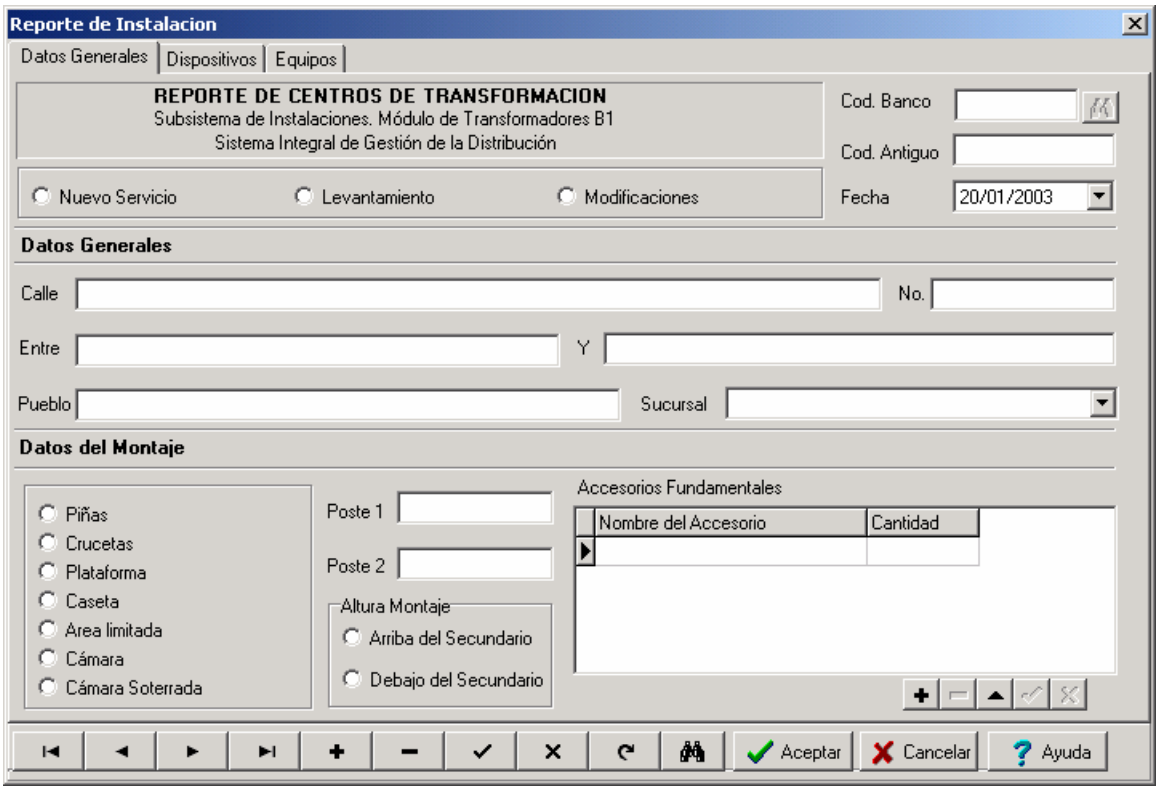


Figura 3.3. Interfaz del caso de uso de Instalar Banco.

Toma de Carga: Esta opción tiene el objetivo de registrar todos los parámetros medidos en la toma de carga a los Bancos de Transformadores, aquí se recogen datos de los transformadores, de la toma de carga y del balanceo de la carga.

Consumo por facturación: Esta opción es muy útil, pues permite realizar una toma de carga mensual aproximada a cada Banco de Transformador sin necesidad de instrumento de medición. Pues se aprovecha las lecturas que se realizan por parte del departamento comercial en la facturación eléctrica, teniendo asociado cada cliente al Banco que lo alimenta y aplicando un método sencillo de distribuir la energía que se factura a través de un gráfico de carga característico de cada Banco obteniéndose un gráfico de la potencia de carga diaria del Banco con una exactitud mayor al 80%, como se muestra en la figura 3.3.

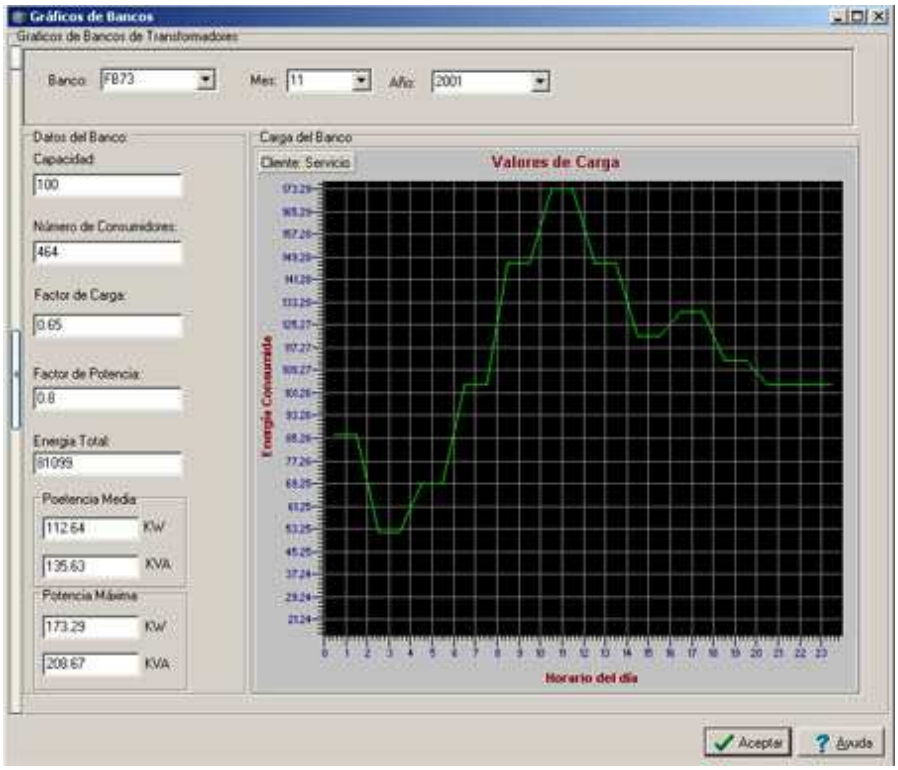


Figura 3.4. Gráfico de un Banco obtenido por el software.

Casos de Uso Realizados sobre los Equipos

Mover Equipo: Aquí es donde se introducen los movimientos de los diferentes equipos, la causa de su movimiento, retiro o ubicación en una instalación diferente, reflejando la instalación o lugar de origen y la de destino.

Necesidad de Transformadores: Esta es la opción que nos permite introducir las necesidades de transformadores, que puedan surgir por diferentes causas.

Verificar Transformador: Tiene como objetivo registrar todos los parámetros de los transformadores de distribución que se verifican en el taller; sea su procedencia debido a mantenimiento, enrollado o nueva adquisición.

Figura 3.5. Interfaz del Caso de Uso Verificar Transformador

Diagnosticar en el Taller: Tiene como objetivo introducir los datos relacionados con el examen final al transformador.

Figura 3.6. Interfaz del Caso de Uso Diagnosticar en el taller

Distribuir Transformadores: Esto nos permite distribuir los transformadores a las diferentes OBEs dependiendo de sus necesidades y las características de los transformadores.

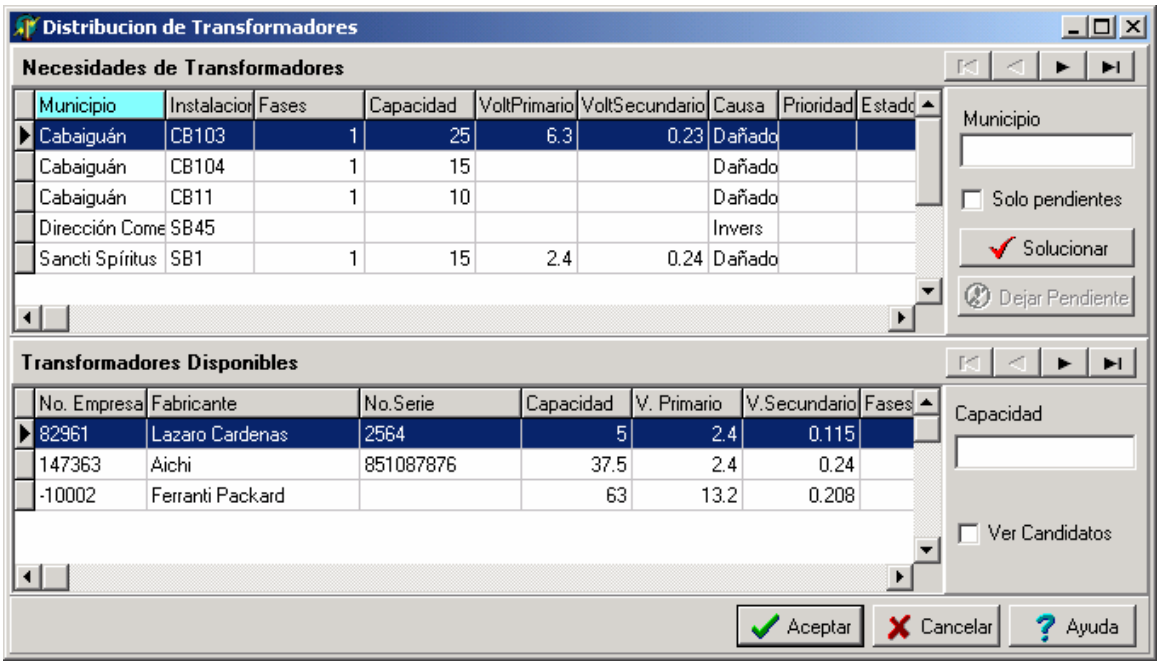


Figura 3.7. Interfaz del Caso de Uso Distribuir Transformador

Captar Reporte de Inspección a Transformador Dañado: Es donde se verifican los transformadores que fallan, se determina e introduce la causa por la cual el transformador se dañó y cual de los elementos del transformador influyó en el daño. Luego de ser introducido, se genera automáticamente una necesidad de transformador en dicha OBE territorial.

Herramientas de Búsqueda y Reportes del sistema

El Software cuenta con potentes herramientas de búsqueda muy flexibles, que permiten buscar equipos o instalaciones con diferentes características que se pueden especificar a la hora de realizar la exploración. Cada búsqueda genera un informe que puede ser impreso.

El Módulo de Transformadores mantiene los datos históricos de todas las operaciones que se realizan, permitiendo hacer numerosos estudios comparativos del comportamiento de las diferentes instalaciones de equipos; así como prever posibles fallas futuras en ellos. Para ello se conformó una serie de reportes que detallan todo lo relacionado con las principales características de las operaciones realizadas sobre las instalaciones, además de los principales parámetros. Dentro de la información que se brinda está la de:

Transformadores dañados en diferentes lapsos de tiempo organizados por: Causa, Circuito o Banco.

Existencia de transformadores organizado por OBE.

Transformadores subcargados o sobrecargados.

Transformadores con valores de tierra alterados.

Comportamiento de grupos de transformadores según la marca y el fabricante.

Otros informes mucho más específicos con información estratégica para la toma de decisiones de las OBEs territoriales.

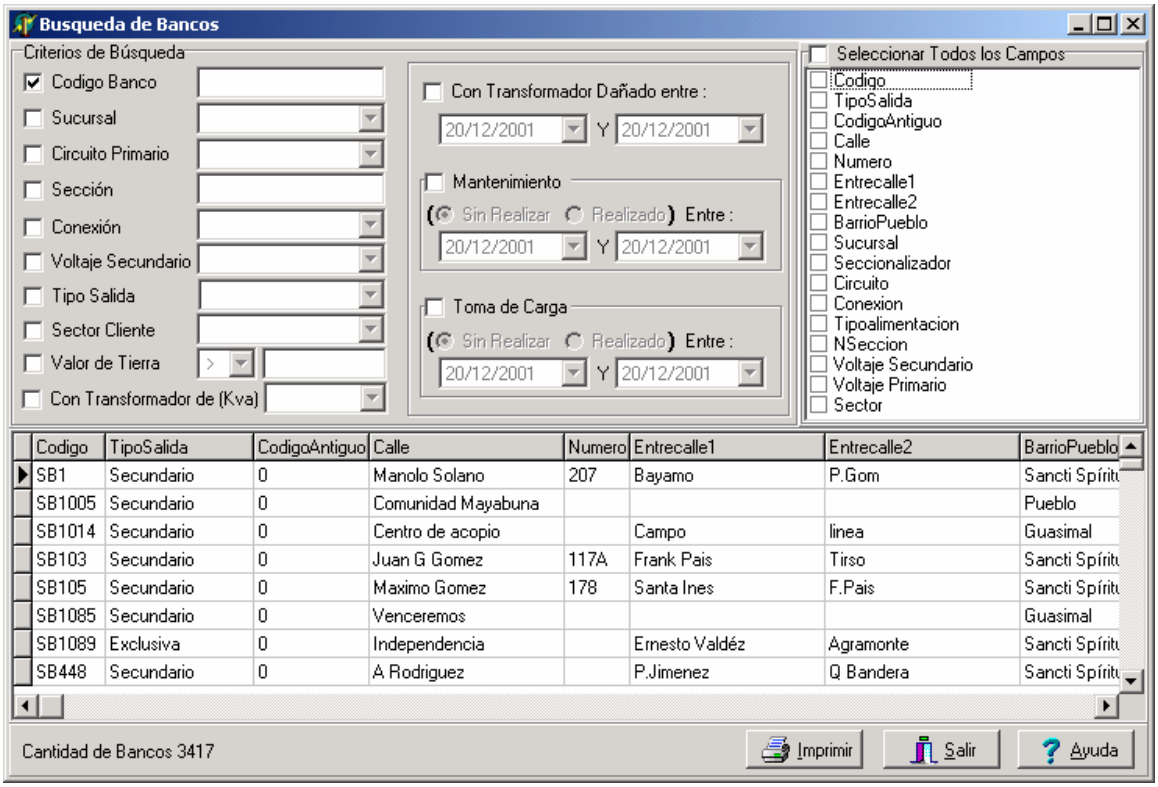


Figura 3.8. Interfaz de Búsqueda de Bancos.

Interfaz cartográfica del sistema

Otra de las formas de visualizar la información que brinda el Módulo de Transformadores es a través de una aplicación del tipo SIG. Mediante esta forma se puede ver y localizar en un mapa los Bancos de Transformadores por circuitos, por voltaje, transformadores por potencia, entre otros parámetros. En los lugares donde se tenga asociado el cliente al poste, puede verse qué clientes se alimentan de un Banco determinado, las rutas de cobro, los circuitos pertenecientes a cada Banco, así como el circuito primario del que se alimenta. Esto es de suma importancia a la hora de agilizar tomas de decisiones, pues cuenta con la ubicación espacial del equipo y la instalación.

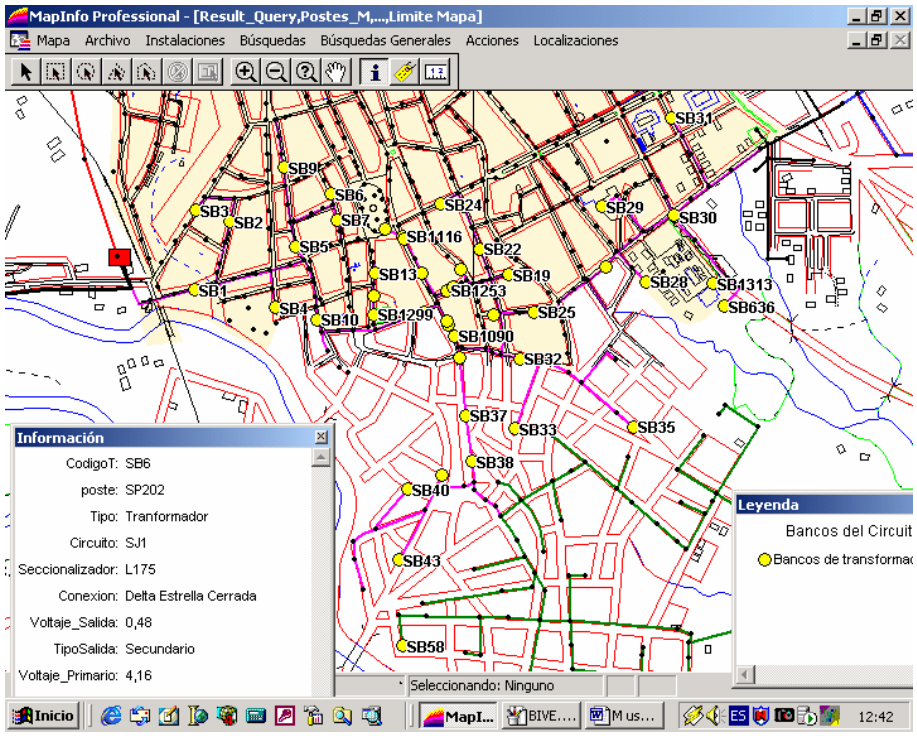


Figura 3.9. Interfaz Cartográfica del Módulo de Transformadores.

Las los diagramas de clases y las consultas de las principales aplicaciones son las siguientes:

APLICACIÓN 1: Verificación del Transformador

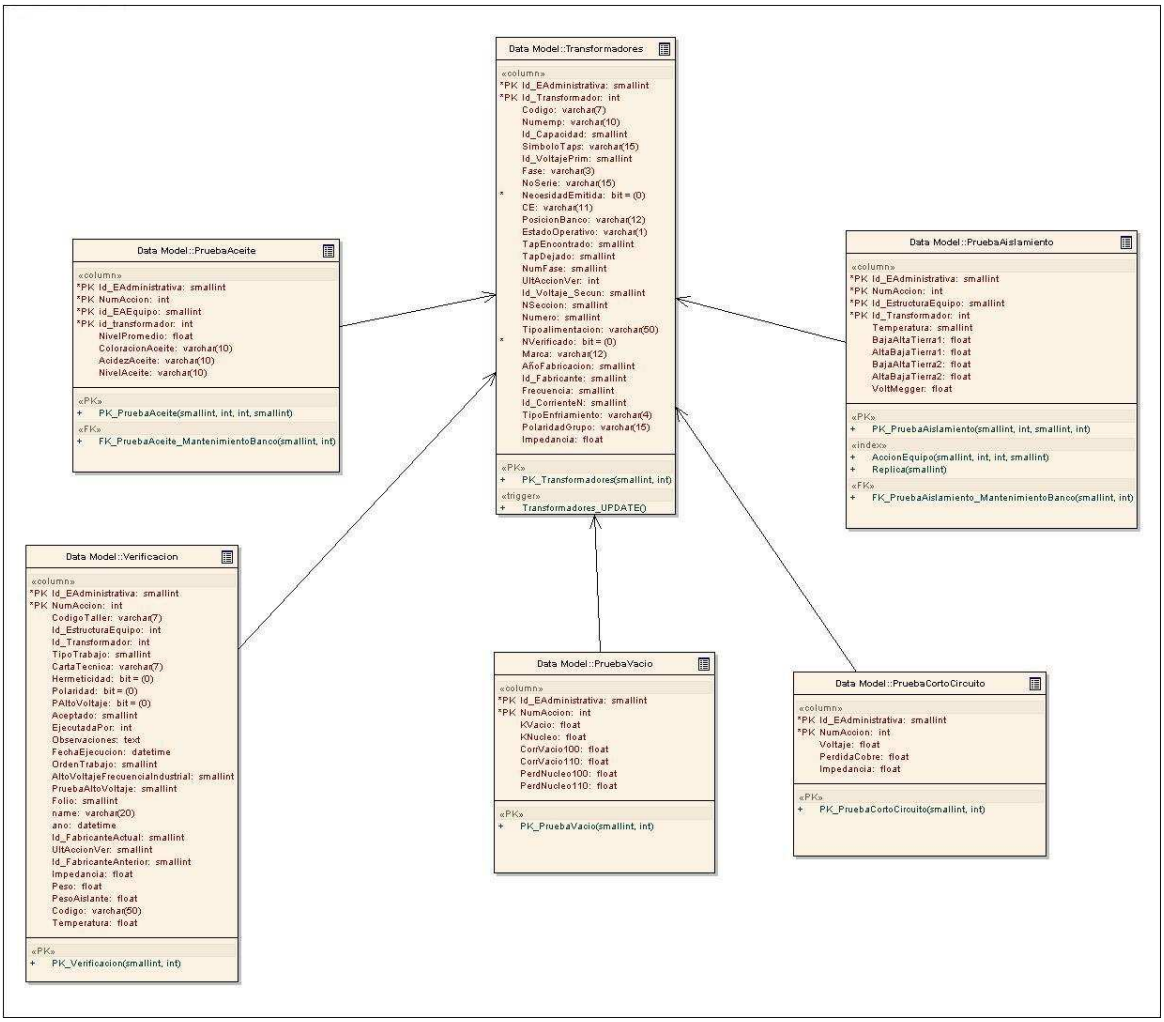


Figura. 3.10. Diagrama de Clases del caso de uso Verificación

Consulta (Sentencia SQL):

SELECT Verificacion.CodigoTaller, EstructurasAdministrativas.Nombre AS EstructuraAdministrativa, Transformadores.Numemp, Verificacion.TipoTrabajo, Verificacion.CartaTecnica, Verificacion.Hermeticidad, Verificacion.Polaridad, Verificacion.PAltoVoltaje, Verificacion.Aceptado, Personal.Nombre AS EjecutadoPor, Verificacion.Observaciones, Verificacion.FechaEjecucion,

Verificacion.OrdenTrabajo, Verificacion.AltoVoltajeFrecuenciaIndustrial, Verificacion.Folio, Verificacion.PruebaAltoVoltaje, Verificacion.name, Verificacion.ano, Verificacion.Temperatura, Verificacion.PesoAislante, Verificacion.Peso, Verificacion.Impedancia, Fabricantes.Nombre AS FabricanteActual, Fabricantes_1.Nombre AS FabricanteAnterior

FROM Verificacion LEFT OUTER JOIN Fabricantes Fabricantes_1 ON Verificacion.Id_FabricanteAnterior = Fabricantes_1.Id_Fabricante LEFT OUTER JOIN Fabricantes ON Verificacion.Id_FabricanteActual = Fabricantes.Id_Fabricante LEFT OUTER JOIN Transformadores ON Verificacion.Id_EstructuraEquipo = Transformadores.Id_EAdministrativa AND Verificacion.Id_Transformador = Transformadores.Id_Transformador LEFT OUTER JOIN EstructurasAdministrativas ON Verificacion.Id_EAdministrativa = EstructurasAdministrativas.Id_EAdministrativa LEFT OUTER JOIN Personal ON Verificacion.EjecutadaPor = Personal.Id_Persona

APLICACIÓN 2: Movimiento de Transformador

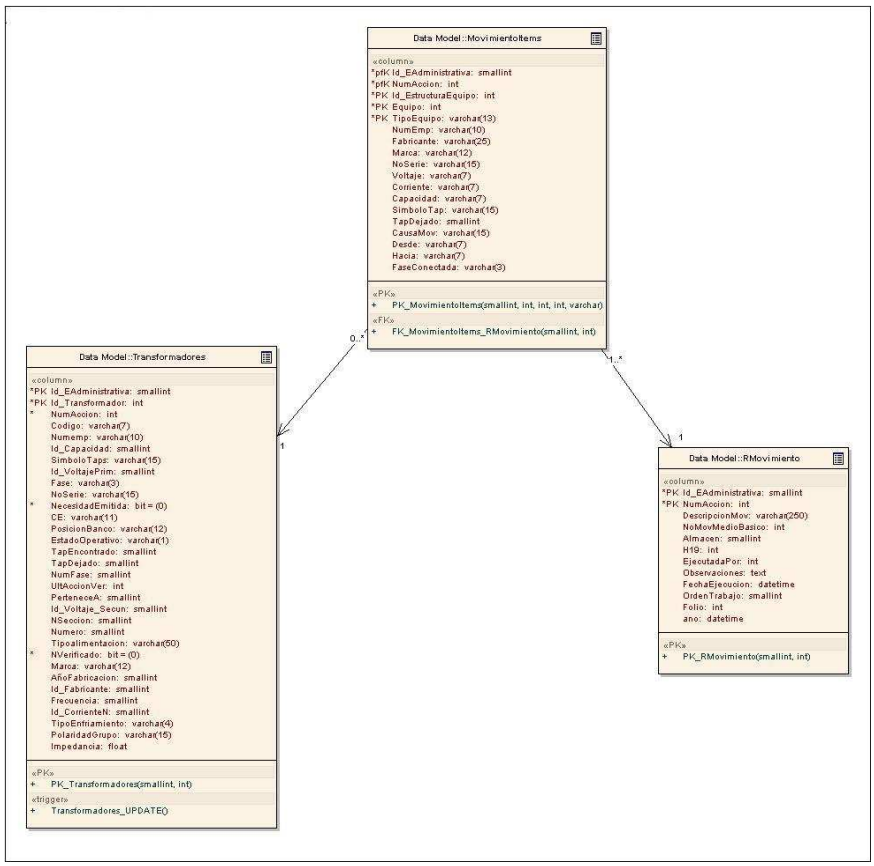


Figura. 3.11. Diagrama de Clases del caso Movimiento de Transformadores

Consulta (Sentencia SQL):

SELECT RMovimiento.DescripcionMov, RMovimiento.NoMovMedioBasico, RMovimiento.Almacen, RMovimiento.H19, Personal.Nombre, RMovimiento.Observaciones, RMovimiento.FechaEjecucion, RMovimiento.Folio, RMovimiento.OrdenTrabajo, RMovimiento.ano, MovimientoItem.NumEmp, MovimientoItem.Marca, MovimientoItem.NoSerie, MovimientoItem.Voltaje, MovimientoItem.Corriente, MovimientoItem.Capacidad, MovimientoItem.SimboloTap, MovimientoItem.TapDejado, MovimientoItem.CausaMov, MovimientoItem.Desde, MovimientoItem.Hacia, MovimientoItem.FaseConectada, Transformadores.Numemp AS NumeroEmpresa, Transformadores.Fase, Transformadores.NoSerie AS NumSerie, Transformadores.Marca AS MarcaT FROM RMovimiento INNER JOIN MovimientoItem ON RMovimiento.Id_EAdministrativa = MovimientoItem.Id_EAdministrativa AND RMovimiento.NumAccion = MovimientoItem.NumAccion INNER JOIN Transformadores ON MovimientoItem.Id_EstructuraEquipo = Transformadores.Id_EAdministrativa AND MovimientoItem.Equipo = Transformadores.Id_Transformador INNER JOIN Personal ON RMovimiento.EjecutadaPor = Personal.Id_Persona

APLICACIÓN 3: Buscar Transformador

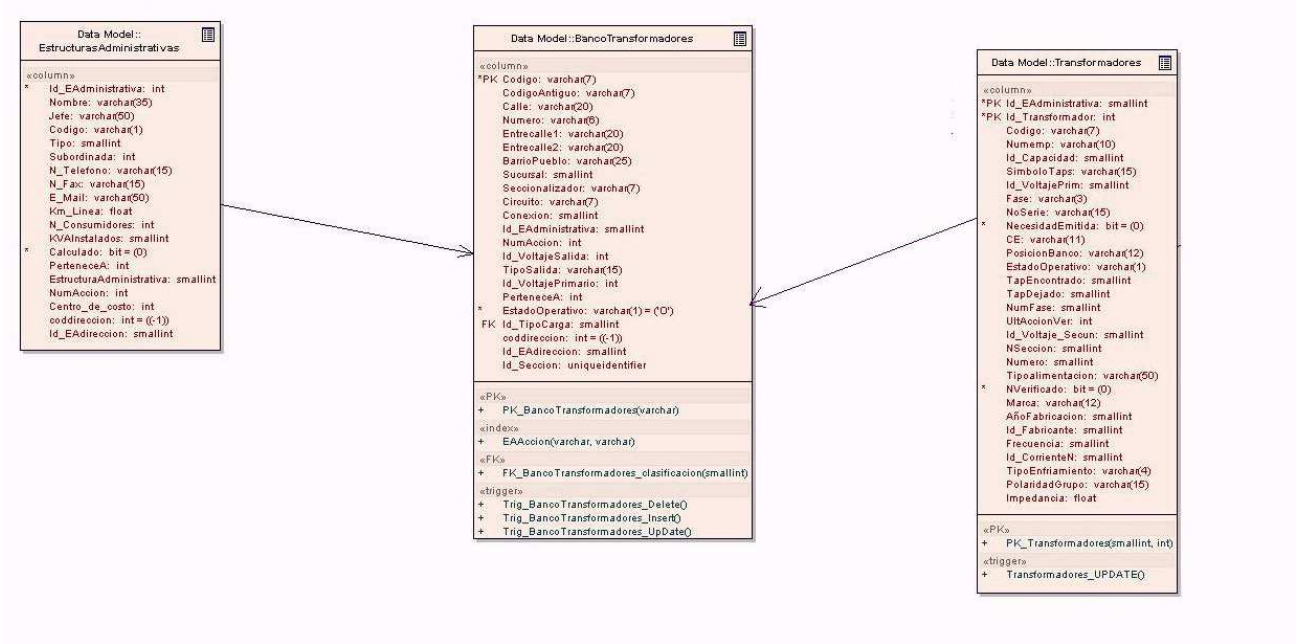


Figura. 3.12. Diagrama de Clases Buscar Transformador

Consulta (Sentencia SQL):

SELECT Transformadores.Codigo, Transformadores.Numemp AS [Num Emp.], Transformadores.Marca, Transformadores.AñoFabricacion AS [Año Fabricacion], Fabricantes.Nombre AS Fabricante, Transformadores.NoSerie AS [No. Serie], Capacidades.Capacidad, VoltajesSistemas_2.Voltaje AS [Voltaje Primario], VoltajesSistemas_1.Voltaje AS [Voltaje Secundario], Transformadores.Fase, Transformadores.NumFase, Transformadores.SimboloTaps AS [Simbolo Taps], Transformadores.Tipoalimentacion AS [Tipo Alimentacion], BancoTransformadores.Circuito, EstructurasAdministrativas.Nombre AS Sucursal, BancoTransformadores.Calle, Banco.Numero AS [Num. Dir], Banco.BarrioPueblo AS [Barrio/Pueblo], Transformadores.Id_Transformador, Transformadores.Id_EAdministrativa, Transformadores.Impedancia
FROM Transformadores INNER JOIN Banco ON Transformadores.Codigo = Banco.Codigo INNER JOIN EstructurasAdministrativas ON Banco.Sucursal = EstructurasAdministrativas.Id_EAdministrativa INNER JOIN Capacidades ON Transformadores.Id_Capacidad =.Id_Capacidad LEFT OUTER JOIN Fabricantes ON Transformadores.Id_Fabricante = Fabricantes.Id_Fabricante LEFT OUTER JOIN VoltajesSistemas VoltajesSistemas_2 ON Transformadores.Id_VoltajePrim = VoltajesSistemas_2.Id_VoltajeSistema LEFT OUTER JOIN VoltajesSistemas VoltajesSistemas_1 ON Transformadores.Id_Voltaje_Secun = VoltajesSistemas_1.Id_VoltajeSistema

En estas consultas se hace referencia a algunos codificadores del SIGERE como son:
VoltajesSistemas
Capacidades
Fabricantes

CONCLUSIONES

Se analizó la teoría existente respecto a las Base de Datos Distribuidas demostrándose la necesidad de una concepción distribuida de la Base de Datos para el control de los transformadores en la OBE de Sancti Spiritus.

Se aplicó el ambiente integrado de herramientas de ayuda al diseño Bases de Datos Distribuidas desarrolladas en el laboratorio de Bases de Datos del Centro de Estudios de Informática de la Universidad Central "Marta Abreu" de Las Villas en el diseño de la Base de Datos para el control de los transformadores. Estas herramientas permitieron obtener esquemas distribuidos de datos hasta el nivel físico.

- ERECASE permitió la modelación del esquema conceptual global
- NETWIZARD obtuvo la información necesaria para la caracterización de la red y los sitios de procesamiento
- APPWIZARD capturó la información sobre las aplicaciones más representativas del problema de modelación del control de transformadores.
- FRAGMENTER realizó la fragmentación de esquemas globales para obtener un conjunto de fragmentos aplicando los diferentes tipos de fragmentación: FHP, FHD y FV.
- ALLOCATOR efectuó la asignación del conjunto de fragmentos a los sitios. Se aplicaron los dos métodos que se brindan, Algoritmo Genético y Q-Learning de Aprendizaje Reforzado, para dar solución al problema de ubicación. Los resultados obtenidos por ambos métodos fueron similares. Se efectuó la asignación física mediante la generación de secuencias de comandos correspondientes al modelo de réplicas soportado por Microsoft® SQL Server ™.

Estos sistemas han sido empleados con éxito en este trabajo, disminuyendo el esfuerzo requerido para realizar el diseño distribuido de la BD, evitando la ejecución manual de métodos de solución a los diversos problemas que se presentan en el diseño de una base de Datos Distribuida de muy alta complejidad.

Se desarrolló un sistema capaz de dar respuesta a la necesidad actual de automatizar la gestión distribuida de los transformadores en la Empresa Eléctrica, controlando toda la información relacionada con los transformadores así como las pruebas y mediciones que se realicen sobre estos equipos.

Además de mantener datos históricos de todas las operaciones que se realizan para estudios comparativos del comportamiento de las diferentes equipos y prever posibles fallas futuras en ellos, también se da la posibilidad de visualizar la información mediante reportes y a través de una aplicación del tipo Sistema de Información Geográfico para ver y localizar en un mapa los Bancos de Transformadores.

Se brinda una arquitectura de software escalable y extensible, aplicable a todas las empresas eléctricas del país, de manera que responda de forma eficiente y fiable a las necesidades de la Unión Eléctrica.

RECOMENDACIONES

Para aplicar el software a otras empresas del país de debe realizar una nueva asignación teniendo en cuenta las características propias de los sitios y la red en el país, así como las aplicaciones más frecuentes en cada territorio. Las herramientas empleadas en el diseño de la Base de Datos permiten realizar esta tarea con un esfuerzo menor. Concretamente, usando el mismo catálogo del diseño de esta aplicación, se deben caracterizar la nueva red y los sitios con NETWIZARD, las nuevas aplicaciones con APPWIZARD y posteriormente usar FRAGMENTER para obtener los nuevos fragmentos que deben ser asignados a los sitios usando ALLOCATOR. Las secuencias de comandos obtenidas deben ser ejecutadas en cada uno de los sitios destino.

BIBLIOGRAFÍA

[1] Apers, P.M.G. (1988): Data Allocation in Distributed Database Systems, ACM Trans. Database Syst., 13 (3): 263-304.

[2] Awerbuch, B.; Bartal, Y.; Fiat, A. (2003): Competitive distributed file allocation, Inf. Comput., 185 (1): 1-40.

[3] Baião, F.A.; Mattoso, M.; Zaverucha, G. (2004): A Distribution Design Methodology for Object DBMS, Distributed and Parallel Databases, 16 (1): 45-90.

[4] Bellatreche, L.; Karlapalem, K.; Li, Q. (1998): Complex Methods and Class Allocation in Distributed Object-Oriented Database Systems. International Conference on Object Oriented Information Systems (OOIS 1998), Paris, France 239-256.

[5] Ceri, S.; Pelagatti, G. (1984): Distributed Databases: Principles and Systems, McGraw-Hill Book Company

[6] Ceri, S.; Pernici, B.; Wiederhold, G. (1987): Distributed Database Design Methodologies, IEEE Database Eng. Bull., 75 (5): 533-546.

[7] Cornell, D.W.; Yu, P.S. (1987): A Vertical Partitioning Algorithm for Relational Databases. In Proceedings of the Third International Conference on Data Engineering. IEEE 30 - 35.

[8] Coulon, C.; Pacitti, E.; Valduriez, P. (2005): Consistency Management for Partial Replication in a High Performance Database Cluster. IEEE International Conference on Parallel and Distributed Systems (ICPADS2005), Fukuoka, Japan.

[9] Chang, P.-Y.; Chen, D.-J.; Kavi, K.M. (2001): File Allocation Algorithms to Minimize Data Transmission Time in Distributed Computing Systems, J. Inf. Sci. Eng., 17 (4): 633-649.

[10] Chaturvedi, A.R.; Roan, J. (1994): Scheduling the allocation of data fragments in a distributed database environment: a machine learning approach, IEEE Transactions on Engineering Management, 41 (2): 194-207.

[11] Daudpota, N.H. (1998): Five Steps to Construct a Model of Data Allocation for Distributed Database Systems, J. Intell. Inf. Syst., 11 (2): 153-168.

[12] Ezeife, C.I.; Barker, K. (1995): A Comprehensive Approach to Horizontal Class Fragmentation in a Distributed Object Based System, Distributed and Parallel Databases, 3 (3): 247-272.

[13] García, C.E.; Rodríguez, A.; González, L.M.; Álvarez, W.A. (2005): ERECASE, una herramienta con validación de diagramas entidad relación. 6to. Simposium Iberoamericano de Computación e Informática SISI 2005. Instituto Tecnológico de Nuevo León, Monterrey, NL, México.

[14] Gu, X.; Lin, W.J.; Veeravalli, B. (2006): Practically realizable efficient data allocation and replication strategies for distributed databases with buffer constraints, IEEE Transactions on Parallel and Distributed Systems, 17 (9): 1001-1013.

[15] Hababeh, I.O.; Bowring, N.; Ramachandran, M. (2003): An integrated strategy for data fragmentation and allocation in a Distributed Database Design. Proceedings of the International Conference on Information Technology and Natural Science (ICITNS). Amman Jordan 268-274.

[16] Hababeh, I.O.; Bowring, N.; Ramachandran, M. (2004): A Method for Fragment Allocation in Distributed Object Oriented Database Systems. Proceedings of the 5th Annual PostGraduate Symposium on The Convergence of Telecommunications, Networking & Broadcasting (PGNet), Liverpool, UK 54 - 59.

[17] Hoffer, J.A.; Severance, D.G. (1975): The Use of Cluster Analysis in Physical Data Base Design. Proceedings of the International Conference on Very Large Data Bases, September 22-24, 1975, Framingham, Massachusetts, USA. ACM 69-86.

[18] Huang, Y.-F.; Chen, J.-H. (2001): Fragment Allocation in Distributed Database Design, Journal of Information Science and Engineering, 17 (3): 491-506.

[19] Iida, S. y otros (2004): A Dynamic Allocation Method of Basis Functions in Reinforcement Learning. In: Webb, G.I.; Yu, X. (eds.): Proceedings of the 17th Australian Joint Conference on Artificial Intelligence AI 2004: Advances in Artificial Intelligence. Springer, Cairns, Australia 272-283.

[20] Irún-Briz, L. y otros (2005): MADIS: A slim middleware for database replication. Proceedings of the 11st International Conference on Parallel Processing (EuroPar 2005). Lecture Notes in Computer Science, Vol. 3648. Springer 349-359.

[21] Johansson, J.M.; March, S.T.; Naumann, J.D. (2000): The effects of parallel processing on update response time in distributed database design. In: Ang, S.y otros (eds.): Proceedings of the Twenty-First International Conference on Information Systems ICIS. ACM, Brisbane, Australia 187-196.

[22] Karlapalem, K.; Pun, N.M. (1997): Query-Driven Data Allocation Algorithms for Distributed Database Systems. In: Hameurlain, A.; Tjoa, A.M. (eds.): Proceedings of the 8th

International Conference on Database and Expert Systems Applications, DEXA '97. Springer, Toulouse, France 347-356.

[23] Lee, J.W.; Baik, D.K. (2004): Database allocation modeling for optimal design of distributed systems, IEICE Transactions on Information and Systems. , E87D (7): 1795-1804.

[24] Lim, S.J.; Ng, Y.-K. (1997): Vertical Fragmentation and Allocation in Distributed Deductive Database Systems, Inf. Syst., 22 (1): 1-24.

[25] Lin, W.J.; Veeravalli, B. (2006): An object replication algorithm for real-time distributed databases, Distributed and Parallel Databases., 19 (2-3): 125-146.

[26] Lin, X.; Orlowska, M.E. (1995): An Integer Linear Programming Approach to Data Allocation with the Minimum Total Communication Cost in Distributed Database Systems, Information Sciences, 85 (1-3): 1-10.

[27] López, Á.V. (2001): NetWizard. Asistente para la caracterización de sitios de procesamiento. Morffi, A.R.; González, L.M.G. (Tutor). Departamento de Ciencia de la Computación. Facultad MFC. UCLV. pp.

[28] Ma, H.; Schewe, K.-D.; Wang, Q. (2005): Distribution design for higher-order data models. Massey University, Department of Information Systems., Palmerston North, New Zealand 50.

[29] Ma, H.; Schewe, K.-D.; Wang, Q. (2006): A Heuristic Approach to Cost-Efficient Fragmentation and Allocation of Complex Value Databases. In: Dobbie, G.; Bailey, J. (eds.): The 17th Australasian Database Conference (ADC2006) Vol. 49. ACM International Conference Archive Proceeding Series vol. 170. Australian Computer Society, Inc., Hobart, Australia.

[30] McCormick, W.T.; Schweitzer, P.J.; White, T.W. (1972): Problem decomposition and data reorganization by a clustering technique, Operations Research

[31] Mei, A.; Mancini, L.V.; Jajodia, S. (2003): Secure Dynamic Fragment and Replica Allocation in Large-Scale Distributed File Systems, IEEE Trans. Parallel Distrib. Syst., 14 (9): 885-896.

[32] Morell, A.; González, L.M.; Rodríguez, A. (2000): Algoritmos para la fragmentación vertical en bases de datos distribuidas. COMPUMAT 2000. 7mo. Congreso de la Sociedad Cubana de Matemática y Computación, Manzanillo, Cuba.

[33] Muthuraj, J.; Chakravarthy, S.; Varadarajan, R.; Navathe, S.B. (1993): A Formal Approach to the Vertical Partitioning Problem in Distributed Database Design. Proceedings of

the 2nd International Conference on Parallel and Distributed Information Systems (PDIS 1993), Issues, Architectures, and Algorithms. IEEE Computer Society, San Diego, CA, USA 26-34.

[34] Navathe, S.; Ceri, S.; Wiederhold, G.; Dou, J. (1984): Vertical partitioning algorithms for database design, ACM Transactions on Database Systems, 9 (4).

[35] Navathe, S.; Ra, M. (1989): Vertical partitioning for database design: A graphical algorithm. International Conference on Management of Data. ACM 440 - 450.

[36] Navathe, S.B.; Karlapalem, K.; Ra, M. (1995): A mixed fragmentation methodology for initial distributed database design. College of Computing, Georgia Institute of Technology, Atlanta, GA.

[37] Őzsu, M.T.; Valduriez, P. (1999): Principles of Distributed Database Systems, Second Edition, Prentice-Hall

[38] Park, S.-J.; Baik, D.-K. (1997): A data allocation considering data availability in distributed database systems. IEEE Computer Society 708-713.

[39] Pérez, J. y otros (2003): Adaptive Allocation of Data-Objects in the Web Using Neural Networks. In: Amedeo, C.; Franco, T. (eds.): Proceedings of the 8th Congress of the Italian Association for Artificial Intelligence, AI*IA 2003: Advances in Artificial Intelligence Springer, Pisa, Italy 154-164.

[40] Pérez, J. y otros (2005): An approach for solving very large scale instances of the design distribution problem for distributed database systems. Proceedings of the 4th International School and Symposium on Advanced Distributed Systems (ISSADS2005). Lecture Notes in Computer Science, Vol. 3563. Springer 33-42.

[41] Pérez, J. y otros (2004): Dynamic Allocation of Data-Objects in the Web, Using Self-tuning Genetic Algorithms. In: Bazzan, A.L.C.; Sofiane, L. (eds.): Proceedings of the 17th Brazilian Symposium on Artificial Intelligence- SBIA 2004, Advances in Artificial Intelligence Springer, São Luis, Maranhão, Brazil 376-384.

[42] Pérez, J. y otros (2003): Adaptive and Scalable Allocation of Data-Objects in the Web. In: Vipin, K.; Gavrilova, M.L.; Chih Jeng Kenneth, T.; L'Ecuyer, P. (eds.): Proceedings of the International Conference on Computational Science and Its Applications - ICCSA 2003, Part I. Springer, Montreal, Canada 134-143.

[43] Pérez, J. y otros (2000): Vertical Fragmentation and Allocation in Distributed Databases with Site Capacity Restrictions Using the Threshold Accepting Algorithm. In: Cairó Battistutti,

O.; Sucar, L.E.; Cantu, F.J. (eds.): MICAI 2000: Advances in Artificial Intelligence, Mexican International Conference on Artificial Intelligence. Springer, Acapulco, Mexico 75-81.

[44] Pramanik, S.; Kao, D.T.; Vineyard, D. (1992): Fragmentation of Recursive Relations in Distributed Databases. Springer 389-404.

[45] Rodríguez, A.; Cárdenas, A.; Castro, M.T.; González, l.M. (2007): Caracterización de la red de comunicación y sitios en el Diseño de Bases de Datos Distribuidas. XIII Convención de Ingeniería Eléctrica CIE´2007. Simposio de Automática y Sistemas Computacionales., Facultad de Ingeniería Eléctrica. Universidad Central de Las Villas. 4.

[46] Rodríguez, A.; González, L.M.; Águila, L.S. (2005): Asignación de fragmentos en Bases de Datos Distribuidas mediante la aplicación de Algoritmos Genéticos, Boletín de la Sociedad Cubana de Matemática y Computación, 3 (1):

[47] Rodríguez, A.; González, L.M.; Cabrera, L.; Morell, A. (2002): ERECASE:Una herramienta de ayuda a la modelación de esquemas conceptuales globales. I Workshop de Bases de Datos, Jornadas Chilenas de Computación JCC 2002. Cámara Chilena del Libro A.G., Universidad de Atacama, Copiapó, Chile 49-58.

[48] Rodríguez, A. y otros (2002): Integración de herramientas de ayuda al diseño de bases de datos distribuidas. I Workshop de Bases de Datos, Jornadas Chilenas de Computación JCC 2002. Cámara Chilena del Libro A.G., Universidad de Atacama, Copiapó, Chile 111-120.

[49] Rodríguez, A.; Rosa, D.; Mainegra, M.; González, L.M. (2007): An Intelligent Agent using Reinforcement Learning to Solve the Allocation Problem in a Distributed Database with Replication. Universidad Central de Las Villas, Santa Clara.

[50] Rosa, D. (2006): Aplicación de técnicas de Inteligencia Artificial en la solución del problema de ubicación en el diseño de bases de datos distribuidas. Rodríguez, A.; González, L.M. (Tutor). Departamento de Ciencia de la Computación. Universidad Central "Marta Abreu" de Las Villas. pp.

[51] Savsar, M.; Al-Anzi, F.S. (2006): Reliability of data allocation on a centralized service configuration with distributed servers Computer Journal, 49 (3): 258-267.

[52] Schewe, K.-D. (2002): Fragmentation of object oriented and semi-structured data. In: Haav, H.-M.; Kalja, A. (eds.): Databases and Information Systems II. Kluwer Academic Publishers 1-14.

[53] Shepherd, J.; Harangsri, B.; Chen, H.L.; Ngu, A.H.H. (1995): A Two-Phase Approach to Data Allocation in Distributed Databases. In: Ling, T.W.; Masunaga, Y. (eds.): Proceedings of

the 4th International Conference on Database Systems for Advanced Applications (DASFAA). World Scientific, Singapore 380-387.

[54] Sun, W.; Shu, J.; Zheng, W. (2004): Dynamic File Allocation in Storage Area Networks with Neural Network Prediction. In: Yin, F.; Wang, J.; Guo, C. (eds.): Proceedings of the International Symposium on Neural Networks: Advances in Neural Networks - ISNN 2004, Vol. II. Springer, Dalian, China 719-724.

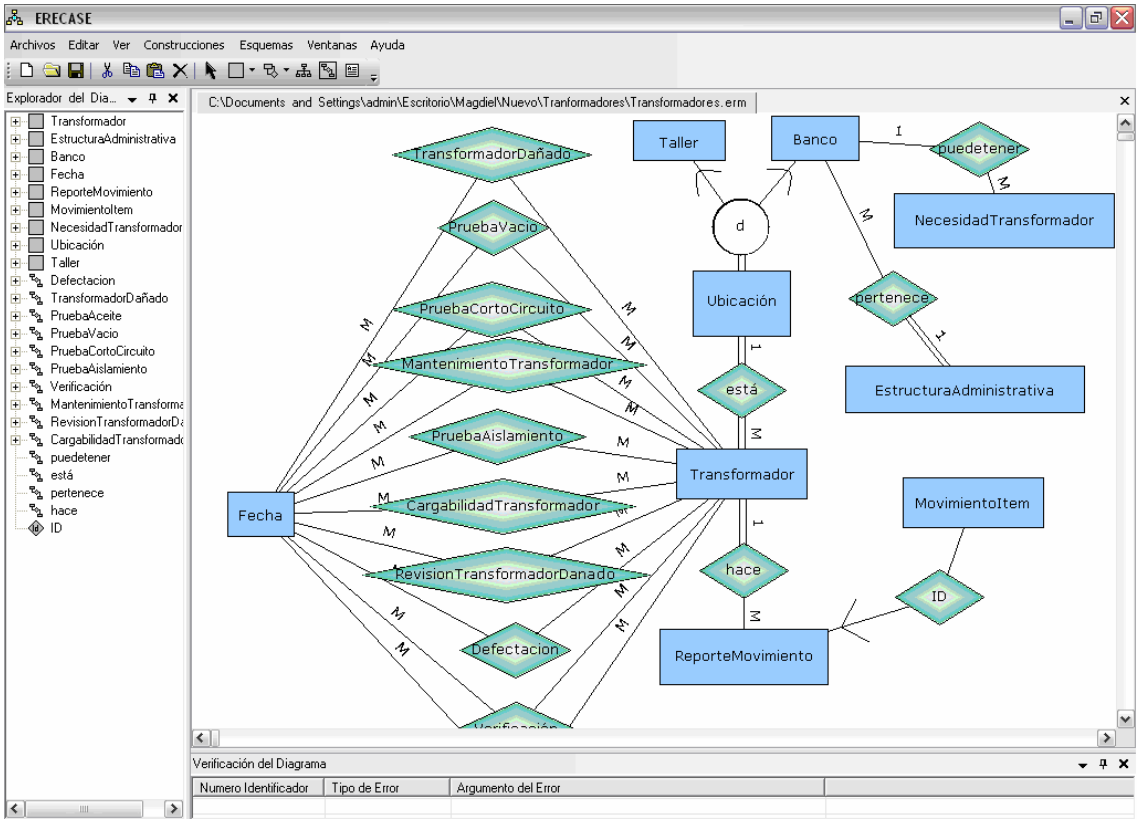
[55] Tamhankar, A.M.; Ram, S. (1998): Database fragmentation and allocation: an integrated methodology and case study, IEEE Transactions on Systems, Man, and Cybernetic Part A, 28 (3): 288-305.

[56] Tsai, H.-P. y otros (2002): SIFA: A Scalable File System with Intelligent File Allocation. Proceedings of the 26th. International Computer Software and Applications Conference (COMPSAC 2002), Prolonging Software Life: Development and Redevelopment IEEE Computer Society, Oxford, England 793-798.

[57] Wolfson, O.; Jajodia, S. (1995): An Algorithm for Dynamic Data Allocation in Distributed Systems, Inf. Process. Lett., 53 (2): 113-119.

[58] Zhou, X.; Zhang, Y.; Orlowska, M.E. (1994): A new Fragmentation Scheme for Recursive Query Processing, Data Knowl. Eng., 13 (2): 177-192.

ANEXO 1. VISTA DE ERECASE



ANEXO 2. ESQUEMAS RELACIONALES GLOBALES PARA EL CONTROL DE TRANSFORMADORES

TRANSFORMADOR (id_EAdministrativa, Id_Transformador, Código, Numemp, Id_Capacidad, Id_VoltajePrim, Fase, NoSerie, NecesidadEmitida, CE, PosiciónBanco, EstadoOperativo, TapEncontrado, TapDejado, NumFase, UltAcciónVer, Id_Voltaje_Secun, NSección, Número, Tipoalimentación, NVerificado, Marca, AñoFabricación, Frecuencia, Id_CorrienteN, TipoEnfriamiento, PolaridadGrupo, Impedancia)

DEFECTACIÓN(Id_Transformador, Id_EAEquipo, Fecha, Dañado, Baja, Causa, Observaciones, Ejecutadopor, OParme, OPenrollado, OPconexiones, OPemsamblado, OPaceite, OPhermeticidad, OPpruebas, Taller)

TRANSFORMADORDAÑADO(Id_Transformador, Id_EAEquipo, Fecha, Instalación, InformadoPor, Fecha, Fase, Causapreeliminar, Tipodaño, CausaDefinitiva, Observaciones)

PRUEBAACEITE(Id_Transformador, Id_EAEquipo, Fecha, NivelPromedio, ColoraciónAceite, AcidezAceite, NivelAceite)

PRUEBAVACÍO(Id_Transformador, Id_EAEquipo, Fecha, KVacío, KNúcleo, CorrVacío100, CorrVacío110, PerdNúcleo100, PerdNúcleo110)

PRUEBACORTOCIRCUITO(Id_Transformador, Id_EAEquipo, Fecha, Voltaje, PérdidaCobre, Impedancia)

PRUEBAAISLAMIENTO(Id_Transformador, Id_EAEquipo, Fecha, Temperatura, BajaAltaTierra1, AltaBajaTierra1, BajaAltaTierra2, AltaBajaTierra2, VoltMegger)

VERIFICACIÓN(Id_Transformador, Id_EAEquipo, FechaEjecución, TipoTrabajo, CartaTécnica, Hermeticidad, Polaridad, PAltoVoltaje, Aceptado, EjecutadaPor, Observaciones, OrdenTrabajo, AltoVoltajeFrecuenciaIndustrial, PruebaAltoVoltaje, Folio, Name , año, Id_FabricanteActual, UltAcciónVer, Id_FabricanteAnterior, Impedancia, Peso, PesoAislante, Código, Temperatura)

MANTENIMIENTOTRANSFORMADOR(Id_Transformador, Id_EAEquipo, FechaEjecución, Fase, EstadoPinturaTanque, EstadoPinturaRótulos, Hermeticidad)

REVISIÓNTRANSFORMADORDAÑADO(Id_Transformador, Id_EAEquipo, FechaEjecución, Código, TipoDaño, BushingPPartido, BushingPFlojo, BushingPFCont, BushingSPartido, BushinfSFlojo, BushingSFCont, Salidero, TanquePerforado, BajantesSMalos, SobrecargaEvidente, CausaDaño, VientosFuertes, Lluvias, TormentasEléctricas, AccidenteTránsito, Derrumbe, contaminación, InstalaciónReciente, Observaciones, EjecutadaPor,

TipoDaño, OT, Folio, año, CorrienteNominal, CorrienteOperación, EstadoDesconectivo, DesconectivoCliente, EstadoCliente, Dilatado, Enrredado, Árboles, AcometidaCC, Tendedera, CCInterno, CantidadConsumidores, DistUltimoConsumidor)

CARGABILIDADTRANSFORMADOR(Id_Transformador, Id_EAEquipo, FechaEjecución, KVAMedido, CodTomaCarga)

MOVIMIENTOITEM (Id_Movimiento, Id_Transformador, Id_EAEquipo, CausaMov, Desde, Hacia, FaseConectada)

REPORTEMOVIMIENTO(Id_Movimiento , FechaEjecución, Descripción, NoMovMedioBásico, H19, EjecutadaPor, Observaciones, OrdenTrabajo, Folio, año)

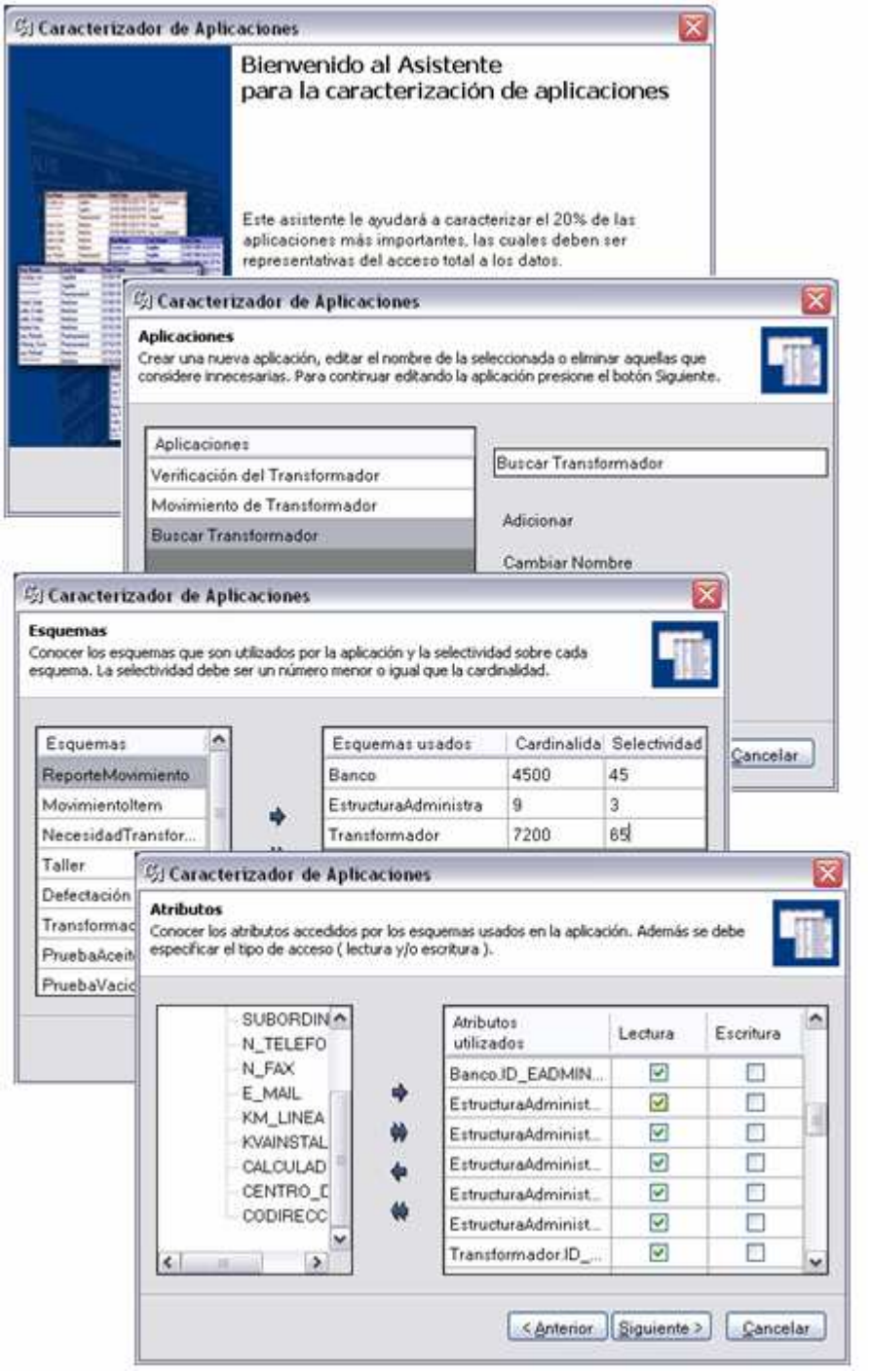
NECESIDADTRANSFORMADOR(Id_Necesidades, Código, id_Capacidad, NumFases, Id_VPrimario, Id_VSecundario, CausaNecesidad, Prioridad, Fecha, Estado, Confirmada, AcciónAutomática, TipoAfectación, TipoServicio)

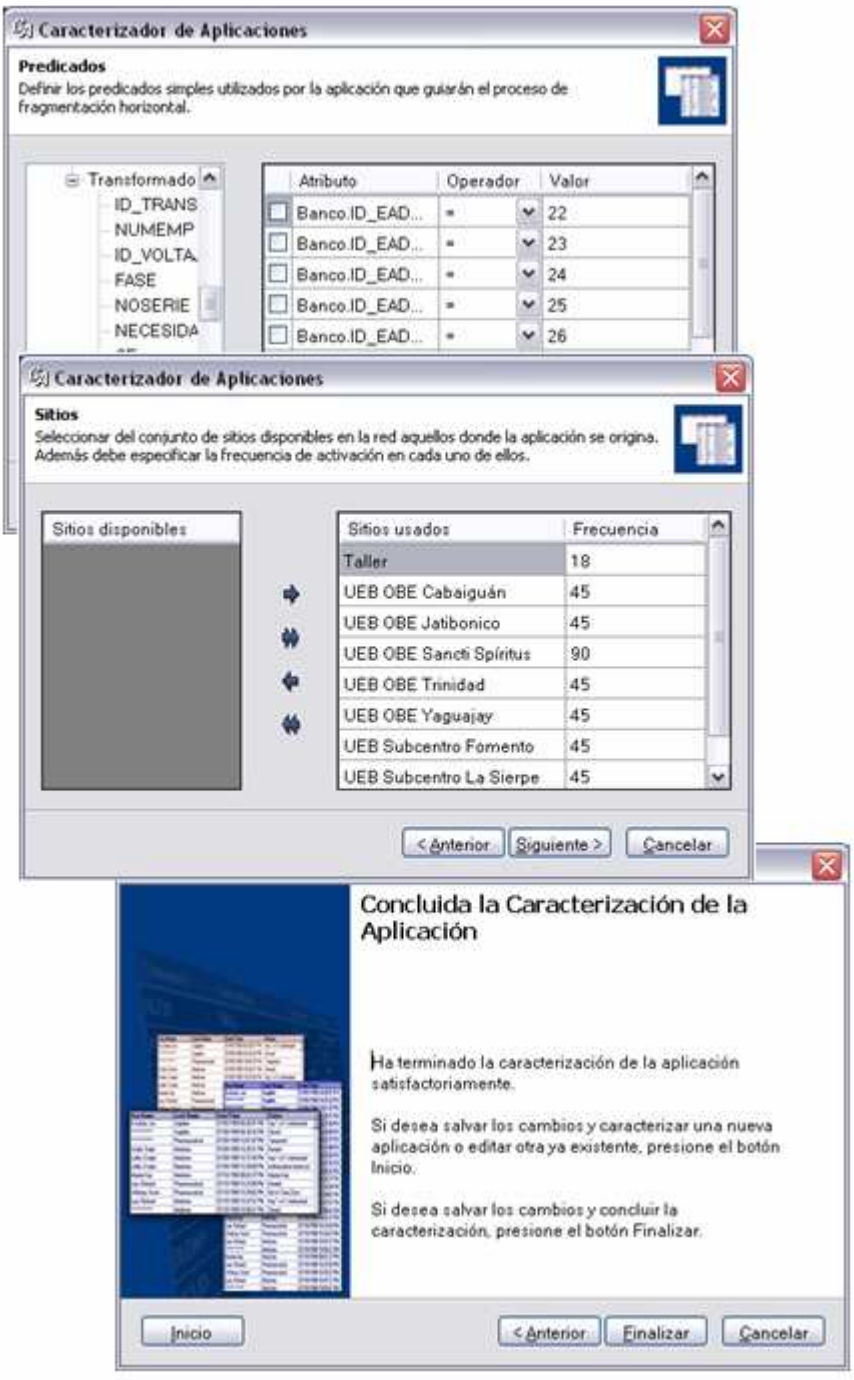
BANCO(Código, CódigoAntiguo, Id_EAdirección, codirección , Secciónalizador, Circuito, Conexión, Tipoalimentación, NSección, Tipoterreno, Id_VoltajeSalida, TipoSalida, Id_VoltajePrimario, Id_EAdministrativa, EstadoOperativo, Id_TipoCarga, NumClientes)

ESTRUCTURAADMINISTRATIVA(Id_EAdministrativa, Nombre, Jefe, Código, Tipo, Subordinada, N_Teléfono, N_Fax, E_Mail, Km_Línea, N_Consumidores, KVAInstalados, Calculado, Centro_de_costo, Id_EAdirección, codirección)

TALLER(Código, N_Teléfono, Id_EAdirección, codirección, Norma)

ANEXO 3. VISTA DE APPWIZARD




Caracterizador de Aplicaciones
Predicados
Definir los predicados simples utilizados por la aplicación que guiarán el proceso de fragmentación horizontal.
Transformado
ID_TRANS
NUMEMP
ID_VOLTA
FASE
NOSERIE
NECESIDA
Atributo
Operador
Valor
Banco.ID_EAD...
22
23
24
25
26
Caracterizador de Aplicaciones
Sitios
Seleccionar del conjunto de sitios disponibles en la red aquellos donde la aplicación se origina. Además debe especificar la frecuencia de activación en cada uno de ellos.
Sitios disponibles
Sitios usados
Frecuencia
Taller
18
UEB OBE Cabaiguán
45
UEB OBE Jatibonico
45
UEB OBE Sancti Spíritus
90
UEB OBE Trinidad
45
UEB OBE Yaguajay
45
UEB Subcentro Fomento
45
UEB Subcentro La Sierpe
45
< Anterior
Siguiente >
Cancelar
Concluida la Caracterización de la Aplicación
Ha terminado la caracterización de la aplicación satisfactoriamente.
Si desea salvar los cambios y caracterizar una nueva aplicación o editar otra ya existente, presione el botón Inicio.
Si desea salvar los cambios y concluir la caracterización, presione el botón Finalizar.
Inicio
< Anterior
Finalizar
Cancelar