PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA
OBTENCIÓN DEL TITULO DE
INGENIERO DE SISTEMAS

"ANÁLISIS DE HERRAMIENTAS CASE APLICADO A UN SISTEMA DE PROVEEDURÍA USANDO LA METODOLOGÍA ADOOSI"

Patricia de las Mercedes Carrillo Sarabia

DIRECTORA DE LA DISERTACIÓN: Ing. Natasha Bayas.

AMBATO, 2002

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA

OBTENCIÓN DEL TITULO DE
INGENIERO DE SISTEMAS

"ANÁLISIS DE HERRAMIENTAS CASE APLICADO A UN SISTEMA DE PROVEEDURÍA USANDO LA METODOLOGÍA ADOOSI"

DIRECTORA: _____
Ing. Natasha Bayas.

REVISORES : _____ _____
Ing. Victor Chuncha. **Ing. Janio Jadán.**

Patricia de las Mercedes Carrillo Sarabia

AMBATO, 2002

AGRADECIMIENTO

Planificar y desarrollar este trabajo de investigación, me ha llevado mucho tiempo, en la recolección de datos, compilación y corrección. Fue una tarea feliz aunque difícil y debo agradecer a todos aquellos cuya contribución lo hizo posible.

A mis padres y hermanos, que una y otra vez me brindaron amor, sugerencias, ideas y un constante apoyo emocional.

A la Ing. Natasha Bayas Directora de Disertación, cuya intervención resulto fundamental en cada una de las etapas y seguimiento del proyecto.

A los Ing. Janio Jadán y Víctor Chuncha Revisores de la Disertación , quienes siempre me guiaron con eficacia y perseverancia para poder concluir con la elaboración de mi tesis.

A mis compañeros de trabajo quienes permanentemente me dieron fortaleza y guía para la culminación de este trabajo.

A todos ellos y a mis maestro que con sus sabias enseñanzas, supieron formar mi persona y mi espíritu, les agradezco de corazón

DEDICATORIA

El esfuerzo invertido en este trabajo va dedicado a DIOS por haberme dado el don divino de la existencia.

A mis queridos padres, quienes día a día con amor, sacrificio y olvidándose de si mismo supieron guiarme por el camino de la rectitud, honestidad, lealtad para alcanzar la culminación de mi carrera .

A mis hermanos, que han sabido ayudarme en cada instante de mi vida.

Para ellos, este trabajo en el que sintetizo todo mi esfuerzo, dedicación y cariño

Patricia Carrillo Sarabia

CAPITULO I

1. HERRAMIENTAS CASE

1.1 INTRODUCCIÓN

Los humanos han evolucionado grandemente en su comportamiento, estructura social y muchos otros valores, el concepto de usar herramientas para facilitar las tareas aún prevalece. Esto puede ser apreciado en diversas disciplinas tales como Medicina, Ingeniería Mecánica, Arquitectura, etc. Una de las últimas disciplinas reclutadas por este concepto es la Ingeniería de Software.

Ciertamente el término "Ingeniería del Software" fue expuesto por primera vez en el año 1968 en una conferencia de la OTAN(Organización del Tratado del Atlántico Norte). En esa conferencia se reveló la existencia de una "crisis de software". Eso reconocía la problemática de desarrollo de software y aspiraba a encontrar soluciones.

Muchos de los intentos en aquel entonces no produjeron soluciones reales. Los expertos identificaron incorrectamente, que la producción del código era la razón primordial de dichos problemas. Debido a la falta de éxito, a finales de los 70’s comenzó un cambio en el modo de atacar los problemas de especificaciones, diseño y administración. Este período introdujo una variedad de técnicas que enfocaban las fases iniciales del ciclo de desarrollo del software. Las empresas privadas y las administraciones públicas entraron en los años 90 en una era de los cambios sin precedentes: aceleración tecnológica, nuevas exigencias sociales, fusiones, adquisiciones etc.

En este contexto turbulento las estructuras, los métodos y los hombres debían realizar un proceso de adaptación permanente. Para asegurarse la perennidad, cada organización buscaba gobernar la evolución.

En los servicios de Informática la incidencia del contexto turbulento, fue sin duda, cada vez más poderosa y determinante. Habían más peticiones de información, ya que la información es el elemento estratégico por excelencia para gobernar la evolución.

- Los equipos de estudio y desarrollo debían equiparse para asegurar una mejor reacción, más eficaz en las tareas de mantenimiento (garantía de funcionamiento de las aplicaciones, su evolución, su mejora) de los Sistemas de Información.
- Además el progreso tecnológico se aceleraba. El aumento de la potencia de los micro-ordenadores, la aparición y consolidación de las redes, los Sistemas de Gestión de Base de Datos Relacionadas y Distribuidas, llevaban consigo una evolución mayor de la Arquitectura de los Sistemas de Información. Era imperativo construirlos de forma sencilla y adaptable a las novedades tecnológicas de aquella época.
- La utilización óptima de los recursos humanos (analistas, conceptores, responsables de mantenimiento, usuarios finales) pasaba por una libre circulación de ideas y experiencias. La única forma de adaptarse en permanencia a las necesidades de la empresa, obligaba a utilizar las competencias de cada uno de forma flexible, y esto solo se produce eficazmente por la adopción generalizada de normas estándares, es decir, por la racionalización del trabajo.
- En estas condiciones, la aportación de la tecnología CASE (Ingeniería de Software Asistida por Computadora) reviste un carácter vital para las empresas. Por la que

definimos como la utilización de procedimientos y herramientas para construir sistemas de información.

El desarrollo de esa utilización disciplinada se ha producido en tres ejes fundamentales:

- Herramientas de concepción y de análisis sobre micro-ordenadores destinados a modelizar los sistemas de información y que están en el origen del término CASE.
- Diccionarios de Datos, o bases de reglamentación, frecuentemente ligados a un sistema de Gestión de Base de Datos particular.
- Generadores de Programas, pocos de estos sistemas pueden tomar a su cargo la totalidad de la cadena de producción de programas, desde su concepción hasta el mantenimiento evolutivo de 5 a 10 años de duración, al que antes hemos hecho referencia. En la generalidad de los casos, se trata de cajas de herramientas que proporcionan soluciones parecidas a las necesidades del usuario final. Si algunos generadores de lenguajes de cuarta generación permiten producir rápidamente programas, no pueden sin embargo, construir aplicaciones de la dimensión necesaria e indispensable para la gestión de la empresa moderna.

Toda esta tecnología CASE ha llegado a nuestro país en la década de los 80s. Pero ha llegado de repente con la siguiente confusión producida por una mala digestión del nuevo término.

Los esfuerzos en el uso de **CASE** han mostrado muy poca productividad inmediata debido a la falta de educación requerida. De cualquier modo, la calidad del desarrollo de sistemas **CASE** ha sido mayor que la esperada. El desarrollo de sistemas con **CASE** tiende a tener pocos errores de análisis y diseño y las pruebas al sistema toman mucho menos tiempo.

Además el mantenimiento de sistemas ha mostrado reducciones significativas debido a la habilidad de hacer cambios para diseñar en lugar de hacer código. Se espera que la productividad de desarrollo se incremente eventualmente, cuando los desarrolladores se acostumbren a usar las herramientas.

¡Lo primero que se debe hacer es elegir una metodología!

La parte más importante de cualquier herramienta **CASE** es su metodología de desarrollo. Si los desarrolladores no están siguiendo estrictamente una metodología, las herramientas automatizadas no ayudarán mucho.

Varias compañías están esperando al **CASE** perfecto ya que ellos no pueden decidir que herramienta comprar. Sin embargo debido a que los productos **CASE** están evolucionando constantemente, es muy difícil escoger la herramienta óptima, no así la metodología de desarrollo. Por lo tanto es recomendable que las herramientas y metodologías se seleccionen con plena seguridad de que eso es lo que realmente se necesita.

CASE es una tecnología relativamente nueva, y hay diferentes puntos de vista acerca de cuando y donde está mejor empleada.

La idea básica del CASE (Ingeniería de Software Asistida por Computadora) es la de apoyar cada fase del ciclo de desarrollo con un conjunto de herramientas que ahorren tiempo y dinero. Algunas herramientas CASE se concentran en apoyar fases iniciales del ciclo de desarrollo.

Gracias a los avances tecnológicos hoy en día se cuenta con herramientas que nos permiten reducir costos de mantenimiento, mejorar la calidad del software, acelerar el

proceso de desarrollo, hacer prácticas las técnicas estructurales y de objetos, aumentar la productividad a través de la automatización de determinadas tareas como la generación de códigos y la reutilización de objetos o módulos

Es importante resaltar que las herramientas actuales permiten generar objetos modelo "estático" y modelo "funcional", más no el modelo "dinámico".

1.2 DEFINICIÓN DE LAS HERRAMIENTAS CASE

CASE es una filosofía que se orienta a la mejor comprensión de los modelos de empresa, sus actividades y el desarrollo de los sistemas de información. Esta filosofía involucra además el uso de programas que permiten construir los modelos que describen la empresa, visualizar el medio en el que se realizan las actividades, llevar a cabo la planificación, el desarrollo del Sistema Informático, desde la planificación, pasando por el análisis y diseño de sistemas, hasta la generación del código de los programas y la documentación.

Las herramientas CASE son un complemento de la caja de herramientas del ingeniero de software . Las herramientas CASE proporcionan al ingeniero la posibilidad de automatizar actividades manuales y de mejorar su visión general de la Ingeniería.

Las herramientas CASE abarcan todos los pasos del proceso de software, y también aquellas actividades generales que se aplican a lo largo de todo el proceso. CASE combina un conjunto de bloques de construcción que comienzan en el nivel del hardware y del software de sistema operativo y finaliza en las herramienta individuales.

En el contexto CASE se entiende por enciclopedia a la base de datos que contiene informaciones relacionadas con las especificaciones, análisis y diseño del software. En está base de datos se incluyen las informaciones de DATOS GRAFICOS y REGLAS.

Las herramientas CASE permiten aumentar la productividad de las áreas de desarrollo y mantenimiento de los sistemas informáticos. Mejorar la calidad del software desarrollado, reducir tiempos, costes de desarrollo y mantenimiento del software. Mejorar la gestión y dominio sobre el proyecto en cuanto a su planificación, ejecución y control. Mejorar el archivo de datos (enciclopedia) de conocimientos (know-how) y sus facilidades de uso, reduciendo la dependencia de analistas y programadores. Automatizar el desarrollo del software, la documentación la generación del código, el chequeo de errores, la gestión del proyecto.

Permitir la reutilización del software, la portabilidad del software, la estandarización de la documentación. Integrar las fases de desarrollo (ingeniería del software) . Facilitar la utilización de las distintas metodologías que desarrolla la propia ingeniería del software.

Las HERRAMIENTAS CASE son un conjunto de métodos, utilidades técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases.

El empleo de herramientas CASE permiten integrar el proceso de ciclo de vida :

- Análisis de datos y procesos integrados mediante un repositorio.
- Generación de interfaces entre el análisis y el diseño.

- Generación del código a partir del diseño.
- Control de mantenimiento.

En la figura 1.1 se representa el Ciclo de vida de un Sistema

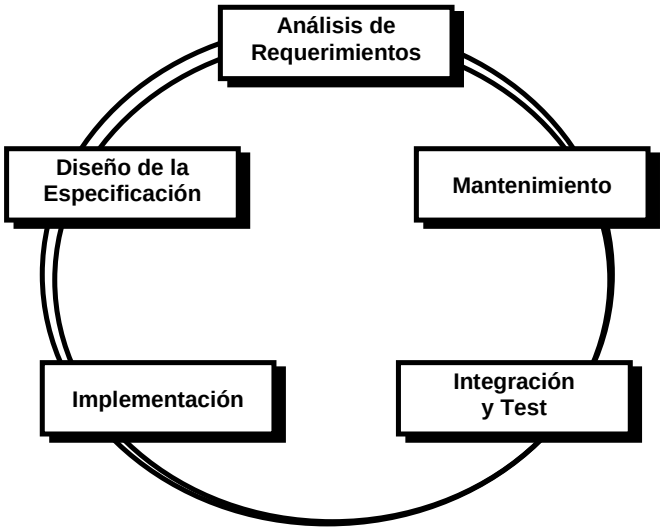


Figura 1.1 Ciclo de Vida de un Sistema

Actualmente, la tendencia en el desarrollo de software está enfocada hacia las microcomputadoras como plataformas de ingeniería de software, que se interconectan mediante redes para que puedan comunicarse de forma efectiva. La base de datos del proyecto (también denominada biblioteca del proyecto o depósito de software), está disponible a través de un servidor de archivos en red que es accesible desde todas las estaciones de trabajo. Un sistema operativo que gestiona el hardware, la red y las herramientas, mantiene todo el entorno unido.

La arquitectura de entorno, compuesta por la plataforma hardware y el soporte del sistema operativo (incluida la red y la gestión de la base de datos), constituye la base del CASE. Pero el entorno CASE, en sí mismo, necesita otros componentes. Un conjunto de servicios de portabilidad constituyen un puente entre las herramientas CASE y su marco de

integración y la arquitectura de entorno. El marco de integración es un conjunto de programas especializados que permite a cada herramienta CASE comunicarse con las demás, para crear una base de datos de proyectos y mostrar una apariencia homogénea al usuario final (el ingeniero de software). Los servicios de portabilidad permiten que las herramientas CASE y su marco de integración puedan migrar a través de diferentes plataformas hardware y sistemas operativos, sin grandes esfuerzos de adaptación.

Las herramientas CASE son una combinación de herramientas software (aplicaciones) y de metodologías de desarrollo :

- Las herramientas permiten automatizar el proceso de desarrollo del software.
- Las metodologías definen los procesos automatizar.

1.3 COMPONENTES Y FUNCIONALIDAD DE LAS HERRAMIENTAS CASE

1.3.1 Repositorio

Es la base de datos central de una herramienta CASE. El repositorio amplía el concepto de diccionario de datos para incluir toda la información que se va generando a lo largo del ciclo de vida del sistema, como por ejemplo: componentes de análisis y diseño (diagramas de flujo de datos, diagramas entidad - relación, esquemas de bases de datos, diseños de pantallas), estructuras de programas, algoritmos, etc. En algunas referencias se le denomina Diccionario de Recursos de Información.

Las características más importantes de un repositorio son:

- **Tipo de información.** Que contiene alguna metodología concreta, datos, gráficos, procesos, informes, modelos o reglas.
- **Tipo de controles.** Si incorpora algún módulo de gestión de cambios, de mantenimiento de versiones, de acceso por clave, de redundancia de la información. La gestión de cambios y el mantenimiento de versiones, ayudarán en el caso de que convivan diferentes versiones de la misma aplicación o se tengan que realizar cambios en la versión en producción y en la de desarrollo, simultáneamente.
- **Tipo de actualización.** Si los cambios en los elementos de análisis o diseño se ven reflejados en el repositorio en tiempo real o mediante un proceso por lotes (batch). Esto será importante en función a la necesidad de que los cambios sean visibles por todos los usuarios, en el acto.
- **Reutilización de módulos para otros diseños.** El repositorio es la clave para identificar, localizar y extraer código para su reutilización.
- **Posibilidad de exportación e importación** para extraer información del repositorio y tratarla con otra herramienta (formateo de documentos, mejora de presentación) o incorporar al repositorio, información generada por otros medios.
- **Interfaces automáticas con otros repositorios** o bases de datos externos.

1.3.2 Módulos de diagramación y modelización

Algunos de los diagramas y modelos utilizados con mayor frecuencia son:

- Diagrama de flujo de datos.
- Modelo entidad - interrelación.
- Historia de la vida de las entidades.
- Diagrama Estructura de datos.
- Diagrama Estructura de cuadros.

- Técnicas matriciales.

Algunas características referentes a los diagramas son:

- **Número máximo de niveles** para poder soportar diseños complejos.
- **Número máximo de objetos** que se pueden incluir para no encontrarse limitado en el diseño de grandes aplicaciones.
- **Número de diagramas distintos en pantalla** o al mismo tiempo en diferentes ventanas.
- **Dibujos en formato libre** con la finalidad de añadir comentarios, dibujos, información adicional para aclarar algún punto concreto del diseño.
- **Actualización del repositorio por cambios en los diagramas.** Siempre resulta más fácil modificar de forma gráfica un diseño y que los cambios queden reflejados en el repositorio.
- **Control sobre el tamaño, fuente y emplazamiento de los textos** en el diagrama.
- **Comparaciones entre gráficos de distintas versiones.** De esta forma será más fácil identificar qué diferencias existen entre las versiones.
- **Inclusión de pseudocódigo** que servirá de base a los programadores para completar el desarrollo de la aplicación.
- **Posibilidad de deshacer el último cambio** facilitando que un error no conlleve perder el trabajo realizado.

1.3.3 Generador de código

Normalmente, se suele utilizar sobre ordenadores personales o estaciones de trabajo, por lo que el paso posterior del código al host puede traer problemas, al tener que compilar en ambos entornos.

Las características más importantes de los generadores de código son:

- **Lenguaje generado.** Si se trata de un lenguaje estándar o un lenguaje propietario.
- **Portabilidad del código generado.** Capacidad para poder ejecutarlo en diferentes plataformas físicas y/o lógicas.
- **Generación del esqueleto del programa o del programa completo.** Si únicamente genera el esqueleto será necesario completar el resto mediante programación.
- **Posibilidad de modificación del código generado.** Suele ser necesario acceder directamente al código generado para optimizarlo o completarlo.
- **Generación del código asociado a las pantallas e informes de la aplicación.** Mediante esta característica se obtendrá la interface de usuario de la aplicación.

1.3.4 Módulo generador de documentación.

El módulo generador de la documentación se alimenta del repositorio para transcribir las especificaciones allí contenidas.

Algunas características de los generadores de documentación son:

- **Generación automática** a partir de los datos del repositorio, sin necesidad de un esfuerzo adicional.
- **Combinación de información textual y gráfica,** lo que hace más fácil su comprensión.
- **Generación de referencias cruzadas.** Con ello se podrá localizar fácilmente en qué partes de la aplicación se encuentra un determinado objeto o elemento, con el fin de analizar el impacto de un cambio o identificar los módulos afectados por un determinado error.

- **Ayuda de tratamiento de textos.** Facilidad para la introducción de textos complementarios a la documentación que se genera de forma automática.
- **Interface con otras herramientas:** procesadores de textos, editores gráficos, etc.

1.4 BLOQUES BASICOS DE UNA HERRAMIENTA CASE

La ingeniería del software asistida por computadora puede ser tan sencilla como herramienta que preste su apoyo para una única actividad de ingeniería de software. O bien puede ser compleja como todo entorno que abarque herramientas, una base de datos, personas, hardware, una red, sistemas operativos, estándares y otros mil componentes mas.

Cada bloque de construcción forma un fundamento para el siguiente, estando las herramientas situadas en la parte superior del montón. Es importante tener en cuenta que el fundamento de entornos CASE efectivos tiene relativamente poco que ver con las herramientas de ingeniería del software en sí. Mas bien, los entornos que tienen éxito para la ingeniería del software se construyen una arquitectura de entorno que abarca un hardware adecuado y un software de sistema adecuado.

Las arquitecturas del entorno, que constan de una plataforma hardware y de un apoyo de sistema operativo (incluyendo el software de red y de gestión de la base de datos), constituyen los fundamentos de CASE. Pero el entorno CASE en sí requiere otros bloques de construcción. Existe un conjunto de servicios de portabilidad que proporciona un puente entre las herramientas CASE y su *marco de referencia de integración* y la arquitectura del entorno. El marco de referencia de integración es una colección de programas especializados que capacitan a las herramientas CASE individuales para comunicarse entre sí, para crear una base de datos del proyecto, y para mostrar el mismo aspecto al usuario final (ingeniero del software). Los *servicios de portabilidad* permiten

que las herramientas CASE y su marco de referencia de integración migren entre distintas plataformas del hardware y el sistema operativo sin un mantenimiento que resulten significativo.

E n la figura 1.2 se representa los bloques de construcción de una herramienta CASE

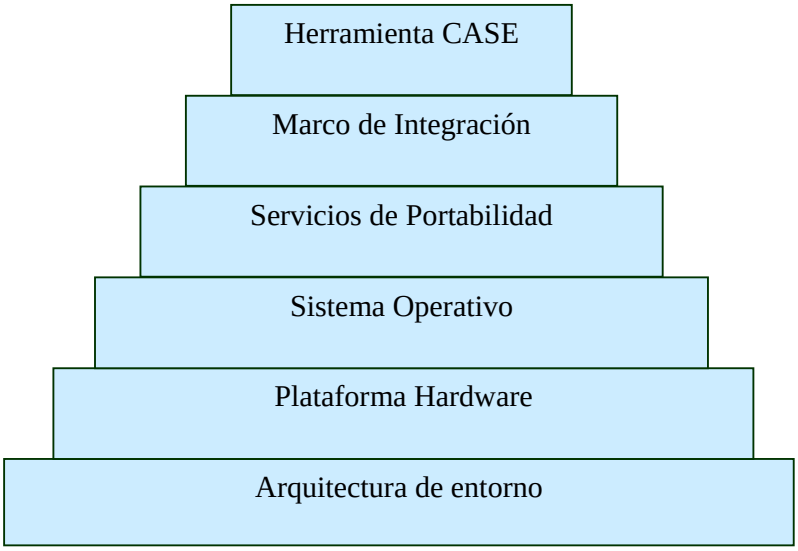


Figura 1. 2 Bloques de Construcción de una herramienta CASE

1.5 CLASIFICACIÓN DE LAS HERRAMIENTAS CASE

No existe una única clasificación de herramientas CASE y, en ocasiones, es difícil incluirlas en una clase determinada.

Podrían clasificarse atendiendo a:

- Las plataformas que soportan.
- Las fases del ciclo de vida del desarrollo de sistemas que cubren.
- La arquitectura de las aplicaciones que producen.

- Su funcionalidad.

Las herramientas CASE, en función de las fases del ciclo de vida abarcadas, se pueden agrupar de la forma siguiente:

1.5.1 Herramientas integradas

- **I-CASE** (Integrated CASE, CASE integrado): abarcan todas las fases del ciclo de vida del desarrollo de sistemas. Son llamadas también CASE workbench. Tienen un repositorio y aportan técnicas estructuradas para todas las fases del ciclo de vida. Estas son las características que les confieren su mayor ventaja: una mejora de la calidad de los desarrollos. Sin embargo, no todas ellas son modernas en el sentido de aprovechar la potencia de las estaciones de trabajo o la utilización de lenguajes de alto nivel o técnicas de prototipo.

1.5.2 Herramientas que comprenden algunas fases del ciclo de vida de desarrollo de software

- **Herramientas de alto nivel, U-CASE** (Upper CASE - CASE superior) o front-end, orientadas a la automatización y soporte de las actividades desarrolladas durante las primeras fases del desarrollo: análisis y diseño. Una estrategia posible es utilizar una U-CASE para análisis y diseño, combinada con otras herramientas más modernas para las fases de construcción y pruebas. En este caso, habría que vigilar cuidadosamente la integración entre las distintas herramientas.
- **Herramientas de bajo nivel, L-CASE** (Lower CASE - CASE inferior) o back-end, dirigidas a las últimas fases del desarrollo: construcción e implantación.

- **Juegos de herramientas o toolkits**, son el tipo más simple de herramientas CASE. Automatizan una fase dentro del ciclo de vida. Dentro de este grupo se encontrarían las herramientas de reingeniería, orientadas a la fase de mantenimiento.

Otra posible clasificación, utilizando la funcionalidad como criterio principal, es la siguiente:

1.5.3 Herramientas de planificación de sistemas de gestión

Sirven para modelar los requisitos de información estratégica de una organización. Proporcionan un "metamodelo" del cual se pueden obtener sistemas de información específicos. Su objetivo principal es ayudar a comprender mejor cómo se mueve la información entre las distintas unidades organizativas. Estas herramientas proporcionan una ayuda importante cuando se diseñan nuevas estrategias para los sistemas de información y cuando los métodos y sistemas actuales no satisfacen las necesidades de la organización.

1.5.4 Herramientas de análisis y diseño.

Permiten al desarrollador crear un modelo del sistema que se va a construir y también la evaluación de la validez y consistencia de este modelo. Proporcionan un grado de confianza en la representación del análisis y ayudan a eliminar errores con anticipación. Se tienen:

- Herramientas de análisis y diseño (Modelamiento).
- Herramientas de creación de prototipos y de simulación.
- Herramientas para el diseño y desarrollo de interfaces.
- Máquinas de análisis y diseño (Modelamiento).

1.5.5 Herramientas de programación.

Se engloban aquí los compiladores, los editores y los depuradores de los lenguajes de programación convencionales. Ejemplos de estas herramientas son:

- Herramientas de codificación convencionales.
- Herramientas de codificación de cuarta generación.
- Herramientas de programación orientadas a los objetos.

1.5.6 Herramientas de integración y prueba

Sirven de ayuda a la adquisición, medición, simulación y prueba de los equipos lógicos desarrollados. Entre las más utilizadas están:

- Herramientas de análisis estático.
- Herramientas de codificación de cuarta generación.
- Herramientas de programación orientadas a los objetos.

1.5.7 Herramientas de gestión de prototipos

Los prototipos son utilizados ampliamente en el desarrollo de aplicaciones, para la evaluación de especificaciones de un sistema de información, o para un mejor entendimiento de cómo los requisitos de un sistema de información se ajustan a los objetivos perseguidos.

1.5.8 Herramientas de mantenimiento

La categoría de herramientas de mantenimiento se puede subdividir en:

- Herramientas de ingeniería inversa.
- Herramientas de reestructuración y análisis de código.
- Herramientas de reingeniería.

1.5.9 Herramientas de gestión de proyectos

La mayoría de las herramientas CASE de gestión de proyectos, se centran en un elemento específico de la gestión del proyecto, en lugar de proporcionar un soporte global para la actividad de gestión. Utilizando un conjunto seleccionado de las mismas se puede: realizar estimaciones de esfuerzo, coste y duración, hacer un seguimiento continuo del proyecto, estimar la productividad y la calidad, etc. Existen también herramientas que permiten al comprador del desarrollo de un sistema, hacer un seguimiento que va desde los requisitos del pliego de prescripciones técnicas inicial, hasta el trabajo de desarrollo que convierte estos requisitos en un producto final. Se incluyen dentro de las herramientas de control de proyectos las siguientes:

- Herramientas de planificación de proyectos.
- Herramientas de seguimiento de requisitos.
- Herramientas de gestión y medida.

1.5.10 Herramientas de soporte

Se engloban en esta categoría las herramientas que recogen las actividades aplicables en todo el proceso de desarrollo, como las que se relacionan a continuación:

- Herramientas de documentación.
- Herramientas para software de sistemas.
- Herramientas de control de calidad.
- Herramientas de bases de datos.

1.6 OPCIONES DE INTEGRACION DE LAS HERRAMIENTAS CASE

Las herramientas CASE pueden ser integradas de muchas formas. En un extremo se utiliza una herramienta CASE de forma aislada. Se crea un número limitado de elementos de configuración de software (documentos, programas o datos) que se manipulan mediante una única herramienta y cuya salida tiene el formato de copia de pantalla y/o documentación gráfica. En cierto sentido, el enlace con el resto del entorno de desarrollo se realiza mediante copias en papel que gestiona el ingeniero.

Pocas herramientas CASE se utilizan en forma aislada. Se suele disponer de las siguientes opciones :

1.6.1 Intercambio de datos

La mayoría de las herramientas permiten exportar datos en forma de archivo sin estructura con un formato conocido. Esto permite un intercambio de datos punto a punto entre las distintas herramientas CASE, utilizando normalmente un "filtro" de transmisión intermedio como lo muestra la figura 1.3

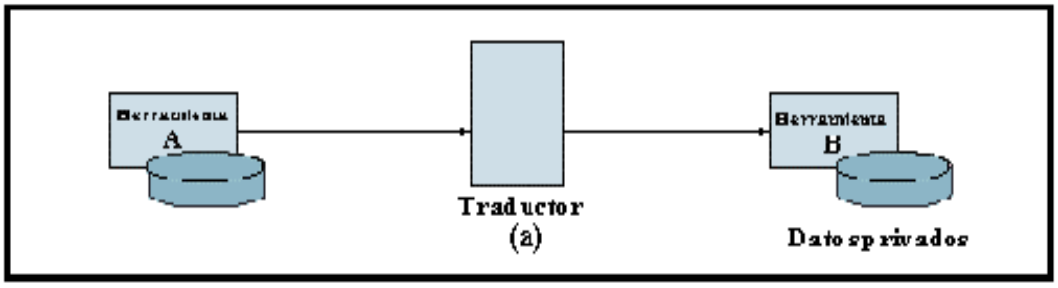


Figura 1.3 Intercambio de Datos

La desventaja del intercambio de datos punto a punto está en que, a menudo, sólo parte de los datos exportados es utilizable por la herramienta receptora, ya que no fue diseñada para ser totalmente compatible. Además, a medida que evoluciona el software, la necesidad de transferir archivos cada vez que se hace un cambio pequeño puede llevar mucho tiempo. Las versiones pueden quedar "desfasadas" fácilmente, perdiéndose la posibilidad de transferencia, la cual suele ser en un único sentido. No hay posibilidad de que los cambios se reflejen en ambos sentidos y, es difícil hacer comprobaciones cruzadas de documentos y mantener la integridad de la configuración a través de las distintas herramientas que se estén utilizando.

1.6.2 Acceso común a herramientas

Permite al usuario utilizar distintas herramientas de forma similar, por ejemplo a través de un menú desplegable del gestor de ventanas del sistema operativo. En un entorno multitarea, un usuario podría abrir simultáneamente varias herramientas, coordinando manualmente sus entradas y comparando las representaciones de diseño a medida que evolucionan(ver figura 1.4). Por ejemplo, el usuario podría visualizar un diagrama de flujo de datos, un diagrama de estructura, un diccionario de datos y un segmento de código fuente, todos mantenidos por diferentes herramientas.

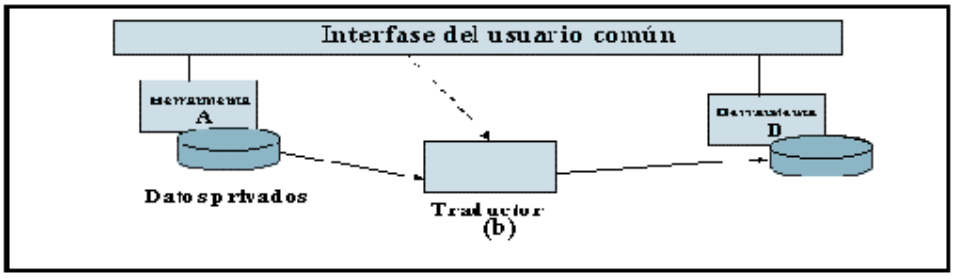


Figura 1.4 Acceso Común a una Herramienta

Patricia Carrillo S.
P.U.C.E.S.A

1.6.3 Integración de Datos

Gestión común de datos. Los datos de distintas herramientas se pueden mantener en una única base de datos lógica, que puede estar físicamente centralizada o distribuida. Hay una modalidad de fusión que permite combinar el trabajo de varias personas trabajando en diferentes partes de una aplicación. Aunque los datos generados por las distintas herramientas se gestionan de forma conjunta en el nivel de gestión de datos comunes, las herramientas no conocen de forma explícita las estructuras de datos y la semántica de representación del diseño de las demás. Consecuentemente, se requiere una etapa de traducción (normalmente ejecutada manualmente) para permitir que una herramienta utilice la salida generada por otra.

Datos compartidos. Las herramientas del nivel de datos compartidos tienen estructuras de datos y semántica compatible, pudiendo intercambiar datos sin necesidad de una etapa de traducción. Cada herramienta se diseña para ser compatible con las demás. Por esta razón, la mayor parte del intercambio de datos se da entre herramientas de un único fabricante o en casos en los que se han establecido relaciones estratégicas, entre distintos fabricantes para generar un conjunto de datos integrado, a veces, a petición de clientes importantes.

Interoperabilidad. Las herramientas que combinan las características de acceso común y la capacidad de compartir datos, tienen la capacidad de interoperación.

Esto representa el mayor nivel de integración entre herramientas diferentes. Sin embargo, hay otras propiedades del entorno global CASE que se pueden añadir para mejorar la efectividad del proceso de desarrollo de software.

En la figura 1.5 se representa la Integración de datos

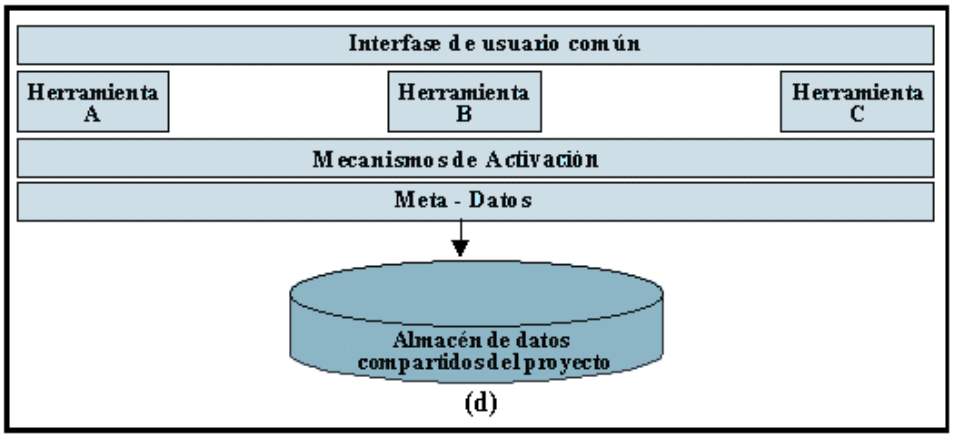


Figura 1.5 Integración de Datos

1.6.4 Integración total

Para alcanzar la integración total del entorno CASE se necesitan dos características más: gestión de metadatos y capacidad de control. Los metadatos representan información sobre los datos de ingeniería generados por las distintas herramientas CASE. Esta información incluye:

- Definiciones de objetos (tipos, atributos, representaciones y relaciones válidas).
- Relaciones y dependencias entre objetos de granularidad arbitraria (por ejemplo un proceso en un diagrama DFD, una entidad única o un fragmento de código de una subrutina).
- Reglas de diseño del software (p. ejemplo: las distintas formas válidas de dibujar y equilibrar un diagrama de flujo de datos).
- Procedimientos (fases estándar, hitos, informes, etc.) y sucesos (revisiones, finalizaciones, informes de problemas, peticiones de cambios, etc.) del flujo de trabajo (proceso).

Normalmente, la parte de reglas y procedimientos de los metadatos se definen en forma de base de reglas, para facilitar su modificación según evoluciona el proceso de desarrollo del software. Por ejemplo, un nuevo método de diseño podría alterar las reglas de representación y cambiar los estándares del proceso de trabajo seguido hasta el momento.

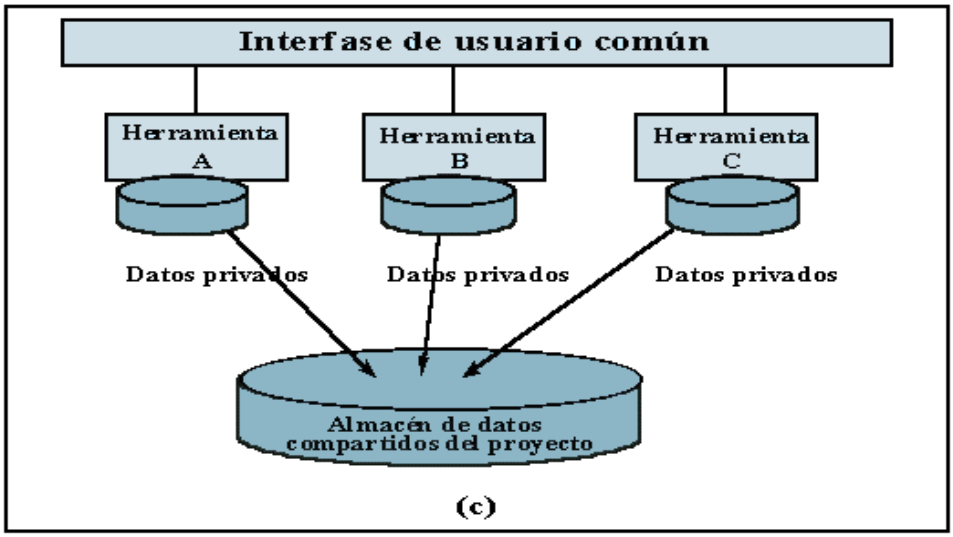
La capacidad de control permite que cada herramienta pueda notificar al resto del entorno (a otras herramientas, al gestor de metadatos, al gestor de datos, etc.) la ocurrencia de sucesos significativos, así como enviar peticiones para la realización de acciones a otras herramientas y servicios por medio de un activador. Por ejemplo, una herramienta de gestión de configuración que haga una comprobación cruzada de la consistencia de documentos. La capacidad de control ayudará a mantener la integridad del entorno y proporcionará, también, un medio para automatizar procesos y procedimientos estándar. El activador puede estar incorporado en un entorno cerrado o puede estar visible para las distintas herramientas, a través de una interface de programación y un mecanismo de paso de mensajes.

La tecnología CASE tendrá el mayor impacto si se integra a proyectos de innovación tecnológica que hoy en día contemple:

- Interfaces de programación visual.
- Soluciones cliente-servidor.
- Manejo de múltiples Bases de Datos.
- Independencia de la plataforma de hardware y software.
- Reingeniería de proceso de negocios.

En la figura 1.6 se muestra un esquema de Integración Total

Figura 1.6 Esquema Integración total



1.7 TIPOS DE HERRAMIENTAS CASE

- Herramientas de Seguimiento de Requisitos
- Herramientas de Métricas
- Herramientas de Documentación
- Herramientas para Software de Sistemas.
- Herramientas de Gestión de Base de Datos
- Herramientas de Bases de Datos y de Configuración de Software.
- Herramientas Pro/Sim
- Herramientas para el Diseño y Desarrollo de Interfaces
- Herramientas de Codificación Convencionales
- Herramientas de Codificación de Cuarta Generación

- Herramientas de Programación Orientadas a Objetos

1.8 CARACTERÍSTICAS DE LAS DIFERENTES HERRAMIENTAS CASE

1.8.1 HERRAMIENTAS DE SEGUIMIENTO DE REQUISITOS

La característica principal de esta herramienta es proporcionar un enfoque sistemático para el aislamiento de requisitos, comenzando por la solicitud del cliente de una propuesta o especificaciones.

1.8.2 HERRAMIENTAS DE MÉTRICAS

Las métricas de software mejoran la capacidad del administrador para controlar y coordinar el proceso del software y la capacidad del ingeniero para mejorar la calidad del software que se produce. La métricas y herramientas de medida actuales se concentran en procesos, proyectos y características del producto.

1.8.3 HERRAMIENTAS DE DOCUMENTACIÓN

Las herramientas de producción de documentación y autoedición se utilizan en casi todos los aspectos de la ingeniería del software y representan una oportunidad muy interesante para todos los que desarrollan software. No es raro que una empresa emplee el 20 o el 30 por ciento de su esfuerzo de desarrollo en la documentación. Por esta razón, estas herramientas constituyen una opción importante para aumentar la productividad.

Las herramientas de documentación suelen estar unidas a otras herramientas CASE por medio de una interfaz de datos suministrada por el vendedor. Por ejemplo, muchas herramientas de análisis y diseño están unidas a uno o varios sistemas de autoedición, de tal forma que los modelos y textos creados durante el análisis y el diseño puedan ser

transmitidos a una herramienta de documentación y añadidos a la especificación creada utilizando la misma herramienta de documentación.

1.8.4 HERRAMIENTAS PARA SOFTWARE DE SISTEMAS

El CASE es una tecnología de estaciones de trabajo. Por esto, el entorno CASE debe soportar software de redes de comunicación de alta calidad, correo electrónico, boletines electrónicos y otras posibilidades de comunicación. Aunque el sistema operativo más empleado en las estaciones de trabajo de ingeniería es el WINDOWS.

1.8.5 HERRAMIENTAS DE GESTION DE BASES DE DATOS

Esta herramienta sirve como fundamento para establecer una base de datos CASE (depósito), que también se denominará base de datos del proyecto. Dado el énfasis acerca de los objetos de configuración, las herramientas de gestión de bases de datos para CASE puede evolucionar a partir de los sistemas de gestión de bases de datos relacionales(SGBDR) para transformarse en sistema de gestión de bases de datos orientada a objetos(SGBDOO).

1.8.6 HERRAMIENTAS DE BASES DE DATOS Y DE CONFIGURACIÓN DE SOFTWARE

El software de gestión de bases de datos sirve como base para el establecimiento de una base de datos CASE (almacén). Poniendo énfasis en los objetos de la configuración, las herramientas de gestión de bases de datos para CASE pueden evolucionar de los sistemas relacionales a los sistemas basados en objetos.

Las herramientas CASE pueden ayudar en las cinco tareas principales de la configuración del software, identificación y control de versiones, control de cambios, auditoria y gestión de estados. Las base de datos CASE proporciona un mecanismo para identificar cada elemento de la configuración y relacionarlo con otros elementos.

La utilización de bases de datos, herramientas de gestión y configuración y herramientas de inspección de componentes, es el primer paso hacia una biblioteca de software que estimulará la reutilización de componentes de software.

1.8.7 HERRAMIENTAS PRO/SIM

Las herramientas de creación de prototipos y de simulación (PRO/SIM) proporcionan al ingeniero de software la capacidad de predecir el comportamiento de un sistema de tiempo real antes de que sea construido. Además, le permiten desarrollar prototipos de sistemas de tiempo real que proporcionen al cliente una visión general de la función, de la operación y de la respuesta, antes de la codificación final.

Muchas herramientas PRO/SIM tienen la capacidad de generar código para Ada y para muchos otros lenguajes de programación que se harán cada vez más sofisticados a medida que estas herramientas evolucionen.

1.8.8 HERRAMIENTAS PARA EL DISEÑO Y DESARROLLO DE INTERFACES

Las herramientas de diseño y desarrollo de interfaces son, en realidad un conjunto de componentes de software, tales como menús, botones, estructuras de ventanas, iconos, mecanismos de visualización, controladores de dispositivos y otros elementos de este tipo. Sin embargo, estos conjuntos de herramientas están siendo reemplazados por herramientas

para desarrollar prototipos que permiten la creación rápida en pantalla de interfaces sofisticadas ajustadas al estándar elegido para el software.

1.8.9 HERRAMIENTAS DE CODIFICACIÓN CONVENCIONALES

Hace tiempo, las únicas herramientas de las que disponía un ingeniero de software eran las herramientas de codificación convencionales.

Hoy día, las herramientas convencionales siguen existiendo en primera línea de desarrollo del software, pero están respaldadas por todas las otras herramientas CASE.

1.8.10 HERRAMIENTAS DE CODIFICACIÓN DE CUARTA GENERACIÓN

La tendencia hacia la representación de aplicaciones de software en niveles más altos de abstracción ha hecho que muchos diseñadores utilicen herramientas de codificación de cuarta generación. Los sistemas de consulta a bases de datos, los generadores de código y los lenguajes de cuarta generación han cambiado la forma en que se desarrollan los sistemas. No hay duda de que el objetivo final del CASE es la generación automática de código, estos es, la representación de sistemas a un nivel de abstracción más alto que el de los lenguajes de programación convencionales. Idealmente, estas herramientas de generación de código no sólo traducirán la descripción de un sistema a un programa operativo, sino que también ayudarán a verificar la corrección de las especificaciones del sistema, de tal forma que la salida resultante satisfaga los requisitos del usuario.

1.8.11 HERRAMIENTAS DE PROGRAMACIÓN ORIENTADAS A OBJETOS

La programación orientada a los objetos es una de las tecnologías más actuales de la ingeniería del software. Por esta razón, los vendedores de sistemas CASE están lanzando el mercado nuevas herramientas para el desarrollo del software orientado a objetos.

Los entornos de programación orientados a los objetos suelen estar unidos a lenguajes de programación específicos (C++, Eiffel, Objective-C o Smalltalk) Un entorno orientado a objetos típico incorpora características de las interfaces de tercera generación (ratón, ventanas, menús desplegables, operaciones sensibles al contexto, multitarea, etc.) con funciones especializadas como la del "inspector", una función que permite al ingeniero de software examinar todos los objetos contenidos en unas bibliotecas de objetos para determinar si pueden ser reutilizados en la aplicación actual.

1.9 ESTRATEGIAS DE IMPLANTACIÓN DE HERRAMIENTAS CASE

Para realizar la implementación de una herramienta CASE podemos seguir los siguientes pasos:

- Identificar la magnitud de problemas a resolver en la Institución.
- Identificar el nivel estratégico que deben tener los sistemas.
- Evaluar los recursos de hardware y software disponibles en la Institución y el medio.
- Evaluar el nivel del personal.
- Efectuar un estudio de costo-beneficio definiendo metas a lograr.
- Elegir las herramientas apropiadas para la Institución.
- Establecer un programa de capacitación de personal de sistemas y usuarios

- Elegir una aplicación que reúna la mayor parte de los siguientes requisitos:

    - Gran impacto de resultados.
    - Disponibilidad de recursos.
    - Mínimo nivel de riesgos.
    - Máxima colaboración de usuarios.
    - Tamaño reducido de solución.

Se establecerá interfaces de compatibilidad de los nuevos sistemas que deben convivir con los sistemas anteriores.

Una vez que se ha hecho una breve introducción a las herramientas CASE en este capitulo, se hará un estudio comparativo de las herramientas CASE en el siguiente capítulo.

CAPITULO II

ESTUDIO COMPARATIVO DE LAS HERRRAMIENTAS CASE

En el presente trabajo se describen las principales herramientas que ayudan al desarrollo de Sistemas de Información, existentes en la actualidad. También se describe su funcionalidad y las características más relevantes, con la finalidad de ayudar en la elección de la herramienta CASE adecuada.

2.1 BPWIN

La revolución que se está produciendo en el ámbito de la información cambia radicalmente la manera de hacer negocios de las empresas. Competir en la "era Internet" implica evolucionar rápidamente para hacer frente a nuevas oportunidades, riesgos y expectativas más sofisticadas de los clientes. Hoy en día, el cambio constante no es una excepción, sino la norma. Con la creciente complejidad de los procesos eBusiness, necesita una solución que proporcione una visión integrada de las operaciones de la empresa.

La solución de modelación de procesos empresariales de **Bpwin** proporciona el marco para comprender estos procesos, determinando el impacto de eventos empresariales y definiendo la forma de interacción de los procesos con los datos que circulan por la empresa.

BPwin nos va ha permitir documentar de manera clara los elementos más importantes de nuestra organización como que actividades son necesarias, cómo se realizan y qué recursos consumen, lo cual nos proporciona una visión exacta, no solo de qué es lo que hace nuestra organización, sino si lo hace de forma eficiente. BPwin proporciona un marco de trabajo

para poder representar y entender los procesos de negocio, determinando el impacto de los diferentes sucesos y definiendo cómo los procesos interactuan unos con otros mediante flujos de información permitiéndonos identificar actividades poco eficientes o redundantes.

2.1.1 CONCEPTO

BPwin es una potente herramienta utilizada para analizar, registrar y mejorar los procesos empresariales complejos. La modelización de procesos nos ayuda ha entender las relaciones entre las actividades más importantes del sistema que queremos analizar. Estas técnicas se han desarrollado para facilitar la comunicación y la captura de información de los expertos en el dominio objeto de estudio. BPwin integra en una misma herramienta las metodologías ***IDEFO, DataFlow diagraming e IDEF***, integrando tres perspectivas clave para poder cubrir las necesidades tanto de la modelización BPR como de la modelización de sistemas de ingeniería.

Con la modelización de funciones (IDEFO), se analiza sistemáticamente el negocio, centrándose en las tareas (funciones) que se realizan de forma regular, las políticas de control que se utilizan para asegurar que esas tareas se realizan de forma correcta, los recursos (tanto humanos como materiales) que se utilizan para realizarla, los resultados de la tarea (salidas) y las materias primas (entradas) sobre las que la actividad actúa tal como se muestra en la figura 2.1

Figura 2. 1 Funciones IDEFO

Los DFD´s (Data flow) suelen ser utilizados en el diseño de software de ordenador, centrándose en el flujo de información entre las diferentes actividades llegando al detalle de poder describir cómo se deben almacenar los datos para maximizar la velocidad de acceso y minimizar el espacio de almacenamiento.

IDEF3 se centra en un proceso en particular, analizando las tareas que lo involucran. Su principal objetivo es proporcionar a los expertos en el dominio un método estructurado y claro a través del cual poder describir situaciones como una secuencia ordenada de sucesos así como describir cualquier objeto participante.

Mediante **BPwin** se pueden dividir modelos de procesos complejos en partes más fáciles de gestionar, lo que permite a los creadores de las simulaciones centrarse en áreas de interés específicas. Finalmente, estas numerosas perspectivas se deben reconciliar y unificar para proporcionar una visión única y coherente de la empresa.

2.1.2 CARACTERÍSTICAS DE BPWIN

- **Cumple los estándares FIPS del Gobierno de los Estados Unidos. BPwi**n, que se utiliza en 500 empresas Fortune, el Departamento de Defensa y otras agencias del Gobierno de los Estados Unidos, cumple todos los Estándares federales de procesamiento de información (FIPS, Federal Information Processing Standards) para la modelación de procesos.
- **BPwin** automatiza muchas tareas que normalmente se asocian a la creación de modelos de procesos, y proporcionan el rigor semántico necesario para ofrecer resultados correctos y coherentes. El resaltado de los objetos le guía a medida que elabora el modelo, de modo que se eliminan varios errores habituales en la creación de modelos.
- **BPwin** soporta diagramas de barras que proporcionan un mecanismo eficaz para visualizar y optimizar procesos complejos de eBusiness. Dichos diagramas organizan procesos complejos a través de fronteras funcionales y permiten visualizar procesos, funciones y responsabilidades con el flujo correspondiente.
- Mediante el nuevo marco de diccionario, la entrada y gestión de información de los modelos se puede realizar de forma rápida y sencilla. Esta interfaz de hoja de cálculo es de fácil aplicación y proporciona un elegante mecanismo para poblar los modelos, independientemente de si se introducen los datos manualmente o se importan.
- Las estructuras de las empresas influyen en gran medida en la forma de definir y realizar los procesos eBusiness. **BPwin** soporta la definición explícita de funciones que definen y clasifican las tareas o trabajos dentro de un proceso eBusiness. **BPwin** crea gráficos de empresa basándose en funciones definidas por el usuario

2.1.3 COMPONENTES Y SU FUNCIONABILIDAD

BPwin proporciona la coordinación y reutilización integrada para técnicas de modelación de procesos empresariales (IDEF0), de flujos de trabajo (IDEF3) y flujos de datos (DFD).

2.1.3.1 Análisis métrico de costes y rendimiento. **BPwin** ofrece el soporte total para la estimación de costes basada en las actividades, optimizado para el análisis de procesos. La generación de informes exhaustivos y la interfaz bidireccional con herramientas ABC específicas facilitan a las empresas la implementación de una estrategia de gestión basada en actividades.

2.1.3.2 Generador de plantillas de informes. El Generador de plantillas de informes (RTB, Report Template Builder) es un nuevo generador de informes que se suele incluir en **ERwin** y **BPwin** para crear informes y sitios Web. Puede definir plantillas de informes que se pueden aplicar a cualquier modelo. Este enfoque de una única definición que se reutiliza en todos los procesos permite definir y facilitar rápidamente estándares de informes. RTB soporta numerosos formatos, incluyendo RTF, HTML y texto sin formato.

2.1.3.3 Interfaz de simulación. BPwin ofrece una interfaz para software de simulación. La simulación permite estudiar los efectos del cambio de forma dinámica. Permite probar distintos ejemplos antes de la implementación de modo que garantiza la obtención de una solución óptima para las necesidades de una empresa.

2.1.3.4 Abriendo un Editor en BPWIN: Seleccionar del menú Editor la opción Definición del Modelo, los campos que contiene esta pantalla son:

- En el campo “Proyect Name” se coloca el nombre del proyecto que aparecerá en la parte superior del diagrama. Colocar “Curso Procesamiento de Datos”.
- El campo “definición” se debe describir qué representa el modelo y qué constituye.

- El campo “alcance” se debe especificar el alcance del modelo (ignorar esta opción).
- El campo “Puntos. de vista” define desde cual puntos de vista el modelo es definido. Diferentes puntos de vista producen diferentes resultados.
- En el campo status se le asigna un estado seleccionando una de las opciones de los botones de estado (seleccionar working).
- En el campo Time Frame se le asigna un tiempo de construcción seleccionando TOBE.
- En Model Name se coloca el nombre del modelo, debe ser un nombre representativo, claro.
- Seleccione del menú principal la opción Editor, ahora haga click en

Diagram Definition, una vez hecho esto aparecerá una ventana con las siguientes opciones:

- Model Name and Project Name: nombre del modelo y nombre del proyecto (ambos campos son constantes desde el Editor de def. Del Modelo).
- Diagram Name: coloque el nombre del diagrama.
- Author Name: Este campo contiene el autor especificado en el diálogo de def. del modelo.
- C-Name: Número de creación cronológico que se puede utilizar para identificar unívocamente un diagrama y trazar su historia
- Used At: Referencia a otro diagrama relacionado. Este valor se despliega en la parte derecha superior del diagrama.
- Creation and Revition Dates: Son las 2 primeras fechas que se especifican cuando se crea el sistema.

- Page Number: Número de página. El valor se despliega en la parte inferior derecha del borde del diagrama.
- Diagram Text: Descripción textual del diagrama.

2.1.4 BENEFICIOS DE BPWIN:

- **Asegura la eficacia operativa** mediante la evaluación de las operaciones empresariales actuales utilizando potentes herramientas de modelación.
- **Mejora los procesos eBusiness** mediante la formulación y evaluación de respuestas alternativas a las presiones del mercado.
- **Elimina rápidamente actividades no productivas** mediante la comunicación rápida e intuitiva de cambios operativos. Las actividades ineficaces, costosas o redundantes se pueden detectar fácilmente y, como consecuencia, es posible mejorarlas, sustituirlas o eliminarlas de acuerdo con los objetivos de la empresa.
- **Ahorra tiempo y dinero** analizando como los cambios afectarán al negocio, determinando la mejor solución y mostrando la solución a los sistemas de información existentes.

ERWIN

Mediante ERwin, las empresas pueden visualizar estructuras complejas de datos y activos de información de inventarios, así como establecer estándares de todas las empresas para la gestión de datos. Permite automatizar de forma inteligente procesos de diseño y sincronizar el modelo con el diseño de bases de datos. Los modeladores pueden utilizar ERwin para diseñar sistemas de transacción, mercados y almacenes de datos en un entorno integrado. ERwin ayuda a las empresas.

ERwin potencia su integración a través de toda la empresa mediante ModelMart de Computer Associates, un sistema de modelización que permite a los diseñadores de bases de datos, desarrolladores de aplicaciones y usuarios finales, compartir la información de modelos de ERwin.

CONCEPTO

ERwin es una herramienta de diseño de bases de datos que le ayudará a diseñar, generar y mantener aplicaciones de bases de datos de calidad y alto rendimiento para cliente/servidor, *web/intrane*t, así como también aplicaciones de D*ata Warehousin*g.

La herramienta Erwin no solo ayuda a diseñar modelos de datos lógicos, también construye automáticamente estructuras de datos físicos con la información del diagrama.

ERwin le permitirá visualizar la estructura apropiada, elementos clave y diseño optimizado de su base de datos. ERwin genera automáticamente tablas y miles de líneas de procedimientos y código para bases de datos de principales. Cuando el modelo de datos

esta listo para usarse, simplemente se selecciona el servidor donde se quiere construir la base de datos y se eligen las opciones de generación de esquema que se quieran incorporar.

ERwin no es solo una herramienta de diseño de base de datos, sino que es una herramienta de desarrollo de RDBMS que permite generar automáticamente tablas y miles de líneas de código de procedimientos almacenados y triggers de todas RDBMS líderes del mercado. Su funcionalidad de "Complete Compare" permite el desarrollo interactivo manteniendo sincronizado el modelo con la base de datos en todo momento.

Su tecnología de "comparación completa" le permite un desarrollo de tal forma que su modelo esté siempre sincronizado con la base de datos. Mediante su integración con los entornos de desarrollo más potentes, ERwin también acelera la creación de aplicaciones centrada en datos.

2.2.2 CARACTERISTICAS

- Incrementa la productividad proporcionando un entorno gráfico de aplicación fácil que simplifica el diseño de bases de datos y automatiza muchas tareas que requieren mucho tiempo. **ERwin** aumenta la velocidad de creación de bases de datos de transacción y almacenes de datos de alta calidad y rendimiento.
- **ERwin** soporta la definición y el mantenimiento de estándares mediante el diccionario de dominio, el editor de estándares de denominación y el editor de estándares de tipos de datos. El diccionario de dominio contiene atributos reutilizables y asegura la aparición de nombres y definiciones coherentes en todos los diseños de bases de datos. Mediante el editor de estándares de denominaciones el usuario puede crear un glosario de palabras.

- Proporciona la flexibilidad para generar el modelo de datos que satisface las necesidades de la empresa. **ERwin** soporta modelos lógicos y físicos, además del modelo lógico/físico del **ERwin** tradicional. **ERwin** mantiene la información de las relaciones y la historia de todo el diseño y permite que el usuario asimile rápidamente el impacto de los cambios de una capa a la siguiente.
- El diseño físico de una base de datos coincide rara ves con el diseño original de datos lógicos. Las limitaciones empresariales imponen la necesidad de modificar tablas para cumplir los requisitos de rendimiento de las aplicaciones actuales de eBusiness. La tecnología de transformación de **ERwin** permite implementar este tipo de cambios y a la vez mantener la integridad del diseño original.

2.2.3 COMPONENTES Y SU FUNCIONABILIDAD

Entre los componentes de ERwin se tiene :

ModelMart es un sistema de gestión de modelos que permite a los diseñadores de bases de datos, desarrolladores de aplicaciones y usuarios compartir la información de los modelos de ERwin. (ver figura 2.2)

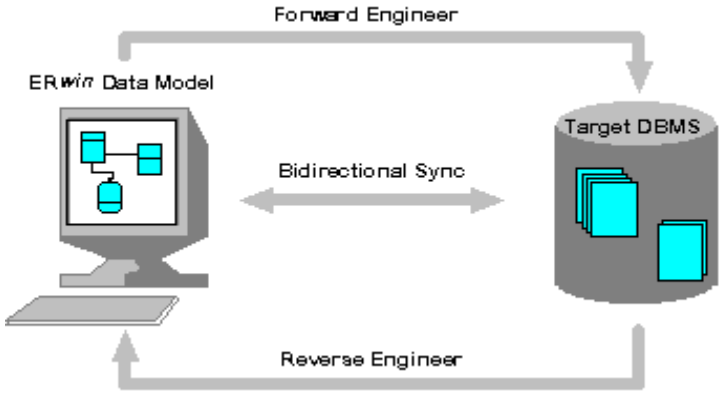
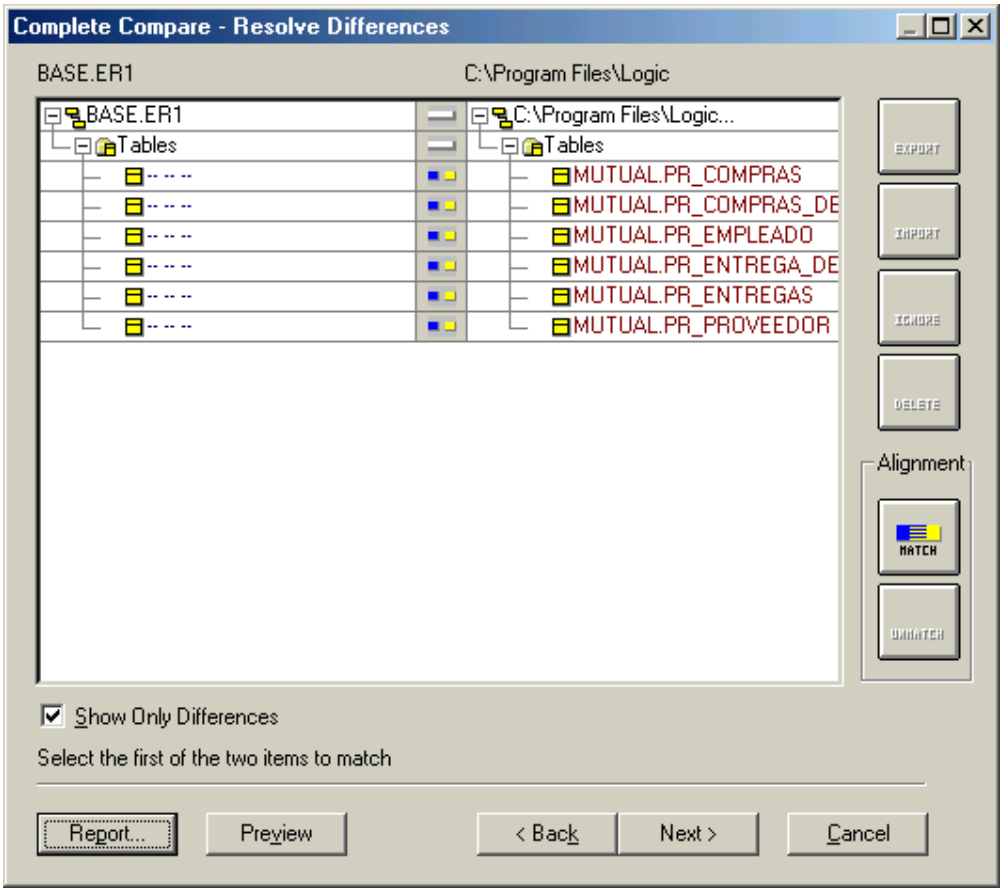


Figura 2.2 Modulo ModelMart de Erwin

La utilidad **"Complete-Compare"** permite que se interactúe con los cambios realizados en la base de datos o en los modelos mediante la comparación de todas las diferencias. ERwin genera automáticamente scripts de alteración para modificar cualquier base de datos preservando los datos.

La utilidad "Complete-Compare" mantiene el modelo y la base de datos sincronizados en todo momento. (VER FIGURA 2.3)

Figura 2.3 Ventana Complete Compare



Erwin combina bases de datos *back-end* y desarrollo de aplicaciones *front-end* en un ambiente unificado. Tiene soporte para multi-clientes, Erwin genera formas de entrada de datos en Visual Basic, DataWindows de Power Builder y PROGESS SmartObjects del

mismo modelo de datos, logrando que los desarrolladores incorporen aplicaciones altamente productivas en tres de los ambientes de desarrollo de bases de datos. (Ver Figura 2.4)

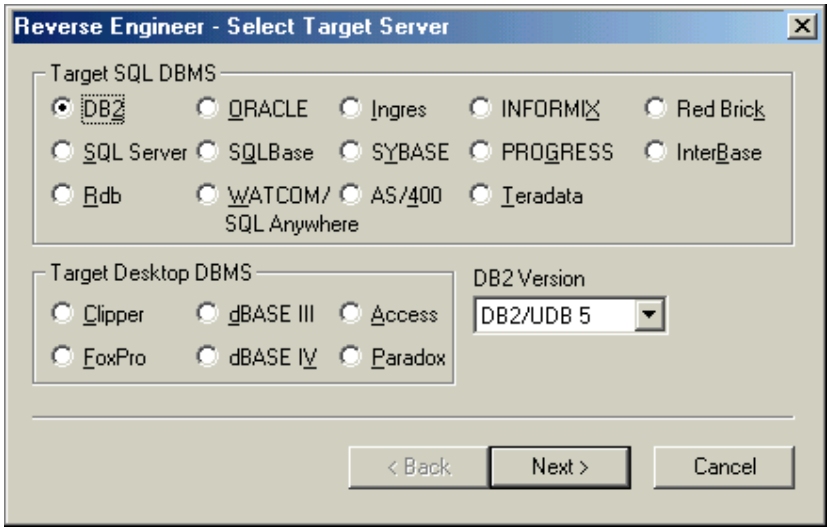


Figura 2.4 Selección de Servidor

Erwin extiende el editor estándar **Column Property Editor** de tal forma que se pueden asignar rápidamente propiedades de columna del lado del cliente, tales como tipo de control por omisión. Despliega formato y reglas de validación de cliente para cada columna y genera formas de entrada de datos en uso y otros componentes de aplicación directamente del mismo modelo Erwin que crea la base de datos *back-en*d (ver figura 2.5)

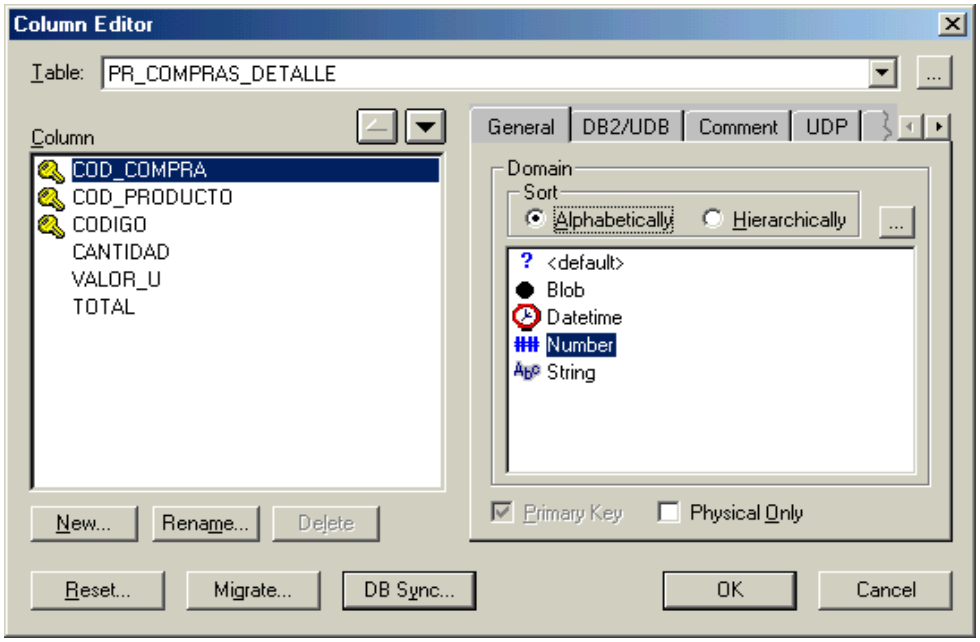


Figura 2.5 Column Editor

Para simplificar aún más el desarrollo de aplicaciones en Visual Basic, Logic Works también ofrece **DataBOT(tm)**, un robot de software avanzado que genera dinámicamente todo el código de acceso de datos SQL requeridos, permitiendo hasta que los programadores novatos creen rápidamente aplicaciones sofisticadas de bases de datos de alto desempeño en los ambientes actuales

2.2.4 BENEFICIOS

- **Se comunica de forma más eficaz** permitiendo que los DBA y desarrolladores compartan y reutilicen modelos, además de poder representar los activos amplios y complejos de datos mediante un formato de fácil aplicación y mantenimiento.
- **Proporciona respuestas más rápidas a las necesidades empresariales en evolución** considerando el impacto del cambio en los activos de información de la empresa y permitiendo la implementación rápida de cambios.

2.2 ER/Studio

ER/Studio es la herramienta ideal para cualquier profesional involucrado en el diseño y/o mantenimiento de bases de datos, tales como administradores de bases de datos, desarrolladores de bases de datos, arquitecto de datos e integradores de sistemas.

ER/Studio está orientado a diseñar y documentar nuevas bases de datos, ya sea para sistemas de procesamiento de transacciones en línea (OLTP) o Data Warehouses, y para mantener, modificar y documentar bases de datos existentes.

ER/Studio es una herramienta visual para modelado de bases de datos en 32 bits para Windows 3.1x, Windows NT y Windows 95. ER/Studio facilita un diseño de alta calidad a un precio muy ventajoso. ER/Studio permite hacer ingeniería-inversa en sus bases de datos con gran precisión y eficiencia, generando código que le permitirá controlar y agilizar sus proyectos de desarrollo y mantenimiento.

2.3.1 CONCEPTO

Es una herramienta de modelado de datos fácil de usar, para el diseño y construcción de bases de datos a nivel físico y lógico.

Ofrece potentes capacidades de diseño lógico, sincronización bidireccional de diseños físicos y lógicos, ingeniería inversa precisa de base de datos y facilidades de información y documentación basadas en HTML.

ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los

diseños físicos y lógicos, construcción automática de bases de datos, documentación y fácil creación de reportes.

ER/Studio ayuda a crear un diseño lógico que puede transformarse en cualquier número de diseños físicos. Como resultado, se puede mantener un diseño lógico normalizado mientras se desnormalizan los diseños físicos para su desempeño. ER/Studio mantiene ligas entre todos los niveles de su diseño por lo tanto puede ER/Studio revisa la normalización y la compilación con la sintaxis de la plataforma de la base de datos. ER/Studio permite tomar por omisión las opciones para todos los diagramas así como realizar cambios al momento de la ejecución.

2.3.2 CARACTERISTICAS

- *Seguridad Del Depósito*: Los administradores de la seguridad del depósito pueden crear los perfiles comprensivos que ayudan a gobernar los privilegios generales de la accesibilidad.
- *Seguridad Del Diagrama*: Los administradores pueden establecer seguridad del objeto-nivel , tiene la capacidad de agregar, de corregir, o de suprimir diagramas específicos.
- *Acceso simultáneo del modelo y del objeto*: Permite la colaboración en tiempo real entre los modelos de bajo nivel y los modelos de objetos. Por ejemplo, más de un modelo puede trabajar en elementos del mismo objeto, tales como un atributo de una entidad, en el mismo tiempo.

- *Resolución Del Conflicto*: Los utilizadores simples e inteligentes de los interfaces con el descubrimiento de las diferencias identificadas entre el modelo de un utilizador específico y los cambios que han ocurrido simultáneamente al modelo

2.3.3 COMPONENTES Y FUNCIONABILIDAD

2.3.3.1 ER/Studio Repository.

Elimina redundancias , administra datos, modela datos para alcanzar usos de más alta calidad ER/Studio Repository distribuye el trabajo a través del MODELING TEAM MEMBERS facilitando de esta manera un ambiente para que ER/Studio Repository pueda ofrecer seguridades mediante la implementación de usuarios.

Facilita un rápido acceso a los diagramas. La creación de diagramas es clara y rápida. Tiene la posibilidad de realizar diagramas con desempeño rápido.

2.3.3.2 Explorer Navigator

Facilita el trabajo hasta con los diagramas más grandes. Se usa el *browser* Explorer para encontrar y seleccionar entidades. Un solo click inmediatamente enfoca una ventana de diagrama.

Se pueden desplegar los modelos de datos usando la notación IDEF1X o IE.

Genera objetos de base de datos: vistas, procedimientos almacenados, defaults, reglas, y tipos de datos de usuario, lo cual ayuda a la auto ordenación de tipos de objetos para eliminar errores de dependencia al construir la base de datos.

Tiene una opción para generar código fuente o para construir bases de datos. Soporte para crear bases de datos para Servidores SQL; y otra, para incluir código SQL y verificar la creación de objetos. Además de la opción para incluir encabezados de comentarios

Para una mejor comprensión de cómo usar ER/Studio y definir sus propios datos a continuación se detallarán los pasos a seguir:

- Abrir la capeta Data Dictionary en el Explorer
- Primero agregar en nuevo Attachment (ver figura 2.6), en el attrachement se crea la información necesaria de un sistema tal como la fecha, nombre del archivo etc.

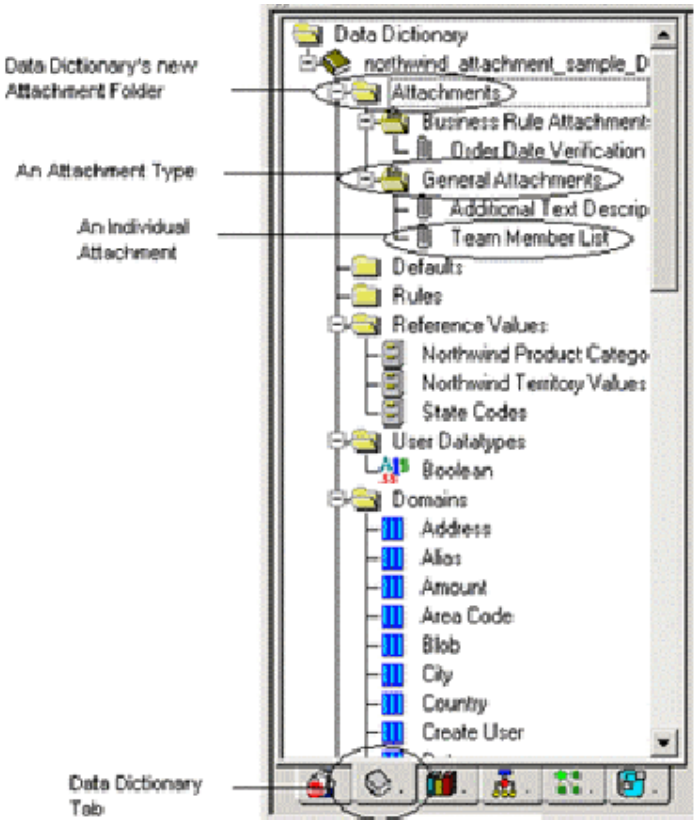


Figura 2.6 Data Dictionary

Patricia Carrillo S.
P.U.C.E.S.A

En el attachment se generan los como se muestra en la figura 2.7

- Diagram
- Model
- Sub model
- Entity/Table
- Attribute/Column
- View
- Relationship
- Index
- Subtype Cluster
- Storage Objects
- Trigger
- Procedure Function
- Rule
- Default
- User Data type
- Domain
- Reference Value

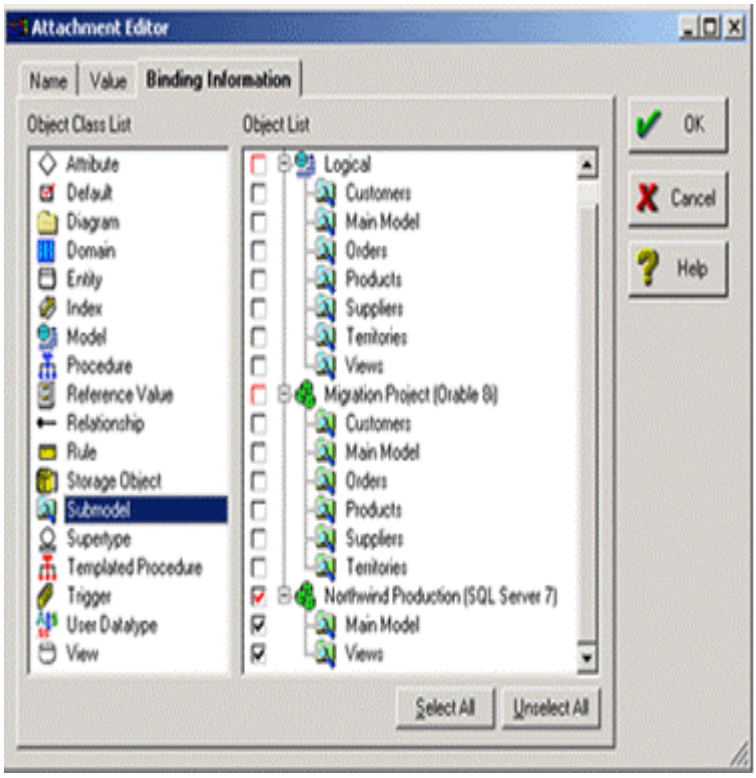
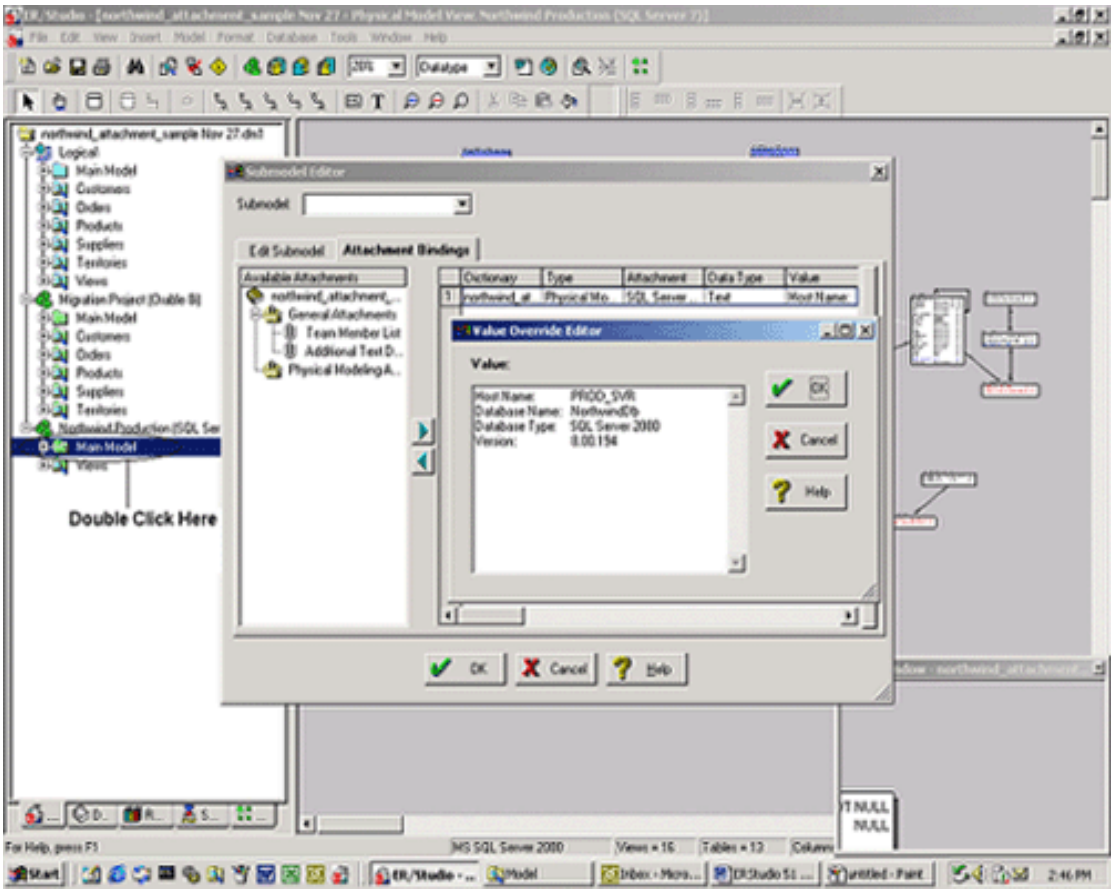


Figura 2.7 Attachment Editor

2.3.3.3 Soporte para crear bases de datos para Servidores SQL; y otra, para incluir código SQL y verificar la creación de objetos. Además de la opción para incluir encabezados de comentarios (Ver figura 2.8)

.

Figura 2.8 Modelador de Base de Datos



2.3.4 BENEFICIOS

- ER/Studio ofrece una poderosa funcionalidad para diseñar bases de datos.
- Permite generar y construir esquemas de bases de datos, para documentar y comunicar diseños de sistemas, y para hacer mantenimiento e ingeniería de inversa a los sistemas actuales.

2. 4 SYSTEM ARCHITECT

System Architect (S.A) es una herramienta CASE que provee de soporte para técnicas variadas para el desarrollo de sistemas de información. Dichas técnicas están asociadas a las principales metodologías actualmente en uso.

Es la única herramienta que integra, en un solo producto multi-usuario, el líder de la industria en soporte de las grandes áreas de modelos, incluyendo modelos de procesos de negocios, orientados a objetos y componentes de modelaje con UML, modelos de datos relacionales, y análisis estructurado y diseño. Toda la funcionalidad existente dentro de System Architect le ofrece soporte nativo para Microsoft VBA

2.4.1 CONCEPTO

System Architect es una herramienta poderosa de modelado estructurado de datos, tiene la capacidad de identificar y clasificar personal para autorizar su entrada al sistema. Los usuarios de red trabajan en un diagrama de proyecto y una llave de registro de diccionario de datos.

System Architect proporciona todos los elementos para diseñar un nuevo sistema o modificar un sistema actual. Es posible crear modelos lógicamente normalizados y modelos de datos físicamente desnormalizados usando el conjunto de herramientas de System Architect. También se puede crear un modelo conceptual de las entidades y especificar su relación con otras. Al avanzar el proyecto, se pueden incluir llaves primarias, atributos, reglas, constraints de integridad referencial, *triggers* personalizados, y cualquier otra información que se elija para mantenerla en el modelo.

Si se diseña un nuevo sistema usando un diccionario amplio de datos es posible especificar los requerimientos de los datos antes de comenzar el modelado, mientras se está construyendo el modelo, o después de haber completado el diseño lógico.

Si se está modificando un sistema existente es posible usar la ingeniería de reverso de System Architect para crear un diagrama de modelo de datos físicos para el sistema actual. System Architect crea automáticamente un DER de un modelo de datos físico. Entonces se puede modificar el DER, creando un modelo lógico normalizado del nuevo sistema. Una vez que se ha completado el diseño lógico, se pueden generar modelos físicos. Si se planea implementar una base de datos desnormalizada, se puede documentar el proceso de desnormalización usando diagramas *Local View* (Vista Local). System Architect mantiene ligas entre el modelo lógico, las vistas lógicas, y el modelo físico; por lo tanto los cambios al modelo lógico se reflejan automáticamente en el modelo físico.

Al final, se tienen dos modelos físicos separados: uno del sistema actual y otro del sistema propuesto.

También, una vez que se ha completado el modelo lógico, se pueden ejecutar una serie de reglas revisadas y reportes de normalización para validar la integridad del diseño. System Architect prueba las Formas Normales: Primera, Segunda, Tercera, y Boyce Codd.

2.4.2 CARACTERISTICAS

- Define propiedades para cualquier entrada de diccionario, incluyendo definiciones, símbolos y diagramas.
- Construye ligas entre varios objetos del diccionario.

- Especifica y define requerimientos, planes de prueba, cambio de requerimientos, objetivos de negocios, metas, y más.
- Especifica que símbolos o grupo de símbolos son afectados.

2.4.3 COMPONENTES Y FUNCIONABILIDAD

La herramienta *Leveling Automatically* nivela diagramas y usa un mecanismo simple para cambiar la herencia en cualquier dirección. Automáticamente crea *Decomposition Diagrams* (Diagramas descompuestos) de la herencia del *Data Flow Diagrams* (Diagramas de flujo de datos).

Permite la edición de un diagrama en cualquier modo de vista, seleccionar y mover objetos individualmente o usando el ratón para obtener la porción del diagrama que se desee, y cambiar el tamaño objetos individuales proporcionalmente o no proporcionalmente usando el ratón.

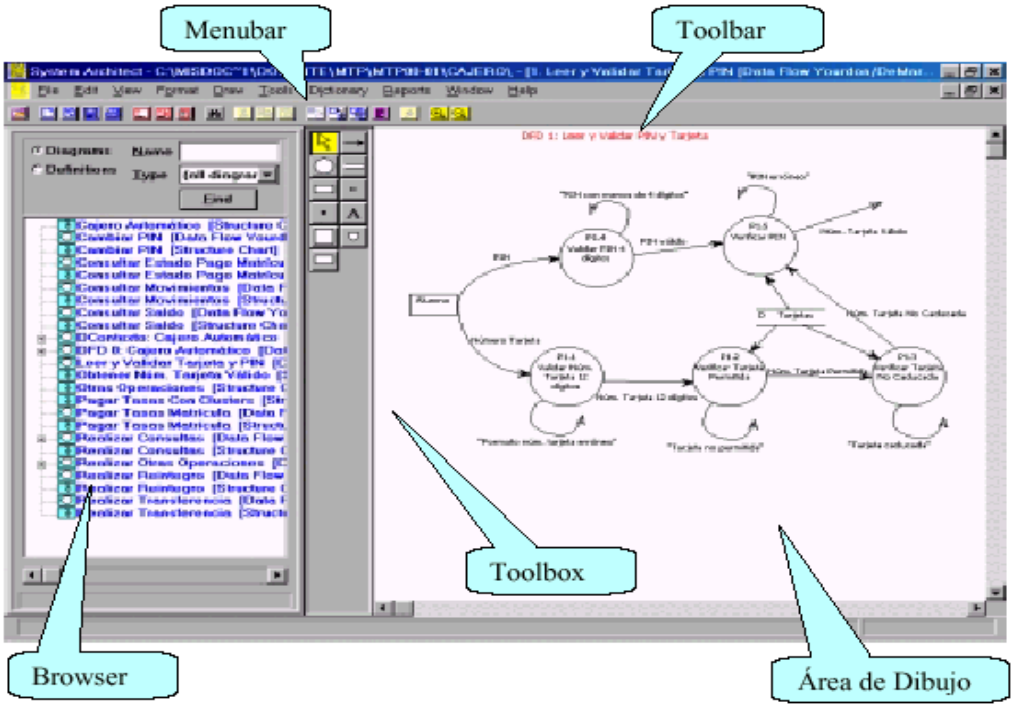
Toda la información introducida mediante la herramienta es almacenada en un directorios, el cual S.A. se denomina ENCICLOPEDIA(repositorio en terminología CASE).

Una Enciclopedia puede contener información de uno o màs sistemas en desarrollo. La herramienta provee mecanismos para importar y exportar información entre enciclopedias. Físicamente, una enciclopedia en S.A contiene:

- Una base de datos relacional compuesta de dos tablas y algunos indices.
- Un fichero por cada diagrama
- Un metafile por cada diagrama
- Cuatro ficheros que determinen la configuración de la enciclopedia
- Un fichero que determinan la configuración de la enciclopedia

- Un fichero de bloqueo si se está ejecutando la versión de System Architect en red.

La primera entrada que hay que proporcionar es el Audi Id, un identificador del usuario para propósitos de auditoria. La pantalla de inicio de sesión se presenta como se muestra en la figura 2.9 en la cual se indican las principales secciones. El Browser permite acceder a la enciclopedia, la información registrada está agrupada en diagramas y definiciones. El toolbox ofrece el conjunto de símbolos que pueden ser dibujados en un diagrama y varían dependiendo del tipo de diagrama que este activo.



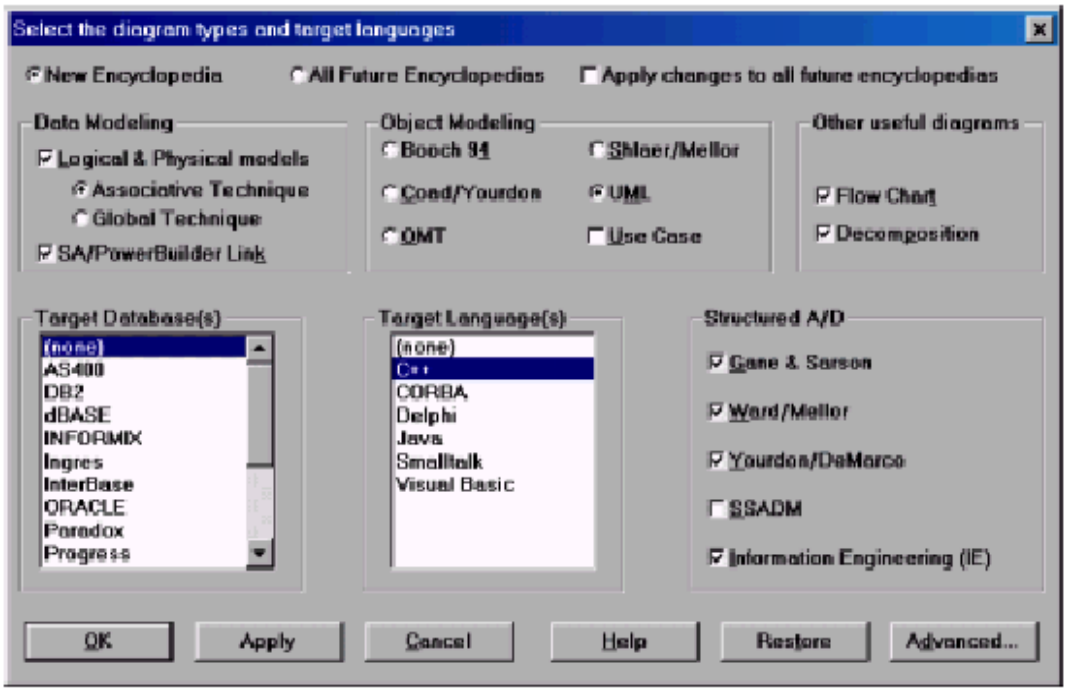
(

Figura 2.9 Secciones de Pantalla

Básicamente el trabajo de análisis y diseño con la herramienta System Architect. consiste en editar diagramas y definiciones. Cada elemento gráfico de un diagrama tiene una definición y a su vez las definiciones pueden tener otras definiciones.

Para crear una enciclopedia se selecciona FILE Enciclopedia Open, en el cuadro de dialogo, luego se debe introducir el nombre de la enciclopedia(directorios) que se va a crear. Crear una enciclopedia implica crear el directorio(si este no existe) y generar en él todos los ficheros iniciales para la enciclopedia .

En la figura 2.10 se seleccionan las técnicas que se utilizarán. Cada técnica tiene asociado un conjunto de diagramas. Además se establece el sistema gestor de base de datos y el lenguaje de programación. Estas elecciones determinan ciertas propiedades relacionadas con la generación automática del diseño físico de la base de datos y de plantillas de código en el lenguaje de programación . Con el botón ADVANCED se accede a un cuadro de dialogo donde es posible modificar la lista detallada de diagramas y propiedades disponibles.

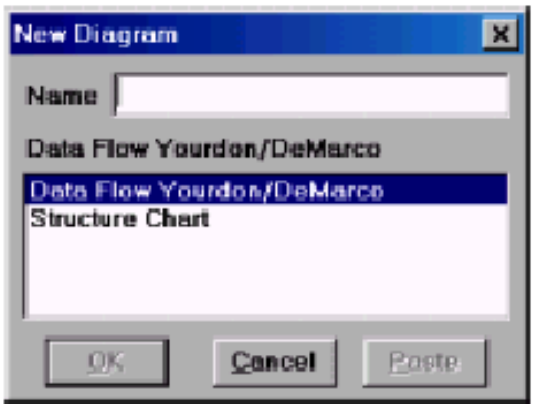


SsFigura 2.10 Selección de técnicas utilizadas en la enciclopedia

Patricia Carrillo S.
P.U.C.E.S.A

Mediante la opción *Menu Bar* | New Diagram o usando el correspondiente botón en el *Toolbar*, se puede acceder a la pantalla de dialogo(figura 2.11) . En ella introducir el nombre de diagrama a construir y seleccionar su tipo. Posteriormente se activará una ventana de dibujo para el diagrama y aparecerá la Toolbox asociada al tipo de diagrama .

Figura 2.11Pantalla de Dialogo



Básicamente, un diagrama está compuesto de símbolos (los disponibles en la *Toolbox*) . La forma de conectar símbolos puede ser modificado accediendo a Format | Símbolo | Style | line. Cada símbolo puede ser descrito por una definición. Estas definiciones constituyen parte de la enciclopedia de la herramienta y pueden ser compartidas por diferentes diagramas. System Architect mantiene independientemente entre las propiedades de los símbolos dibujados y las definiciones asociadas. Es decir, al dibujar un símbolo no es obligatorio proporcionarle propiedades, ni definiciones.

Para efectuar alguna operación sobre uno o más símbolos, estos se seleccionan y posteriormente se debe escoger la operación a realizar en el *Menú bar* o presionando el botón derecho de ratón; en este caso aparece una lista de opciones similar a la que se muestra en la figura 2.12

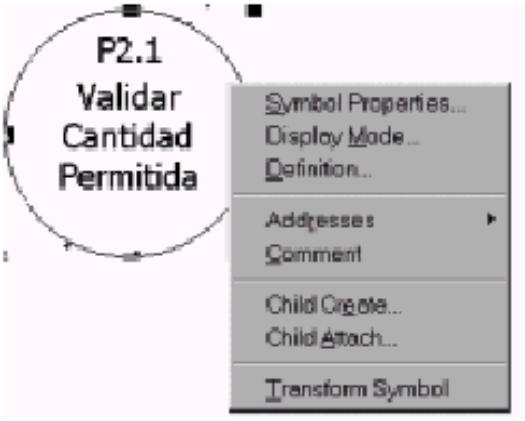


Figura 2.12 Selección de Símbolos

2.4.4 BENEFICIOS

- Soporta la mayoría de los paquetes de red incluyendo Novell, 3Com, Banyan, DecNet, LAN Manager, STARLAN y otras.
- Permite construir, optimiza y manejar bases de datos .
- Para cualquier tipo de proyecto en el que se este trabajando, SA proporciona flexibilidad para completar el trabajo.
- Se pueden elegir modos de despliegue en cualquier tiempo durante el proceso de diseño: conceptual, basado en llaves, totalmente atribuido, o despliegue físico.

2.5 POWER DESIGNER

PowerDesigner es una *suite* de aplicaciones de Powersoft para la construcción, diseño y modelado de datos a través de diversas aplicaciones. "Sybase PowerDesigner es una suite de herramientas de modelado de software que son accesibles vía un formato común, permitiendo a los grupos de desarrollo y los administradores de bases de datos trabajar conjuntamente y rápidamente para crear aplicaciones sólidas. Así mismo, la utilización de PowerDesigner requiere algo de capacitación inicial para miembros que manejan Tecnología de la Información

Esta herramienta de modelamiento combina la funcionalidad del producto líder de diseño de base de datos con un potente modelador de objetos basado en UML, brindando el primer entorno integrado de análisis y diseño objeto/relacional.

PowerDesigner soluciona la falta de correspondencia existente en aplicaciones persistentes orientadas a objetos en una base de datos relacional. Además PowerDesigner también ayuda a tomar los primeros pasos en el desarrollo de aplicaciones distribuidas usando diagramas basados en UML para modelar los objetos de la aplicación y luego generar objetos no visuales PowerBuilder y Java.

2.5.1 CONCEPTO

Es una herramienta para crear bases de datos y aplicaciones cliente/servidor basadas o no en *We*b. Permite a los diseñadores de aplicaciones complejas de cliente/servidor tener una descripción general de los procesos particulares para comprender mejor a la organización.

Exporta información del modelo físico y extiende atributos al diccionario de 4GL. Importa atributos extendidos de PowerBuilder. Soporta definición de atributos extendidos para PowerBuilder, Progress, Uniface, PowerHouse y NS-DK.

Ofrece un acercamiento de diseño para optimizar las estructuras de las bases de datos. Capturando el flujo de datos de su organización, puede crear un modelo conceptual y físico de la base de datos. La técnica de diseño a dos niveles permite separar lo que se desea diseñar de lo que se desea implementar.

2.5.2 CARACTERISTICAS

- Mediante el incremento del modelo de la base de datos, AppModeler genera instantáneamente objetos, componentes *data-warehouse*, y hasta aplicaciones básicas listas para ejecutarse inmediatamente en PowerBuilder, Power++, Visual Basic, Delphi, y Web-based objects.
- El AppModeler permite a los desarrolladores: diseñar modelos de bases de datos físicas o crearlas instantáneamente a través de la ingeniería de reversa de bases de datos existentes, generar, documentar y mantener bases de datos, generar rápidamente objetos de aplicación y componentes de datos.
- Genera objetos personalizables de PowerBuilder y componentes basados en modelos de bases de datos físicos y plantillas que se encuentran dentro de las librerías de clases de su elección. Genera objetos ventana y ventana de datos basadas en tablas, vistas y relaciones de llaves primarias-foráneas.
- Genera y hace ingeniería de reverso a los atributos. Incluye plantillas personalizables para la librería PowerBuilder Foundation Class (PFC).

- Genera formas basadas en tablas, vistas, y relaciones de llaves primarias-secundarias.
- Genera proyectos basados en modelos de propiedades.
- Genera controles tales como menús, listas, etc.

2.5.3 COMPONENTES Y FUNCIONABILIDAD

PowerDesigner cuenta con herramientas para la creación y control de diagramas como son: *Off-page Connecto*r; que representa los flujos de entradas y salidas en un proceso, *Business Rules* que define las reglas de uso para Procesos, Almacenamiento de datos, Entidades externas, y Flujos de datos; y CRUD Matrix, que define el efecto de un proceso de datos en términos de Crear, Leer, Actualizar, y Borrar operaciones (CRUD)

2.5.3.1 PowerDesigner Appmodeler.

Mediante el incremento del modelo de la base de datos, AppModeler genera instantáneamente objetos, componentes *data-war*e, y hasta aplicaciones básicas listas para ejecutarse inmediatamente en PowerBuilder, Power++, Visual Basic, Delphi, y Web-based objects.

El AppModeler permite a los desarrolladores: diseñar modelos de bases de datos físicas o crearlas instantáneamente a través de la ingeniería de reversa de bases de datos existentes, generar, documentar y mantener bases de datos.

PowerDesigner es una *suite* de aplicaciones de Powersoft para la construcción, diseño y modelado de datos a través de diversas aplicaciones. Esta *suite* cuenta con los siguientes productos:

a) PowerDesigner Processanalyst.

b) PowerDesigner DataArchitect.

c) PowerDesigner WarehouseArchitect.

d) PowerDesigner MetaWorks.

e) PowerDesinger VieWer**.**

a) PowerDesigner ProcessAnalyst.

Permite analizar el flujo de datos de toda la empresa, a través de los departamentos hasta el usuario final.

b) PowerDesigner DataArchitect.

Data Architect proporciona capacidades de modelado de datos tradicional, incluyendo diseño de bases de datos, generación, mantenimiento, ingeniería de reversa (ver figura 2.13 y documentación para arquitecturas de bases de datos. Permite que los diseñadores de bases de datos creen estructuras de datos flexibles, eficientes y efectivas para usar una ingeniería de aplicación de bases de datos.

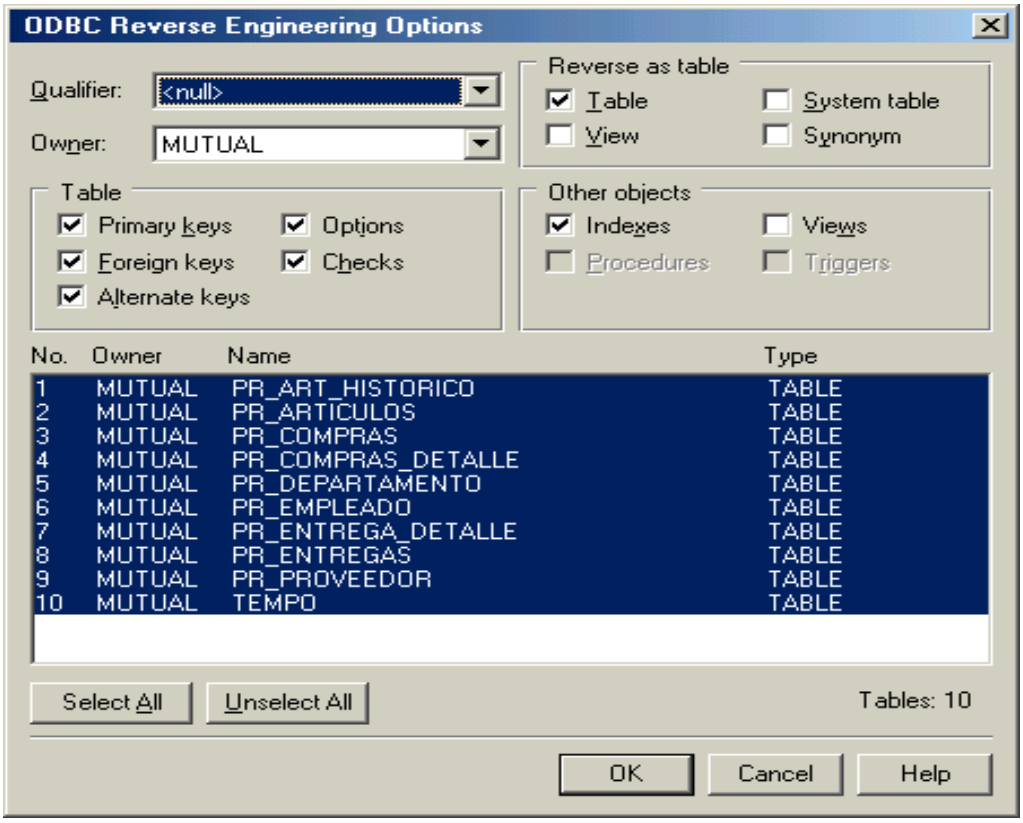


Figura 2.13 ODBC

También proporciona un diseño conceptual de modelo de datos, generación automática de modelo de datos, diseño de normalización física, sistema de manejo de bases de datos múltiples (DBMS) y soporte de herramientas de desarrollo, y elementos de reportes con presentación y calidad.

A continuación se detalla los pasos a seguir para poderse conectar a una base de datos desde **PowerDesigner DataArchitect.**

- En el menú se escoge Dictionary , luego aparece un submenú en el cual al dar clic en Generate Physical Model aparece una ventana en donde se escoge la base de datos a la cual se quiere conectar (ver figura 2.14) y se habilita en el menú la opción DATABASE en la cual se escoge connect y aparace una ventana donde

permite seleccionar el nombre de la base de DATOS a la cual se va ha conectaremos (ver figura 2.15)

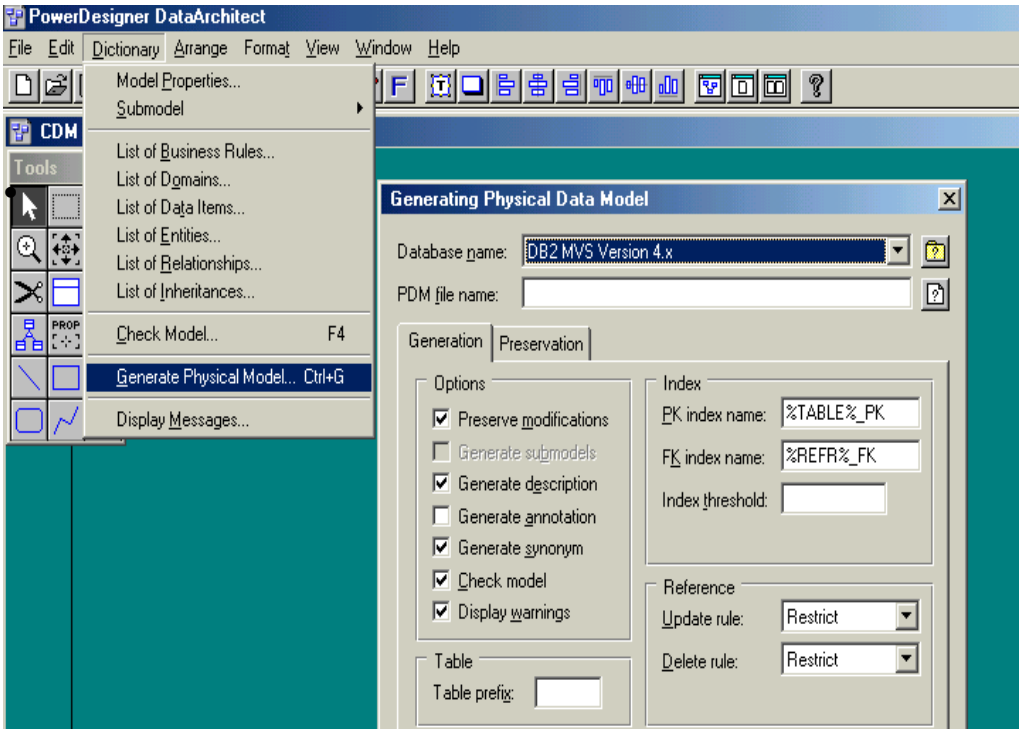


Figura 2.14 Pantalla para seleccionar la Base de Datos

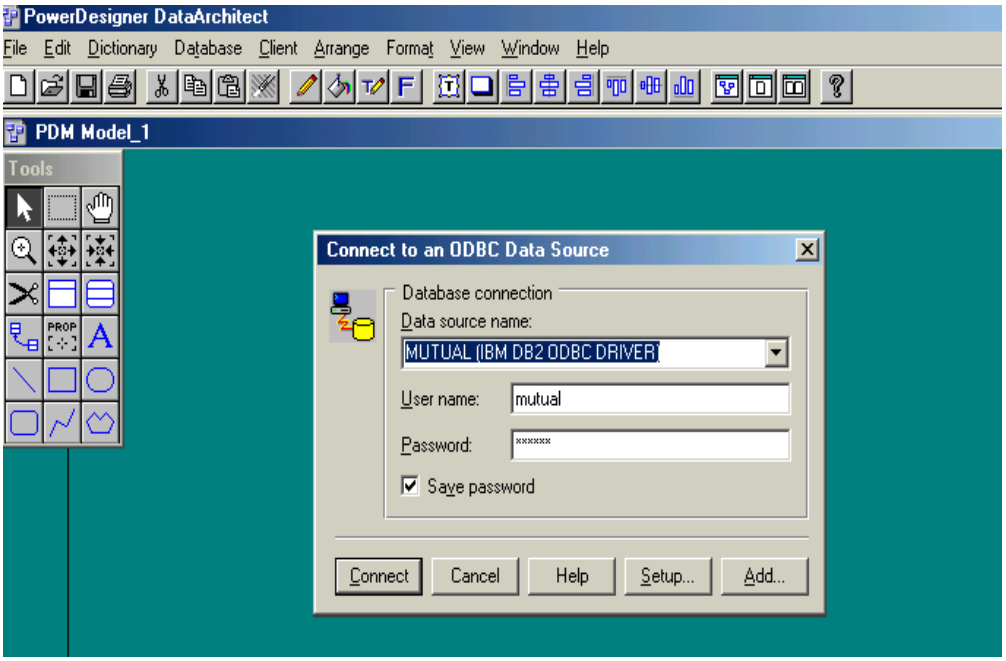


Figura 2.15 Ventana de Conexión

c) PowerDesigner WareHousea Architect

Provee un poderoso *datawarehousing* para el diseño e implementación de una base de datos. Cuenta con soporte para bases de datos tradicionales DBMS y bases de datos en plataformas de sistemas analíticos usando modelados dimensionales, particionamiento y agregación. También cuenta con un alto desempeño en el indexamiento de esquemas.

d) PowerDesigner MetaWorks

Permite fácilmente ver y compartir la información del modelado de datos con una definición constante de objetos. También puede comparar y mezclar dos modelos de datos paso a paso.

e) PowerDesinger Viewer.

Crea reportes de los modelos físicos, conceptuales y procesos del modelado de la base de datos. También permite generar reportes para Internet en HTML. Este producto cuenta con demos directos de sitio de Sybase en Internet para su evaluación.

2.5.4 BENEFICIOS

- Permite ahorrar dinero al habilitar diseños de aplicaciones rápidamente.
- Mejora la calidad del software y reduce los costos en el mantenimiento de aplicaciones.

2. 6 ORACLE DESIGNER

Oracle Designer forma parte de una familia de productos orientados a la creación de aplicaciones Cliente/Servidor escalables, portables y fáciles de desarrollar tanto a nivel de grupos de trabajo, departamental como corporativo; de la forma más productiva posible.

El esquema **Cliente/Servidor** se compone del elemento **servidor** quien se encarga de mantener almacenada la información en la base de datos, ya sea para consultas y modificaciones por parte del elemento cliente o para almacenar aquella nueva información enviada por este. Además se encarga de atender todos los requerimientos de información o procesamiento de esta información solicitada por cada uno de los clientes mientras estos lo requieran, manejar los esquemas de seguridad, mantener la integridad de la información y administrar la concurrencia sobre la información almacenada en el servidor.

Cuenta además con el elemento **cliente** quien toma los requerimientos entregados por el usuario por medio de la aplicaciones y los envía al servidor. Una vez el cliente ha recibido una respuesta del servidor, ya sean los datos solicitados o la negación de estos, le entrega el resultado al usuario final. Con una característica adicional, y es que el procesamiento de esta información puede ser ejecutado ya sea del lado del servidor o ya sea del lado del cliente, lo cual nos permite **particionar** las aplicaciones. Típicamente en el esquema Cliente/Servidor el elemento cliente se encuentra remoto o como parte de una red de área local.

Y finalmente debe existir el **medio de comunicación** entre los dos elementos anteriormente descritos, el cual esta dirigido por el **protocolo de comunicación** quien se encarga de tomar la información solicitada por el cliente y empaquetarla para ser transmitida por el medio de comunicación hacia el servidor y viceversa, el cual puede ser cable coaxial, línea satelital, línea telefónica, etc. Dentro de los protocolos de comunicación más conocidos tenemos: TCP/IP, IP/SPX, DECNET, NAMEDPIPES, etc.

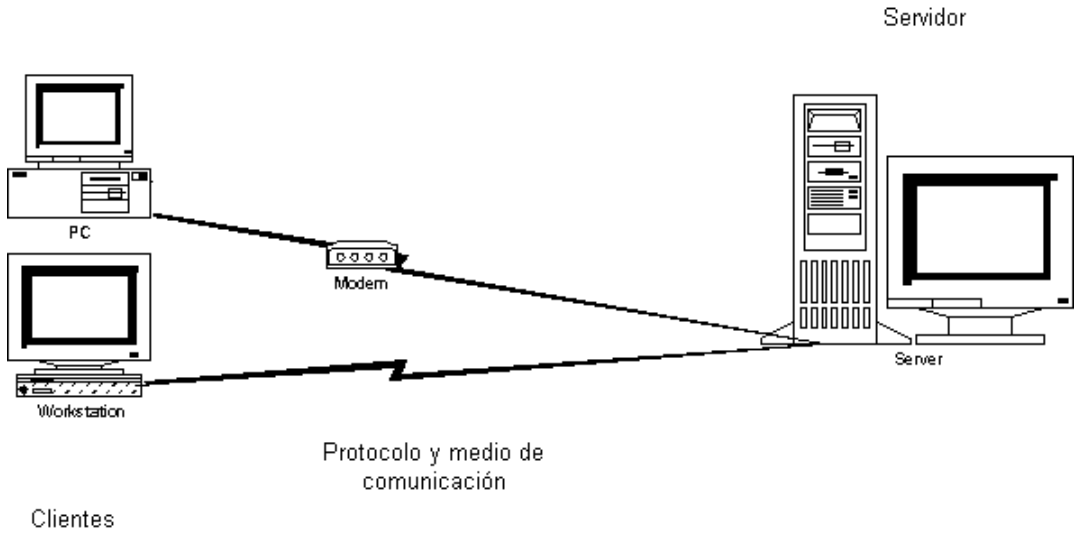


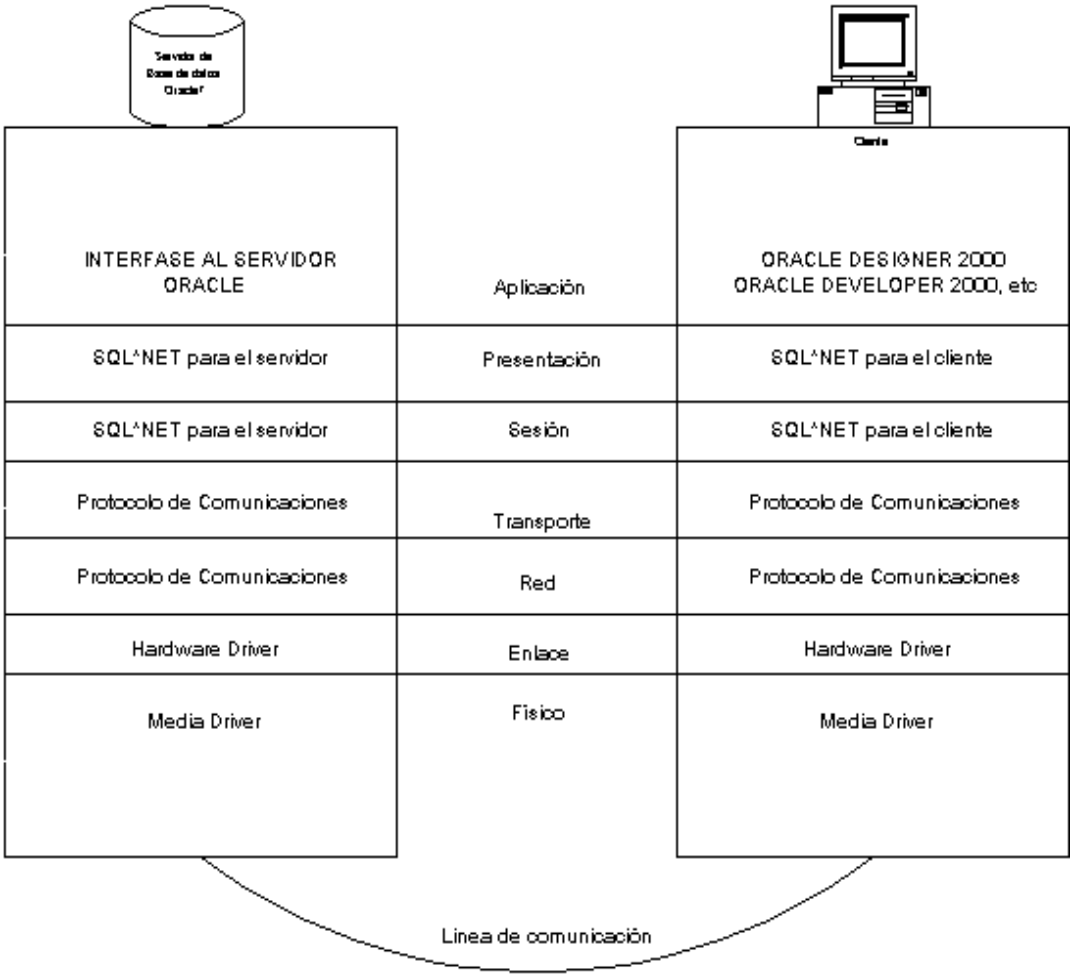
Figura 2.16 Detalla cada uno de los componentes de un ambiente Cliente-Servidor

Oracle Designer es una herramienta CASE que le permite MODELAR procesos complejos y a partir de esto y de otros generados en procesos de INGENIERÍA EN REVERSO - realizados por el mismo Designer, generar temas orientados al manejo de información almacenado en Bases de datos de cualquier proporción. Cuenta con un repositorio de información común, soporte a cualquier metodología de desarrollo, un ambiente de desarrollo cliente/servidor unificado y una gran cantidad de herramientas gráficas para el modelamiento y generación de grandes aplicaciones.

2.6.1 CONCEPTO

Oracle Designer es una herramienta CASE cliente/servidor compuesta por diagramadores en el lado del cliente y un repositorio multiusuario en el lado del servidor. Esta arquitectura permite modelar aplicaciones en equipos de trabajo dado que la seguridad, integridad, consistencia y concurrencia de la información almacenada en este esta completamente controlada por el manejador de base de datos Oracle y administrada por el usuario desde el Cliente.

Figura 2.17 indica como trabaja Designer y cuales son cada uno de sus componentes.



Oracle Designer es una herramienta CASE que abarca todo el ciclo de vida de la creación de cualquier sistema de información, incluyendo la generación del código necesario para la puesta en producción. Además soporta cualquier metodología de desarrollo que quiera utilizarse para el desarrollo del mismo incluyendo: ORACLE CASE METHOD, Buisness Process Reengineering (BPR), Ingeniería en Reverso, Bases de datos distribuidas, etc.

Cuenta con Diagramadores y Herramientas para cada etapa del desarrollo completamente integradas entre ellas, las cuales obtienen la información de un repositorio de información común, que permite acceso a múltiples usuarios en forma simultánea sobre la misma aplicación; esto facilita la creación de aplicaciones corporativas trabajando en equipos de trabajo.

2.6.2 CARACTERÍSTICAS

Oracle Designer, tiene muchas facilidades tales como:

- Manejar múltiples versiones de una aplicación.
- Mantener copias "congeladas" de estas.
- Crear perfiles de usuario dentro del repositorio separados de los esquemas de seguridad de la base de datos.
- Generación de reportes sobre el estado actual de cualquier elemento de la aplicación .
- Ingeniería en reverso de aplicaciones existentes.
- Extraer información de otras herramientas CASE e incluirlas dentro de la definición de las aplicaciones, etc.

2.6.3 COMPONENTES Y FUNCIONABILIDAD

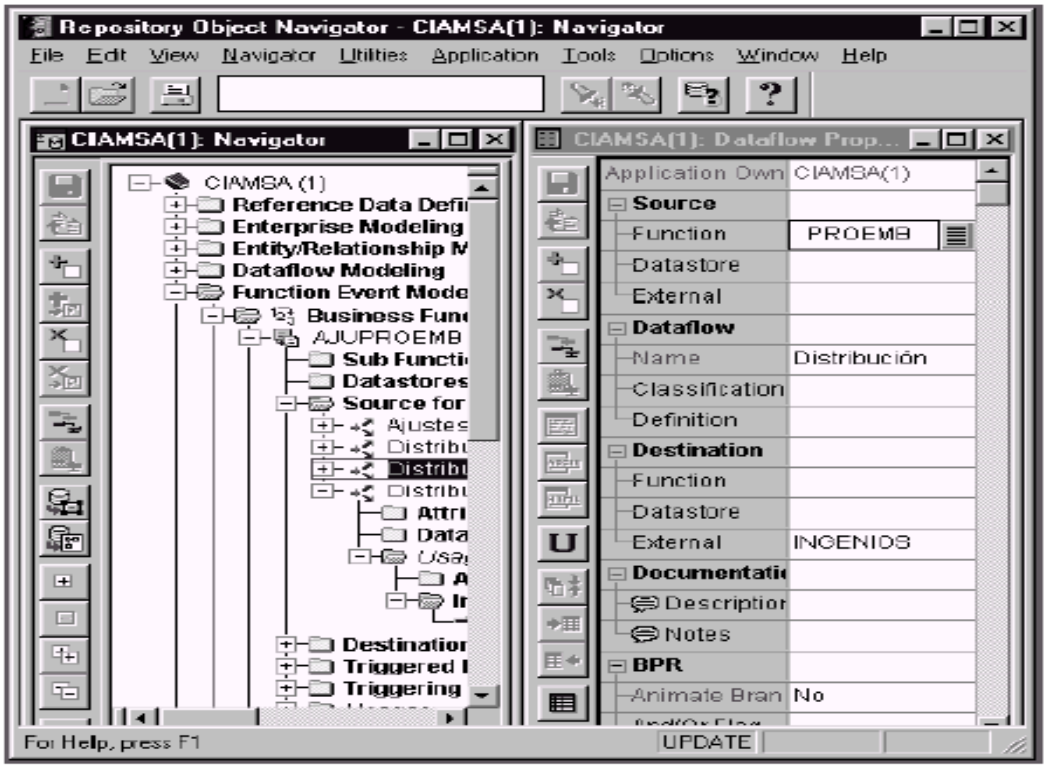
2.6.3.1 EL REPOSITORIO DE ORACLE DESIGNER

La información referente a un proyecto específico se almacena como definiciones (lo que es un conjunto estructurado de detalles) en el repositorio de Oracle Designer.

Las definiciones están compuestas de propiedades (los detalles) así que el proceso de alimentación de una definición requiere de llenado de dichas propiedades.

Como se podrá notar aunque la definición de este elemento esta completa existen algunas propiedades vacías lo cual es muy típico del repositorio. Algunas propiedades como las de etiqueta (Label) y la de definición corta (Short Definition) , pero el resto de propiedades normalmente pueden tener un valor por defecto o simplemente estar vacías.

Figura 2.18 Repository Object Navigator



El problema radica en que algunas de las propiedades no obligatorias pueden ser muy importantes para posteriores etapas del ciclo de vida del proyecto y que los valores que se coloquen en la etapa de análisis del proyecto pueden radicalmente afectar la forma como Oracle Designer maneja el diseño y la generación de código.

2.6.3.1.1 Flujo de la Información

La información del repositorio de Oracle Designer fluye a otras áreas por medio de un conjunto de herramientas y utilidades. Se pueden clasificar todas las definiciones del repositorio en dos tipos principales (especialmente para análisis y diseño)

DATOS: entidades, atributos, relaciones, tablas, columnas y constraints.

PROCESOS: Funciones, flujos de datos, almacenamientos de datos, módulos, usos de datos y asociaciones de módulos. Cada uno de estos tipos tiene definiciones separadas dentro del repositorio de Oracle Designer y algunos de ellos son copiados a otras definiciones por medio de las utilidades de Oracle Designer

En la Figura 2.19 se puede observar gráficamente como fluye la información en Oracle Designer.

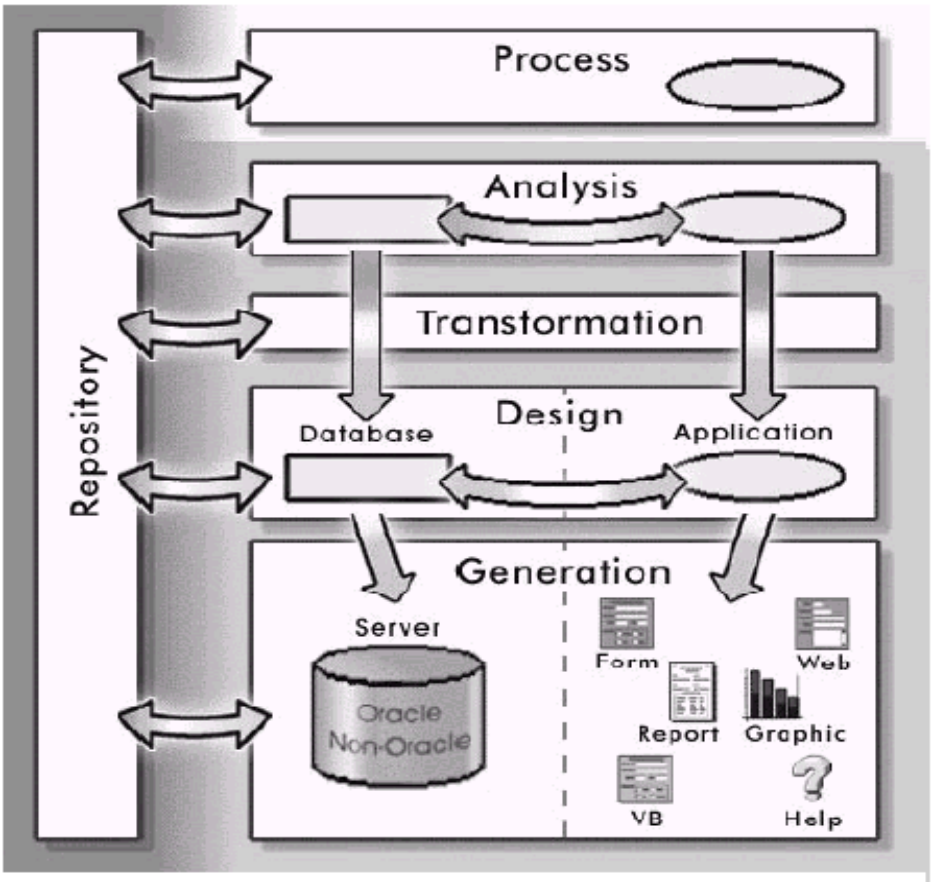


Figura 2.19 Flujo de la información en ORACLE DESIGNER

2.6.3.1.1.1 Análisis a Diseño

Las definiciones de análisis (Entidades y Funciones) del repositorio sirven como un modelo lógico del negocio. Oracle Designer copia los valores de las propiedades a dos tipos de objetos de diseño:

Datos: Tablas, columnas, llaves foráneas y constraints.

Procesos: Formas, reportes, módulos manuales.

Existen diferentes diagramadores en Oracle Designer que muestran de diferentes maneras la información de análisis y diseño con los tipos de datos y procesos.

Por el lado de los datos estos son transformados por la utilidad de transformación de la base de datos (Database Design Transformer DDT) la cual se encarga de crear las tablas y las definiciones de las columnas lo cual hace convirtiendo las entidades en tablas y los atributos en columnas. La utilidad también resuelve las relaciones muchos a muchos que por algún motivo no se resolvieron en el análisis, para ello crea una nueva tabla con una llave primaria compuesta por las llaves primarias de las dos entidades involucradas en la relación.

La utilidad detecta si una entidad no tiene un identificador primario lo soluciona creando una llave primaria (llamada Surrogate key) la cual es llenada por medio de una secuencia de Oracle la cual también es generada por la utilidad. El DDT ((Database Design Transformer) también resuelve los arcos que hayan sido creados en la fase de análisis y ofrece varias opciones.

De otro lado están los procesos o sea los módulos de la aplicación, el Application Design Transformer (ADT) crea las definiciones de los módulos tomando la información de las definiciones de las funciones lo cual produce módulos candidatos (módulos que todavía no están listos para la generación) los cuales pueden ser luego aceptados como módulos de la aplicación utilizando la utilidad RON o el editor de diseño (Design Editor).

Luego que el módulo ha sido aceptado este ya puede ser utilizado para la generación de código con alguno de los generadores. El ADT también crea el uso de datos del módulo basado en la información del uso de datos de las funciones de esta forma el ADT tiene información para generar bloques para las formas o grupos para los reportes. Finalmente el

ADT también toma información de las funciones y sus asociaciones para generar un menú inicial con dicha información.

Todo esto es valido para cualquier metodología que sé este utilizando dentro del proyecto como puede ser Estrategia-Análisis-Diseño ó Análisis-Diseño ó RAD. En la versión actual de Oracle Designer aún no se soporta totalmen- te la metodología UML (el UML en Oracle Designer será abordado en un próximo articulo) pero en las siguientes versiones dicho soporte va a ser cada vez mayor.

2.6.3.1.1.2 Diseño a Generación

En esta parte se cubre todo lo referente a la parte de diseño que a la postre va a ser utilizado por los generadores de código basados en la información almacenada previamente en el repositorio. El generador de la base de datos se encarga de producir todos los scripts necesarios para generar los diferentes objetos de la base de datos (tablas, vistas, índices, sinónimos y secuencias, entre otros), esto también incluye todas las características propias de una base de datos distribuida (snapshots). Los generadores de Forms, Reports, WebServer, Visual Basic y Ms Help producen código basado en la información definida en los módulos que previamente han sido almacenados en el repositorio y los cuales pueden haber sido obtenidos por medio de la utilidad ADT. Esto quiere decir que muchas de las propiedades que se han definido en una tabla pasan a un módulo el cual forma parte de la aplicación, es importante resaltar la importancia en la definición del uso de los datos en esta etapa.

Adicionalmente encontramos la utilización de librerías de objetos los cuales pueden ser utilizados por los generadores así como la definición de una serie de preferencias en las cuales puedo definir estándares de programación. El uso adecuado de estos elementos

puede permitir la construcción de grandes aplicaciones sin necesidad de escribir ni una sola línea de código y almacenar la lógica de la aplicación en el repositorio de datos.

2.6.3.1.1.3 Objetos de la Base de Datos a Diseño

En esta categoría la información fluye desde los objetos de la base de datos hacia el repositorio, en otras palabras es lo que se conoce como ingeniería de reversa o captura de diseño y para ello Oracle Designer provee una utilidad que se encarga de rescatar los objetos de la base de datos real y colocarlos en términos de definiciones del repositorio de datos y más aun en un estado de diseño. En este punto es importante resaltar que la utilidad es muy sensitiva a los constraints que la base de datos tenga definidos al momento de hacer la captura, esto quiere decir que si se selecciona una tabla y no se seleccionan las tablas con las que tienen relación no se puede pretender capturar las definiciones de los constraints de las llaves foráneas.

2.6.3.1.1.4 Módulos de la Aplicación a Diseño

En el flujo de información que existe en Oracle Designer también tenemos el que existe entre los módulos de una aplicación ya realizada y el repositorio. Esto quiere decir que podemos recuperar el diseño de formas y reportes que previamente hayan sido construidos con el mismo Oracle Designer (versiones anteriores) o manualmente. Suena muy interesante cierto pero la verdad es que sólo se puede aspirar a que recupere toda la información referente al uso de datos dentro de los módulos, esto quiere decir que todo lo referente al layout del módulo no se mantendrá si posteriormente se utilizan los generadores.

2.6.3.2 DISEÑO Y ANÁLISIS

Es interesante ver como Oracle Designer permite fluir la información a etapas muy primarias del ciclo de vida de un sistema y por ello permite pasar información de la etapa de diseño a la de análisis por medio de una utilidad (Table to Entity Retrofit) la cual permite seleccionar de una lista de tablas candidatas las que se desean que se conviertan en entidades (esto solo es para tablas que no tienen asociada una entidad o sea que no sirve para conciliar definiciones entre entidades y tablas).

2.6.3.3 DISEÑO DE LA BASE DE DATOS

2.6.3.3.1 Database Design Wizard(DDW)

Esta herramienta de Designer ejecuta en forma automática el paso del modelo entidad/relación a sus respectivas componentes en la etapa de diseño. El database design wizard se encarga, además del paso de entidades a tablas, de relaciones a condiciones de integridad, de identificadores únicos a llaves primarias y de atributos a columnas, de resolver ciertas características del modelo entidad/relación tales como:

- Relaciones Muchos a Muchos
- Subtipos y Supertipos
- Arcos.

2.6.3.3.2 Data Diagrammer(DD)

Por medio de este diagramador puede plasmar la definición de los componentes generados por el database design wizard en forma gráfica. Aquí puede ver e incluir nueva información sobre: tablas, vistas, snapshots, condiciones de integridad, secuencias, índices, etc, lo cual le permitirá refinar el modelo de datos que va a manejar su aplicación.

2.6.3.4 ORACLE SERVER GENERATOR

El Oracle Server Generator se encarga de crear automáticamente todo el esquema de la base de datos almacenado en el repositorio de Designer.

- Crea la definición de: Bases de datos, Tablas, Columnas, Índices, Condiciones de integridad, vistas, sinónimos, snapshots.
- Soporta standard ANSI SQL y ORACLE SQL
- Creación de los procedimientos almacenados Oracle7
- Creación de los esquemas de seguridad Oracle7
- Creación de todo el esquema y objetos necesarios para bases de datos distribuidas

Dado que Oracle Designer es un repositorio de información, sobre el cual puede coexistir más de un proyecto, más de un grupo de trabajo y más de una aplicación, este cuenta con herramientas tanto de administración como de trabajo que facilitan la labor de administración y definición de la información allí almacenada.

2.6.3.5 REPOSITORY ADMINISTRATION UTILITY

Dado que Designer es una herramienta CASE que permite el acceso de múltiples usuarios simultáneamente y la creación de más de una aplicación a la vez, esta cuenta con una herramienta de administración del repositorio la cual tiene las siguientes ventajas:

- Interfase gráfica Fácil de utilizar
- Permite la creación de usuarios para trabajo dentro del repositorio
- Define los esquemas de seguridad para cada uno de los usuario que van a trabajar en este programa

- Maneja la definición de objetos internos repositorio dentro de la base de datos tales como: Vistas, Índices, Procedimientos, etc.

2.6.4 BENEFICIOS

Al usar Oracle Designer las empresas y los usuarios obtendrán:

- Desarrollo de Escala Empresarial.
- Reingeniería de Procesos de Negocios.
- Sistemas Visuales y Modelajes de Diseño.
- Desarrollo de Sistemas Dirigido por Modelo.
- Repositorio Abierto / Diagramador de Relaciones.
- Generación de Aplicación de Cliente.
- Entrega de diseño de Cliente/Servidor de segunda generación .
- Modelador de Jerarquía de Funciones.
- Modelador de Flujo de Datos.
- Funciones de Generación de Aplicación Developer/2000.
- Generación de Visual Basic.

2.7 RATIONAL ROSE

La complejidad de los proyectos de software hoy en día, el constante cambio de requerimientos y la falta de una documentación durante el proceso desarrollo provoca que los proyectos se retrasen en tiempo y se incrementen en costo.

La solución a esta problemática es implantar una arquitectura de desarrollo que permita dar seguimiento a los proyectos desde su etapa de requerimientos, hasta su implantación.

Rational Rose ofrece un Proceso Unificado (RUP) para el desarrollo de los proyectos de software, desde la etapa de Ingeniería de Requerimientos hasta la etapa de pruebas. Para cada una de estas etapas existe una herramienta que ayuda en la administración de los proyectos, Rational Rose es la herramienta para la etapa de análisis y diseño de sistemas.

Para mantener este liderazgo en el mercado, Rational ha lanzado una nueva versión de Rose, *Rose* orientada a aplicaciones de comercio electrónico Esta nueva versión ayuda a los desarrolladores de software a construir mejores productos en menor tiempo permitiendo que sus aplicaciones ingresen al mercado más rápidamente, da un excelente soporte en el manejo de cambios durante el ciclo de vida del proyecto y mejora la comunicación entre los miembros del equipo.

Rational Rose es una herramientas CASE Orientadas a Objetos líderes en cuanto a penetración en el mercado. En lo que se refiere a volumen de ventas por año. En México, actualmente se utiliza en corporativos importantes y por empresas de consultoría encargadas de construcción de software de alto nivel. Cuenta, además con gran aceptación en el ámbito académico (UNAM, IPN, ITESM, ITAM, FAR)

2.7.1 CONCEPTO

Rational Rose es una herramienta Orientada a Objetas con plataforma independiente que ayuda a la comunicación entre los miembros del equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas. Una de las grandes ventajas de Rose es que utiliza la notación estándar en la arquitectura de Software (UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común. Otra ventaja de Rose es que los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros componentes del proyecto. Además Rose soporta la construcción de componentes en lenguajes como C++, VisualBasic, Java, Ada. Por todo lo anterior Rose es la herramienta de Análisis, Diseño, Modelado y Construcción de software Orientado a Objetos líder en el mercado.

2.7.2 CARACTERISTICAS

A continuación se detallan algunas nuevas características incluidas en Rose , como:

Integración entre WinDNA y Microsoft VisualStudio

Mejoras en la generación de código con Java y aplicaciones CORBA

Integración con ClearCase

2.7.3 COMPONENTES Y SU FUNCIONABILIDAD

2.7.3.1 Integración entre Windna y Visual Studio

Los arquitectos y desarrolladores de software que diseñan y construyen aplicaciones distribuidas en Internet utilizan las tecnologías de Microsoft y WinDNA, por esto Rose :

- Permite modelar y generar objetos COM además de definir interfaces usando Visual Basic o Visual C++, ya que los componentes puedan ser implementados en

cualquiera de los dos lenguajes y las interfaces puedan definirse en un lenguaje y ser usadas por un componente en otro lenguaje.

- Mejora la ingeniería Round-trip, la cual es soportada por Visual C++ y Visual Basic 6.0 incluyendo: WebClass, DHTML y Data Connections.
- Uso de ingeniería en reversa para visualizar y reutilizar clases e interfaces existentes en formato binario (.ddl, .tib, .ocs, .exe). Las interfaces y clases obtenidas de la ingeniería en reversa pueden usarse en la construcción de nuevos componentes de Visual Basic y Visual C++.
- En la generación de código para VisualBasic, es posible transformar clases estereotípicas en código real, los templates pueden especificarse con propiedades y métodos opcionales o requeridos, el cuerpo del código puede ser personalizado y los templates pueden ser usados como patrones de código.
- Para la construcción de transacciones rápidas y robustas, de componentes data-ware en Visual Basic se permite la utilización de MTS (Microsoft Transaction Server) y ADO (ActiveX Data Objects). Además captura patrones de código común para la creación de clases MTS y ADO-ware.

2.7.3.2 Mejoras ROSE-JAVA-CORBA.

Rose es una herramienta indispensable en las compañías con desarrollo de soluciones para el comercio electrónico, puesto que:

Soporta múltiples estilos de generación de código dando a los codificadores flexibilidad para establecer sus propios estándares.

- El formato para los comentarios dentro del código puede ser personalizado.

- Generación automática de Java Doc incluyendo los Java Beans, con lo que se invierte menos tiempo en la documentación del código.
- Hace ingeniería en reversa de clases en Java en .jar, .cab y .zip
- Ayuda a construir aplicaciones robustas, utilizando el JDK 1.2 (Java Deployment Kit)
- El modelo y el código son sincronizados automáticamente.
- Soporte completo para CORBA 2.2, ayudando de esta manera a las empresas a estar dentro de los estándares.

2.7.3.3 Integración con el nuevo clearcase

La integración con el nuevo ClearCase permite tener un repositorio robusto con el almacenamiento de todos los artifacts de requerimientos, código fuente y documentación. Esto permite al equipo tener un ambiente multiusuario fuerte donde cada miembro pueda administrar y controlar sus cambios realizados dentro del ciclo de desarrollo. Esta nueva integración Rose 2000:

- Permite llevar con control de versiones en códigos generados en C++ y Ada.
- La interfaz común entre Windows y Unix permite la transición de archivos entre plataformas.
- Permite ejecutar comandos de ClearCase directamente desde el Rose.

2.7.4 BENEFICIOS

- Permite una mejor comunicación de los miembros del equipo.
- El Web Publisher permite a los usuarios de Rose tener sus modelos en archivos html, con lo que es posible que los modelos puedan ser accesados por todos miembros del proyecto, sin necesidad de que tengan instalado Rose en sus Pc's.

2.8 GENEXUS

Los sistemas son cada día más complejos por varias razones: hoy los empresarios necesitan y exigen bases de datos corporativas por que una de sus mayores prioridades es la información para apoyar la toma de decisiones y, paralelamente, en los últimos diez años hemos pasado de sistemas con pantallas de textos y diálogos muy modestos a sistemas gráficos con diálogos muy sofisticados. GeneXus genera los sistemas que necesitamos para la plataforma que más nos convenga.

Y, en el futuro, ante la aparición de nuevas tecnologías, todo el conocimiento atesorado por GeneXus le servirá a los clientes para regenerar sus sistemas. Simplemente deberán utilizarse los nuevos generadores GeneXus que incluyan esas nuevas tecnologías. Aprender a utilizar esta herramienta, va más allá del conocimiento de las primitivas del lenguaje de definición, puesto que lo más importante será la asimilación de esta nueva metodología

2.8.1 CONCEPTO

LA HERRAMIENTA INTELIGENTE GENEXUS es una nueva metodología y filosofía en las herramientas tipo CASE. para el Diseño, Desarrollo y Mantenimiento de Aplicaciones, misma que se concentra en:

- Generar y Administrar una Base de Conocimiento, con la utilización de Inteligencia Artificial y Diseño Asistido.
- Usar las técnicas CASE para generar bases datos relacionales y programas fuentes, bajo diferentes plataformas de ejecución.

La Herramienta Inteligente ***GENEXUS***, está basada en C++ y Prolog para un mejor desarrollo incremental de software. ***GENEXUS*** posee características que la hacen única, pues no se limita a la automatización parcial de aplicaciones como las herramientas tradicionales de desarrollo, sino que además, automatiza todo aquello que es automatizable; esto es, normalización, diseño, generación y mantenimiento de las bases de datos, programas de la aplicación y generación de Bases de Conocimiento.

Los beneficios que se obtienen con *GENEXUS* son muchos: va desde una interfaz estándar para todas las aplicaciones, esto es, prototipos, programas para las plataformas más difundidas en el mercado, hasta un rápido retorno de la inversión, basado en el aumento de la productividad de los analistas hasta **10 veces** en la etapa de desarrollo de aplicaciones y al menos **20 veces** en el mantenimiento de las mismas.

2.8.2 CARACTERÍSTICAS

- GeneXus simplifica toda esta complejidad con él se describe, de una forma muy natural nuestra realidad a nivel conceptual, a nivel de conocimiento puro e independientemente de los vaivenes de la tecnología.

2.8.3 COMPONENTES Y FUNCIONABILIDAD

2.8.3.1 Módulo Modelador / Prototipador

Es de carácter general y puede ser ejecutado en un microcomputador o en una red de microcomputadores. Este módulo permite la especificación, prototipación y generación para producción en microcomputador o en red de microcomputadores, en arquitectura

File/Server y en varios lenguajes: Java, Visual Basic, Visual Foxpro, Foxpro for Windows, Clipper y Foxpro para DOS.

GENEXUS genera prototipos de modelos (aplicaciones) que estructural y funcionalmente son idénticos a los de Producción, es decir, las pruebas hechas en el Prototipo son validadas y serán ejecutadas exactamente como se las hará en Producción.

Los elementos básicos de este módulo, entre otros son:

- Ambiente interactivo para la definición de objetos y especificación de requerimientos.
- Ambiente completo de prototipación de aplicaciones en el microcomputador.
- Automáticamente diseña la base de datos y mantiene la Base de Conocimiento.
- Permite definir redundancia (desnormalización) y genera las rutinas necesarias para mantenerlas.
- Automáticamente genera los programas de definición y/o de reorganización de la base de datos en Prototipo (microcomputador) y en Producción (Plataforma selecionada).
- Automáticamente genera los programas asociados a los objetos definidos, para prototipar totalmente la aplicación y para Producción (Plataforma de ejecución que Ud. escoge).

- Automáticamente genera los informes sobre los impactos que se ejecutarán en la base de datos, según los cambios que se requieren sobre las versiones anteriores existentes.

- Facilita la distribución/verificación de consistencia/consolidación de conocimiento entre aplicaciones desarrolladas por separado (equipos de desarrollo locales ó externos).

- Permite generar pantallas interactivas en la Internet con Visual Basic, Java y RPG/400 en Servidores Unix y AS/400.

- Genera Transacciones de Data Warehousing

2.8.3.2 Módulo Generador CLIENTE / SERVIDOR

De acuerdo a la definición de la aplicación probada y aprobada en el Prototipo, se puede seleccionar la plataforma de trabajo. Al momento ***GENEXUS*** puede trabajar con las siguientes **Bases de Datos**: Sql Server, DB2/400, DB2/6000, DB2/2, Oracle e Informix; así mismo, están disponibles los siguientes **Clientes**: Visual Basic, Foxpro for Windows, Visual Foxpro y C/Sql (Ver figura 2.20).

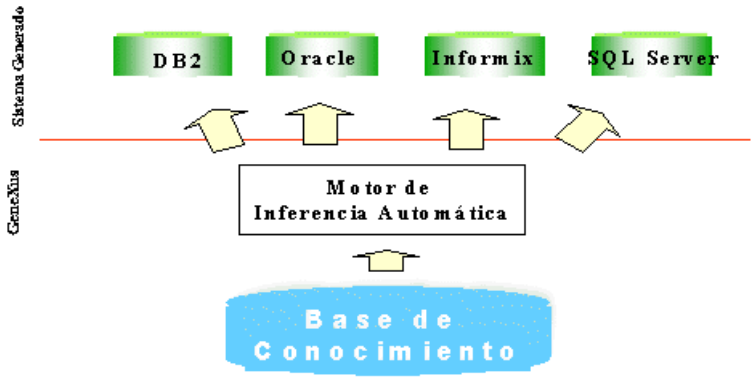


Figura 2.20 Sistema Generado

2.8.3.3 Módulo Generador para AS / 400:

De acuerdo a la definición de la aplicación probada y aprobada en el Prototipo, se puede seleccionar el lenguaje de generación en arquitectura Centralizada para el AS/400, pudiendo ser: RPG ó Cobol.

2.8.3.4 Módulo Generador C / S JAVA:

Este módulo le permite poner en producción las aplicaciones desarrolladas en una arquitectura Cliente/Servidor ó Internet con las mismas bases de datos del generador anterior, para lenguaje JAVA.

2.8.3.5 Módulo Generador Metadata Manager (DATA WAREHOUSE):

GENEXUS es la herramienta que permite diseñar y generar el Data Warehouse y los procedimientos de carga de datos desde la Base de Datos Operativa al Data Warehouse.

EL Metadata Manager permite realizar la administración automática de la Metadata, de manera de permitir el acceso dinámico desde el OLAP, en tiempo de ejecución.

2.8.4 BENEFICIOS

2.8.4.1 PARA EL USUARIO FINAL:

- Lo torna un participante esencial en el diseño y prototipación de sus aplicaciones y, como consecuencia, éstas se ajustan mucho mejor a sus necesidades.
- No hay sorpresas de último momento, no hay errores de programación que aparezcan en el momento más inoportuno.

2.8.4.2 Para el Analista:

- Lo libera de las tareas tediosas y mecánicas (en las cuales se cometen muchos errores) como diseño de la base de datos y programación y le permite dedicar su atención a lo esencial: entender el problema del usuario.

- Prototipación oportuna de todas las partes de la aplicación evitan la acumulación de trabajo, presiones e incertidumbres en el "día D".

- Propagación automática de cambios tanto en la base de datos como en los programas lo cual disminuye dramáticamente los costos de mantenimiento y le permite dedicarse a tareas creativas.

- Permite incorporar nuevas tecnologías en forma inmediata, con un mínimo estudio de las mismas.

2.8.4.3 Para el Empresario

- Le permite reaccionar más rápidamente ante las nuevas necesidades del mercado como, por ejemplo, el comercio electrónico y, como consecuencia, obtener ventajas comparativas para su negocio.

- Le permite un rápido retorno de la inversión, aumentando la productividad de los analistas hasta 10 veces en el desarrollo de aplicaciones y hasta 20 veces en su mantenimiento.

- Le permite escoger libremente la plataforma que se adapta mejor a sus necesidades de hoy y, en la medida que estas necesidades cambien, cambia de plataforma sin necesidad de re-invertir en el desarrollo de sus sistemas.

La sistematización del conocimiento, además de aumentar fuertemente la productividad, protege a la empresa cuando las personas que desarrollaron las aplicaciones dejan de estar disponibles.

En la tabla 2.1 detallamos las plataformas, requerimientos técnicos y de sistema operativo que las diferentes herramientas Case necesitan para su funcionamiento e instalación.

HERRAMIENTA CASE	PLATAFORMAS	REQUERIMIENTOS	
		TECNICO	SISTEMA OPERATIVO
BPWIN	DB2 SQL SERVER ORACLE SYBASE AS/400 INFORMIX CLIPPER FOXPRO ACCES PARADOX DBASE III DBASE IV	10 MB espacio en disco duro 32 MB RAM Ratón Microsoft o compatible Monitor VGA, SVGA	Windows 95,98,NT
ERWIN	DB2 SQL SERVER ORACLE SYBASE AS/400 INFORMIX CLIPPER FOXPRO ACCES PARADOX DBASE III DBASE IV	10 MB espacio en disco duro 16 MB RAM Procesador 486 Pentium o SUM SPARC	Windows 95,98,2000,NT Solaris 2.x
ER/Studio	ORACLE SYBASE MS SQL SERVER IMB DB2 UDB INFORMIX ONLINE INTERBASE SQL ANYWHERE ACCESS 2000 VISUAL FOXPRO	17 MB espacio en disco duro 32 MB RAM	Windows NT
SYSTEM ARCHITECT	AS/400 DB2 DBASE INFORMIX ORACLE PARADOX	10MB en disco duro 12 MB RAM Procesador 486 o mayor	

HERRAMIENTA CASE	PLATAFORMAS	REQUERIMIENTOS	
		TECNICO	SISTEMA OPERATIVO
POWER DESIGNER	DB2 SQL SERVER ORACLE SYBASE AS/400 INFORMIX CLIPPER FOXPRO ACCES PARADOX DBASE III DBASE IV	10 MB espacio en disco duro 12 MB RAM Procesador 486 o mayor Monitor VGA 10 MB espacio en disco duro 8 MB RAM Procesador 486 o mayor Monitor VGA	Windows 3.1,95, NT
ORACLE	DB2 SQL SERVER ORACLE SYBASE AS/400 INFORMIX CLIPPER FOXPRO ACCES PARADOX DBASE III DBASE IV	1 GB en disco duro 32 MB RAM Pentium 90 MHz SQL Net Client 2.3.4 SQL Plus 3.3.4	Windows 95,98,NT
RATIONAL ROSE	DB2 SQL SERVER ORACLE SYBASE	64 MB RAM Pentium 150MHZ	Windows 5,98,NT,2000,ME
GENEXUS	DB2 SQL SERVER ORACLE SYBASE INFORMIX AS/400 RPG/400 UNÍS	30 MB en disco duro 24 MB en memoria Procesador Pentium	Windows NT UNÍS LINUX

Tabla 2.1 Tabla de plataformas y requerimientos de las diferentes herramientas CASE

CAPITULO III

ANÁLISIS DEL SISTEMA

3.1 INTRODUCCIÓN

La Mutualista AMBATO fue constituida en abril 16 de 1963, en la ciudad de Ambato. Es una entidad de derecho privado cuya finalidad social es promover la formación de capitales para destinarlos bajo su propia administración, a la solución del problema habitacional del país y al mejoramiento del bienestar familiar de su asociados.

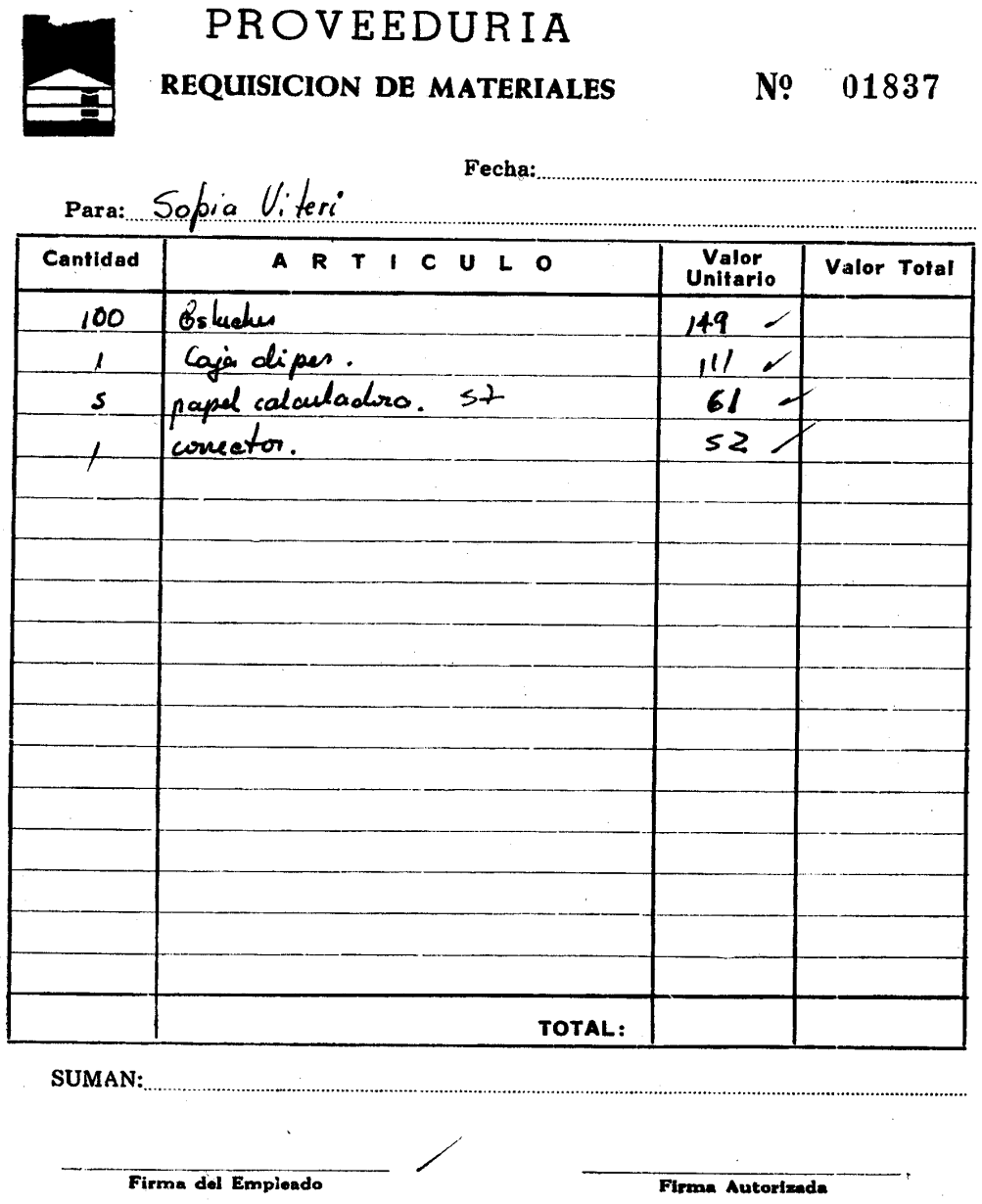
Las actividades y operaciones que realiza la Mutualista están regidas por la Ley General de Instituciones del Sistema Financiero

Para dar una mejor atención al público la Mutualista Ambato esta conformada por los departamentos de Atención al Cliente, Ahorros a la Vista, Ahorros a Plazo, Crédito, Cartera, Trabajo Social, Departamento Técnico, Contabilidad, Auditoria, Computo.

El Departamento de Contabilidad para mayor control de sus procesos, debe realizar Inventarios de compras de suministros de oficina e implementos de limpieza que realiza la Mutualista así como también la entrega de dichos artículos por parte de los empleados que laboran en la Institución

3.2 DEFINICION DE REQUERIMIENTOS DEL SISTEMA

Anteriormente la Mutualista Ambato llevaba su control de Inventarios en Fox para el D.O.S pero con el cambio del nuevo milenio este sistema quedo inutilizado, posteriormente este control lo realizaban manualmente, por medio de un boletín de PROVEEDURÍA (ver figura 3.1)



PROVEEDURIA

REQUISICION DE MATERIALES Nº 01837

Fecha:

Para: Sofia Viteri

Cantidad	A R T I C U L O	Valor Unitario	Valor Total
100	Esferos	149 ✓	
1	Caja diper.	111 ✓	
5	papel calculadora. 57	61 ✓	
1	corrector.	52 ✓	
	TOTAL:		

SUMAN:

Firma del Empleado

Firma Autorizada

Figura 3.1 Boletín de Proveeduría

en el cual registraban todas las entregas a los empleados y las compras archivaban las facturas , debiendo al final del mes pasar una hoja Excel lo cual les llevaba demasiado tiempo.

Con el fin de facilitar el trabajo de las personas que están a cargo del CONTROL DE INVENTARIO, se ha elaborado el SISTEMA DE PROVEEDURÍA el mismo que controlará todo lo referente a COMPRAS de artículos a los distintos proveedores y ENTREGAS de los mismos a los empleados de la institución.

Obteniendo finalmente :

- INVENTARIO FISICO el mismo que le permitirá a la persona encargada de Proveeduría saber con que cantidad de artículos cuentan para poder realizar oportunamente los pedidos a los distintos proveedores.
- Qué compras se realizaron en el mes
- Saber qué articulo y qué cantidad es lo que se entregó a cada empleado que labora en la institución.

Las funciones básicas que realizará este sistema son las siguientes :

- Registrar artículos, proveedores, empleados, compras y entregas .
- Calcular el valor del articulo automáticamente.
- Por medio de un trigger al realizar una compra de un artículo este aumentara en el inventario al igual que cuando se realizase una entrega a una empleado este controlará que se disminuya en el inventario.
- Realizar un prorrateo de precios.

3.3 METODO DE ANÁLISIS

El modelo de objetos siempre es necesario si vamos a hacer el Análisis Orientado a Objetos. Estrictamente, podemos dividir la etapa de análisis dentro de la metodología OMT(es una metodología OO de desarrollo de software basada en una notación gráfica

para representar conceptos OO. La metodología consiste en construir un modelo del dominio de aplicación e ir añadiendo detalles a este modelo durante la fase de diseño) en tres fases: modelo de objetos que representa la estructura estática de la información, modelo dinámico que indica la secuencia de eventos y modelo funcional. Sin embargo, si consideramos la fase inicial de conceptualización, debemos incluir también los casos de uso que muestra las transformaciones de datos.

Para poder diseñar los diagramas que se utiliza en el Análisis Orientado a Objetos se ha escogido POWER DESIGNER porque nos permite diseñar y generar esquemas de base datos a través de un verdadero modelamiento de base de datos relacionales de dos niveles (conceptual y físico) basado en métodos probados. A partir de un diagrama de clase, PowerDesigner automáticamente genera y realiza ingeniería reversa de ambientes populares como Java(incluyendo EJB 2.0), XML, Servicios Web, C++, PowerBuilder, VisualBasic y más a través de un generador personalizable.

3.3.1 CASOS DE USO

Un caso de uso está formado por una serie de interacciones entre el sistema y un *actor* (una entidad externa, ejerciendo un rol determinado), que muestran una determinada forma de utilizar el sistema. Cada interacción comienza con un evento inicial que el actor envía al sistema y continua con una serie de eventos entre el actor, el sistema y posiblemente otros actores involucrados.

Un caso de uso puede ser descrito en lenguaje natural o mediante diagramas de interacción de objetos.

A continuación en la tabla 3.1 se detalla en lenguaje natural los actores y los procesos que intervendrán en el sistema

ACTORES	PROCESOS
Usuario	Ingresa Artículos
Usuario	Ingresa Proveedores
Usuario	Ingresa Empleados
Usuario	Registra las compras
Usuario	Registra las entregas
Proveedor	Entrega los artículos pedidos
Empleado	Recibe los productos

Tabla 3.1 Lenguaje Natural

En las siguientes figuras (3.2,3.3 y 3.4) se presenta los actores y procesos mediante diagramas de interacción los cuales serán necesarias para el Análisis del sistema de proveeduría

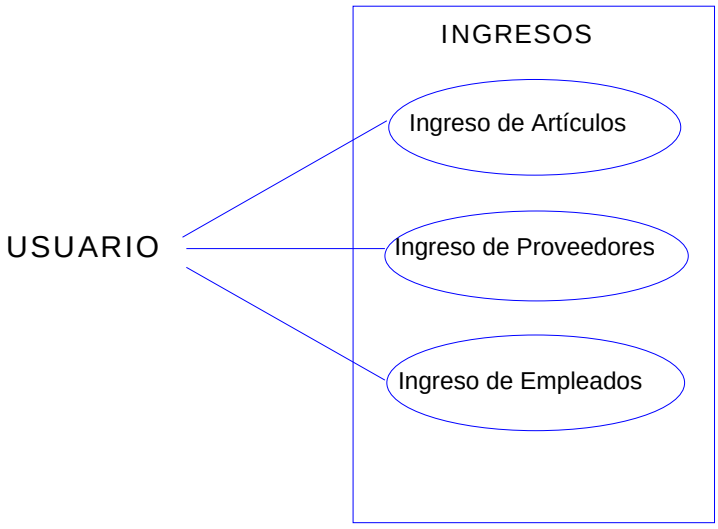


Figura 3.2 Diagrama de interacción de Ingresos

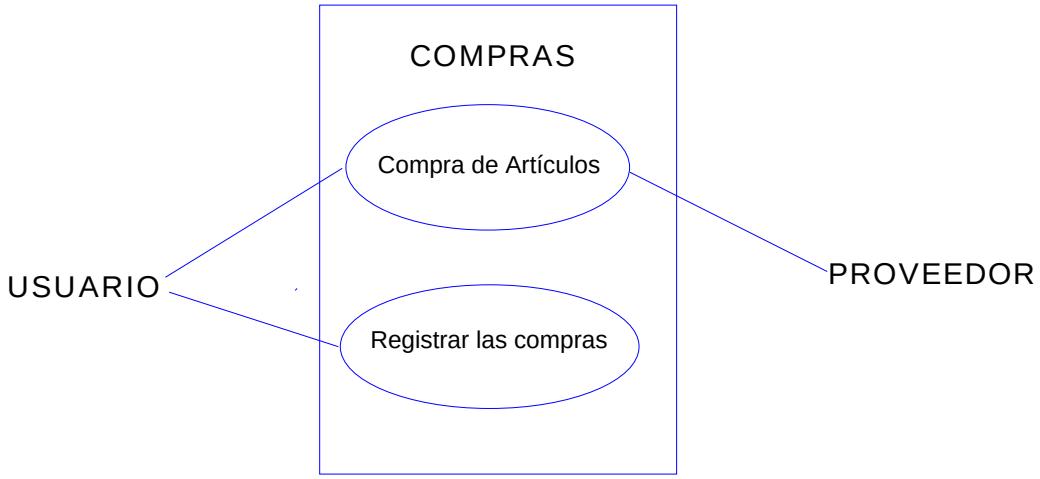


Figura 3.3 Diagrama de interacción de Compras

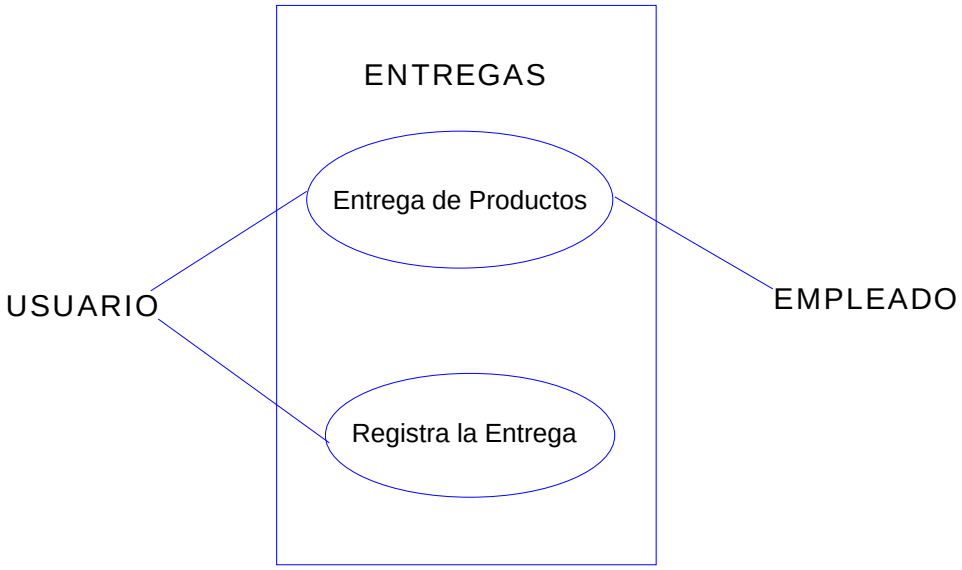


Figura 3.4 Diagrama de interacción de Entregas

3.3.2 MODELO DE OBJETOS

El modelo Objetos describe la estructura de los objetos de un sistema: su identidad, sus relaciones con otros objetos, sus atributos y sus operaciones.

El modelo de objetos se representa gráficamente con diagramas de objetos y diagramas de instancias, respectivamente.

A continuación se detalla los elementos que puede contener el modelo de objetos.

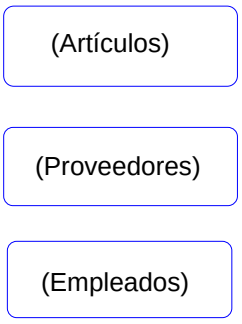
3.3.2.1 Objetos

Un objeto es un concepto, una abstracción o una cosa con unos límites definidos y que es relevante para el problema en cuestión. Una característica de los objetos es que tienen identidad y son distinguibles. Aunque dos objetos tengan los mismos valores para todos sus atributos son diferentes.

La mayoría de las instancias de una clase derivan su individualidad de tener valores diferentes en alguno/s de sus atributos o de tener relaciones con instancias diferentes. No obstante pueden existir instancias con los mismos valores de los atributos e idénticas relaciones.

El símbolo gráfico para representar instancias es un rectángulo de esquinas redondeadas. Dentro del rectángulo figura la clase a la que pertenece la instancia (entre paréntesis) y los valores de sus atributos.

Los objetos que intervienen en el sistema de PROVEEDURÍA son los siguientes :



3.3.2.2 Clases.

Una clase o clase de objetos es una abstracción que describe un grupo de instancias con propiedades (atributos) comunes, comportamiento (operaciones) común, relaciones comunes con otros objetos y (lo que es más importante) **una semántica común**.

El símbolo gráfico para representar clases es un rectángulo, en el que figura el nombre de la clase. Las clases se representan en los diagramas de clases, que son plantillas que describen un conjunto de posibles diagramas de instancias. Describen, por tanto el caso general.

3.3.2.3 Atributos

Un atributo es un dato contenido en todas las instancias de una clase. Cada atributo tiene un valor para cada una de las instancias. Varias clases pueden tener atributos comunes (por e ej. *nombre*, en las clases *Persona* y *Calle*) pero cada atributo debe ser único dentro de una clase.

Los atributos tienen que ser datos, no objetos. La diferencia entre unos y otros reside en la identidad: los objetos tienen identidad, pero los atributos no.

Los atributos se representan en el segundo área de los símbolos de clase e instancia. En las clases, figurará el nombre del atributo, el tipo y el valor por defecto. En las instancias, el valor del atributo para ese objeto determinado.

3.3.2.4 Operaciones

Una operación o método es una función o transformación. Cada operación lleva implícito un objeto sobre el que se va a realizara la operación. El comportamiento de la operación depende de la clase del objeto destino. Todos los objetos de una clase comparten las mismas operaciones o métodos. Las operaciones figuran en la tercer área del símbolo de las clases.

En la siguiente figura 3.5 vemos algunos de los elementos que intervienen en el Modelo de Objetos para el desarrollo del SISTEMA DE PROVEEDURÍA

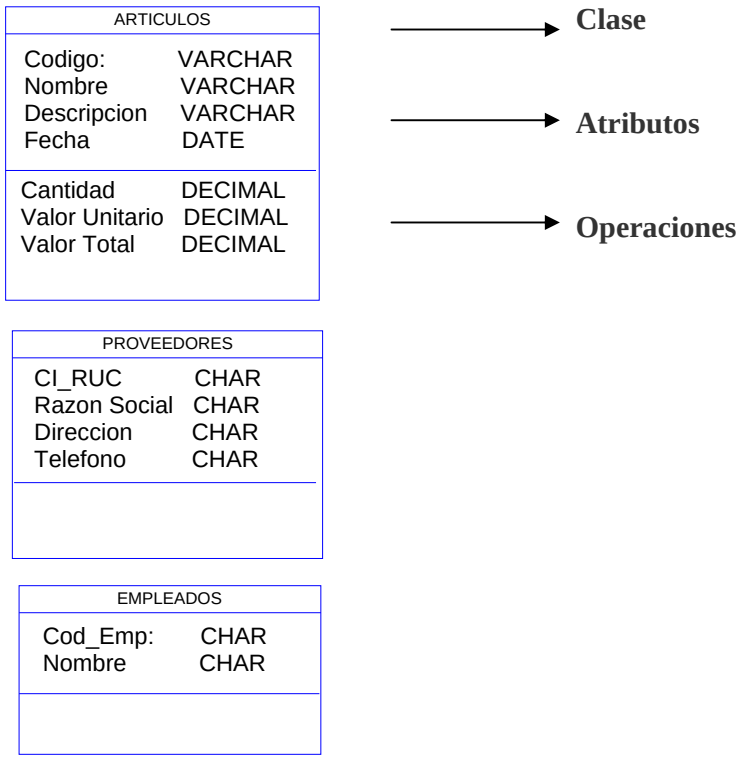


Figura 3.5 Componentes del Modelo de Objetos

3.3.2.5 Enlace

Es una conexión entre dos o mas instancias (objetos) como se muestra en la figura 3.6

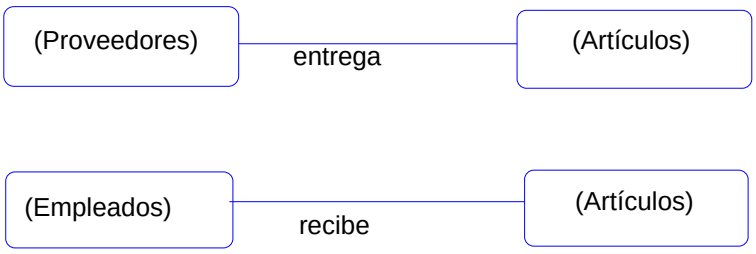


Figura 3.6 Representación de enlaces

3.3.2.6 Asociaciones

Una Asociación describe un conjunto de enlaces de la misma forma que una clase describe un conjunto de instancias como se representa en la figura 3.7

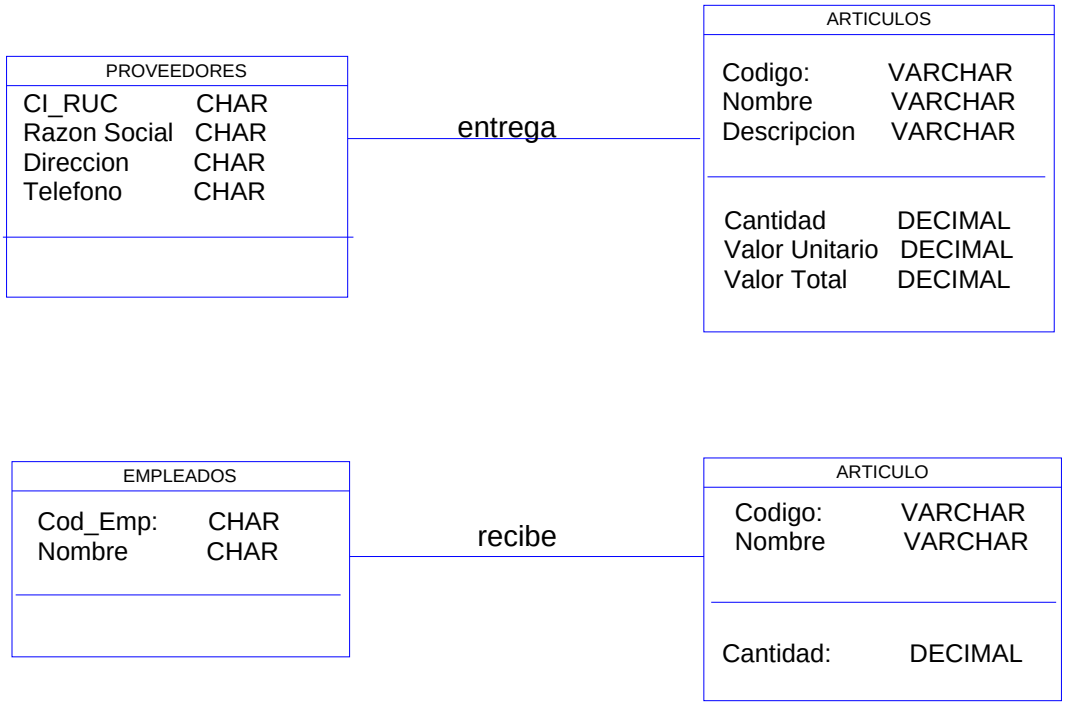


Figura 3.7 Representación de Asociaciones

3.3.2.7 Multiplicidad

Para representar el número de instancias de cada clase que pueden participar en una asociación utilizaremos la siguiente notación en cada extremo de la asociación:

○ Opcional. La asociación puede relacionar 0 ó 1 instancias de la clase

● Muchos. Significa de 0 a N.

3 Exactamente 3.

2,4 Dos o cuatro.

2-4 De dos a cuatro.

4+ Más de cuatro

En el Modelo Entidad Relación se puede observar la representación de multiplicada (ver figura 3.8)

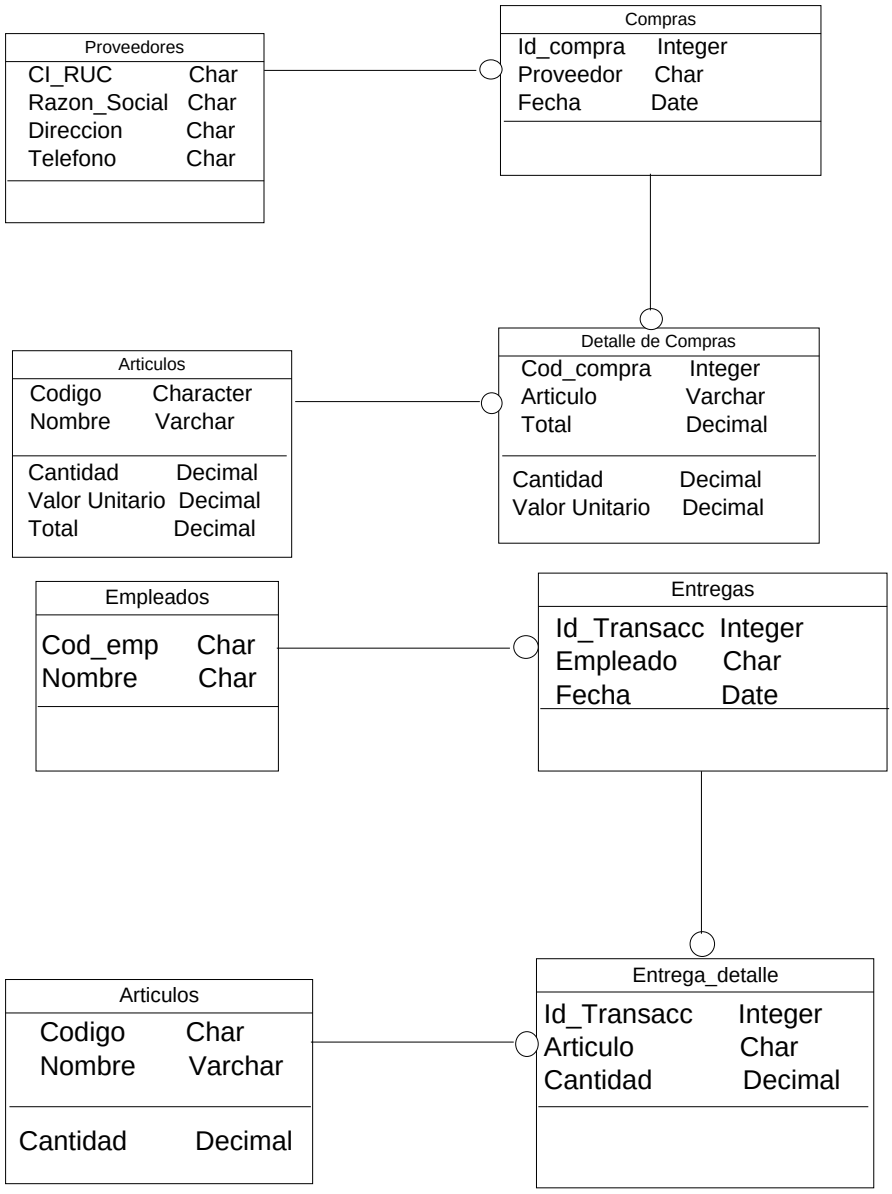


Figura 3.8 Representación de Multiplicidad

3.3.3 MODELO DINAMICO

El modelo dinámico describe las iteraciones temporales entre objetos representados (varios estímulos que ocurren y la respuesta del sistema a los estímulos). Los diagramas de eventos expresan un aspecto de comportamiento que es compartido por los objetos en una clase . No hay un diagrama de eventos para cada clase con comportamientos dinámicos ; la colección de diagramas de eventos Inter. actuantes conforma el **Modelo dinámico** (ver figura 3.10,3.11 y 3.12)

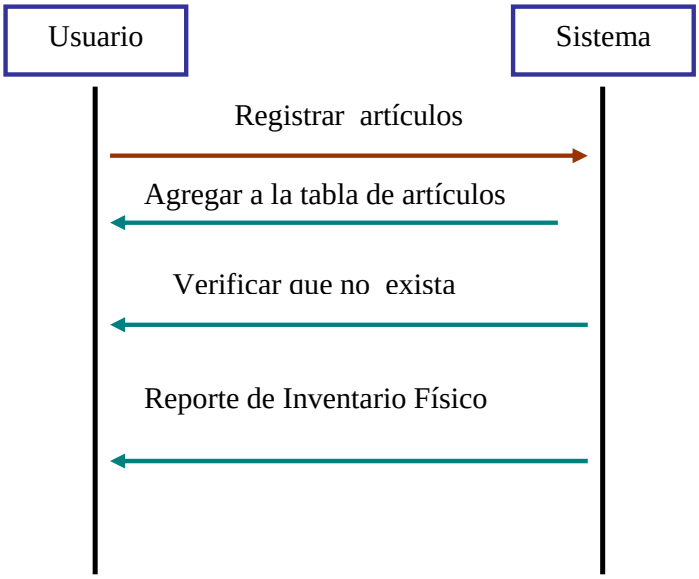


Figura 3.10 Escenario para Registrar Artículos

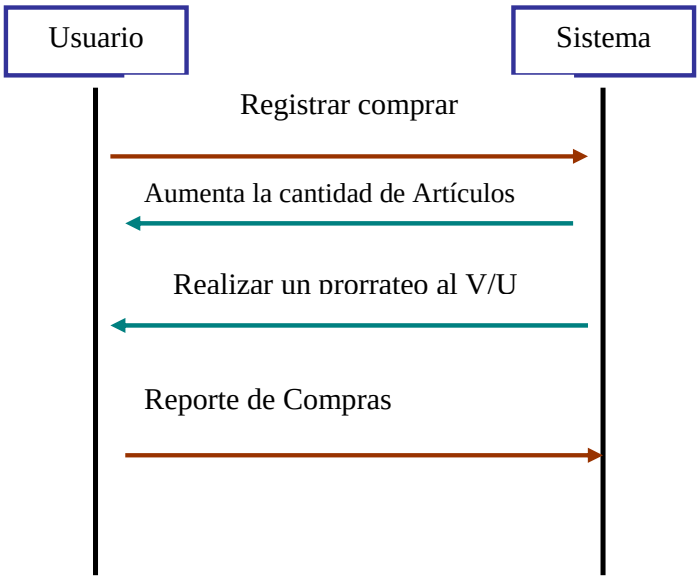


Figura 3.11 Escenario para Registrar Compras

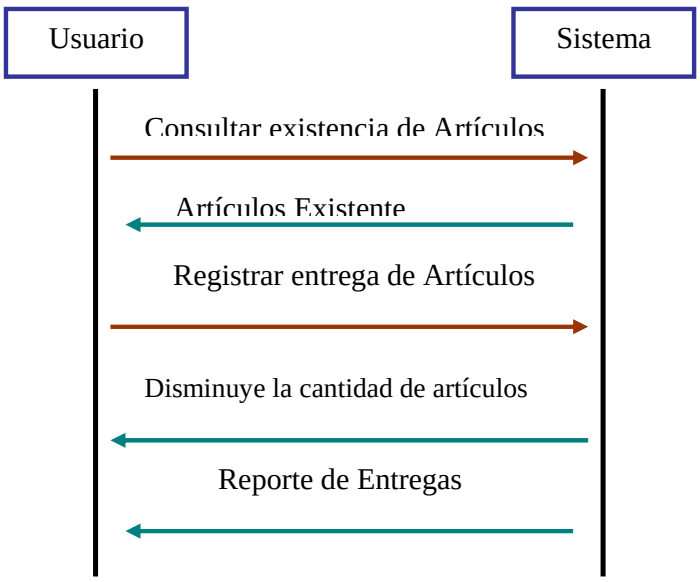


Figura 3.12 Escenario para la Entrega de Artículos

Patricia Carrillo S.
P.U.C.E.S.A

3.3.4 MODELO FUNCIONAL

El Modelo funcional describe las computaciones que se realizan en un sistema, mostrando como se derivan los valores de salidas a partir de las entradas El Modelo funcional se puede expresar mediante algún lenguaje formal, aunque la mayoría de los casos basta utilizar el lenguaje natural.

En las siguientes figuras se representa en lenguaje formal las entradas, proceso y salidas de cada uno de los módulos que intervendrán en el sistema de proveeduría .

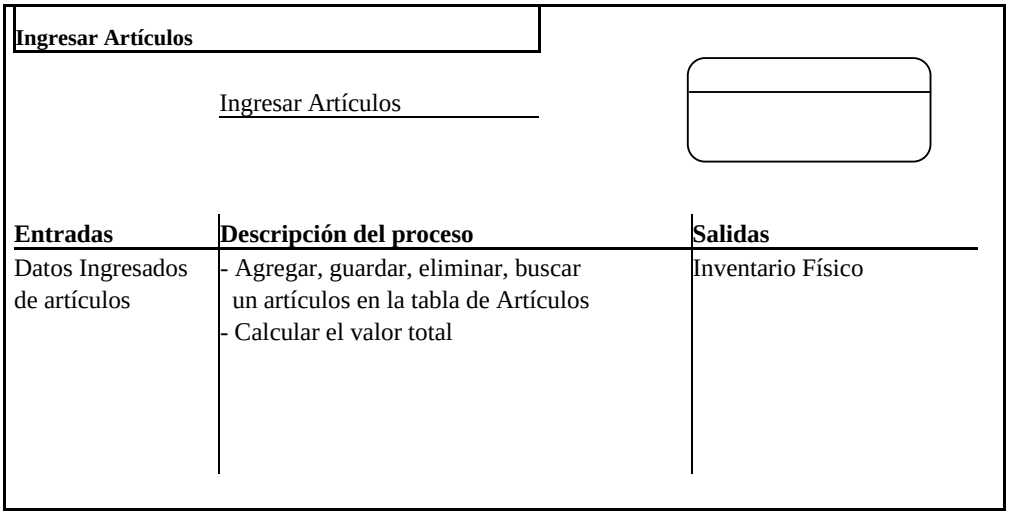


Figura 3.13 se representan las entradas, procesos y salidas del ingreso de Artículos

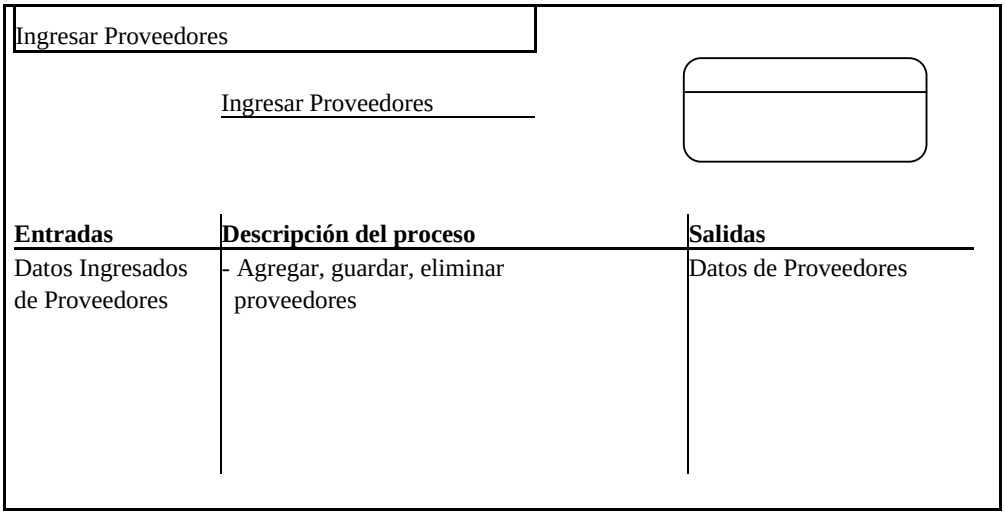


figura 3.14 se representan las entradas, procesos y salidas del ingreso de Proveedores

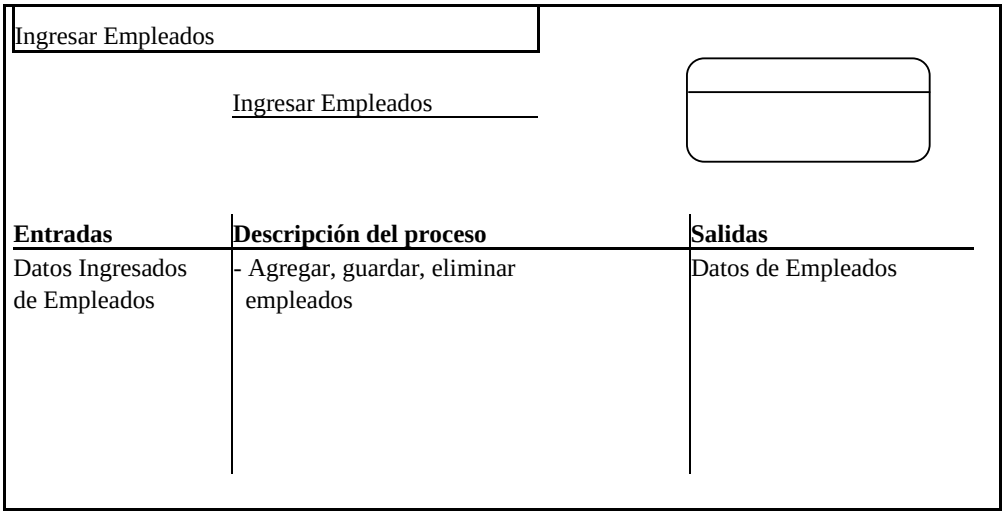
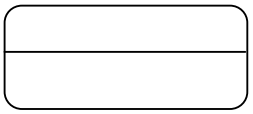


figura 3.15 se representan las entradas, procesos y salidas del ingreso de Empleados

Registrar Compras

Registrar Compras

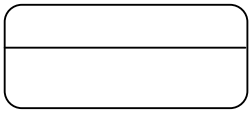


Entradas	Descripción del proceso	Salidas
- Ingresar código de la compra - Escoger el proveedor al cual se Lo realizo la compra -Escoger el producto comprado	- Calcular el valor unitario de acuerdo a la cantidad y el valor total - Si el producto comprado ya existe y tiene un valor de compra diferente al registrado anteriormente se realiza un prorrateo lo cual se lo realizará por un trigger	Reporte de Compras realizadas a los distintos proveedores

figura 3.16 se representan las entradas, procesos y salidas para registrar compras

Registrar Entregas

Registrar Entregas



Entradas	Descripción del proceso	Salidas
- Ingresar código de la entrega - Escoger el empleado al cual se Lo realizo la entrega -Escoger el producto entregado	- La cantidad del articulo disminuirá Lo cual será controlado por un Trigger	Reporte de entregas A los empleados de la Institución

figura 3.17 se representan las entradas, procesos y salidas registrar entregas

CAPITULO IV

DISEÑO DEL SISTEMA

4.1 INTRODUCCION

Es el proceso de determinar una implementación efectiva y eficiente que realice las funciones y tenga la información del análisis de dominio, es la fase donde se define la arquitectura del sistema.

El objetivo del diseño de sistema es refinar el modelo del análisis, los pasos que se llevan acabo son:

4.1.1 Organizar el sistema en subsistemas.

4.1.2 Seleccionar como se va a administrar el almacenamiento de datos.

4.1.3 Seleccionar la implementación del control del software

4.1.1 Organizar el sistema en subsistema

A continuación se detalla los subsistemas que intervendrán en el SISTEMA DE PROVEEDURÍA.

En las figura 4.1 se esquematiza el ingreso de artículos, esta pantalla permite ingresar los artículos que no existan en la tabla de artículos.

Figura 4.1 Ingreso de Artículos

En la figura 4.2 se representa la pantalla, ingreso de proveedores en donde se registra los proveedores que no existan en la tabla de proveedores.

Figura 4.2 Ingreso de Proveedores

En la figura 4.3 se representa la pantalla ingreso de empleados, esta pantalla permite el ingreso de los empleados que no existan en la tabla de empleados

Figura 4.3 Ingreso de Empleados

En la figura 4.4 se observa la pantalla de registro de compras , la cual permite registrar las compras realizadas a los distintos proveedores.

Figura 4.4 Registro de Compras

En la figura 4.5 se representa la pantalla de registro de entregas, en donde se almacenan las entregas de cada empleado

Figura 4.5 Registro de Entregas

4.1.2 Seleccionar como se va a administrar el almacenamiento de datos.

Para el almacenamiento de datos se ha escogido el administrador de Base de Datos Relacionales IBM/DB2 versiòn 6, que es la base de datos con la cual actualmente trabaja el sistema de producción de la Mutualista Ambato.

DB2 Universal Database es un sistema de gestión de bases de datos relacionales que tiene la potencia suficiente para satisfacer las demandas de grandes organizaciones y es lo suficientemente flexible para operar en pequeñas y medianas empresas.

La principal característica de DB2 es su capacidad de acceso a datos heterogéneos, proporcionando a las aplicaciones la posibilidad de funcionar con información procedente

de diversas fuentes como si se tratara de una sola base de datos, sin las limitaciones tradicionales de localización y origen .

4.1.3 Seleccionar la implementación del control del software

- Para ingresar al SISTEMA DE PROVEEDURÍA se debe conectar a la base de datos por medio de un password
- Otro de los controles del sistema es que no permite ingresar datos repetidos, en caso de haber datos repetidos aparecerán mensajes de advertencias como por ejemplo este código ya existe.
- Cuando no se ha registrado correctamente los datos el sistema se indicara que ocurrió un error en la transacción y esta no se realizó

4.2 DISEÑO DE OBJETOS

Su objetivo es refinar el modelo del análisis y proporcionar una base detallada para la implementación tomando en cuenta el ambiente en que se implementará

Los pasos que se realizan en el diseño de objetos son los siguientes:

4.2.1 Se busca una operación

Se define una operación para cada suceso del modelo funcional y dinámico, en las tablas 4.1,4.2,4.3,4,4,4.5 se describe los procedimientos para insertar, guardar , eliminar y calcular el valor total de un artículo .

```
procedure TFArticulos.NuevoExecute(Sender: TObject);
begin
datos.TArticulos.Insert; end;
end.
```

Tabla 4.1 Procedimiento para insertar

```
procedure TFArticulos.GuardarExecute(Sender: TObject);

begin

with datos do

    begin
        if TArticulos.Active then
        begin
            try
            TArticulos.ApplyUpdates;
            TArticulos.CommitUpdates;
            except
                application.messagebox('Ocurrio un error la transacción no se realizo
!!','Error',MB_ICONERROR);
            end;
        end;
        TArticulos.Active:=false;
        TArticulos.Active:=true;

    end;
```

Tabla 4.2 Procedimiento para guardar

```
procedure TFArticulos.EliminarExecute(Sender: TObject);
begin
      if application.MessageBox('Desea Eliminar el registro?','Advertencia',
      MB_YESNO+MB_ICONQUESTION)=6 Then
      begin
      with datos do
      begin
       TArticulos.Delete;
       datos.data.StartTransaction;
       try
        datos.TArticulos.ApplyUpdates;
        datos.data.Commit;
       except
        datos.data.Rollback;
        raise;
       end;
       datos.TArticulos.CommitUpdates;
      end
end;
end;
```

Tabla 4.3 Procedimiento para eliminar

```
TFArticulos.DBEvalorExit(Sender: TObject);
begin
if  (DBEcantidad.field.value <>null) and (DBEvalor.field.value <>null)  THEN
DBEtotal.field.value:=(DBEcantidad.Field.value*DBEvalor.Field.Value);
end;
```

Tabla 4.5 Procedimiento para calcular el valor total

4.2.2 Se diseña la implementación de asociaciones

Para facilitar el diseño de implementación de asociaciones se lo ha elaborado en PowerDesigner, herramienta CASE que facilita el diseño de base de datos, se escogió esta herramienta Case por la facilidad de adquirirlo y su fácil funcionamiento.

En la figura 4.6 se representa el modelo entidad relación de la base de datos el cual se lo genero en el DataArchitec de Power Designer

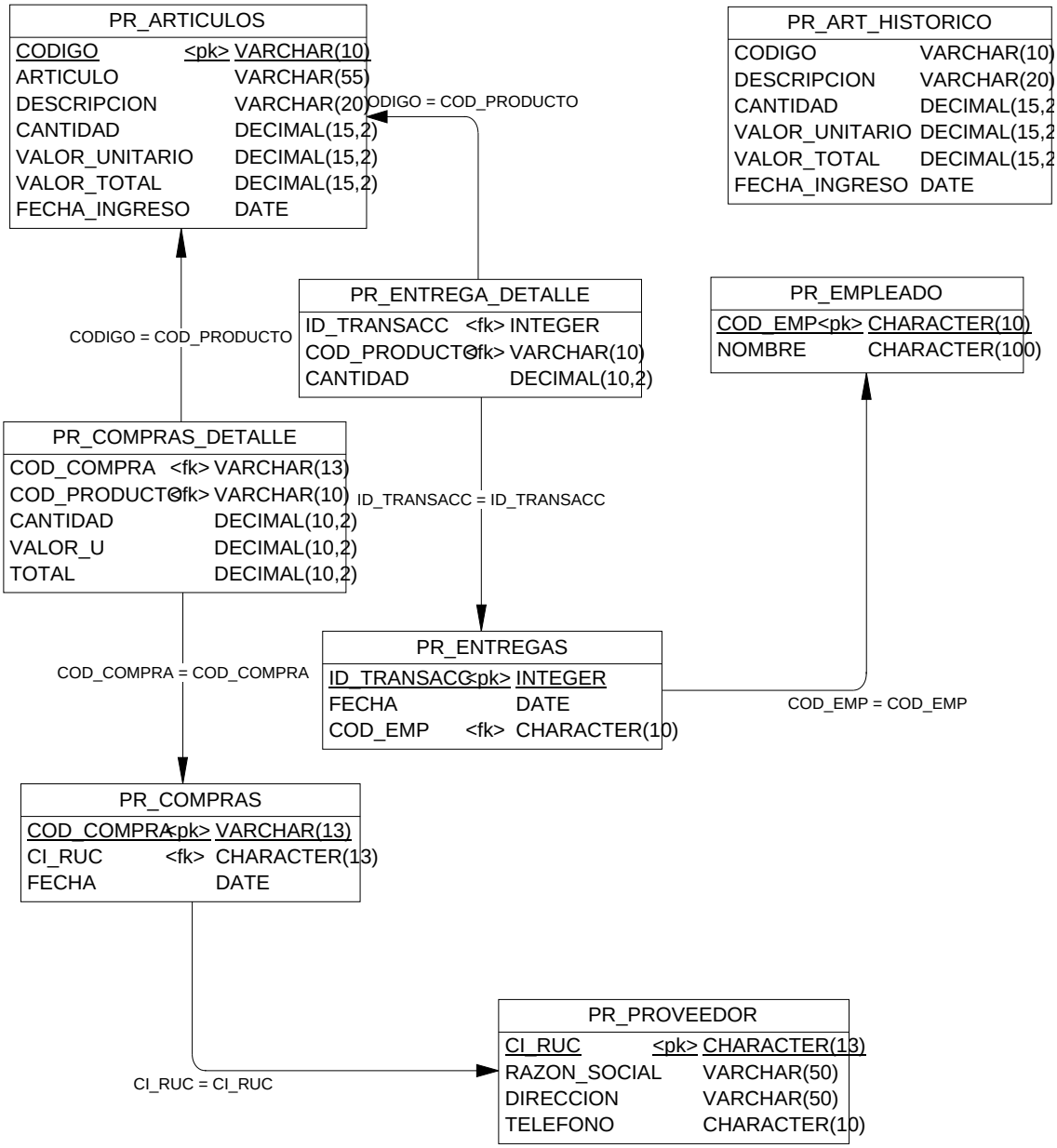


Figura 4.6 Modelo Entidad Relación de la base de Datos del Sistema de Proveeduría

CONCLUSIONES

Después de haber realizado el Análisis de Herramientas CASE , he llegado a las siguientes conclusiones:

- Todas las herramientas CASE tienen particularidades que las hacen mejores y deseables.
- Mantienen una sincronización con el código fuente.
- Reducen el riesgo de errores.
- Optimizan el tiempo total de Desarrollo en un sistema
- Las herramientas CASE ayudan al diseño más que a la documentación
- Permiten migrar datos sin grandes esfuerzos de adaptación entre las diferentes plataformas de hardware y sistemas operativos
- Las Herramientas Case PowerDesigner y Rational Rose permiten realizar todas las fases del ciclo de vida de un sistema orientado a Objetos.
- Tanto el análisis como el diseño del sistema de proveeduría fueron realizados en PowerDesigner por cuanto es de fácil manejo, bajo costo y de rápida instalación.

RECOMENDACIONES

Es recomendable utilizar una herramientas CASE porque:

1. Ayudan al Diseño de Base de datos con facilidad evitando de esta manera escribir sentencias muy largas y complicadas.
2. Se recomienda usar en la elaboración del ciclo de vida de un sistema Orientado a Objetos POWER DESIGNER o RATIONAL ROSE.

   PowerDesigner porque nos permite diseñar y generar el esquema de la base de datos a través de un verdadero modelamiento de bases de datos relacionales de dos niveles conceptual y físico. Gracias a su componente MODEL OBJECT .PowerDesigner completa el análisis y diseño usando técnicas UML (Lenguaje de Modelación unificado). A partir de un diagrama de clase, PowerDesigner automáticamente genera y realiza ingeniería reversa de ambientes populares como Java, XML, Servicios Web, C++, PowerBuilder(r), VisualBasic(r) y más, a través de un generador personalizable

   Rational Rose tiene un conjunto de herramientas de modelado visual para el desarrollo de soluciones robustas y eficientes a necesidades de negocio reales. Mediante esta herramienta se abarcan los conceptos de UML necesarios para un desarrollo óptimo. Rational Rose es capaz de generar partes del código para utilizar en el desarrollo como son declaración de clases, de atributos y de operaciones, facilitando de este modo, el trabajo del desarrollador

GLOSARIO

ADOOSI. Metodología de Análisis y Diseño Orientado a Objetos de Sistemas Informáticos.

CASE. (Ingeniería de Software asistidas por computadoras). Las herramienta CASE están dirigidas a los programadores. Proporcionan automatización en fases del desarrollo de programas o facilidades de documentación y comprensión.

CLIENTE/SERVIDOR. Se le suele llamar así a la arquitectura a dos capas, es decir, una capa servidor, u ordenador que contendrá los datos y los programas gestores asociados, y capas clientes, u ordenadores que se dirigiran al anterior para obtener la información.

CLIENTE quien toma los requerimientos entregados por el usuario por medio de la aplicaciones y los envía al servidor.

CORBA. Principalmente es una arquitectura o norma para el desarrollo de aplicaciones distribuidas. Es decir, permite definir la comunicación entre ordenadores mediante la utilización de "objetos" entendidos dentro de los sistemas de Programación Orientada a Objetos.

DATAWINDOW . Una datawindow es un objeto que contiene (entre otras cosas) una instrucción Select de SQL, y una representación visual de los datos que trata la misma.

DBA (Adiministrador de base de datos) La persona encargada del mantenimiento de la base.

DFD (Diagrama de Flujo de Datos) Los diagramas de flujos de datos también son llamados Carta de Burbujas.

DIAGRAMA. Es una representación gráfica que sirve para esquematizar un diseño cuando se trata la información.

DBMS (Sistema de manejo de bases de datos múltiples).

FIPS (Federal Information Processing Standards).

METADATOS.- Los metadatos consisten en información que caracteriza datos. Los metadatos son utilizados para suministrar información sobre datos producidos.

MULTI-USUARIO. Multiusuario significa que varios usuarios pueden trabajar directamente en el mismo módulo y para un mismo contribuyente o compañía.

OMT. Es una metodología Orientada a Objetos de desarrollo de software basada en una notación gráfica para representar conceptos.

RDBMS (*Relational Data Base Management System*) significa Sistema de Administración de Bases de Datos Relacionados.

SCRIPT. Son programas que se ejecutan en un servidor Web dedicados a procesar las peticiones que le llegan de los navegadores.

SERVIDOR. Es quien se encarga de mantener almacenada la información en la base de datos.

TRIGGERS(Trigger significa en inglés gatillo, disparador). Los triggers permiten ejecutar comandos Un trigger es un fragmento de código.

UML (Lenguaje de Modelación unificado*).* Es la unificación de metodologías de análisis y diseño destinadas a **Objetos**.

BIBLIOGRAFÍA

LIBROS

PRESMAN ,Ingeniería de Software , 4ta edición

KENDAL Y KENDAL, Análisis y Diseño de Sistema, 3ra Edición

TANENBAUM Andrew, Redes de Computadores

INTERNET

http://coqui.lce.org/erporto/sis-ab/metod/case.htm

http://ceds.nauta.es/Program/case1.htm

http://pegaso.org/aplicaciones/case.html

http://www.cs.queensu.ca/Software-Engineering/case.html

http:// inei.gob.pe/cpi/bancopub/libfree/lib667/co2.html

http://docencia.dgsca.unam.mx/cursos/cursos/temarios/sistdinfor/SIO3.html

http://www.uco.es/~ma1lurui/web-ccia/HCASE.html#principio

http://ceds.nauta.es/Program/case.htm

http://members.es.tripod.de/klauzen/notas.htm#is

http://www.inei.gob.pe/cpi/bancopub/libfree/lib615/

http://ca.com/channel/emea/

http://www.logicworks.com

http://www.abits.com.mx/Fabs/Rational/Rational.htm

http://www.elcomercioperu.com.pe/Pcwtema/Html/2000-10-17/csinternacio0021.html

http://www.usmp.edu.pe/publicaciones/boletines/fics/info7/bpwin.htm

http://sistemas.dgsca.unam.mx/publica/pdf/casestru.pdf

http://Monografias.com

INDICE

PAG

CAPITULO I HERRAMIENTAS CASE

1.1 Introducción..... 1
1.2 Definición de herramienta Case..... 5
1.3 Componentes..... 8
1.3.1 Repositorio..... 8
1.3.2 Módulo de diagramación y modelización 9
1.3.3 Generador de Código..... 10
1.3.4 Módulo Generador de Documentación..... 11
1.4 Bloques básicos de una herramienta Case..... 12
1.5 Clasificación de las herramientas Case..... 13
1.5.1 Herramientas Integradas..... 14
1.5.2 Herramientas que comprende algunas fases del ciclo de vida..... 14
1.5.3 Herramientas de planificación de Sistemas de Gestión..... 15
1.5.4 Herramientas de Análisis y Diseño..... 15
1.5.5 Herramientas de Programación..... 16
1.5.6 Herramientas de Integración..... 16
1.5.7 Herramientas de Gestión de Prototipos..... 16
1.5.8 Herramientas de Mantenimiento..... 16
1.5.9 Herramientas de Gestión de Proyectos..... 17
1.5.10 Herramientas de Soporte..... 17
1.6 Opciones de integración de las herramientas Case..... 18
1.6.1 Intercambio de Datos..... 18
1.6.2 Acceso Común a Herramientas..... 19
1.6.3 Integración de Datos..... 20
1.6.4 Integración total..... 21
1.7 Tipos de herramientas CASE..... 23
1.8 Características de las herramientas CASE..... 24
1.8.1 Herramientas de seguimiento de requisitos..... 24
1.8.2 Herramientas Métricas..... 24

Patricia Carrillo S.
P.U.C.E.S.A

PAG

1.8.4 Herramientas para Software de sistemas..... 25
1.8.5 Herramientas de Gestión de Base de Datos..... 25
1.8.6 Herramientas de Base de Datos y configuración de Software..... 25
1.8.7 Herramientas PRO/SIN..... 26
1.8.8 Herramientas para el Diseño y desarrollo de interfaces..... 26
1.8.9 Herramientas de codificación Convencional..... 27
1.8.10 Herramientas de codificación de Cuarta Generación..... 27
1.8.11 Herramientas de programación Orientada o Objetos..... 27
1.9 Estrategias de Implementación de herramientas CASE..... 28

CAPITULO II ESTUDIO COMPARATIVO DE LAS HERRAMIENTAS CASE

2.1 BPWIN
2.1.1 Concepto..... 31
2.1.2 Características..... 33
2.1.3 Componentes y Funcionabilidad..... 34
2.1.4 Beneficios..... 36

2.2 ERWIN
2.2.1 Concepto..... 37
2.2.2 Características..... 38
2.2.3 Componentes y Funcionabilidad..... 39
2.2.4 Beneficios..... 42

2.3 ER/Studio
2.3.1 Concepto..... 43
2.3.2 Características..... 44
2.3.3 Componentes y funcionabilidad..... 45
2.3.4 Beneficios..... 48

PAG

2.4 SYSTEM ARCHITECT

2.4.1 Concepto..... 49

2.4.2 Características..... 50

2.4.3 Componentes y Funcionabilidad..... 51

2.4.4 Beneficios 55

2.5 POWER DESIGNER

2.5.1 Concepto..... 56

2.5.2 Características..... 57

2.5.3 Componentes y Funcionabilidad..... 58

2.5.4 Beneficios..... 62

2.6 ORACLE DESIGNER

2.6.1 Concepto..... 65

2.6.2 Características..... 66

2.6.3 Componentes y Funcionabilidad..... 67

2.6.4 Beneficios..... 75

2.7 RATIONAL ROSE

2.7.1 Concepto..... 77

2.7.2 Características..... 77

2.7.3 Componentes y Funcionabilidad..... 77

2.7.4 Beneficios..... 79

2.8 GENEXUS

2.8.1 Concepto..... 80

2.8.2 Características..... 81

2.8.3 Componentes y Funcionabilidad..... 81

2.8.4 Beneficios..... 84

PAG

CAPITULO III ANÁLISIS DEL SISTEMA

3.1 Introducción..... 89
3.2 Definición y requerimientos del sistema..... 89
3.3 Método de Análisis..... 91
3.3.1 Casos de Uso..... 92
3.3.2 Modelo de Objetos..... 94
3.3.3 Modelo Dinámico..... 100
3.3.4 Modelo Funcional..... 102

CAPITULO IV DISEÑO DEL SISTEMA

4.1 Introducción..... 105
4.2 Diseño de Objetos..... 109