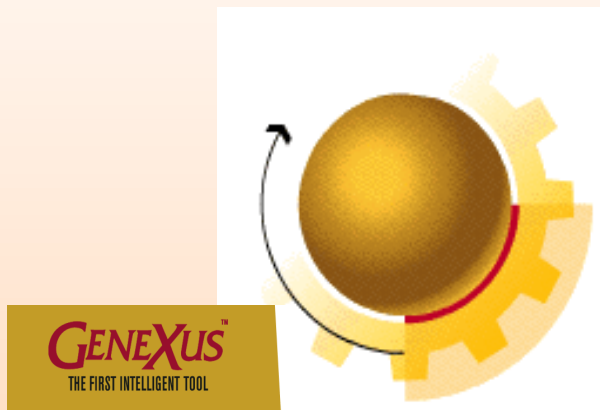


CAPÍTULO I

INTRODUCCIÓN A LAS HERRAMIENTAS CASE



- ❖ Generalidades
- ❖ Definiciones de Herramientas CASE
- ❖ Clasificación de las Herramientas CASE
- ❖ Objetivos de las Herramientas CASE
- ❖ Estructura de las Herramientas CASE
- ❖ Proceso de Desarrollo de Software con Herramientas CASE
- ❖ Ponderaciones para la Utilización de Herramientas CASE
- ❖ Requerimientos para Adquirir Herramientas CASE
- ❖ Estrategias para la Correcta Implantación de Herramientas CASE
- ❖ Recuperación de Costos de las Herramientas CASE

1.1. GENERALIDADES

El principal objetivo de los desarrolladores de aplicaciones es: elaborar software de calidad, a precios cómodos y en tiempos mínimos, de tal manera que sus aplicaciones sean compatibles para la mayoría de: plataformas, bases de datos arquitecturas y frontales existentes.

Para lograrlo se ha buscado herramientas que permitan: solucionar, consolidar, acoplar y unir los diferentes requerimientos que surgen de los compradores de sistemas. Una de las principales alternativas es la utilización de Herramientas CASE, donde el desarrollador desde: un ambiente gráfico de diseño, obviando las partes automatizables, proyectándose hacia una cultura de programación ordenada, enfocado al desarrollo de su sistema cumpliendo los estándares recomendados por la ingeniería de software, puedan desarrollar soluciones de gran nivel competitivo.

El desarrollador de sistemas debe ser polifuncional, y no centrarse en un solo lenguaje de programación y una base de datos específica; debe estar preparado para desarrollar soluciones capaces de ejecutarlas en diferentes arquitecturas, sin que esto requiera cambios de fondo en la estructura medular del sistema maestro o de referencia.

Para comprender este preámbulo de una mejor manera se presenta el siguiente ejemplo: El cliente A, desea un sistema de contabilidad, y dispone de: SQL Server, Visual Basic y Windows NT. El cliente B, quiere el mismo sistema de contabilidad realizado con: Informix para Linux, y Java. La semana siguiente otro cliente C, desea un sistema de contabilidad similar para: AS/400 con DB/2 en el servidor y Foxpro para Windows en el cliente.

Si se quiere satisfacer todos estos requerimientos, se tiene dos alternativas: 1. Contratar a tres desarrolladores diferentes que sean expertos en cada una de las arquitecturas requeridas. 2. Desarrollar una sola aplicación con una Herramienta CASE que sea capaz de generar todas las aplicaciones con los requisitos mencionados.

De hecho, lo más conveniente es la solución dos, ya que se elaboraría un solo programa o prototipo y se lo generaría para cada arquitectura requerida, el gráfico 1.1 y 1.2 muestra como una herramienta CASE es capaz de generar aplicaciones para múltiples lenguajes y bases de datos diferentes.

Los conceptos y las formas básicas de desarrollo convencional de sistemas son de mucha importancia, por ejemplo: el diseño de las bases de datos desde un lenguaje SQL puro o el desarrollo de una aplicación con programación netamente en C++ o en HTML; el buen programador debe conocer solidamente todos estos métodos, para tener una concepción adecuada sobre la automatización de los mismos por parte de las Herramientas CASE.

Cuando el desarrollador de aplicaciones a estudiado a fondo estas formas universales de desarrollo ya está en capacidad de adoptar una Herramienta CASE para la construcción de soluciones de software.

Gráfico 1.1

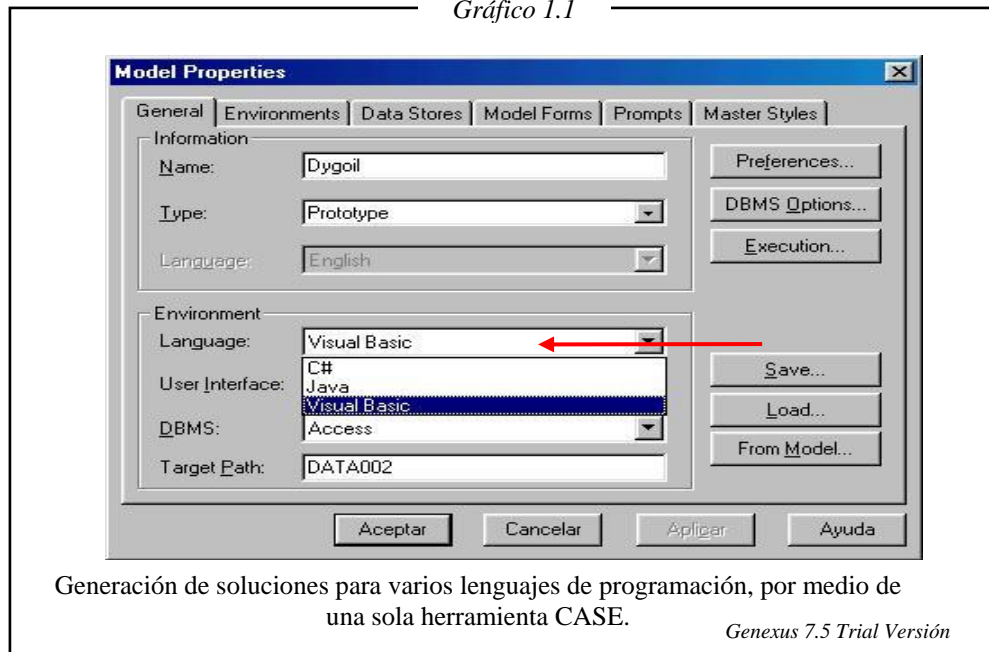
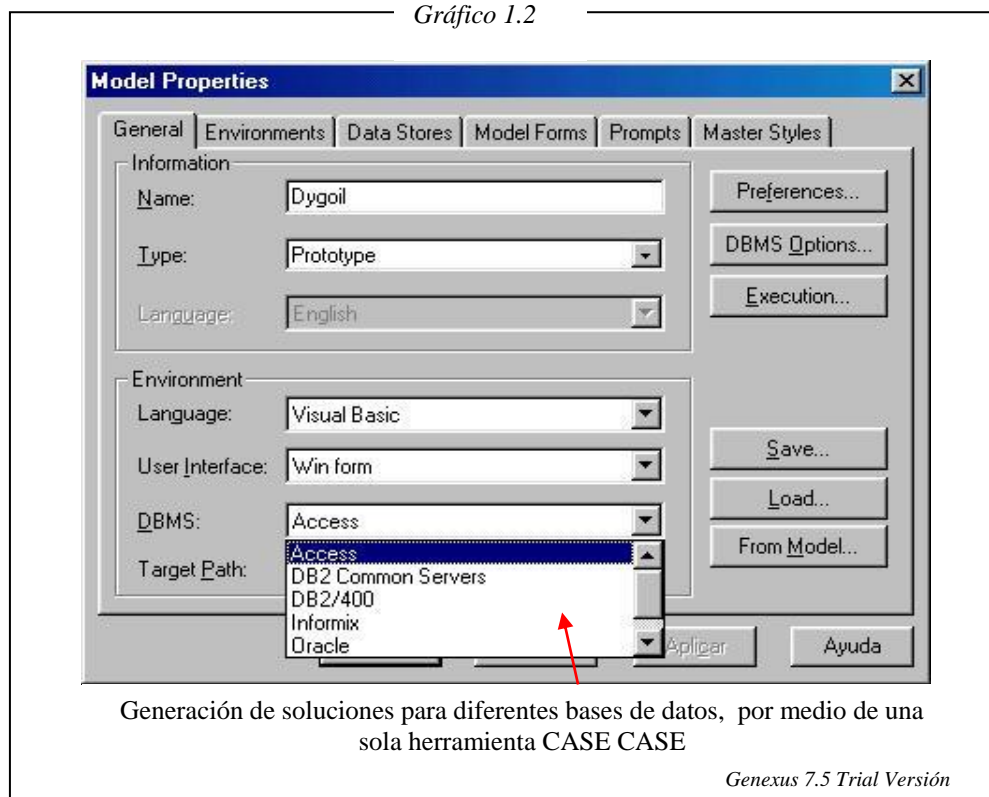


Gráfico 1.2



Otro aspecto importante del por qué se enfoca la utilización de Herramientas CASE, es el ámbito laboral de los Ingenieros de Sistemas, al egresar estos profesionales deben ser entes generadores de empleo, con miras a la creación de su propia empresa desarrolladora. En nuestro medio el desarrollo de aplicaciones con herramientas que permitan satisfacer la mayoría de requerimientos de un cliente, a tiempos mínimos, a precios cómodos, con facilidades de mantenimiento, de utilización, instalación, escalabilidad y que vayan de acuerdo a las necesidades cotidianas, todavía no es muy común, y es una alternativa de trabajo que está en pleno auge para su explotación.

En definitiva la tecnología CASE, proporciona un conjunto de herramientas semiautomatizadas y automatizadas, ligadas a una cultura ordenada y sistemática de desarrollo, que está enfocada a conseguir la generación automática de programas desde una especificación netamente a nivel de diseño. [lib002] [www002] [www003] [www027]

1.2. DEFINICIONES DE HERRAMIENTAS CASE

CASE: Computer Aided Software Engineering. (Ingeniería de Software Asistida por Computadora.)

CASE: "Son un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases." (Calderi Irivict) [www003]

CASE: "Es la automatización del software." (Carma Mac Clure) [www004]

Una Herramienta CASE es un programa especializado en el control y desarrollo de aplicaciones informáticas, siguiendo alguna de las metodologías más extendidas como los diagramas de control de flujo de Yourdon o los diagramas Entidad / relación de P.P. Chen para la normalización de bases de datos. En grandes equipos de desarrollo, el sistema queda centralizado a través de un diccionario de datos o "repositorio" que coordina los desarrollos de todos los participantes.

Henry David Crockett (Portland State University) "Las Herramientas CASE se ven simplemente como herramientas que cualquiera puede escoger y utilizar (como un martillo) para desarrollar un sistema de información, su selección e implementación casi siempre llevará a una reducida productividad y calidad. La selección e implementación de Herramientas CASE son procesos de múltiples etapas que permiten errores fatales en cada etapa. Uno de los errores más comunes es escoger una herramienta CASE que apoye un método desconocido para los diseñadores".

Alan Chimura (CASE Associates) dice "Las Herramientas CASE incluyen manejadores, métodos, técnicas, disciplina, e instrucciones, todos trabajando juntos. Definir Herramientas CASE menos ampliamente y presentarlo sin un suficiente entorno de apoyo es un acto de negligencia".

Las Herramientas CASE abarcan cada etapa del proceso de ingeniería y cada actividad que se desarrolla a lo largo del mismo. Las Herramientas CASE forman un conjunto de bloques que comienzan en el nivel del hardware y del sistema operativo y acaban en cada una de las herramientas del sistema. [www005]

Las Herramientas CASE se refieren a herramientas para el desarrollo de sistemas que constan de cinco componentes: herramientas de diagramación, depósito de información, generadores de interfaces, generadores de código y herramientas de administración. Las Herramientas CASE hacen hincapié en las actividades de alto nivel, aunque el objetivo a largo plazo es abarcar las actividades de análisis, diseño y desarrollo. [www006]

CASE: Es 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 a automatizar. [www008]

1.2.1. EL PROCESO DE DESARROLLO DE SOFTWARE

El proceso de desarrollo de software consiste en una serie de pasos bien definidos, que seguidos adecuadamente, conducen a un software mantenible y bien diseñado, aún así, muchas organizaciones olvidan las fases de análisis y diseño a favor de comenzar inmediatamente la implementación de código.

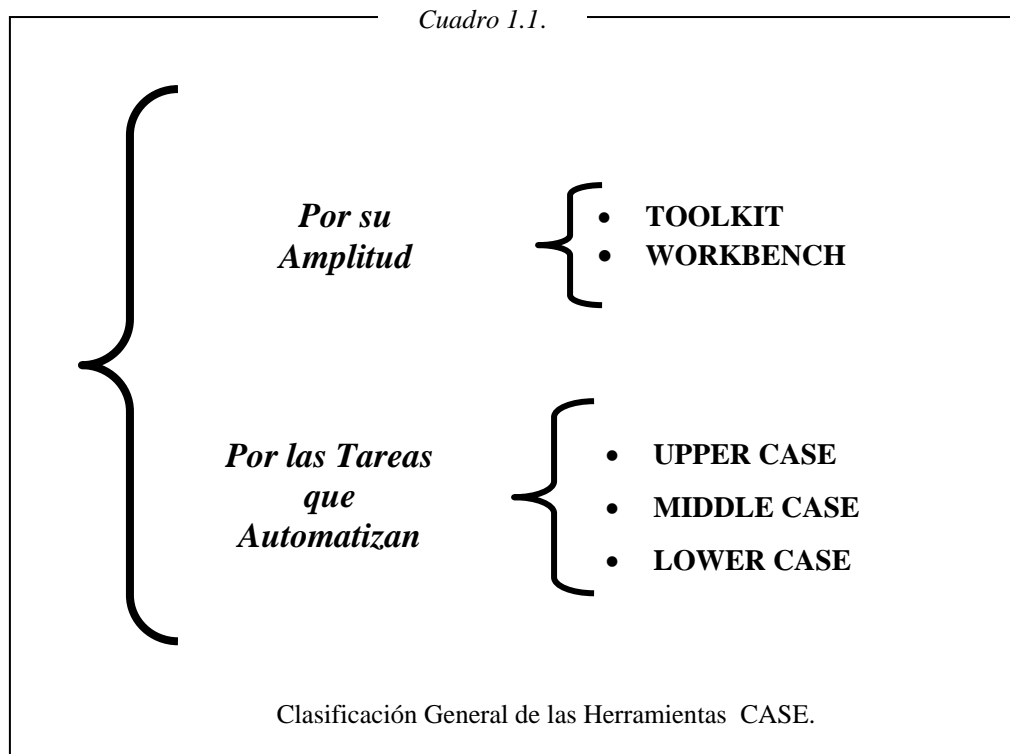
Es más positivo pensar en el desarrollo de software, no como un proceso lineal, sino como un ciclo, aunque el paso de una fase a otra se realiza en un sentido, también pueden existir vueltas atrás en determinados momentos, especialmente cuando aparecen requerimientos de usuario ocultos en las primeras fases, a continuación se muestra cuales son las principales facetas para desarrollar software. [www027]

1. Análisis de Requerimientos
2. Diseño de la Especificación (Prototipo)
3. Implementación (Producción)
4. Integración, Tes y Documentación.
5. Mantenimiento
6. Reingeniería

1.3. CLASIFICACIÓN DE LAS HERRAMIENTAS CASE

Las Herramientas CASE por su complejidad, no tienen una clasificación específica de sus tipos, varios autores las clasifican de diferente manera de acuerdo a la forma de ver e interpretar las cosas. Una de las más importantes clasificaciones de Herramientas CASE se observa en el cuadro 1.1. [www003]

1.3.1 CLASIFICACIÓN GENERAL DE LAS HERRAMIENTAS CASE



POR SU AMPLITUD

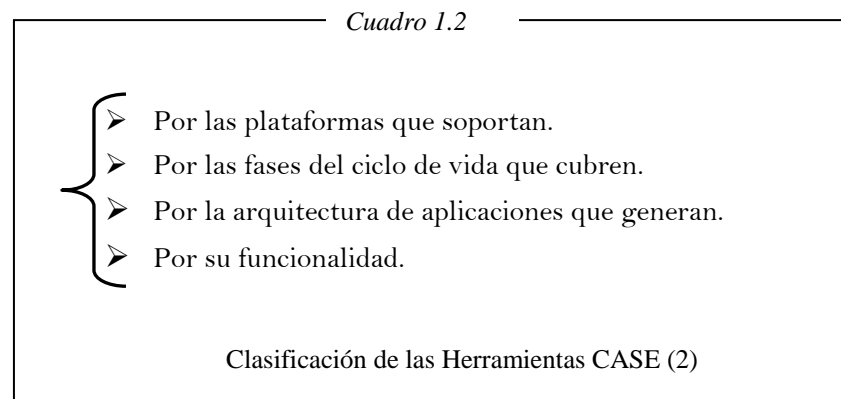
- **TOOLKIT**: Es una colección de herramientas integradas que permiten automatizar un conjunto de tareas en algunas de las fases del ciclo de vida del sistema informático: planificación estratégica, análisis, diseño o generación de programas, un ejemplo de estas herramientas es: Power Designer de Sybase.
- **WORKBENCH**: Son conjuntos integrados de herramientas que dan soporte a la automatización del proceso completo de desarrollo del sistema informático; permiten cubrir todo el ciclo de vida; el producto final aportado por ellas es un sistema en código ejecutable, tal es el caso de GENEXUS de Artech.

POR LAS TAREAS QUE AUTOMATIZAN

- ***UPPER CASE***: Planificación estratégica, requerimientos de desarrollo funcional de planes corporativos, como Vicio de Microsoft.
- ***MIDDLE CASE***: Análisis y diseño, como Designer de Oracle.
- ***LOWER CASE***: Generación de código, test e implantación, como Genexus de Artech.

1.3.2. OTROS TIPOS DE CASE

Como se muestra en el cuadro 1.2 las Herramientas CASE también pueden clasificarse de la siguiente manera: [www003]



POR LAS PLATAFORMAS QUE SOPORTAN

- Case Uniplataforma
- Case Multiplataforma

POR LAS FASES DEL CICLO DE VIDA QUE CUBREN

- I-CASE. Integrated CASE, CASE integrado: Abarcan todas las fases del ciclo de vida del desarrollo de sistemas. Son llamadas también CASE workbench.
- TOOLKIT. Herramientas que cubren alguna de las partes del desarrollo.

POR LA ARQUITECTURA DE APLICACIONES QUE GENERAN

- Case Centralizadas.
- Case Cliente Servidor. (Dos Capas)
- Case Cliente Servidor. (Multicapa)

1.3.3. CASE QUE GENERAN UNA PARTE DEL CICLO DE VIDA

- ***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.
- ***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 solo una fase dentro del ciclo de vida del sistema. [www003]
- ***Herramientas I – CASE.*** Se basan en una metodología, tienen un repositorio y aportan técnicas estructuradas para todas las fases del ciclo de vida, con esto se logra mayor calidad de desarrollo, 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 prototipéo.

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.

1.3.4 TIPOS DE CASE POR SU FUNCIONALIDAD

- ***Herramientas de Planificación de Sistemas de Gestión.*** Sirven para modelar los requisitos de información estratégica de una organización. Proporcionan un "meta modelo" 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.
- ***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 interfases.
 - ❖ Máquinas de análisis y diseño.

-
-
- ***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 (4GL)
 - ❖ Herramientas de programación orientadas a los objetos

 - ***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

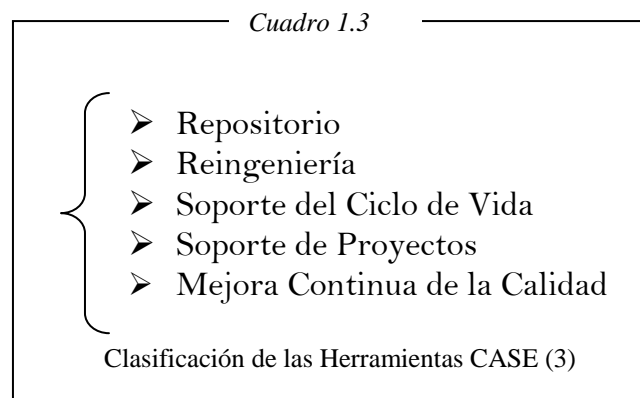
 - ***Herramientas de Gestión de Prototipos.*** Los prototipos son utilizados ampliamente en el desarrollo de aplicaciones, para la evaluación de especificaciones de un sistema de información, o para un mejor entendimiento de cómo los requisitos de un sistema de información se ajustan a los objetivos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

 - ***Herramientas de Mantenimiento.*** La categoría de las herramientas de mantenimiento se puede subdividir en:
 - ❖ Herramientas de ingeniería inversa
 - ❖ Herramientas de reestructuración y análisis de código
 - ❖ Herramientas de reingeniería

 - ***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 de sistemas, hacer un seguimiento que va desde los requisitos del pliego de prescripciones técnicas iniciales, 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

- **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: [www027]
 - ❖ Herramientas de documentación
 - ❖ Herramientas para software de sistemas
 - ❖ Herramientas de control de calidad
 - ❖ Herramientas de bases de datos

1.3.5 OTRA CLASIFICACIÓN DE LAS HERRAMIENTAS CASE



- **REPOSITORIO.** Funcionan en torno a un repositorio central, siendo éste el núcleo fundamental que contiene todas las definiciones de objeto y sus relaciones. Los objetos pueden ser especificaciones del sistema en forma de diagramas de flujo de datos, diagramas entidad-relación, esquemas de bases de datos, diseños de pantallas, etc. El repositorio es un concepto más amplio que el de diccionario de datos y soporta a los demás grupos de funciones. No es fácil encontrar en el mercado productos CASE con funcionalidades estrictamente a las de repositorio, ya que, a pesar de su innegable importancia, tienen un carácter auxiliar de los demás grupos de funciones. Cualquier sistema Case poseerá un repositorio propio o bien, trabajará sobre un repositorio suministrado por otro fabricante o vendedor.
- **REINGENIERÍA.** Los sistemas CASE permiten establecer una relación estrecha y fuertemente formalizable entre los productos generados a lo largo de las distintas fases del ciclo de vida, permitiendo actuar en el sentido especificaciones y código (ingeniería "directa") y también en el contrario (ingeniería "inversa"). Ello facilita la realización de modificaciones en la fase más adecuada en cada caso y su traslado a las demás. Al conjunto de facilidades proporcionadas por la ingeniería "directa" e "inversa" se le denomina "reingeniería".

- **SOPORTE DEL CICLO DE VIDA.** El ciclo de vida de una aplicación o de un sistema de información se compone de varias etapas, que van desde la planificación de su desarrollo hasta su implantación, mantenimiento y actualización. Aunque el número de fases puede ser variable en función del nivel de detalle que se adopte, pueden de modo simplificado, identificarse las siguientes:

- ❖ Planeamiento
- ❖ Análisis y Diseño
- ❖ Implantación (programación y pruebas)
- ❖ Mantenimiento y actualización

Los sistemas CASE pueden cubrir la totalidad de estas fases o bien especializarse en alguna(s) de ellas. En este último caso se puede distinguir sistemas de "alto nivel" ("Upper Case"), orientados a la autonomía y soporte de las actividades correspondientes a las dos primeras fases y, sistemas de "bajo nivel" ("Lower Case"), dirigidos hacia las dos últimas. Los sistemas de "alto nivel" pueden soportar un número más o menos amplio de metodologías de desarrollo.

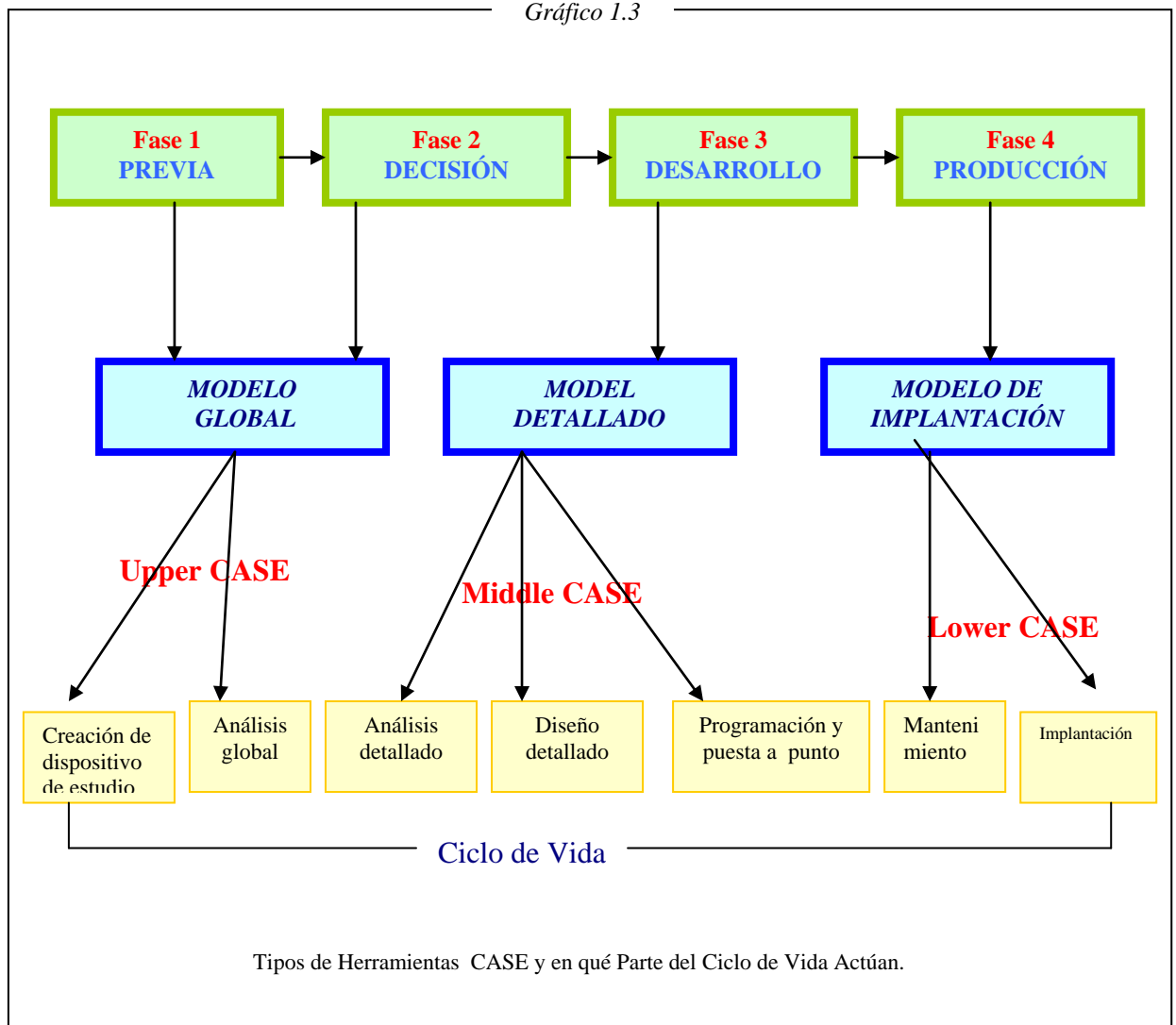
- **SOPORTE DE PROYECTOS.** Este tipo de funciones hace referencia al soporte de actividades que se producen durante el desarrollo, derivadas fundamentalmente del trabajo en grupos, tales como facilidades de comunicación, soporte a la creación, modificación e intercambio de documentación, herramientas personales, controles de seguridad, etc. Los sistemas CASE pueden conceder a estas cuestiones una importancia variable por lo cual el soporte de proyecto constituye un factor de diferenciación.
- **MEJORA CONTINUA DE LA CALIDAD.** Aunque frecuentemente se asocia a los sistemas CASE con la mejora de la productividad en el desarrollo de aplicaciones, debe tenerse en cuenta que una de las principales ventajas estriba también, en la mejora de la calidad de los desarrollos realizados. Determinados sistemas CASE enfatizan más sobre este punto que sobre el anterior, introduciendo herramientas que permiten ejercer un control intenso de garantía de calidad del software desarrollado desde las primeras fases de su ciclo de vida. [www003] [www027]

1.3.6. VENTAJAS Y DESVENTAJAS DEPENDIENDO DEL TIPO DE HERRAMIENTA CASE UTILIZADA

TIPO	VENTAJAS	DESVENTAJAS
Upper CASE	Se utiliza en arquitecturas para PC y es aplicable en diferentes entornos Menor Costo	Mejora la calidad pero no la productividad. Permite la integración del ciclo de vida.
Lower CASE	Mejora la productividad a corto Plazo. Buen soporte al mantenimiento.	No garantiza la persistencia en niveles corporativos. No garantiza la eficiencia de análisis y diseño. No permite la integración del ciclo de vida.
I – CASE	Integra el ciclo de Vida. Mejora la productividad a mediano plazo. Buen soporte de mantenimiento. Mantiene la persistencia en niveles corporativos.	No es eficiente para niveles simples, sino para complejos. Depende del hardware y software. Costos elevados.

1.3.7. CASE EN EL CICLO DE VIDA DE UN SISTEMA

En el gráfico 1.3 se puede observar qué partes del ciclo de vida pueden generar los diferentes tipos de Herramientas CASE. [www004]



1.4. OBJETIVOS DE LAS HERRAMIENTA CASE

- Aumentar la productividad de las áreas de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la calidad del software desarrollado.
- Reducir tiempos y costos 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 o enciclopedia de conocimientos y sus facilidades de uso, reduciendo la dependencia de analistas a los programadores.
- Automatizar:
 - ❖ El desarrollo del software
 - ❖ La documentación
 - ❖ La generación del código
 - ❖ El chequeo de errores
 - ❖ La gestión del proyecto
 - ❖ La generación del modelo de datos
 - ❖ La generación de diagramas
 - ❖ La generación de pantallas
- Permitir:
 - ❖ La reutilización del software
 - ❖ La portabilidad del software
 - ❖ La estandarización de la documentación
- Integrar las fases de desarrollo con ingeniería del software y Herramientas CASE.
- Facilitar la utilización adecuada y ordenada de las distintas metodologías, que propone la ingeniería del software. [www004] [www006] [www008] [www0027]
- Crear software de calidad, partiendo desde una metodología sistemática de desarrollo.
- Someter al desarrollador a las tareas específicas de análisis y diseño.
- Automatizar las partes automatizables.

1.5. ESTRUCTURA DE LAS HERRAMIENTAS CASE

Es muy difícil especificar cuál es la estructura exacta de una Herramienta CASE, ya que no todas tienen las mismas características unas de otras, a continuación se presenta la estructura más general a manera de niveles o módulos. (cuadro 1.4)

Cuadro 1.4

- Módulo de Repositorio
- Módulo de Diagramación y Modelamiento
- Módulo de Prototipado
- Módulo de Generación de Código
- Módulo de Generación de Documentación

Estructura General de una Herramienta CASE

1.5.1. MÓDULO DE REPOSITORIO O ENCICLOPEDIA

En el contexto de las Herramientas CASE se entiende por enciclopedia a la base de datos que contiene toda la información relacionadas con: las especificaciones, análisis y diseño de la aplicación, definiendo a cada objeto de la siguiente manera:

- **Datos:** Elementos, atributos (campos), asociaciones (relaciones), entidades (registros), almacenes de datos y estructuras.
- **Procesos:** Procesos, funciones y módulos.
- **Gráficos:** Diagramas de flujo de datos, diagramas entidad relación, diagramas de descomposición funcional, diagramas de estructura de árbol y diagramas de clases.
- **Reglas:** Reglas de gestión de métodos y comportamiento de datos y procesos.

El 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. En algunas referencias se le denomina *Diccionario de Recursos de Información* y en otros casos se le llama **Base de Conocimiento**, si a una Herramienta CASE se la considera como un Sistema Experto, el repositorio sería la fuente de alimentación de la información.

La mayoría de Herramientas CASE poseen un repositorio propio o bien trabajan sobre un repositorio suministrado por otro fabricante o vendedor, es preferible trabajar con herramientas que tengan su propio repositorio ya que se adaptan a la tecnología propia de la herramienta y no tienen que heredar metodologías de otro fabricante.

Apoyándose en la existencia del repositorio se efectúan comprobaciones de integridad y consistencia, para que no existan los siguientes errores: [www003]

- Que no existan datos no definidos.
- Que no existan datos autodefinidos o sea datos que se emplean en una definición pero que no han sido definidos previamente.
- Que todos los alias o referencias a un mismo dato, empleando nombres distintos sean correctos y estén actualizados.

CARACTERÍSTICAS DE UN REPOSITORIO

Tipo de Información. Que contenga alguna metodología concreta, para formar: datos, gráficos, procesos, informes, modelos y reglas.

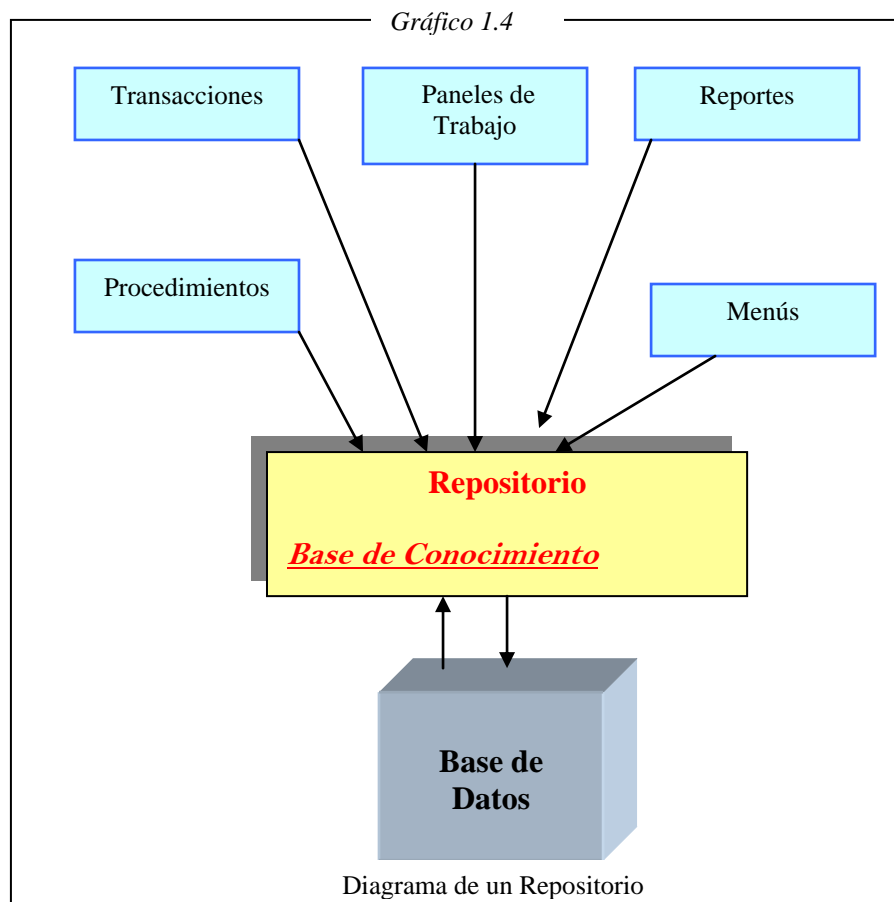
Tipo de Controles. Si incorpora algún módulo de gestión de cambios, de mantenimiento de versiones, de acceso por clave, de redundancia de la información. La gestión de cambios y el mantenimiento de versiones, ayudarán en el caso de que convivan diferentes versiones de la misma aplicación o se tengan 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 (match). 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.

Interfases Automáticas con Otros Repositorios y Bases de Datos Externas. Ayudan a la migración de aplicaciones, bases de datos, datos, y en si a la migración de toda la aplicación, ya sea como procesos de reingeniería o como procesos de cambios de versión de la herramienta, se toma como procesos de reingeniería a las aplicaciones y bases de datos realizadas con otras herramientas y se las acopla a la tecnología de una Herramienta CASE y los procesos de cambio de versión actúan cuando se quiere migrar aplicaciones de una versión más antigua a una versión más actual, sin que esto lleve a un cambio de fondo en la programación de la aplicación; en el gráfico 1.4. se observa el módulo de repositorio.



1.5.2. MÓDULO DE DIAGRAMACIÓN Y MODELAMIENTO

Es la parte del sistema, donde el programador diseña la realidad, dicho de otra manera diseña lo que desea ver propiamente en su programa, luego de esto la Herramienta CASE automáticamente genera los diferentes diagramas, como: [www003]

- Diagramas de flujo de datos
- Diagramas entidad relación
- Historia de vida de las entidades
- Diagramas de estructura de datos
- Diagrama de estructura de cuadros
- Diagramas de estructura de árboles
- Técnicas matriciales

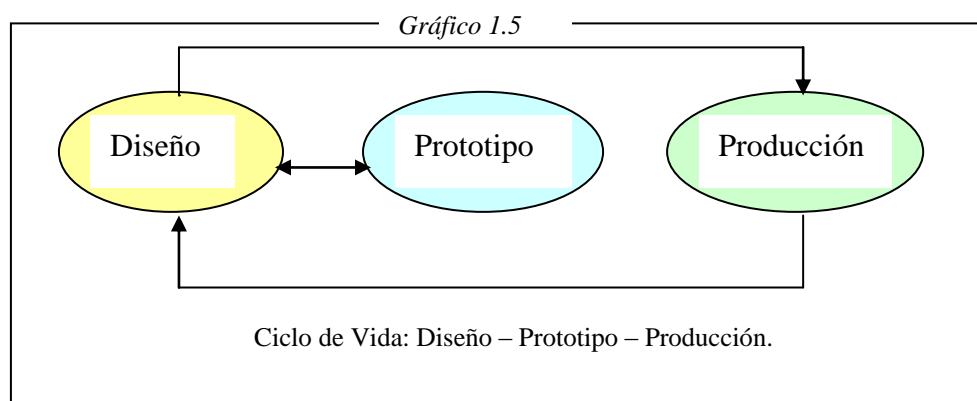
CARACTERÍSTICAS QUE SE DEBEN CONSIDERAR EN LOS DIAGRAMAS

- 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 pseudo código 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.

1.5.3. MÓDULO DE PROTOTIPADO

El objetivo principal de esta herramienta es poder mostrar al usuario desde los momentos iniciales del diseño, el aspecto que tendrá la aplicación una vez desarrollada. Esto facilitará a que el software cuente con la mayoría de las necesidades que pueda tener el usuario, antes de que la aplicación entre a producción.

El programador diseñara el software conjuntamente con el usuario, haciéndole ver como se van a presentar en pantalla los criterios requeridos por él. En el gráfico 1.5 se observa el ciclo de vida de diseño, prototipo y producción. [LIB001]



La herramienta será mucho más útil, cuanto más rápidamente permita la construcción del prototipo y cuanto más antes, se consiga la implicación del usuario final en el diseño de la aplicación.

Asimismo, es importante poder aprovechar como base el prototipo para la construcción del resto de la aplicación. Actualmente, es imprescindible utilizar productos que incorporen esta funcionalidad por la cambiante tecnología y necesidades de los usuarios.

Los prototipos han sido utilizados ampliamente en el desarrollo de sistemas tradicionales ya que proporcionan una realimentación inmediata, que ayudan a determinar los requisitos del sistema. Las Herramientas CASE están bien dotadas, en general, para crear prototipos con rapidez y seguridad.

En el ciclo de prototipo, el diseñador recorrerá repetidamente el bucle Diseño – Prototipo durante la fase de diseño, construyendo y probando sucesivos prototipos del modelo.

En el ciclo de producción por el contrario, pasará con menos frecuencia el bucle Diseño – Producción, ya que la generación del sistema se realiza solamente cuando el prototipo ha sido totalmente aprobado, o luego de haber instrumentado y probado algún cambio. [LIB001]

1.5.4. MÓDULO DE GENERADORES 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, en los gráficos 1.1 y 1.2 Se puede observar como una Herramienta CASE genera código automáticamente en diferentes lenguajes.

CARACTERÍSTICAS DE LOS GENERADORES DE CÓDIGO

- ***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 interfase de usuario de la aplicación. [www003]

1.5.5. MÓDULO GENERADOR DE DOCUMENTACIÓN

El módulo generador de documentación se alimenta del repositorio para transcribir las especificaciones allí contenidas, a este módulo se suman la ayuda en línea, los help o las ayudas del programa.

CARACTERÍSTICAS DE LOS GENERADORES DE DOCUMENTACIÓN

- 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 en el tratamiento de textos, es la facilidad para la introducción de textos complementarios a la documentación que se genera de forma automática.
- Interfase con otras herramientas: procesadores de textos, editores gráficos, etc.

1.6. PROCESO DE DESARROLLO DE SOFTWARE CON HERRAMIENTAS CASE

1.6.1. ARQUITECTURA JERÁRQUICA DE LAS HERRAMIENTAS CASE

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, constituyen la base de las Herramientas CASE.

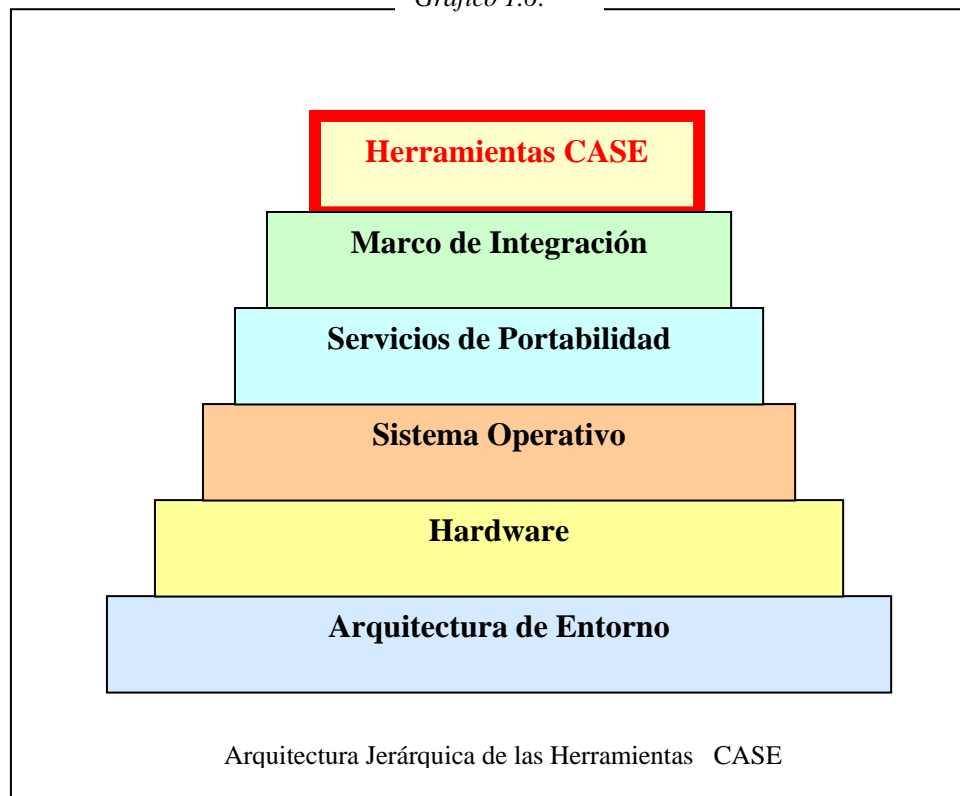
Pero el entorno de las Herramientas CASE en sí mismo necesitan otros componentes; un conjunto de servicios de portabilidad forman un puente entre las Herramientas CASE con su marco de integración y la arquitectura de entorno.

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 y al 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. En el gráfico 1.6. se visualiza la arquitectura jerárquica de las Herramientas CASE.

[www005]

Gráfico 1.6.



La mayoría de las Herramientas CASE no han sido construidas utilizando todos los bloques componentes, muchas de éstas son soluciones puntuales, esto es, una herramienta se utiliza como ayuda en una actividad concreta de ingeniería de software, pero no se comunica directamente con otras herramientas porque no está unida a una base de datos de proyectos; aunque esta situación no es la ideal, una Herramienta CASE puede ser utilizada eficientemente, aún siendo una solución puntual.

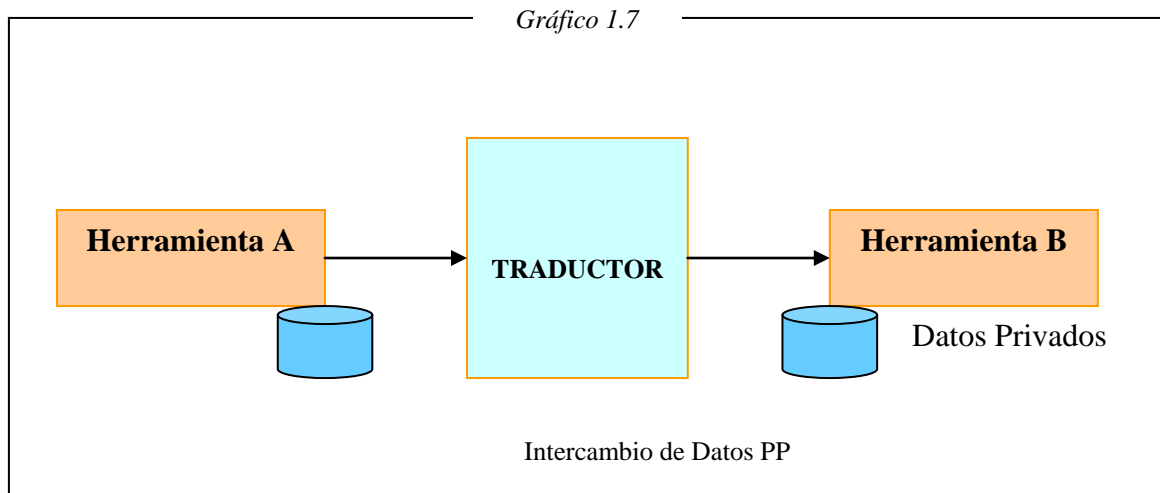
1.6.2. OPCIONES DE INTEGRACIÓN

La meta ideal de las Herramientas CASE es crear herramientas que integren en conjunto los siguientes elementos: [www003]

- Interfases 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

Para esto es necesario estudiar las formas comunes de integración como: el intercambio de datos punto a punto, el acceso común a herramientas, las formas de integración etc.

1.6.2.1 INTERCAMBIO DE DATOS PUNTO A PUNTO



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 (gráfico 1.7), utilizando normalmente un "filtro" de transmisión intermedio.

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.

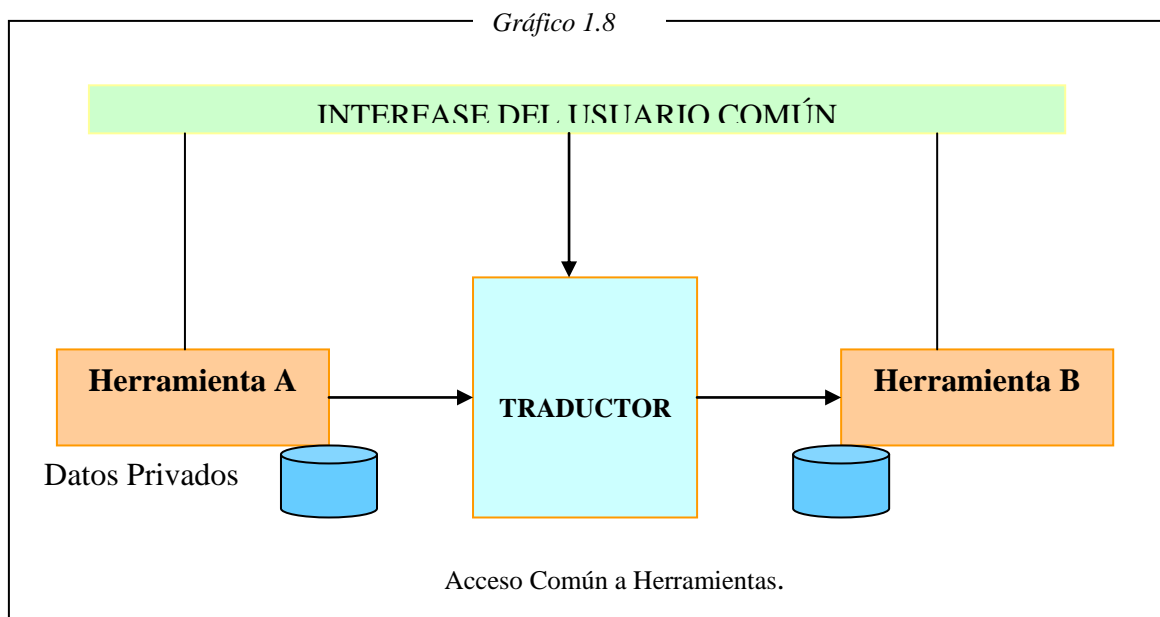
Además, a medida que evoluciona el software, la necesidad de transferir archivos cada vez que se hace un cambio pequeño puede llevar mucho tiempo. Las versiones pueden quedar "desfasadas" fácilmente, perdiéndose la posibilidad de transferencia, la cual suele ser en un único sentido.

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. [ww003]

1.6.2.2. 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 (gráfico 1.8.).



En ejemplo, el usuario puede 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.

En estos entornos, el intercambio de datos de herramienta a herramienta podría simplificarse llamando al procedimiento de traducción a través de un simple menú o de la selección de una macro, pero esta no es la opción más adecuada.

1.6.2.3 INTEGRACIÓN DE DATOS

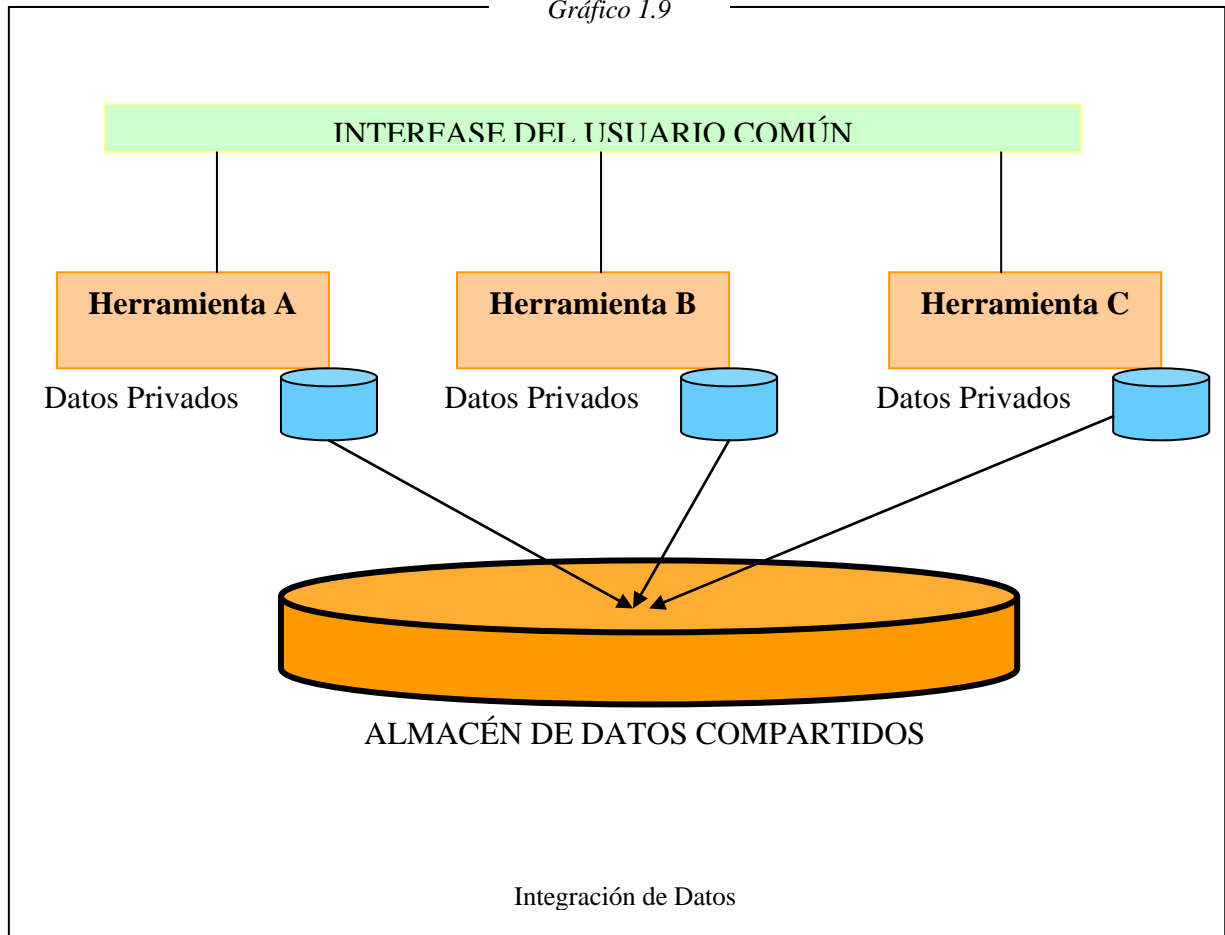
Gestión Común de Datos. Los datos de distintas herramientas se pueden mantener en una única base de datos lógica, que puede estar físicamente centralizada o distribuida. Hay una modalidad de fusión que permite combinar el trabajo de varias personas trabajando en diferentes partes de una aplicación.

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. (gráfico 1.9)

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

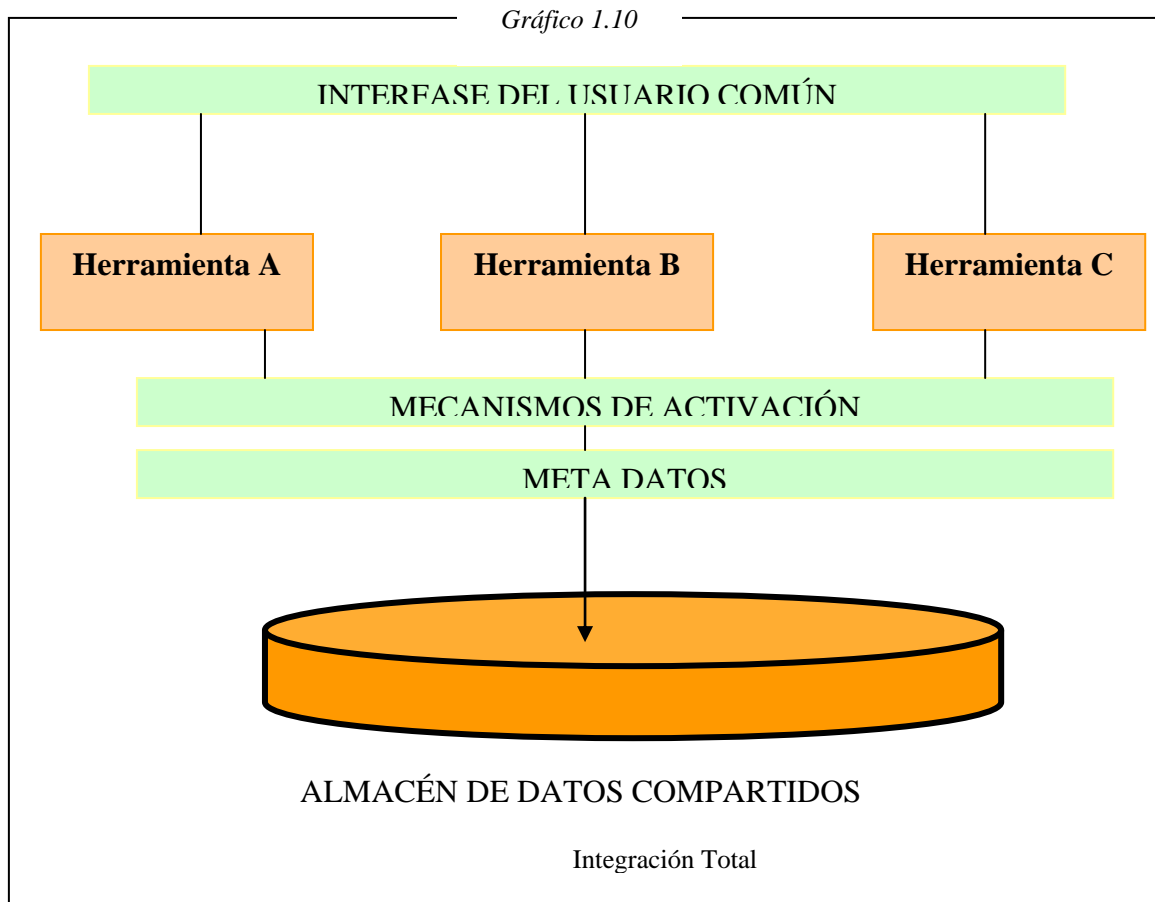
Gráfico 1.9



1.6.2.4. INTEGRACIÓN TOTAL

Para alcanzar la integración total del entorno CASE se necesitan dos características más: gestión de meta datos y capacidad de control (gráfico 1.10.) Los meta datos 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 así: un proceso en un diagrama DFD, una entidad única o un fragmento de código de una subrutina.
- Reglas de diseño del software. Como 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).



Normalmente, la parte de reglas y procedimientos de los meta datos 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 meta datos, 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 interfase de programación y un mecanismo de paso de mensajes.

[ww003]

1.7. PONDERACIONES PARA LA UTILIZACIÓN DE HERRAMIENTAS CASE

Antes de utilizar Herramientas CASE, se debe hacer caso a las siguientes ponderaciones para tener un concepto claro sobre lo que hacen y deberían hacer dichas herramientas.

- Las Herramientas CASE deben: descomponer la complejidad, simplificar, explicar y reducir.
- Deben ser fáciles de: utilizar y comprender por usuarios y personal técnico
- La utilización del Herramientas CASE debe ser más barata y eficiente que los métodos tradicionales para construir software.
- Las especificaciones y el diseño generadores por Herramienta CASE deben ser exactas y concisas representaciones del sistema a ser construido.
- Cada requerimiento en la implantación del software debe ser verificable y seguible hasta el documento original. Criterios de eficiencia en ejecución, límites del sistema y condiciones de error deben ser establecidos como parte del diseño.
- Las especificaciones y el diseño deben ser fáciles de adaptar a medida que las metas de análisis y diseño cambien.
- Las Herramientas CASE deben estar orientada a la representación gráfica de las especificaciones.

1.8 REQUISITOS PARA ADQUIRIR HERRAMIENTAS CASE

La primera etapa que debe abordarse de modo sistemático dentro del proceso de adquisición, es el análisis de las necesidades existentes, que deberán ser satisfechas a través de la implantación de la herramienta que se va a adquirir. El comprador debe identificar lo siguiente:

- Los principales requisitos funcionales que debe cumplir la herramienta.
- El tipo de facilidades de uso que debe prestar.
- Las limitaciones y restricciones que se derivan del entorno de operación previsto.

1.8.1. REQUISITOS FUNCIONALES

En función de los requisitos funcionales se podrá deducir qué tipo de herramienta es la más adecuada. Algunos factores a tener en cuenta y que son comunes en todas estas herramientas son:

- Tipo(s) de plataforma(s) sobre las que deberá funcionar la herramienta, tanto desde el punto de vista del equipamiento lógico como del equipamiento físico.
- Requisitos físicos (espacio en disco, memoria RAM, UCP, etc).
- Necesidad de integración con otras herramientas de ayuda al desarrollo ya existentes.
- Necesidad de acceso simultáneo para diferentes usuarios. Esto puede enfocar la elección hacia una herramienta que permita accesos compartidos a los datos y que cuente con una definición de perfiles de usuario para la protección de información.
- Necesidad de compartir datos con aplicaciones externas. Se valorará más a aquella aplicación que permita exportar sus datos o que almacene la información en un formato de fácil acceso para otra aplicación. [www003]

1.8.2. FACILIDADES DE USO QUE DEBE PRESTAR UNA HERRAMIENTA CASE

- **Funcionalidad Requerida.** Es importante definir con el mayor grado de aproximación, cuáles son las funciones que se le van a pedir a la herramienta. Para ello, es necesario analizar si las necesidades son cubiertas con una herramienta integrada o con una orientada a alguna de las fases del ciclo de vida del desarrollo.
- **Metodología Soportada.** Si en la organización ya existe una metodología y técnicas, la herramienta deberá soportar dicha metodología, así como las técnicas empleadas en cada fase. Si la Herramienta CASE va a servir precisamente para introducir un nuevo método de trabajo, habrá que asegurarse de que dicho método es el adecuado. En ocasiones, para adaptarse a una metodología, es preciso realizar desarrollos adicionales en la herramienta.
- **Generación Automática de Código.** En algunos casos la necesidad predominante del usuario puede consistir en la generación automática de código fuente (programas), a partir de productos del diseño fuertemente formalizados (scripts, formatos, etc.). En tal caso, deberán conocerse los pormenores de tal necesidad, como lenguajes de programación admisibles como salida, generación en tiempo real o en un proceso por lotes, etc.
- **Capacidad de Integración en la Arquitectura Existente.** Habrá que tener en cuenta la plataforma o plataformas diferentes, ordenadores que deberán soportar la Herramienta CASE, su tipología (fabricante, modelo y sistema operativo cuando menos) y las características de la red de interconexión cuando exista. Ello tendrá importancia a la hora de garantizar la compatibilidad del equipamiento existente con los nuevos productos que se van a adquirir. Lo mismo debe hacerse en relación con las herramientas lógicas previamente existentes en esas plataformas, siempre que deban integrarse en mayor o menor medida con

los nuevos productos. Se deberá considerar cuáles son los recursos disponibles en el equipamiento existente para la implantación de la Herramienta CASE en cuestión. Deberán conocerse con el mayor detalle, posibles cuestiones como memoria RAM y espacio en disco necesarios, grado de utilización de la(s) UCP(s) en condiciones normales de operación y de picos de demanda de la nueva herramienta. Este punto es importante de cara a un posible redimensionamiento del equipamiento disponible. Estas mismas consideraciones también deben ser tomadas en cuenta no ya para la propia Herramienta CASE, sino para las aplicaciones desarrolladas con ayuda de dicha herramienta.

- **Modo de Funcionamiento.** Será bueno conocer el modo de funcionamiento: monousuario o multiusuario, así como el grado deseable de centralización de los recursos y funciones asociadas con la administración y operación de la Herramienta CASE que se va a implantar.
- **Personalización del Entorno.** Finalmente, deberán considerarse las necesidades o conveniencias de la personalización del sistema, en función de los diferentes perfiles de usuario de la herramienta.

1.8.3. PROCESO DE ADQUISICIÓN DE HERRAMIENTAS CASE

En la definición del objeto del contrato y los requisitos inherentes al mismo, así como en la valoración y comparación de ofertas de los proveedores, pueden intervenir muchos factores y de muy diversa índole, así:

- Factores de empresa o de institución
- Factores económicos
- Factores técnicos particulares

No obstante, y a título orientativo en este apartado se hace mención de aquellos factores que, entre los anteriores, pueden intervenir en el proceso de adquisición de herramientas de ayuda al desarrollo y cuyo seguimiento debe efectuarse exhaustivamente.

Aparte de las cláusulas que se toman en cuenta cuando se adquiere cualquier tipo de software, las consideraciones en el contrato de adquisición de Herramientas CASE son:

1. Requerimientos para el funcionamiento de la Herramienta CASE.
2. Incumplimiento de los requerimientos.
3. Entrega e instalación de la herramienta.
4. Instalación de la herramienta.
5. Certificación de la instalación.
6. Pruebas de funcionamiento.
7. Informe de fallas durante la prueba de aceptación.
8. Responsabilidad de fallas.
9. Penalidad en caso de no alcanzar el nivel de funcionamiento mínimo.

10. Constancia de aceptación del equipo.
11. Garantía de la herramienta.
12. Asesoría técnica..
13. Capacitación.
14. Información técnica.
15. Honorabilidad del proveedor.
16. Honorabilidad del contacto.

1.9. ESTRATEGIAS PARA LA CORRECTA IMPLANTACIÓN HERRAMIENTAS CASE

En esta parte se describe cuales son las formas correctas de implantar una Herramienta CASE en ambientes corporativos; la utilización de esta tecnología implica un cambio considerable en las organizaciones, ya que se debe seguir una serie de pasos, estrategias, consejos para alcanzar el éxito deseado; estas estrategias son:

Estrategia de Implantación. Se debe comenzar aplicando la herramienta al desarrollo de un proyecto piloto, que no afecte a ningún área crítica y que sea de poca envergadura. Con la experiencia adquirida en este proyecto piloto, se podrá acometer el desarrollo de otros más complejos.

Es importante asegurarse de poder utilizar la nueva herramienta sin tener que volver a escribir las aplicaciones existentes. En el caso particular de implantar por primera vez una Herramienta CASE, es un factor crítico el apoyo del suministrador o de consultores con experiencia en las etapas iniciales, el cual deberá ser suministrado por el proveedor. [www002] [www003]

Requisitos Físicos. Expresado en el modelo de tecnología de arquitectura y características del puesto de desarrollo (procesador, memoria RAM, espacio en disco) y características del puesto de producción para las aplicaciones desarrolladas. Con ello se asegura que se dispone de los equipos necesarios o de que exista la necesidad de compra. Es posible que este factor obligue a la remodelación de todos los equipos y que su costo no sea asumible, el estudio previo de los requisitos físicos ayuda a que en el momento de la implantación no aparezcan costos no presupuestados.

Requisitos Lógicos. Expresado en el Modelo de Tecnología, se debe analizar con especial atención la necesidad de otros módulos no incluidos en el producto ofertado por el vendedor, para el correcto y completo funcionamiento de la herramienta como: compiladores, módulos para trabajo en grupo, parches, software adicional, etc.

Es fundamental comprobar si la herramienta tiene los módulos que incorporan las funcionalidades ofrecidas. Hay que tener cierta precaución cuando se analice un módulo ofertado, ya que hay casos en que para el funcionamiento de dicho módulo, es necesario adquirir otros módulos incluso de diferentes fabricantes.

En las etapas de promoción y demostración de las herramientas, los proveedores muestran su producto funcionando al cien por ciento, con todas las características que tiene la herramienta, de tal manera que el cliente se queda sorprendido por las ventajas de dicho producto; le hacen creer al comprador que la herramienta trae todos los requerimientos incluidos con un costo único solo de la herramienta; pero el momento de ponerse a desarrollar vienen los problemas y los costos adicionales del software utilitario.

Prueba en Condiciones Reales. Si se va a instalar una Herramienta CASE, se debe exigir al suministrador una prueba previa a la adquisición, esta prueba debe realizarse en la propia instalación de destino y si es posible con los equipos de la organización donde se va a adquirir.

La prueba se debe realizar en las condiciones más parecidas a las reales que se puedan conseguir e intentando simular el acceso de un número de usuarios, parecido al esperado. Durante la prueba se deberán evaluar conceptos objetivos y fácilmente medibles.

No todas las herramientas cumplen con las prestaciones indicadas en los manuales, por lo que es aconsejable establecer un período de prueba para explorar la herramienta que se pretende adquirir.

Una vez que en las especificaciones técnicas se hayan definido, la plataforma física y lógica y las necesidades funcionales, durante un período de prueba aceptable, se podrá decidir si la herramienta funcionará o no en su organización.

Dependencia del Proveedor. Hay que evitar esta dependencia. A veces las herramientas llevan integradas partes de la plataforma operativa, lo cual las hace cerradas y propietarias. En el contrato de adquisición se debe contemplar la asesoría técnica, la capacitación y la información técnica. Se debe encontrar el equilibrio entre la productividad de la herramienta y su carácter abierto, por ejemplo: independencia del proveedor y del sistema operativo.

Costo límite de Adquisición. En este apartado hay que analizar las posibilidades que ofrece el suministrador en cuanto a disponer de licencias individuales, grupos de licencia o licencias corporativas. Los costes varían considerablemente en función del tipo de licencia, los costos de Herramientas CASE varían de acuerdo al número de programas, bases de datos y plataformas en las que pueden generar sus soluciones.

Costo de Instalación de las Aplicaciones Generadas. Hay que averiguar si una vez generada la aplicación y a la hora de distribuirla entre los usuarios, es necesaria la instalación de un módulo propiedad del suministrador.

Este módulo en ocasiones no es de libre distribución y es preciso comprarlo. Hay que dejar claro este punto desde un principio, hay que averiguar si la herramienta genera programas ejecutables.

Capacidad de Integración. Hay que tener en cuenta la plataforma o plataformas diferentes en que va a ser instalada la herramienta en cuestión, su tipología (fabricante, modelo y sistema operativo) y las características de la red de interconexión, cuando exista. Igualmente habrá que asegurar la integración con el software ya instalado. La necesidad de la integración con una herramienta CASE determinada, condiciona de forma decisiva la elección de un buen lenguaje de programación. (4GL)

Portabilidad de la Aplicación Generada. Cuando se pretende ejecutar la aplicación generada en diferentes plataformas, es un factor muy importante la portabilidad, tanto del código generado como de las especificaciones del diseño.

En el caso particular de un 4GL, este factor puede convertirse en decisivo si se tiene la intención de instalar la aplicación generada en entornos técnicos diferentes: sistemas operativos, plataformas físicas, interfases gráficas y protocolos de red. Un 4GL será realmente portable si el código generado se ejecuta en diferentes plataformas sin necesidad de adaptar los programas. [www002] [www003]

Capacidad Técnica de la Empresa y de la Asistencia Técnica que Presta. Es recomendable pedir referencias a otros usuarios que venden el mismo producto, como a organizaciones que ya adquirieron el mismo.

Formación del Personal y el Efecto de la Curva de Aprendizaje. Para minimizar el costo de la curva de aprendizaje son importantes factores como la calidad de la formación inicial, la calidad de la documentación de la herramienta, la existencia de ayuda interactiva y la disponibilidad de la herramienta en castellano.

1.10. RECUPERACIÓN DE COSTOS DE COSTOS DE LAS HERRAMIENTAS CASE

Los ambientes corporativos que hoy en día no utilizan metodologías de desarrollo soportada por Herramientas CASE, podrían compararse a empresas constructoras cuyos métodos de construcción se redujesen a la experiencia de sus operarios y cuyas herramientas constructivas fueran los tradicionales picos, palas, carretillas, etc; aunque sus equipos humanos estuvieran integrados por excelentes jefes de obra y oficiales de albañilería, sus "métodos y técnicas artesanales" les impedirían abordar competitivamente cualquier proyecto de construcción actual, con independencia de que el mismo se llevase a cabo con los más modernos materiales. Esta situación que en construcción civil e industrial es asumida y tan evidente, no es trasladable a los desarrollos o construcciones Informáticas.

En la actualidad en muchas empresas e instituciones, aún empleando los más modernos materiales de software informático, como son los lenguajes de desarrollo: Java, XML, PHP, Visual Basic, .NET, etc; se siguen construyendo sus sistemas informáticos con métodos y técnicas convencionales. En estas empresas la única evolución constructiva se ha limitado a pasar de utilizar unos materiales por otros, han cambiado los tradicionales lenguajes como: COBOL, C, etc. por los nuevos como Java o

Visual Basic, pero sus métodos y técnicas constructivas permanecen como las de antaño; en muchos casos se limitan a la experiencia particular de sus técnicos.

En estas organizaciones se sigue construyendo software con costos y tiempos muy altos los nuevos sistemas informáticos, con toda la problemática ya muy conocida, que fue denominada con las palabras anglosajonas de "sistemas heredados", aunque posiblemente la traducción más adecuada en español hubiera sido la de sistemas ruinosos.

La mayoría de las Direcciones de Informática son conscientes de que está bajo su responsabilidad directa cambiar esta situación, puesto que seguir desarrollando hoy en día nuevos sistemas artesanales es un error que implica graves daños para las empresas e instituciones que los están financiando.

Como observará, no se está ante una decisión económica sino humana, siendo conscientes de que lo más costoso y grave para las empresas es no cambiar su metodología y seguir trabajando con los métodos tradicionales.

1.10.1. OBJETIVOS DE LA RECUPERACIÓN DE COSTOS

1.10.1.1. OBJETIVOS CUANTITATIVOS

Ganancia de Productividad de los Analistas. La influencia que a nivel de costos de análisis y diseño que se asume por el uso de Herramientas CASE es muy importante. La ganancia de la productividad de un analista que lleva a cabo sus análisis con la ayuda de Herramientas CASE es superior al 30% y el periodo de entrenamiento y dominio de la herramienta es corto, normalmente inferior a tres meses.

Esta ganancia de productividad es aun mayor cuando en un proyecto participan múltiples analistas o desarrolladores, en estas situaciones muy frecuentes en proyectos de tamaño medio y grande, las Herramientas CASE se convierten además en excelentes herramientas de trabajo en grupo. Esta ganancia de productividad permite prácticamente por si sola la recuperación de las inversiones llevadas a cabo por la adquisición de Herramientas CASE en menos de 18 meses.

Disminución de los Costos de Puesta a Punto de los Nuevos Sistemas Desarrollados. Uno de los principales problemas que están teniendo la mayoría de empresas es el excesivo tiempo de la puesta a punto de los programas en los proyectos en desarrollo. Gran parte de esta problemática está directamente relacionada con un análisis y diseño inicial defectuoso e incompleto. Su repercusión económica en el conjunto del proyecto es muy importante pues obliga a realizar cambios en los procesos ya programados que no hubieran sido precisos si el análisis y diseño se hubieran realizado con amplitud y detalle utilizando Herramientas CASE. A nivel de costos se puede estimar una reducción del 10% en los costos totales de un proyecto realizado con una metodología moderna de desarrollo soportada por Herramientas CASE.

Disminución de los Costos de Mantenimiento de las Aplicaciones. Estos beneficios son los más importantes a largo plazo, para conseguirlos es necesario que los sistemas hayan sido llevado a cabo con el soporte de tecnologías CASE ó si se parte de sistemas ya existentes, que su análisis y diseño se documente en la enciclopedia de las Herramientas CASE.

Cuando una organización ya está trabajando con Herramientas CASE, el mantenimiento se simplifica de forma drástica; cualquier cambio requerido en los procesos o en los datos que se están utilizando es automáticamente evaluado; las Herramientas CASE permiten ver el detalle del impacto de los cambios en todos los procesos de desarrollo por medio de los ciclos diseño, prototipo y producción; sin Herramientas CASE esto es imposible.

1.10.1.2 OBJETIVOS CUALITATIVOS

Disminución de los Tiempos de Desarrollo y de Mantenimiento de los Sistemas Existentes. Para muchas empresas, tan importante como es la disminución de los costos de desarrollo, es la disminución del tiempo de tener disponible los sistemas que precisa para llevar a cabo sus estrategias de negocio.

Mayor Calidad de los Sistemas Desarrollados. El uso Herramientas CASE permite verificar que los requisitos establecidos en cualquier proyecto informático se cumplan correctamente. El control de calidad cuando no se utilizan Herramientas CASE se hace muy difícil de llevar a cabo.

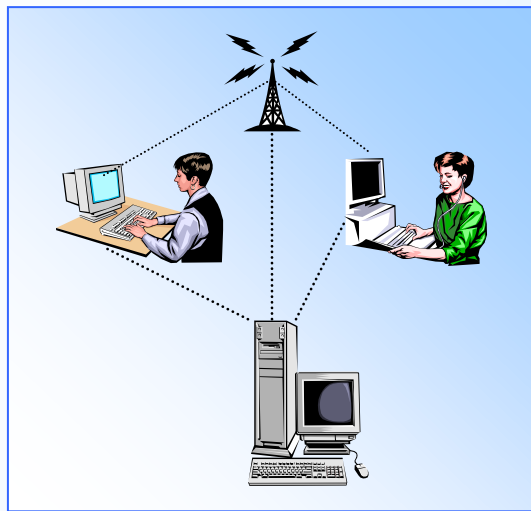
Mejorar la Documentación de los Sistemas Informáticos y Facilitar que esté Permanentemente al Día. Cuando se utiliza, Herramientas CASE, la documentación de los sistemas es un proceso semiautomatizado, puesto que la fuente de la misma es la propia enciclopedia disponible en la Herramienta CASE.

Soporte a la Metodología de Desarrollo Utilizada. Cuando no se dispone de Herramientas CASE el uso de las metodologías en las etapas fundamentales de análisis y diseño se complica excesivamente, en la mayoría de las situaciones se limita al uso de gráficos parciales e incluso en proyectos de alta prioridad a simples requisitos textuales. (comandos) Esta situación trae consigo imprecisiones que con mucha frecuencia repercuten muy negativamente en las fases posteriores de programación, prueba e implantación, haciendo que tanto los tiempos como los costos se desvíen bastante sobre los previstos.

Mejor Seguimiento y Gestión de Proyectos. La utilización de Herramientas CASE facilitan el seguimiento y gestión de los proyectos. Aunque las Herramientas CASE no son en sí mismas herramienta de planificación, si disponen de los mecanismos para incorporar las extensiones que cada organización precise, bien como check-list de actividades, objetos y/o componentes obtenidos en el proceso de desarrollo. [www001]

CAPÍTULO II

SOLUCIONES CLIENTE / SERVIDOR GENERADAS POR HERRAMIENTAS CASE



- ❖ Generalidades de Cliente / Servidor
- ❖ Arquitectura Centralizada
- ❖ Definición de Cliente / Servidor
- ❖ Estructura de Cliente / Servidor
- ❖ Clasificación de Cliente / Servidor
- ❖ Herramientas CASE en Cliente / Servidor
- ❖ Formas de Enlace con Herramientas CASE
- ❖ Reingeniería con Herramientas CASE en C/S
- ❖ Tácticas para Soluciones C/S Generadas por Herramientas CASE
- ❖ Características de una Herramienta CASE C/S
- ❖ Ventajas y Desventajas de Herramientas CASE C/S

2.1. GENERALIDADES DE CLIENTE / SERVIDOR

La historia en la industria de las aplicaciones informáticas, está matizada por constantes innovaciones, todos los cambios que surgen prometen dar soluciones a problemas que se generan en el convivir diario. En este capítulo se analizará el modelo cliente / servidor, y con ello, todas las connotaciones y variantes con que las distintas tecnologías lo implantan, de manera especial las Herramientas CASE.

Una de las principales características, que hay que tomar en cuenta en la arquitectura cliente / servidor, es que se está frente a una plataforma abierta por excelencia. Ciertamente las posibilidades de igualar o nivelar distintos productos como: sistemas operativos, bases de datos, aplicaciones o componentes de varios proveedores, brinda la oportunidad de hacer una gran cantidad de combinaciones de clientes y servidores, ya que las empresas de hoy en día, tienden a distribuir todas sus aplicaciones.

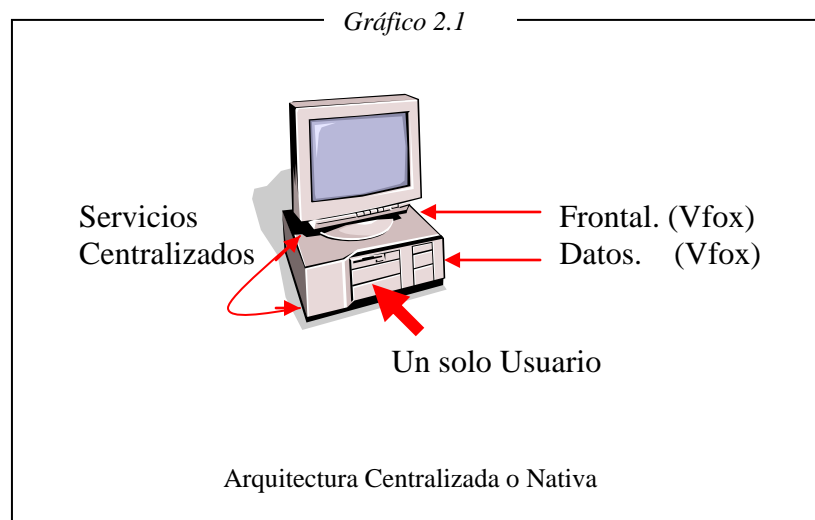
Cliente / Servidor es una tecnología de bajo costo, que proporciona: recursos compartidos, escalabilidad, integridad, encapsulamiento de servicios, seguridad, etc. Pero al igual que toda tecnología, el desarrollo de aplicaciones cliente / servidor requiere que el personal tenga conocimientos, experiencia y habilidades en: procesamiento de transacciones, diseño de base de datos, redes de ordenadores diseño gráfico de interfases, seguridad, distribución de usuarios, etc. [www0015] [www016] [www020] [www027]

2.2. ARQUITECTURA CENTRALIZADA

Antes de empezar a estudiar las diferentes arquitecturas, se procederá a la definición de la palabra arquitectura. Una *Arquitectura* es un conjunto de componentes funcionales, que aprovechando: diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de una organización.

Arquitectura Nativa o Centralizada, es o se da cuando en una aplicación, no se hace diferencias entre servidores y clientes, los datos y los frontales funcionan de manera unificada, o sea en una misma aplicación, en si los componentes de el sistema no se distribuyen. Al no hacer diferencia entre clientes y servidores, no hay necesidad de utilizar herramientas para la conectividad entre ellos.

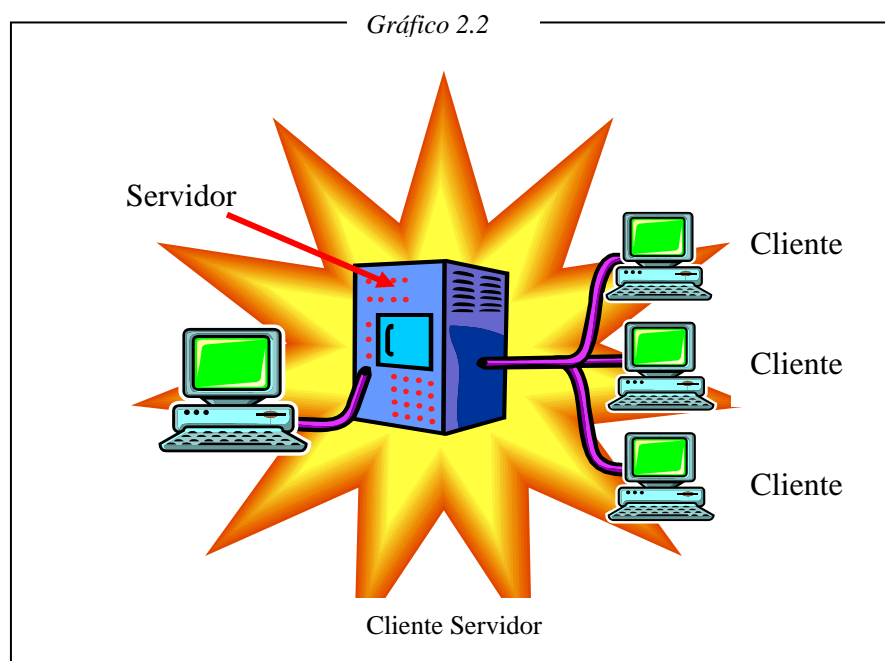
En su ejecución puede estar un solo usuario a la vez o sea son uniusuario y no existe conversación entre equipos. Antes de existir cliente / servidor los programas informáticos se ejecutaban independientemente en cada PC, sin importar que el mismo producto lo utilizarán a la vez un gran número de usuarios, se debía instalar todo el programa en un computador diferente para cada usuario del mismo, entonces no se podía compartir datos de un computador a otro. (gráfico 2.1)



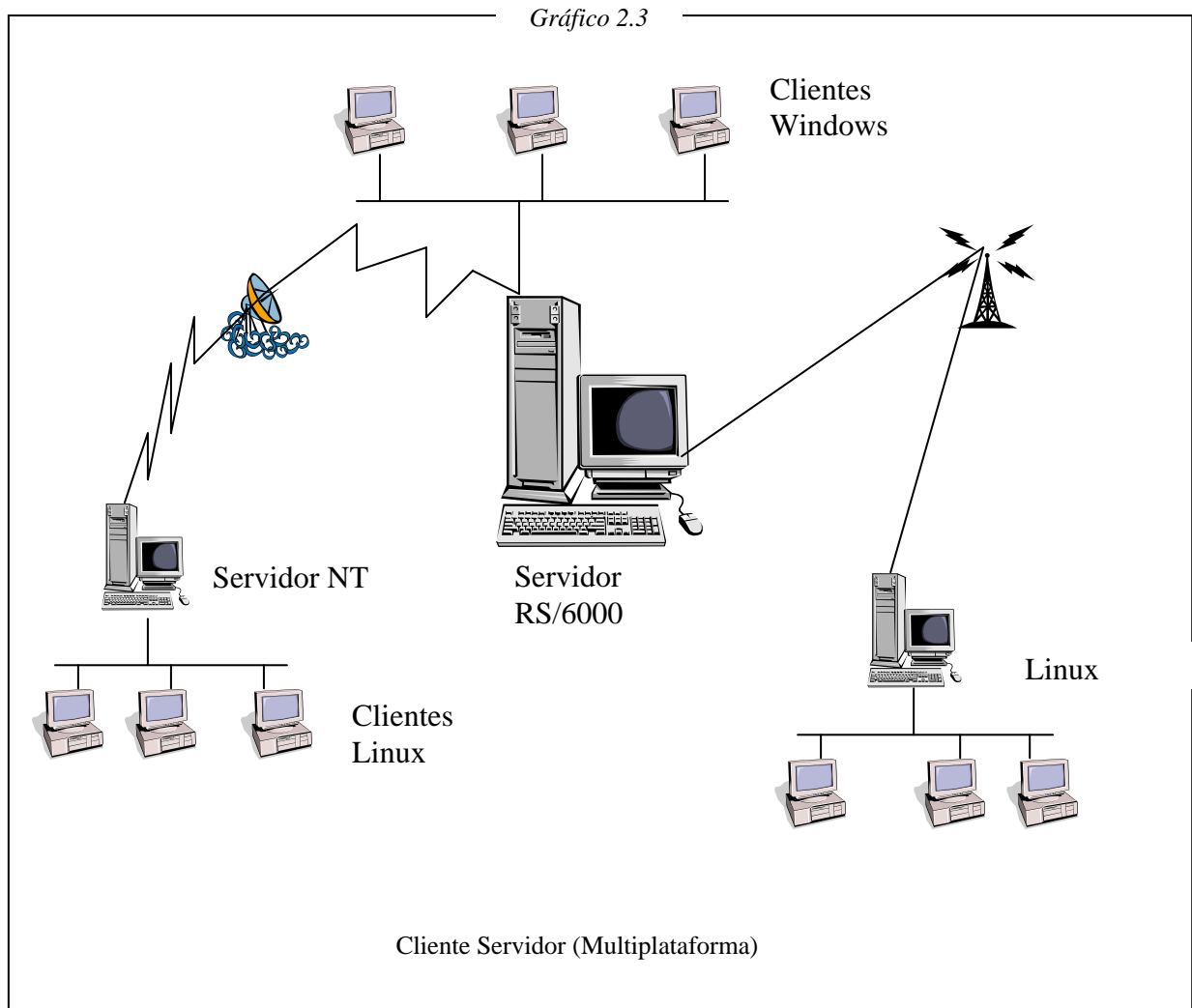
2.3. DEFINICIÓN DE CLIENTE / SERVIDOR

El concepto de cliente / servidor es eminentemente técnico, es una definición que el mercado ha madurado en forma vertiginosa en lo que va de los últimos años.

Su principio básico es muy sencillo. Se tienen aplicaciones en un computador, las cuales están "conversando" con aplicaciones en otro computador; a partir de ese momento se establece un diálogo cooperativo entre los dos computadores.



La idea no hace referencia a un tipo específico de hardware o sistema administrador de base de datos; de hecho en este estudio se explica cómo la idea tras la cual se basa la arquitectura, no solo funciona para aplicaciones acogiendo a bases de datos, sino que existen otras áreas de la computación, que pueden ser susceptibles a la implementación de esta tecnología. Hoy en día los modelos cliente / servidor, se los utiliza para soluciones de: correo electrónico, servicios transaccionales, telefonía móvil, internet, bases de datos, etc. (gráfico 2.2) [www016] [www020]



Un sistema Cliente / Servidor es aquel que coloca la maquinaria de acceso a la base de datos a través de la red en un equipo poderoso central o servidor, y la presentación en un equipo menos poderoso (PC) denominado cliente.

Cliente / Servidor. Es el concepto de interacción más común entre aplicaciones en una red. No forma parte de los conceptos de la Internet como los protocolos IP, TCP o UDP, sin embargo todos los servicios estándares de alto nivel propuestos en Internet funcionan según este Concepto.

La tecnología Cliente / Servidor. Es el procesamiento cooperativo de la información por medio de un conjunto de procesadores, en el cual múltiples clientes distribuidos geográficamente, solicitan requerimientos a uno o más servidores centrales. Desde el punto de vista funcional, se puede definir el modelo cliente servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma.(gráfico 2.3)

Cliente / Servidor. Es un modelo para construir sistemas de información, que se sustenta en la idea de repartir el tratamiento de la información y los datos por todo el sistema informático, permitiendo mejorar el rendimiento del sistema global de información.

2.3.1 SERVIDOR

Servidor, es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LANs o WANs, para proveer de múltiples servicios a los clientes, tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, correo electrónico, internet, etc.

Servidor o Servicio, es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. Al proceso servidor se lo conoce con el término back - end. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos, hoy en día se utiliza mucho los servidores de transaccionales o lógica del negocio; algunas de sus funciones son: [www013] [www018]

- Aceptar los requerimientos de bases de datos que hacen los clientes.
- Procesar requerimientos de bases de datos.
- Formatear datos para transmitirlos a los clientes.
- Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

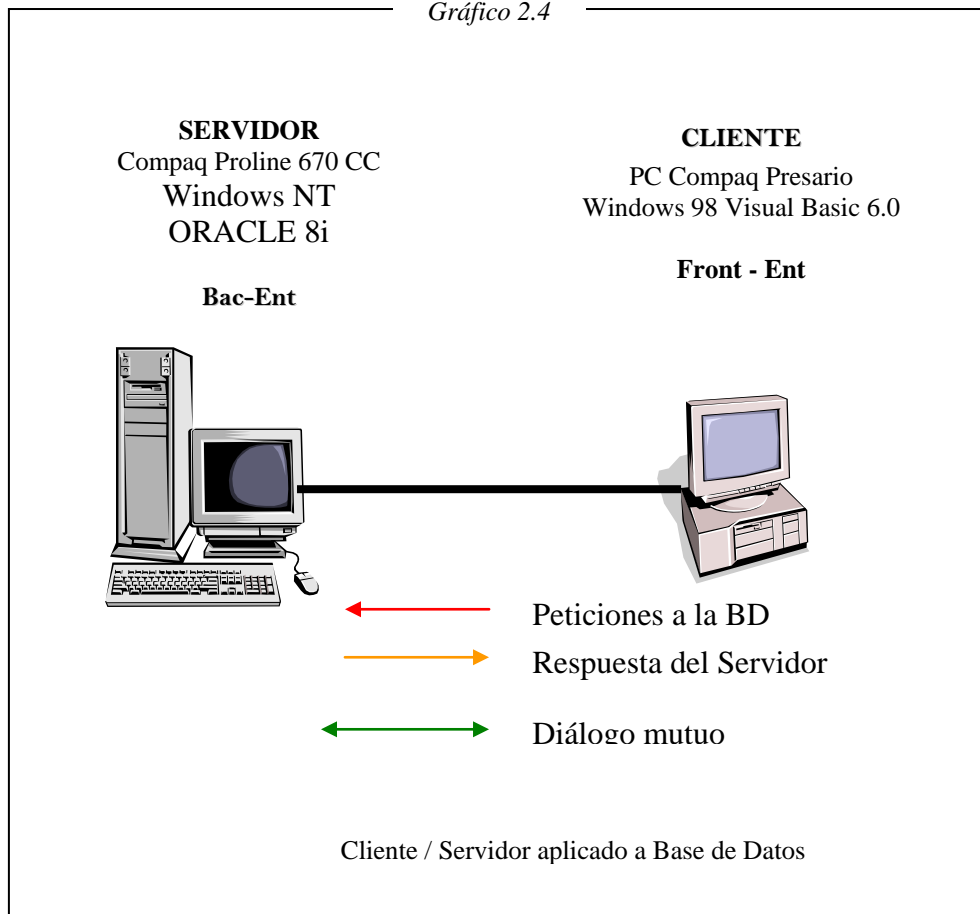
2.3.2 CLIENTE

Cliente, es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

El cliente es el proceso que permite al usuario formular los requerimientos y pasarlos al servidor, (gráfico 2.4) se lo conoce con el término front - end. Normalmente maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de la red, sus funciones son: [www013] [www019]

- Administrar la interfaz de usuario
- Interactuar con el usuario.
- Procesar la lógica de la aplicación y hacer validaciones locales.
- Generar requerimientos de bases de datos.
- Recibir resultados del servidor.
- Formatear resultados.

Gráfico 2.4



2.4. CONCEPTOS FUNDAMENTALES UTILIZADOS EN C/S CON HERRAMIENTAS CASE

La arquitectura Cliente / Servidor como todas las arquitecturas, maneja una serie de términos fundamentales que se deberían conocer para generar de una mejor manera aplicaciones de software, estos conceptos fundamentales están relacionados directamente con la utilización de Herramientas CASE.

2.4.1. MIDDLEWARE

En su definición más simple, middleware es la interfaz que provee la conectividad entre aplicaciones clientes y aplicaciones servidoras, y entre aplicaciones y bases de datos.

Es una capa de software que protege a los desarrolladores de tener que manejar detalles de bajo nivel de diferentes protocolos de comunicación, sistemas operativos y arquitecturas de bases de datos. Este tipo de interfaces incluyen mensajería de red y accesos a bases de datos.

2.4.2 SISTEMA ABIERTO

Es un ambiente en el cual los sistemas y productos de cómputo de diferentes proveedores son capaces de trabajar conjuntamente, para proveer una solución aplicativa a cualquier requerimiento de la organización. También se refiere a la posibilidad de transportar aplicaciones y/o datos desde cualquier sistema de cómputo a otro.

2.4.3. DOWNSIZING

Es la migración de aplicaciones a plataformas de cómputo menores con la intención de obtener mayor flexibilidad, eficiencia, reducción de costos y autosuficiencia para los usuarios.

2.4.4. PROCESO DISTRIBUIDO

Es un modelo de sistemas y/o de aplicaciones, en el cual las funciones y los datos pueden estar distribuidos a través de múltiples recursos de cómputo, conectados en un ambiente de redes LAN o WAN.

En los sistemas cliente / servidor se puede distribuir los recursos en diferentes partes como servidores de: bases de datos, correo electrónico, noticias, FTP, internet, transacciones o aplicaciones.

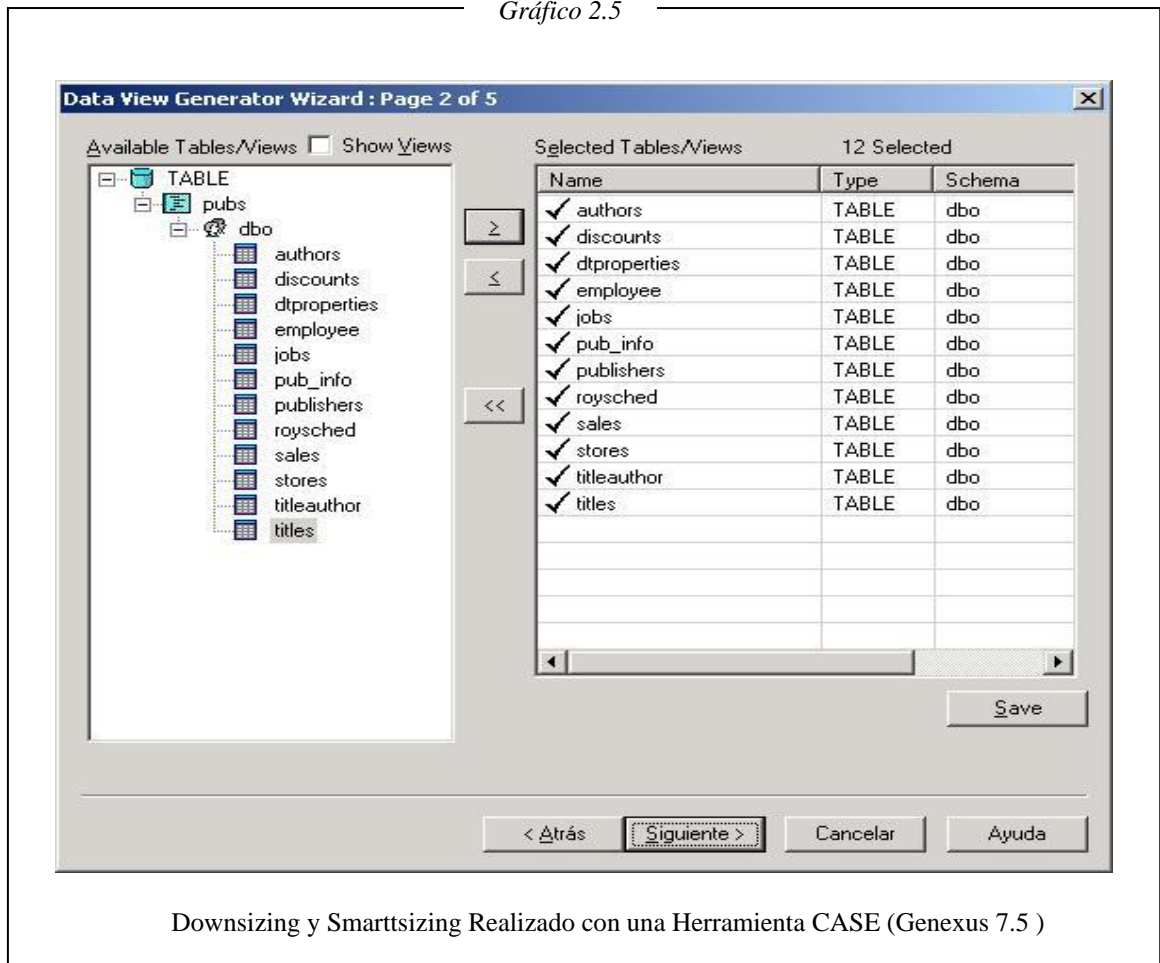
2.4.5. SMARTSIZING

El Smartsizing, a diferencia del downsizing, está basado en la reingeniería de procesos del negocio que reimplementa los sistemas automatizados ya existentes en unos más pequeños o en plataformas basadas en LAN.

El downsizing se concentra en el ahorro de costos y en el incremento de la productividad actual.

En esta última parte, mientras el código de la aplicación puede perfeccionarse, poco o ninguna consideración se le da al proceso en sí. En el gráfico 2.5 Se puede observar en la parte izquierda, la aplicación ya realizada, y en la parte derecha, las tablas que quiere transformar a otras bases de datos o lenguajes de programación, esta tarea está realizada por medio de una Herramienta CASE, lo que comúnmente se conoce como reingeniería.

Gráfico 2.5



El Smartsizing sostiene que la tecnología de la información puede hacer más eficiente los procesos del negocio e incrementar sus beneficios.

La reingeniería de negocios se basa en el uso de tecnología para los flujos de trabajo interno, tales como los ingresos de órdenes y la facturación a clientes. La tecnología de la información puede usarse para incrementar la satisfacción del cliente. Los productos pueden desarrollarse y lanzarse más rápido al mercado, usando la tecnología de la información.

La tecnología de la información es uno de los principales principios de las Herramientas CASE, algunas de las herramientas ya lo tienen como el data view generator de genexus que se muestra en el gráfico anterior.

2.4.6. OUTSOURZING

Se define como la cesión de la responsabilidad en la gestión de los Sistemas de Información de una organización a otra empresa especializada en este tipo de actividades.

En general, Outsourcing es una cesión completa de la gestión, pudiendo incluir al personal técnico informático al equipamiento físico lógico que pudiera existir en el momento de la realización del contrato, de modo que todas las tareas de carácter informático de la organización, pasan a ser realizadas por la empresa contratista.

En ocasiones particulares esta cesión puede hacerse de forma sectorial por ejemplo, puede excluirse al personal informático y, en general, debe ser muy flexible para adaptarse a las necesidades propias de cada organización.

2.4.7. UPSIZING

Es la consolidación de usuarios finales o aplicaciones y datos de redes LANs en plataformas de cómputo mayores, incrementando la facilidad de acceso, capacidad y rendimiento

2.4.8. RIGHTSIZING

Consiste en la selección de tecnologías de información adecuadas para la solución de la problemática de los negocios y servicios, tales como mejor respuesta al mercado, un adecuado servicio a los clientes y ciudadanos y un mayor aprovechamiento en el uso de la tecnología y de los recursos.

Las organizaciones exitosas de hoy en día eligen metodologías soportadas por Herramientas CASE, ya que han demostrado la mejora en la productividad del software desarrollado. [www019]

2.5. ESTRUCTURA DE CLIENTE / SERVIDOR

La gran variedad de posibilidades y modelos Cliente / Servidor, implica que se debe tener en cuenta, una gran cantidad de elementos a considerar y evaluar al momento de enfrentar una solución informática basada en esta arquitectura.

Se puede agrupar básicamente en dos aspectos dicha problemática: ¿Qué plataforma elegir? y ¿Qué herramientas de desarrollo elegir?.

La primera pregunta tiene relación con otras interrogantes internas, como: ¿Qué plataforma cliente elegir? ¿Qué plataforma servidor? ¿Qué clase de middleware? ¿Qué administrador o servidor de base de datos? ¿Sobre qué arquitectura de computación distribuida se tendrá que montar la solución?.

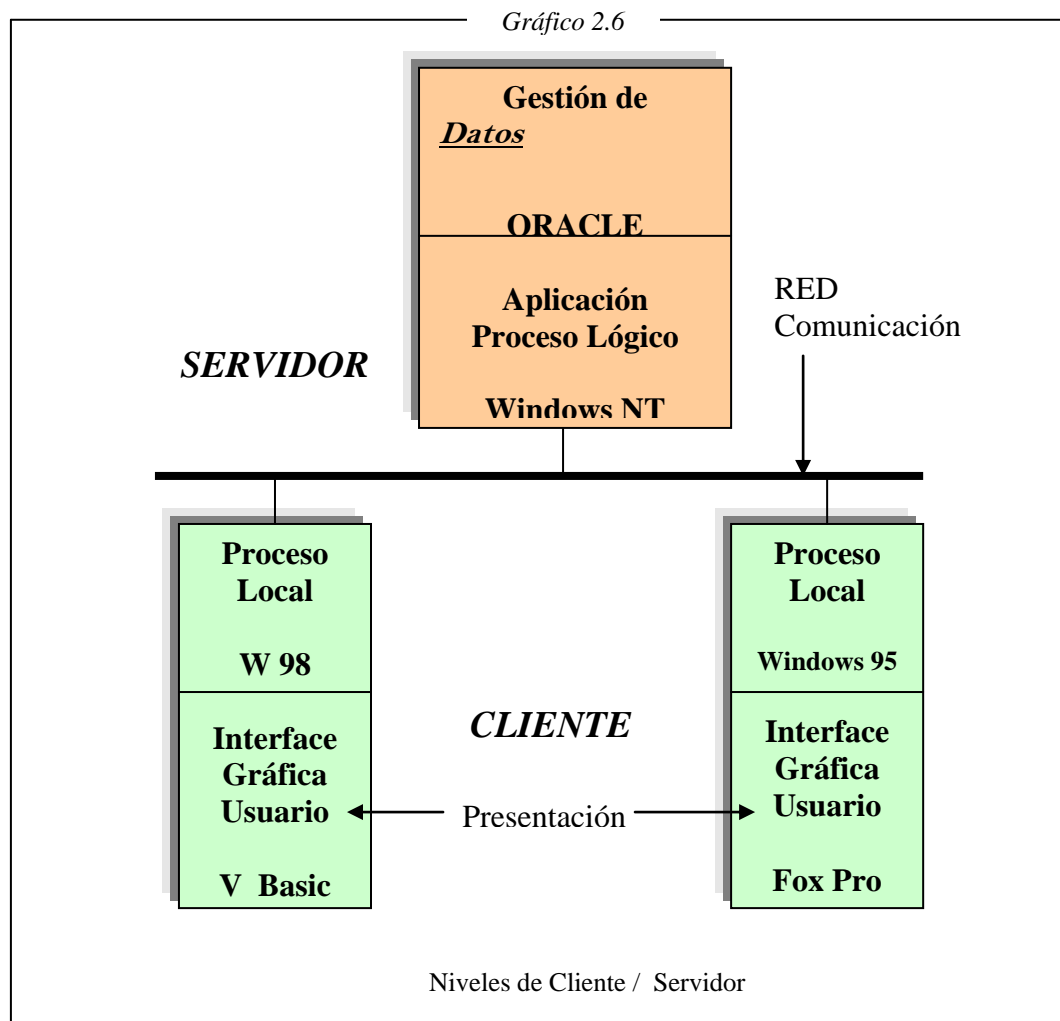
El segundo aspecto tiene relación con la toma de decisiones sobre el área de desarrollo y herramientas de Cliente / Servidor a utilizar, como en cuántas capas se va a desarrollar la aplicación.

Si bien es cierto que la mayor ventaja de esta tecnología es la flexibilidad en cuanto a que se puede elegir entre muchas opciones, esto obliga a tener conocimientos importantes para la integración de las mismas, dado que el desarrollo de aplicaciones Cliente / Servidor requiere del manejo de elementos en el área de diseños de bases de datos, comunicación entre procesos, procesamiento de transacciones, generación de interfaces gráficas de usuarios y para que hablar de Internet, con clientes y servidores distribuidos a lo largo de la red de redes.

Como se ha venido señalando, Cliente / Servidor es un modelo basado en la idea del servicio, en el que el cliente es un proceso consumidor de servicios y el servidor es un proceso proveedor de servicios, además esta relación está establecida en función del intercambio de mensajes que es el único elemento de acoplamiento entre ambos.

De estas líneas se desprenden los tres elementos fundamentales sobre los cuales se desarrollan e implantan los sistemas Cliente / Servidor: el **Proceso Cliente** que es quien inicia el diálogo, el **Proceso Servidor** que pasivamente espera a que lleguen peticiones de servicio y el **Middleware** que corresponde a la interfaz que provee la conectividad entre el cliente y el servidor para poder intercambiar mensajes.

Para entender en forma más ordenada y clara los conceptos y elementos involucrados en esta tecnología se puede aplicar una descomposición denominada: arquitectura de niveles. Esta descomposición principalmente consiste en separar los elementos estructurales de esta tecnología en función de aspectos más funcionales de la misma:



- **Nivel de Presentación:** Agrupa a todos los elementos asociados al componente Cliente
- **Nivel de Aplicación:** Agrupa a todos los elementos asociados al componente Servidor.
- **Nivel de Comunicación:** Agrupa a todos los elementos que hacen posible la comunicación entre los componentes Cliente y servidor.
- **Nivel de Base de Datos:** Agrupa a todas las actividades asociadas al acceso de los datos.

En el gráfico 2.6 se demuestra los niveles de Cliente / Servidor, detallando las consideraciones mencionadas anteriormente. Cabe recalcar que las Herramientas CASE Workbench automatizan todos los niveles de desarrollo Cliente / Servidor. [www016]

2.6. CLASIFICACIÓN DE CLIENTE / SERVIDOR

Más allá de entender los componentes: cliente / middleware / servidor, es preciso analizar ciertas relaciones entre éstos que pueden definir el tipo de solución que se ajusta de mejor forma a las estadísticas y restricciones acerca de los eventos y requerimientos de información que se obtuvieron en la etapa de análisis de un determinado proyecto.

De hecho el analista o líder deberá conocer estos eventos y restricciones del negocio, para a partir de allí, hacer las consideraciones y estimaciones de la futura configuración, teniendo en cuenta aspectos como: la oportunidad de la información, tiempo de respuesta, tamaños de registros, tamaño de bases de datos, estimaciones del tráfico de red, distribución geográfica tanto de los procesos como de los datos, etc.

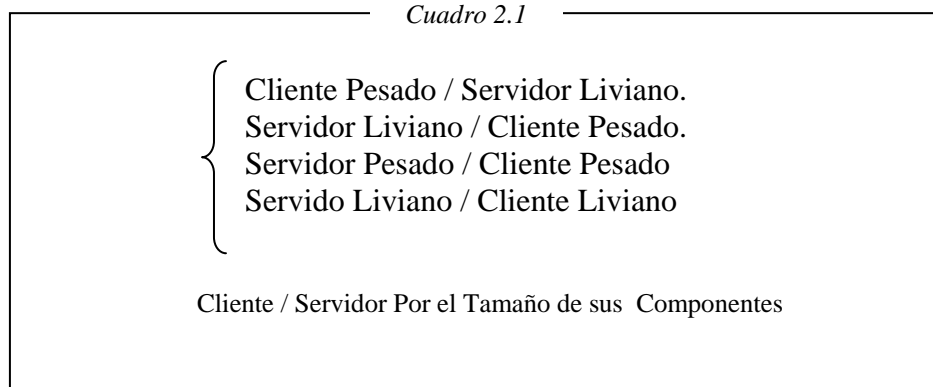
Para esto, se presenta una clasificación de Cliente / Servidor (cuadro 2.2), que ayudará al desarrollador a escoger la opción más adecuada.

2.6.1 POR EL TAMAÑO DE SUS COMPONENTES.

Este tipo de clasificación se basa en los grados de libertad que brinda el modelo Cliente / Servidor para balancear la carga de proceso entre los niveles de presentación, aplicación y base de datos, dependiendo de qué segmento de las capas de software tenga que soportar la mayor o menor carga de procesamiento.

Consideraciones de este tipo son importantes al momento de decidir una plataforma, al punto que se pueda definir la viabilidad o no de las mismas para enfrentar un cierto número de restricciones impuestas por una problemática a resolver. (cuadro 2.1)

Cuadro 2.1

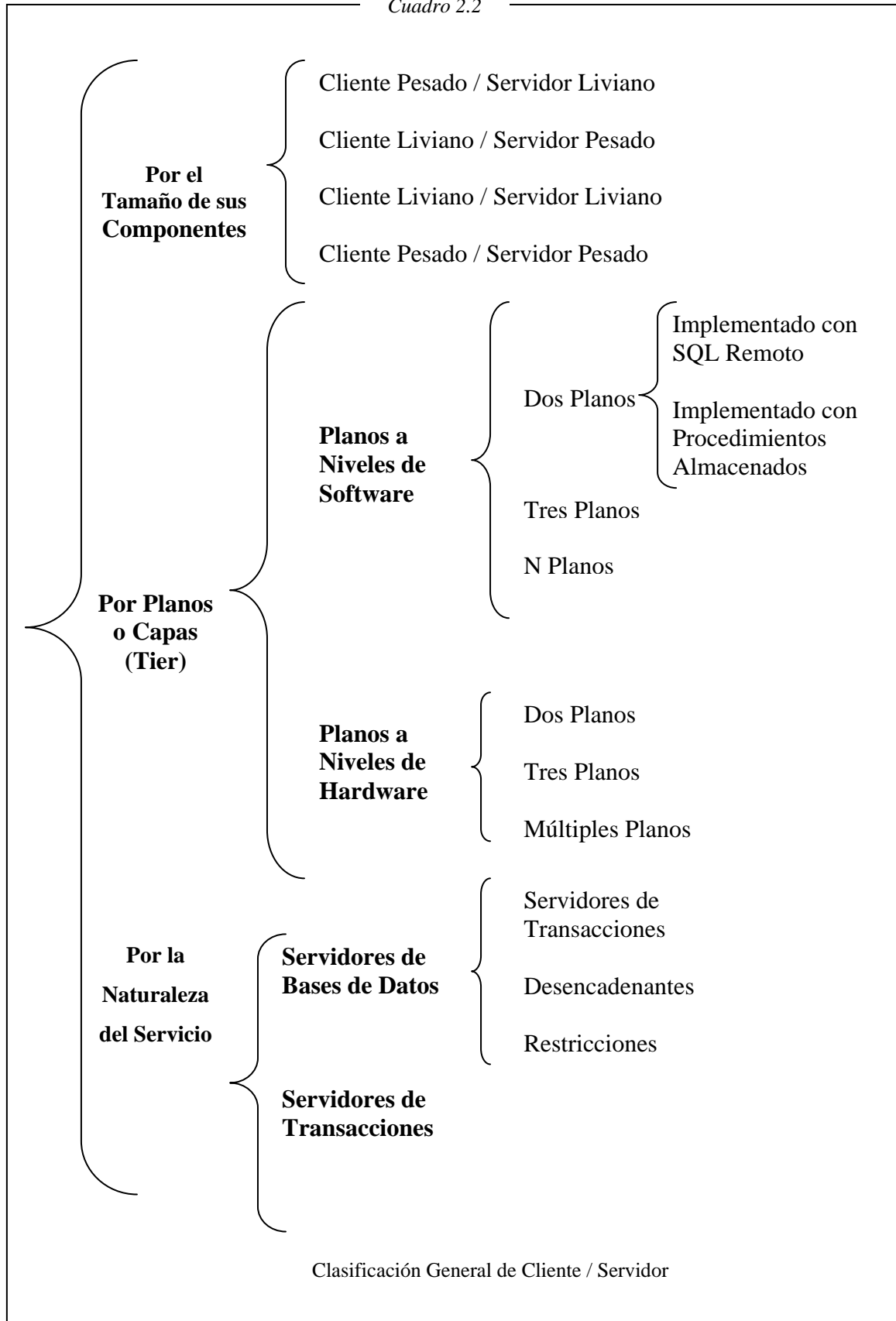


- ***Cliente Pesado – Servidor Liviano (Fat Client -Thin Server).*** En este esquema de arquitectura el grueso de la aplicación es ejecutada en el cliente, es decir, en el nivel de presentación y en el nivel de aplicación corre un único proceso cliente, y el servidor es relegado a realizar las funciones que provee un administrador de base de datos. Para este modelo se consideran frontales grandes o pesados y bases de datos pequeñas o livianas, por ejemplo power builder con SQL Server. En general este tipo de arquitectura tiene mejor aplicación en sistemas de apoyo de decisiones y sistemas de información ejecutiva, tienen pocas posibilidades de aplicarse en sistemas de misión crítica.

- ***Cliente Liviano - Servidor Pesado (Fat Server - Thin Client).*** Este es el caso opuesto al anterior, el proceso cliente es restringido a la presentación de la interfaz de usuario, mientras que el grueso de la aplicación corre por el lado del servidor de aplicación. En general este tipo de arquitectura presenta una flexibilidad mayor como para desarrollar un gran espectro de aplicaciones, incluyendo los sistemas de misión crítica a través de servidores de transacciones y grandes servidores de bases de datos. Un ejemplo puede ser Oracle con Visual Foxpro.

- ***Cliente Liviano - Servidor Liviano / Cliente Pesado - Servidor Pesado.*** Servidor Gordo con Cliente Gordo y Servidor Flaco con cliente flaco, respectivamente. Esta clasificación nivela los dos componentes, tanto clientes como servidores, por ejemplo informix con power builder y SQL con visual basic, en ambos casos el cliente y el servidor tienen igual importancia unos de otros.

Cuadro 2.2



2.6.2. POR PLANOS O CAPAS (TIER)

Una de las más comunes y discutidas distinciones entre las diferentes arquitecturas Cliente Servidor se basan en la idea de planos, la cual es una variación sobre la división o clasificación por tamaño de componentes en especial clientes grandes y servidores amplios.

Esto se debe a que se trata de definir el modo en que las prestaciones funcionales de la aplicación serán asignadas, y en qué proporción, tanto al cliente como al servidor. Dichas prestaciones se deben agrupar entre los tres componentes básicos para Cliente / Servidor: interfaz de usuario, lógica de negocios y los datos compartidos, cada uno de los cuales corresponde a un plano.

Dentro de esta categoría se tiene las aplicaciones en dos planos (two-tier), tres planos (three-tier) y multi planos (multi-tier). Dado que este término ha sido sobrecargado de significados por cuanto se lo utiliza indistintamente para referirse tanto a aspectos lógicos (Software) como físicos (Hardware), aquí se esquematizan ambas posibilidades.

2.6.2.1 PLANOS A NIVEL DE SOFTWARE

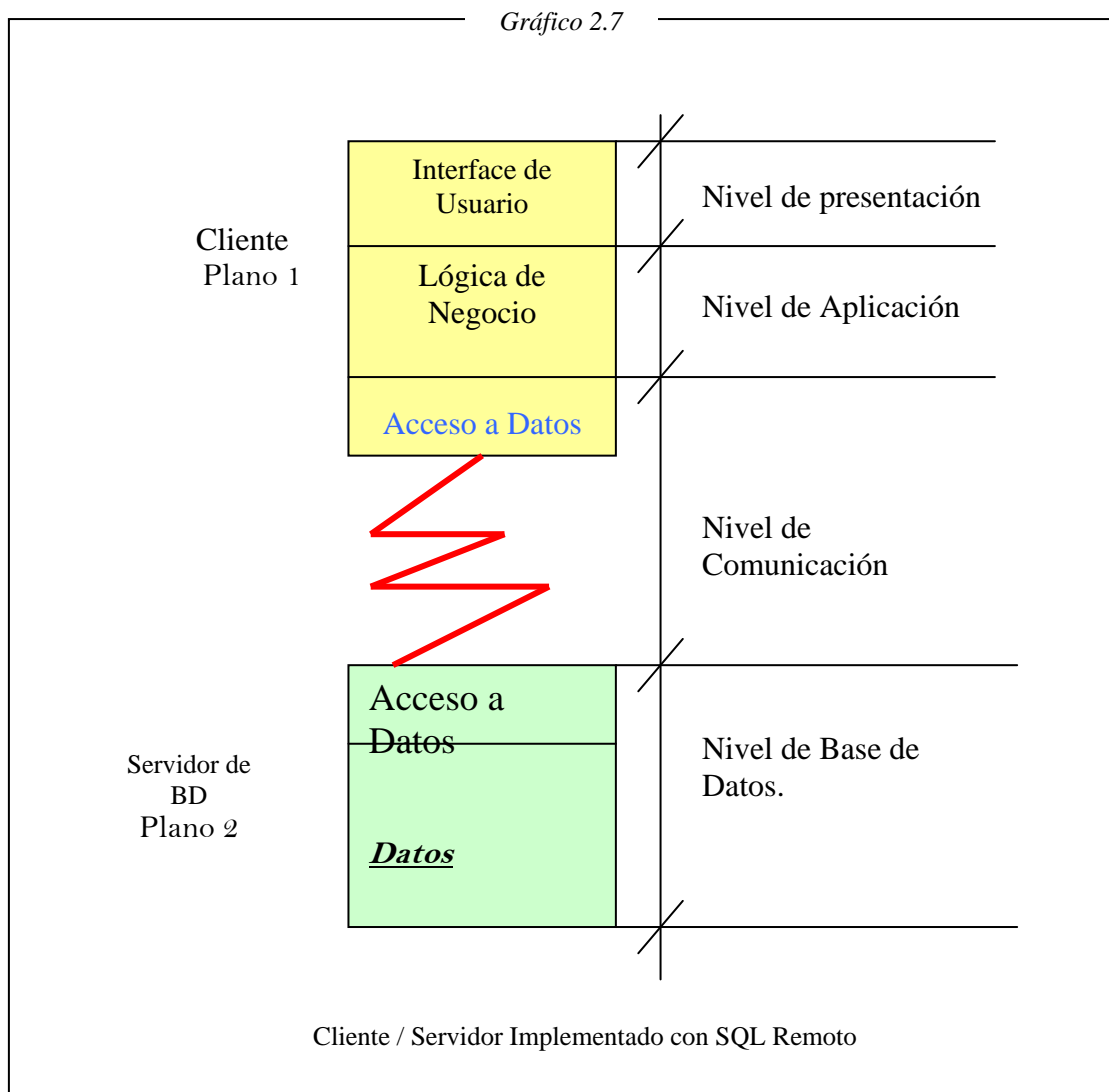
Este enfoque o clasificación es el más generalizado y el que más se ajusta a los enfoques modernos, dado que se fundamenta en los componentes lógicos de la estructura Cliente / Servidor y en la madurez y popularidad de la computación distribuida.

Por ejemplo, esto permite hablar de servidores de aplicación distribuidos a lo largo de una red, y no tiene mucho sentido identificar a un equipo de hardware como servidor, si no más bien entenderlo como una plataforma física sobre la cual pueden operar uno o más servidores de aplicaciones.

Cliente / Servidor Dos Planos. Esta estructura se caracteriza por la conexión directa entre el proceso cliente y un administrador de bases de datos. Dependiendo de donde se localice el grupo de tareas correspondientes a la lógica de negocios se pueden tener a su vez dos tipos distintos dentro de esta misma categoría, los cuales son implementaciones con SQL remoto e Implementaciones con procedimientos almacenados.

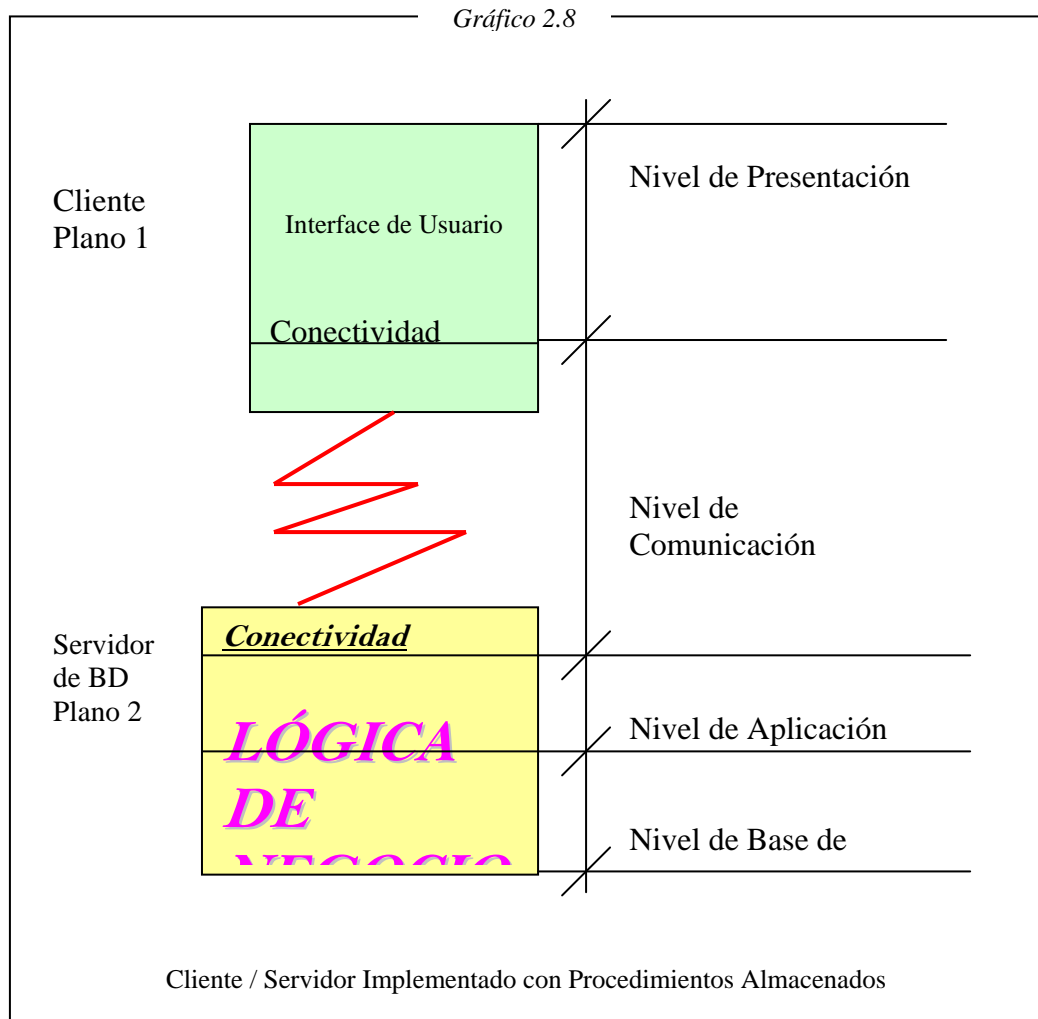
Implementado con SQL Remoto. En este esquema el cliente envía mensajes con solicitudes SQL al servidor de bases de datos y el resultado de cada instrucción SQL es devuelto por la red, no importando si son uno, diez, cien o mil registros. Es el mismo cliente quien debe procesar o validar todos los registros que le fueron devueltos por el servidor de base de datos, según el requerimiento que él mismo hizo, (Gráfico 2.7)

Esto hace que este tipo de estructura se adecúe a los requerimientos de aplicaciones orientadas a los sistemas de apoyo y gestión, pero resultan inadecuados para los sistemas críticos en que se requieran bajos tiempos de respuesta, la lógica de negocio está en el cliente. [www029]



Implementado con Procedimientos Almacenados. En este esquema el cliente envía llamadas a funciones que residen en la base de datos, y es ésta quien resuelve y procesa la totalidad de las instrucciones SQL agrupadas en la mencionada función. (gráfico 2.8)

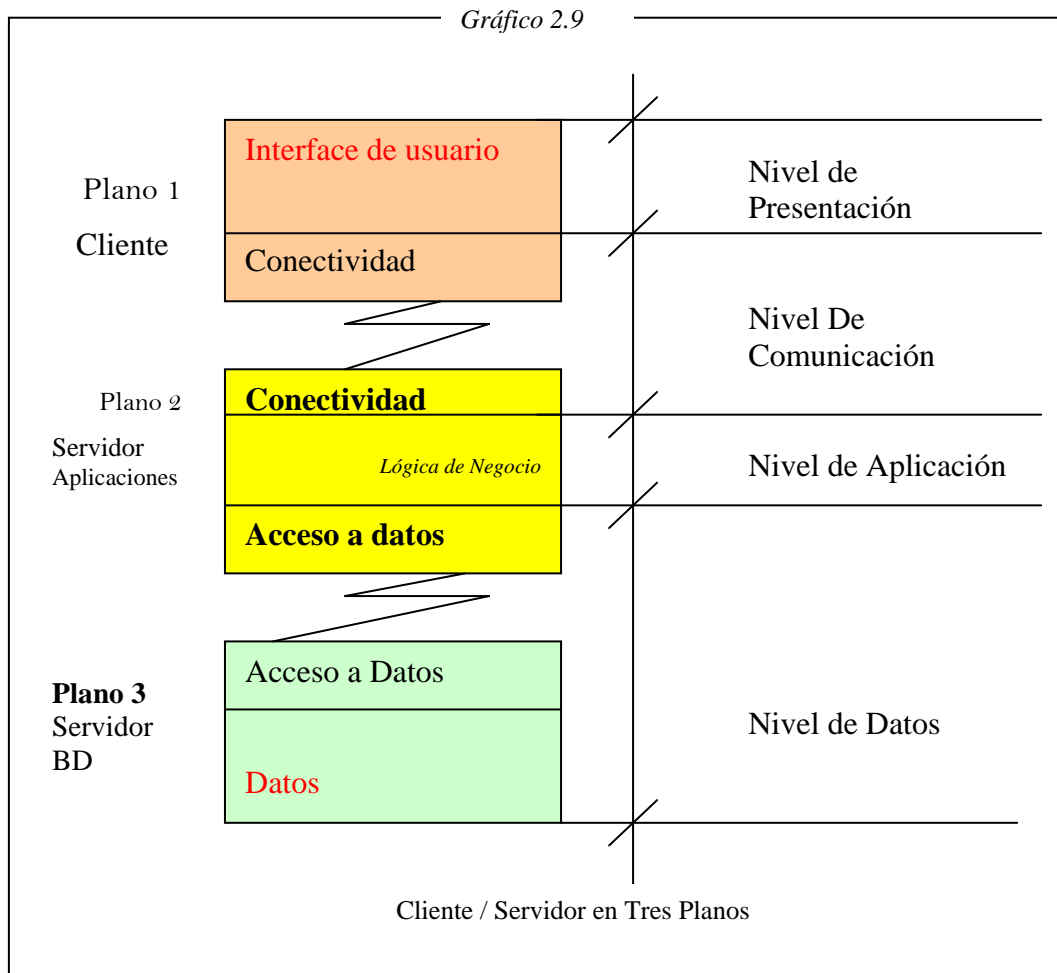
En sí, la base de datos junto con sus funciones procesa todas las peticiones del cliente, la lógica del negocio está en el plano del servidor, exactamente en el servidor de base de datos.



Cliente / Servidor Tres Planos: Esta estructura se caracteriza porque sus aplicaciones se generan en base a dos capas principales de software, más la capa correspondiente al servidor de aplicaciones o lógica del negocio.

Al igual que en la arquitectura dos capas, y según las decisiones de diseño que se tomen, se puede balancear la carga de trabajo entre el proceso cliente y el nuevo proceso correspondiente al servidor de aplicación.

En este esquema el cliente envía mensajes directamente al servidor de aplicación el cual debe administrar y responder todas las solicitudes. Es el servidor, dependiendo del tipo de solicitud, quien accede y se conecta con la base de datos. (gráfico 2.9) [www029]



2.6.2.2 PLANOS A NIVEL DE HARDWARE

Esta clasificación del modelo Cliente / Servidor se basa igualmente en la distribución de los procesos y elementos entre sus componentes, pero centrándose en la parte física del mismo, en el que la administración de la interfaz gráfica se asocia a los clientes PC y la seguridad e integridad de los datos quedan asociados a ambientes mainframe o por lo menos a servidores locales y / o centrales.

2.6.3 POR LA NATURALEZA DEL SERVICIO

Servidores de Bases de Datos: Obviamente la generación de aplicaciones Cliente / Servidor está íntimamente asociada a la utilización de servidores de bases de datos; dependiendo de los requerimientos y restricciones de la organización, se debe elegir entre una arquitectura dos o tres planos.

Según los distintos estándares de SQL (SQL-89, SQL-92, SQL3) el servidor debe proveer un acceso compartido a los datos con los mecanismos de protección necesarios, así como proveer mecanismos para seleccionar resultados dentro de un conjunto de datos, posibilitando un ahorro en todos los procesos, especialmente en los de comunicación. El servidor debe también proveer mecanismos de concurrencia, seguridad y consistencia de datos, basado principalmente en el concepto de transacción en el que todo se realiza, y por lo tanto es permanente.

Los servidores de bases de datos actuales son una mezcla de SQL estándar más otras extensiones propias de cada proveedor. Por ejemplo casi todas las bases de datos están provistas con procedimientos almacenados (stored procedures), desencadenantes (triggers) y restricciones (constraints) pero presentan diferencias importantes en su implementación.

Es claro que esto obedece a presiones comerciales para tratar de extender los mecanismos de bases de datos para que realicen más funciones de las que corresponden a un servidor SQL relacional, con el objeto de tener una mayor participación en el espectro de los sistemas Cliente / Servidor por parte de los proveedores de bases de datos y como una manera de diferenciar sus productos.

Una de las posibilidades de implementar de mejor forma un sistema Cliente / Servidor en dos planos, sin olvidar todas sus restricciones y limitaciones, es a través de **procedimientos almacenados**, que son funciones que agrupan un conjunto de instrucciones y lógica de procedimientos SQL, los cuales son compilados y almacenados en la misma base. Ya se hizo notar que con estos mecanismos los proveedores de bases de datos introducen la lógica de negocios dentro de su servidor propietario; lo lógico sería mantener una independencia entre datos, códigos y reglas de comportamiento, por medio de servidores de aplicaciones, mecanismos que todavía no son tan utilizados.

El rol principal de *los procedimientos almacenados* es proveer la parte servidora de la lógica de una aplicación Cliente / Servidor, es decir *vendría a reemplazar al servidor de aplicaciones en una arquitectura tres planos*; si bien los procedimientos almacenados permiten a un servidor brindar servicios aptos para OLTP (On Line Transaction Processing), no se compara con los rendimientos alcanzados por un servidor de aplicaciones o transaccional.

Desencadenantes. (Triggers) Son mecanismos que permiten realizar acciones automáticamente sobre los datos, las cuales están asociadas a algún evento definido. Normalmente son implementados a través de procedimientos almacenados.

Los eventos a los cuales se hace referencia están asociados a las actualizaciones de tablas mediante sentencias delete, insert o update, y son llamados implícitamente al suceder cualquiera de estos eventos, a diferencia de los procedimientos almacenados que son llamados explícitamente por un proceso cliente.

Restricciones : Al igual que los desencadenantes, son acciones que se realizan asociadas a algún evento determinado y están orientadas a llevar a cabo validaciones más simples de datos. Los tipos de eventos son los mismos que para los desencadenantes.

2.7. HERRAMIENTAS CASE EN CLIENTE SERVIDOR

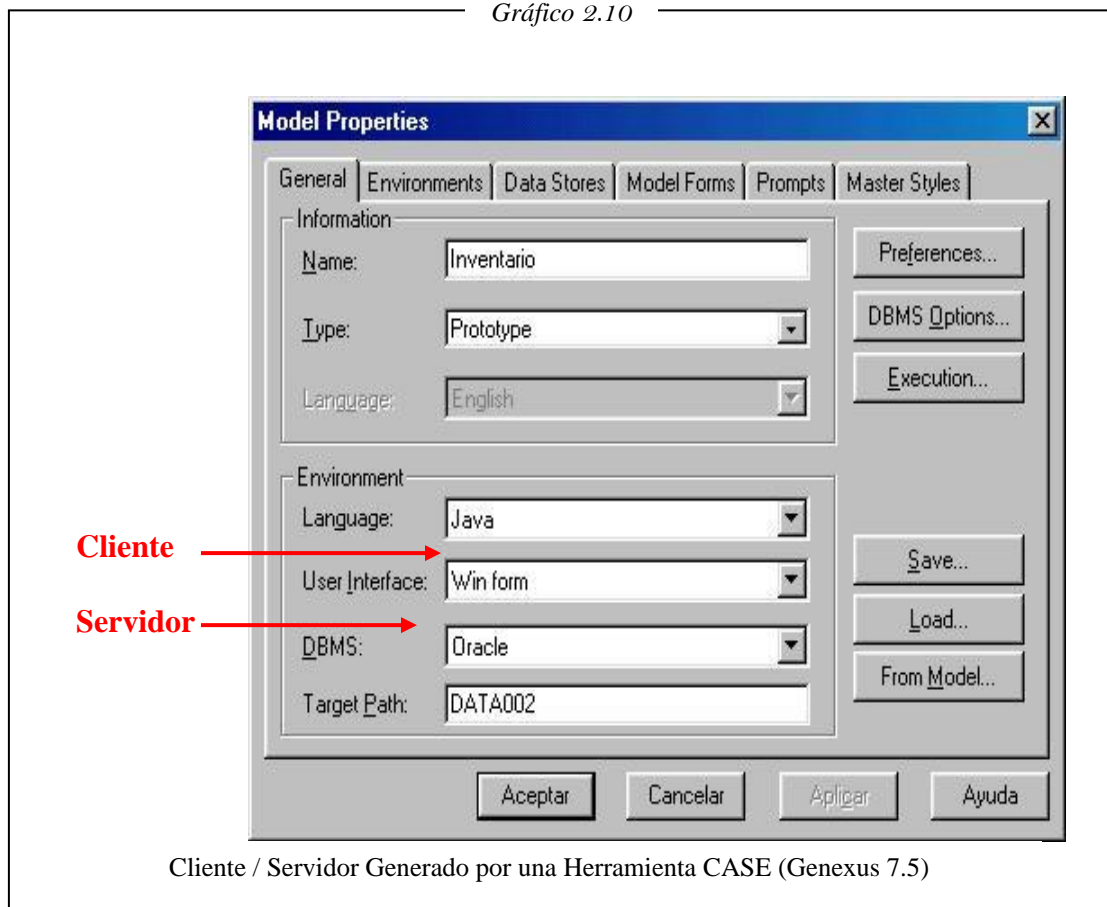
Una Herramienta CASE Cliente / Servidor provee de: modelos de datos, generación de código, registro del ciclo de vida de los proyectos, múltiples repositorios de usuarios, documentación, diseño e implementación de bases de datos, generación de prototipos, procesos de reingeniería, comunicación entre distintos ingenieros, ayuda en línea, entre otras.

En el gráfico 2.10 se observa una aplicación Cliente / Servidor, generada por una Herramienta CASE, en este caso genexus. La figura esquematiza a un cliente Java trabajando para un servidor Oracle, cabe mencionar que el diseño se realiza completamente con la herramienta, y este se encarga de generar todo el proceso de desarrollo de aplicaciones, solamente toca especificar qué lenguaje y a qué base de datos conectarse, en la figura se observa las flechas para los distintos clientes y servidores.

Las principales Herramientas CASE, que generan aplicaciones Cliente / Servidor son: KnowledgeWare's Application Development Workbench, TI's Information Engineering Facility (IEF), Andersen Consulting's Foundation for Cooperative Processing, Genexus, etc.

Es importante esclarecer que las Herramienta CASE C / S a las cuales se enfoca este estudio son de tipo Workbench, ya que solamente aquellas solucionan todo el ciclo de vida de un sistema, y es donde se puede observar directamente la generación de una arquitectura Cliente / Servidor. Lo que no sucede con otros tipos de CASE, que sirven solo para generar alguna parte de ciclo de vida, como por ejemplo el diseño del modelo de datos.

Gráfico 2.10



2.7.1 REQUERIMIENTOS DEL CLIENTE, PARA SOLUCIONES GENERADAS CON HERRAMIENTAS CASE

Además de los requerimientos básicos que solicitan los generadores CASE, como: espacio en disco, cantidad de memoria, versión del sistema operativo, etc. Es necesario tomar en cuenta estos aspectos:

Cada estación de trabajo deberá disponer de una conexión vía TCP/IP con el servidor donde se realizará la compilación, se habla de TCP/IP ya que es el protocolo más utilizado por los sistemas de red, pero dependerá de lo que se esté manejando para instalar el protocolo más adecuado como por ejemplo IPX SPX para novell.

Esta conexión, se utiliza para transferir los fuentes generados al servidor, se recomienda utilizar FTP como protocolo de transferencia y para la ejecución de comandos remotos REXEC que es el requerido para efectuar la compilación de programas. El esquema de transferencia FTP y ejecución remota se basa en la especificación Winsock 1.1 por lo que, prácticamente, cualquier producto que provea soporte TCP/IP puede aprovecharlo. El programa que realiza la transferencia y compilación requiere directamente los servicios de: WINSOCK.DLL. [www027]

2.7.2 REQUERIMIENTOS DEL SERVIDOR, PARA SOLUCIONES C/S GENERADAS CON HERRAMIENTAS CASE

Para estar en condiciones de conectarse desde el cliente al servidor y realizar las tareas de transferencia y compilación se requieren, además de existir un medio físico que los conecte, de los siguientes productos y servicios instalados y en funcionamiento.

TRANSFERENCIA DE FUENTES. La transferencia de fuentes de los programas generados permite tres esquemas básicos: FTP, Copy y Don't transfer.

Los dos primeros asumen que el directorio del modelo no está en el servidor, y que, por consiguiente, se necesita transferir los fuentes allí generados al servidor para poder compilarlos.

El tercer esquema, Don' transfer asume que el directorio mencionado arriba ya se encuentra en el servidor, el cliente lo ve como un disco de red y en consecuencia, no es necesaria la transferencia.

Si se selecciona FTP como esquema de transferencia, es necesario que el servidor tenga disponible y 'escuchando' el servicio de FTP.

Si se selecciona Copy o Don't transfer como esquemas de transferencia, la estación de trabajo debe 'ver' los discos del servidor como discos de red. En otras palabras, el servidor debe funcionar también como servidor de archivos para la estación. En esta situación, no es necesario activar el servicio FTP.

COMPILACIÓN. Con el fin de realizar la compilación de los fuentes generados, se requiere el servicio de ejecución remota REXEC activo en el servidor.

DBMS. Para poder compilar los fuentes generados y armar correctamente programas ejecutables, es necesario hacer referencia a archivos que el proveedor del DBMS distribuye. Estos archivos son: include files (*.h), bibliotecas de rutinas compiladas.

Esta preferencia permite establecer en qué directorio se encuentran los archivos de soporte mencionados. No existe valor por defecto para esta preferencia y debe ser especificada. En caso de no hacerlo, se producirá un error al intentar compilar link-editar en los programas generados.

Algunos DBMS requieren establecer una conexión a una base de datos durante la precompilación para poder validar las sentencias SQL incluidas en los fuentes y / o generar información, en dicha base de datos relativa al programa precompilado. En la tabla 2.1 se demuestra cuales DBMS necesitan una conexión durante la precompilación con Herramientas CASE.

Tabla 2.1

<i>DBMS</i>	<i>Requerimientos</i>
DB2	si
SQL	si /Creación/ Reorganización
Oracle	no
Informix	no

DBMS que requieren conexión para la precompilación

2.8 FORMAS DE ENLACE

Las Herramientas CASE que generan todo el proceso de desarrollo, necesitan mecanismos de conexión desde el cliente hacia la base de datos, estos procesos comunes de enlace se los conoce como drivers: ODBC, JDBC, etc.

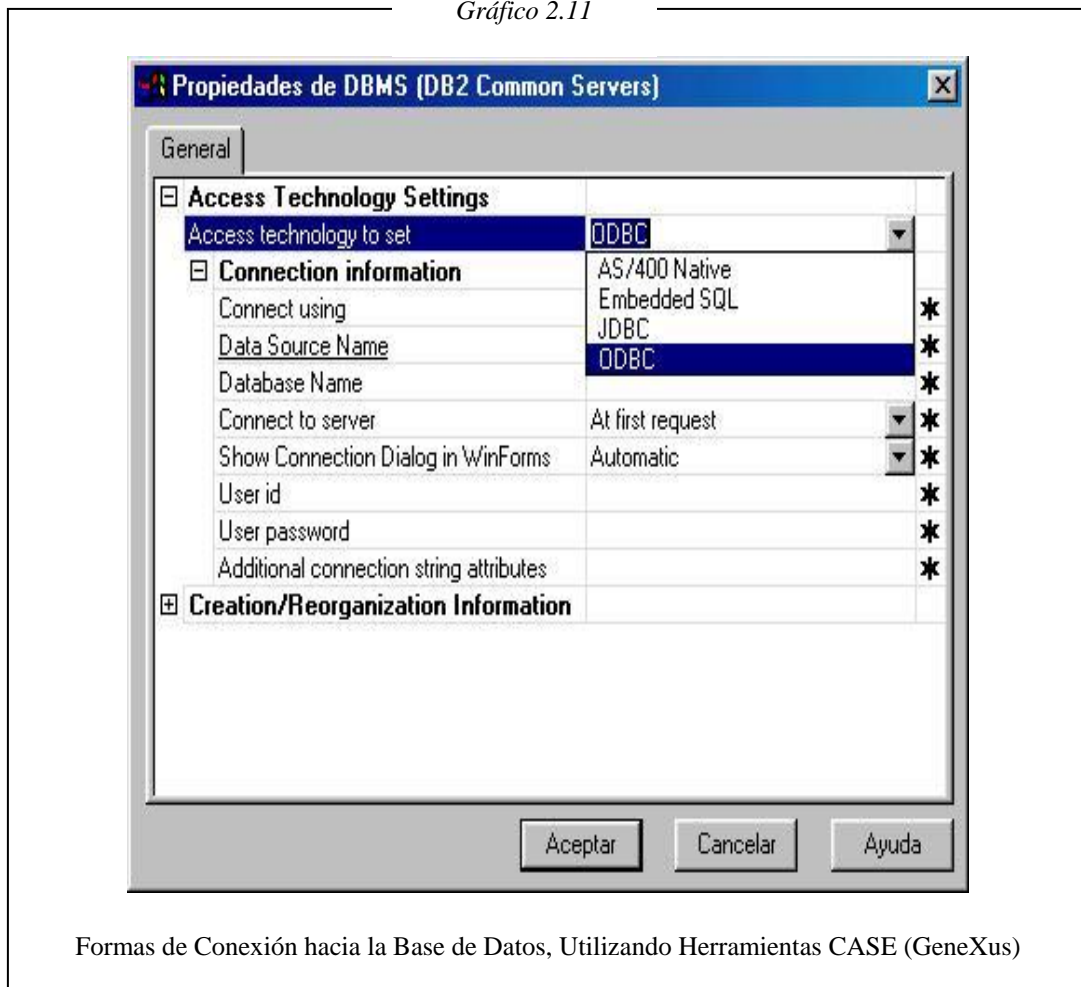
ODBC. Es la interfase de conexión hacia una base de datos más utilizada por las personas encargadas de desarrollar soluciones de software.

Una aplicación Cliente / Servidor necesita estrictamente un driver ODBC o similar para poder existir ya que es el mecanismo principal donde se puede ver la distribución de recursos, de esta manera se puede tener independencia entre los datos y la aplicación.

La tecnología ODBC permite también la conexión mediante tecnología OLEDB, que es una API elaborada con C++ la cual utiliza las interfases de mejor manera para el acceso hacia: datos, funciones y objetos sin importar si se trata de una base de datos normal, o sea no importa si el enlace es hacia: archivos de texto, imágenes, sonidos, direcciones de correo electrónico, etc. Este tipo de tecnología se utiliza mucho con bases de datos objeto – relacionales.

Cuando se genera aplicaciones con Herramientas CASE, se debe especificar las formas de conexión en los primeros pasos del desarrollo, exactamente cuando se empieza a crear la base de conocimiento, en el gráfico 2.11 se observa las formas de conexión utilizando Herramientas CASE.

Gráfico 2.11



JDBC. Es utilizado especialmente por el generador Java, es un conjunto de drivers odbc, es un odbc superior o es el API estándar de acceso a base de datos utilizando clientes Java.

JODBC es un ODBC escrito en el lenguaje JAVA que presta mayores facilidades de: portabilidad, instalación y seguridad, lo que no sucede con ODBC que es escrito en lenguaje C el cual no es portable.

Embedded SQL. Es utilizado por el generador C/SQL definido en el modelo de la Herramienta CASE, cuando la preferencia 'Default Access Method' tiene el valor 'Embedded SQL' se los utiliza cuando los datos están llamados con sentencias de SQL puras. (gráfico 2.11)

AS/400 Native. RPG. Es utilizado por los generadores Rpg y Cobol definidos en el modelo de la Herramienta CASE de referencia (Genexus), se los utiliza para ambientes AS/400 y RS/6000.

Tabla 2.2

ARQUITECTURAS CENTRALIZADAS	
Plataforma	Lenguaje Generado
AS/400	Cobol/400 RPG/400

ARQUITECTURA FILE SERVER		
Plataforma	Lenguaje Generado	
DOS	Foxpro Clipper	DBF
WINDOWS	Foxpro for Windows Visual Basic Visual Fox	DBF Access

ARQUITECTURA CLIENTE / SERVIDOR	
Cliente	Database Server
Foxpro for windows Visual Basic Visual Fox Pro Java C#	MS SQL Oracle UDB Informix DB/2 RS/600 DB2/400

Generación de Soluciones en Diferentes Arquitecturas por medio de una sola
Herramienta CASE (GENEXUS 7.0) [LIB001]

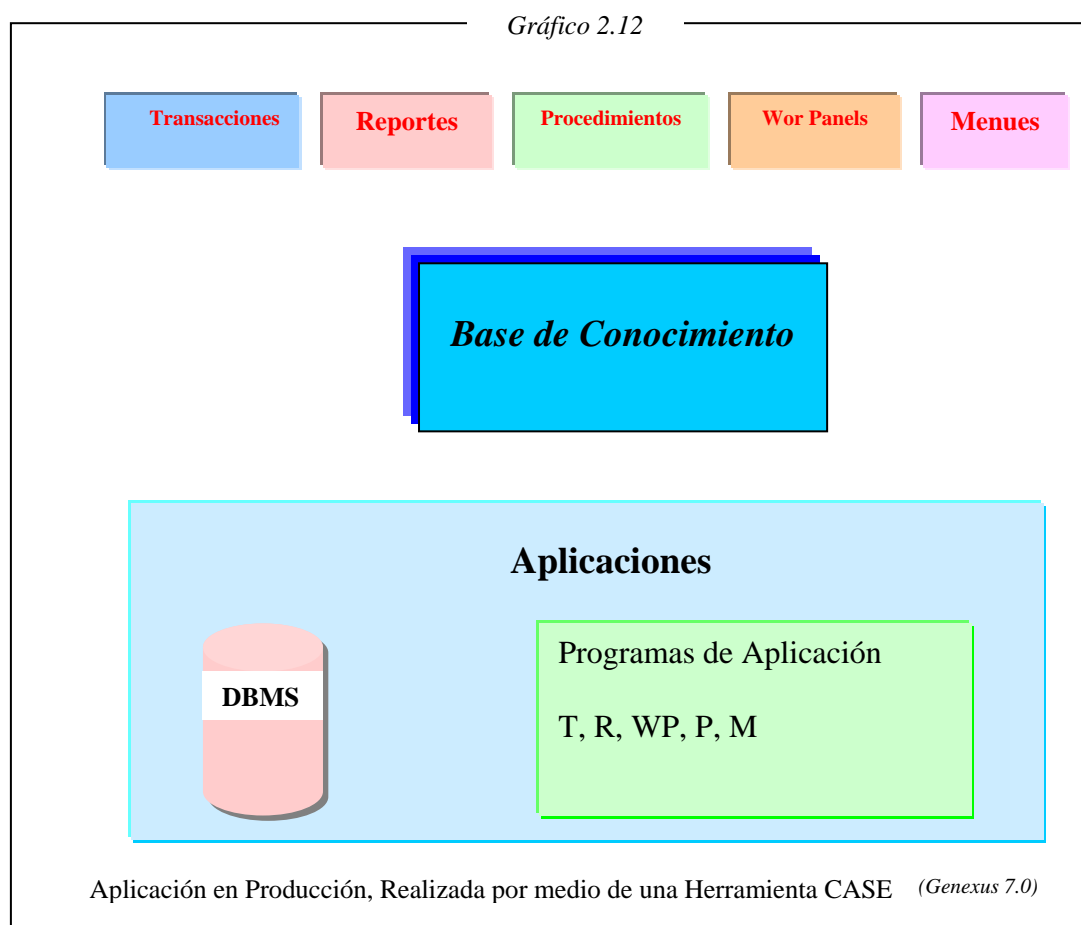
La Tabla 2.2 muestra las diferentes opciones de cómo una Herramienta CASE (Genexus 7.0) puede generar varias aplicaciones en diferentes arquitecturas a partir de una sola especificación netamente a nivel de diseño, esta es una de las principales ventajas de las Herramientas CASE, ya que solo se necesita crear un solo programa y se lo puede generar en cualquier arquitectura requerida. [LIB001]

2.9. REINGENIERÍA Y ANÁLISIS DE IMPACTOS CON HERRAMIENTAS CASE CLIENTE / SERVIDOR

El concepto de reingeniería, está matizado por dos aspectos primordiales; El primero se centra en aquella reingeniería en la cual la Herramienta CASE genera soluciones desde una aplicación ya elaborada, por ejemplo: La reingeniería que realiza una Herramienta CASE de un sistema elaborado previamente con Oracle y Visual Basic a base de programación convencional; entonces el proceso de reingeniería con Herramientas CASE se encarga de absorber toda la información de esta aplicación ya realizada y reprogramarla a la metodología de dicha herramienta.

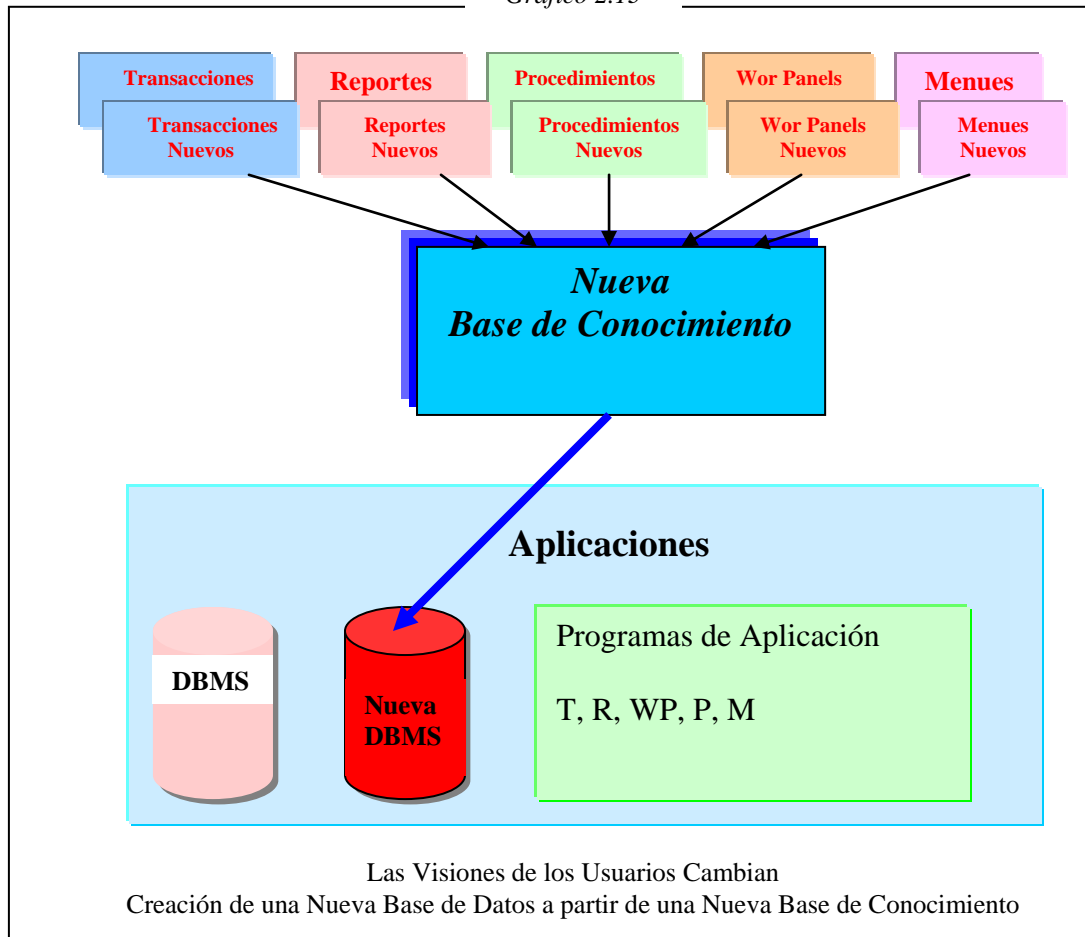
El segundo aspecto de la reingeniería es aquella que se utiliza cuando se quiere realizar algún cambio en la aplicación, también se la considera como fase de mantenimiento y actualización de la aplicación; para esto se debe recorrer los bucles: diseño – prototipo y diseño – producción.

Con la finalidad de entender este proceso y los impactos que puede producir un cambio en toda la aplicación, se empezará observando como está estructurada una aplicación realizada con una Herramienta CASE. (gráfico 2.12)



Una vez que el software está funcionando, los usuarios cambian las visiones y existen muchas modificaciones en la aplicación. Para poder realizar estos cambios (reingeniería y/o mantenimiento) las herramientas como genexus primeramente generan una nueva base de datos, alimentada por la nueva base de conocimiento (gráfico 2.13) [LIB001]

Gráfico 2.13

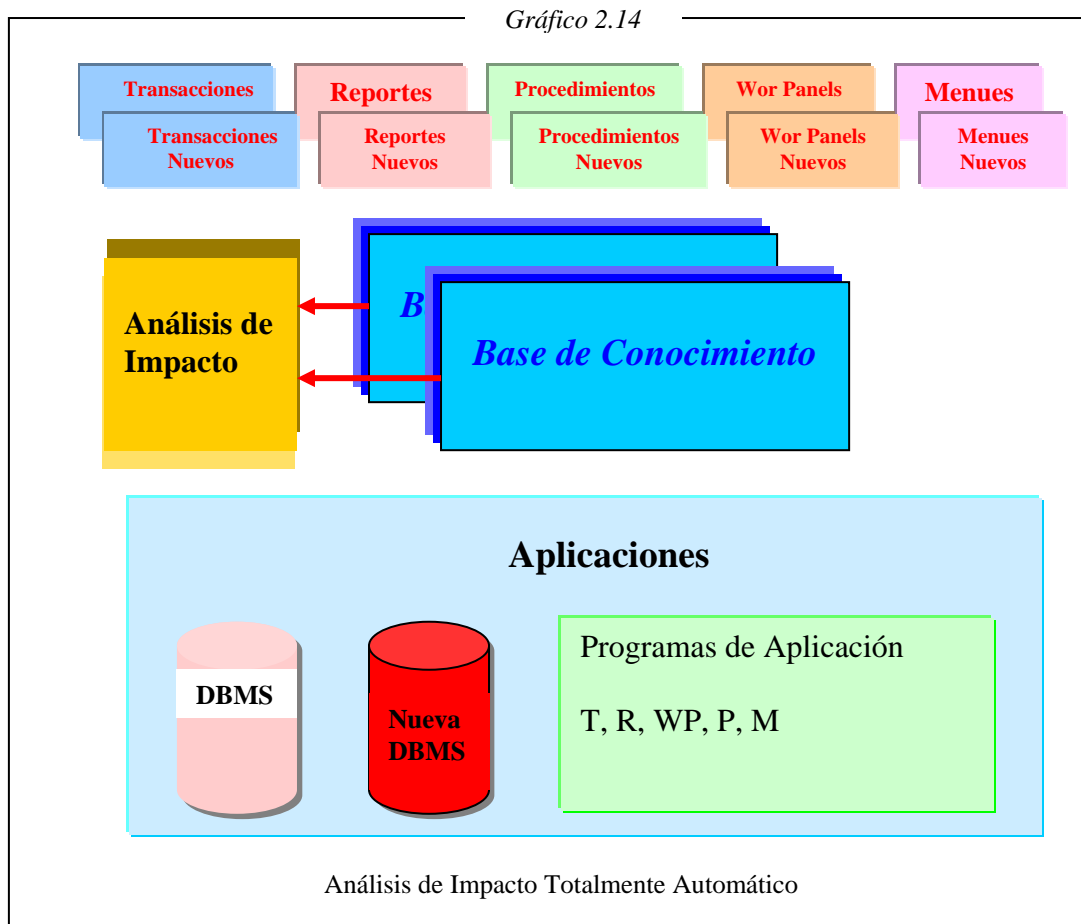


Algunas veces, la nueva base de datos coincide con la anterior, en este caso todos los programas anteriores existentes seguirán siendo válidos, ya que la base de datos antigua sigue siendo parte del sistema, hasta que se realice un análisis de impacto general: a la base, a las formas y a los demás objetos de la aplicación. Este análisis de impacto se genera cuando se pasa de diseño a producción o a prototipo.

Cuando se realiza algún cambio en la aplicación, inclusive se conservan los datos existentes en las tablas, si los cambios realizados concuerdan con los existentes; por ejemplo se puede cambiar de un char de 20 a un char de 50 pero no conservará los datos si cambia un char de 20 por un int de 10.

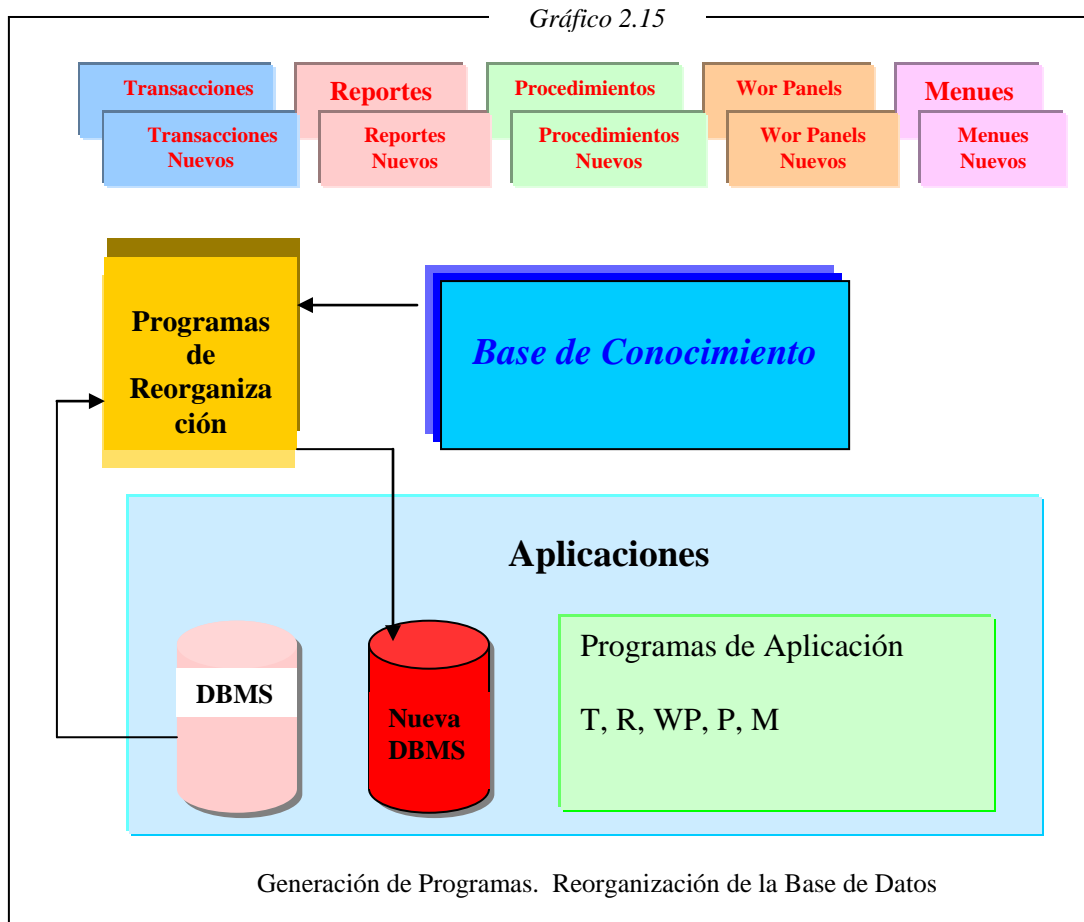
Una vez confirmadas las nuevas visiones de los usuarios, la herramienta procede automáticamente a generar un análisis de impacto sobre los objetos. (gráfico 2.14)

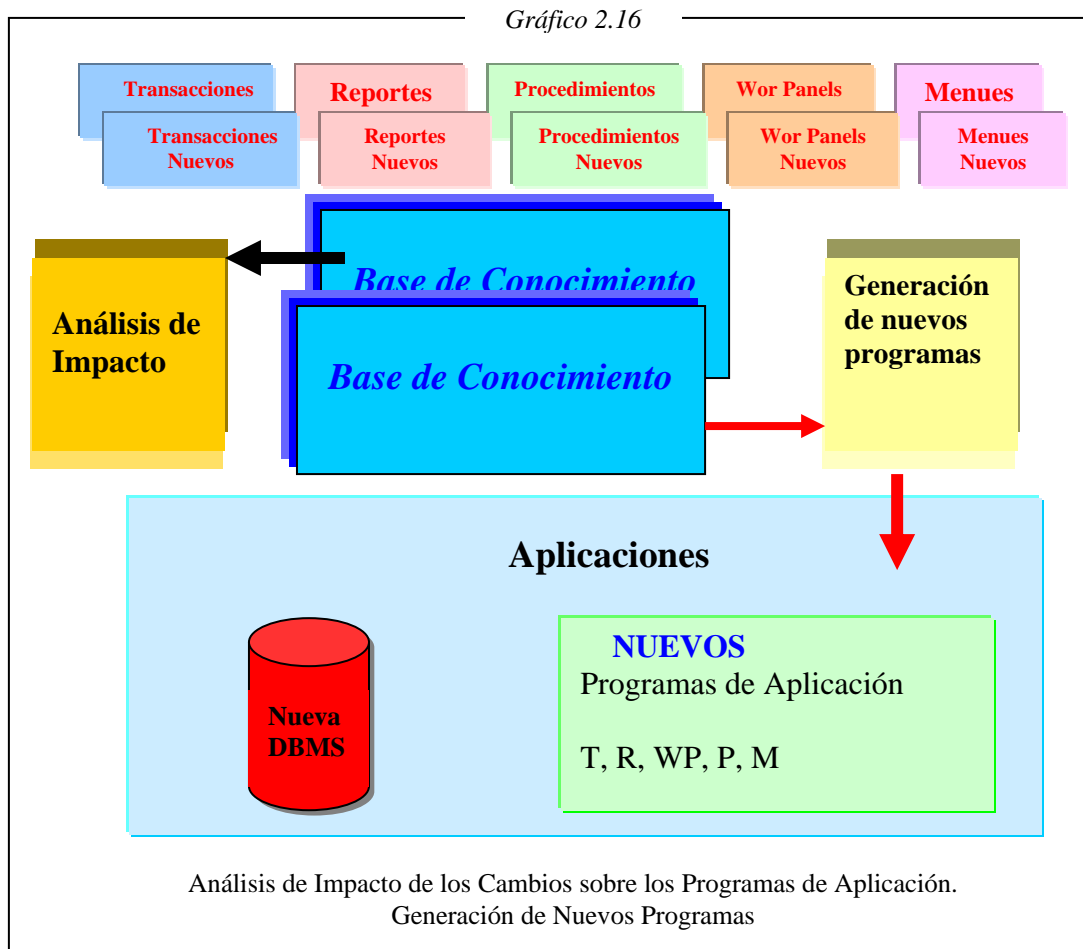
Al realizar el análisis de impacto, la herramienta muestra en pantalla todos los cambios que se realizarán en cada objeto de la aplicación, conjuntamente con los problemas y las pérdidas de datos que se ocasionarán por los cambios realizados, para evitar problemas inesperados se recomienda probar primeramente en modo prototipo.



Después de haber generado el análisis de impacto, la herramienta produce otro informe en el que se visualiza como quedaron los nuevos objetos, ya sean producto de una reorganización o de una creación.

Como todavía estos cambios reposan en la base de conocimiento, habrá que hacer una reorganización de la base de datos, de las transacciones, de los paneles de trabajo, etc para que los cambios surtan efecto en la aplicación. Las Herramientas CASE, empiezan realizando los respectivos cambios por la base de datos. (gráfico 2.15)





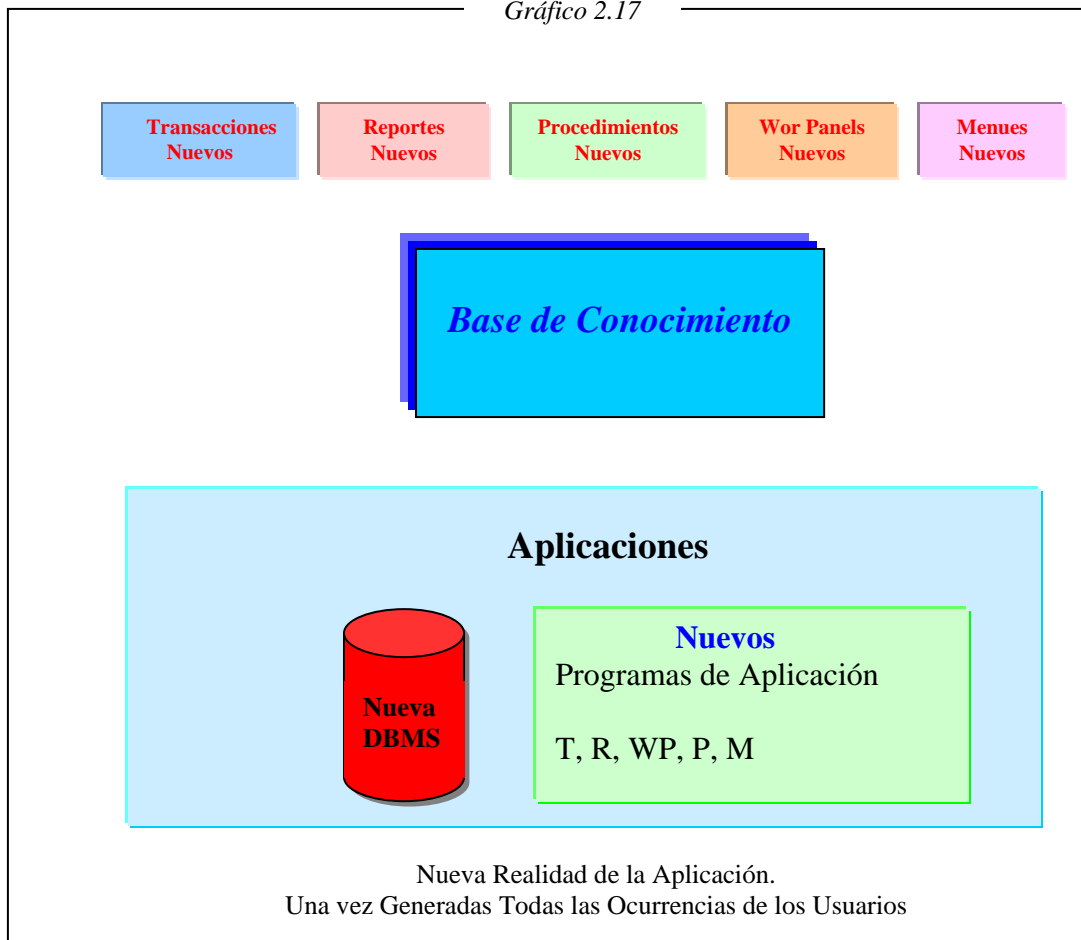
En el gráfico 2.16 se muestra los cambios automáticos sobre los lenguajes de programación como: VB, FP, Java, C#, Fox, etc.

En el gráfico 2.17 se observa a una aplicación final, después de su ciclo de reingeniería y/o mantenimiento, en síntesis es el producto final de una aplicación.

Cabe indicar que estas herramientas son totalmente adaptables a todas las necesidades repentinas de los clientes, es por eso que se le toma a la reingeniería como los procesos de ingeniería inversa y los procesos de mantenimiento y actualización del sistema, enfocado todo a las necesidades y requerimientos esporádicos de los usuarios.

Para poder generar todos estos cambios sin necesidad de reprogramar a toda la aplicación, las Herramientas CASE manejan el concepto de que ninguna base de datos es estable y que todas pueden estar sujetas a cualquier cambio, adaptación o modificación repentina. [LIB001] [LIB002]

Gráfico 2.17



2.10. TÁCTICAS PARA SOLUCIONES CLIENTE / SERVIDOR GENERADAS CON HERRAMIENTAS CASE

- Si tiene un gran volumen de aplicativos ya desarrollados, es conveniente contrastar lo realizado versus las técnicas de análisis y diseño que proporcionan las Herramientas CASE, para ver si es viable o no la aplicación de esta metodología.
- Si tiene presión por resultados a corto plazo, el empleo de un Lower CASE le será de utilidad, si se basa en modelos de datos y procesos claros y definidos.
- Si desea realizar proyectos de gran envergadura es recomendable aplicar Upper y Lower CASE a la vez.
- Si trabaja con archivos de grandes dimensiones, es recomendable que la Herramienta CASE soporte el diseño y generación de la bases de datos, para que no tenga problemas posteriores con el tipo y dimensión de los datos a almacenarse.

- Si no tiene formación y experiencia en el manejo de metodologías CASE es recomendable contar con asesoría especializada, que capacite al personal y supervise los avances de implantación de la nueva tecnología.
- Evalúe la eficiencia del producto, en las pruebas unitarias y de integración, y fundamentalmente en las pruebas de generación, utilice primeramente prototipos aplicados a un sistema piloto.
- Considere los recursos apropiados para usar Herramientas CASE, tanto en la parte física y en la parte lógica.
- Considere el tipo de arquitectura con la que desea trabajar y evalúe si la herramienta genera o no dicha arquitectura.
- Considere la seguridad que le brinda la herramienta, con respecto a posibles atracos y robos de los datos de la aplicación.
- Evalúe las versiones que soporta la herramienta, con respecto a la base de datos y lenguajes de programación.

2.11 CARACTERÍSTICAS DE UNA HERRAMIENTA CASE CLIENTE SERVIDOR.

Proporcionar Topologías de Aplicación Flexibles. La herramienta debe proporcionar facilidades de construcción que permita separar o distribuir la aplicación en muchos puntos diferentes, entre clientes y servidores.

Proporcionar Aplicaciones Portátiles. La herramienta debe generar código para Windows, OS/2, Macintosh, Unix y todas las plataformas de servidores conocidas. Debe ser capaz, a tiempo de corrida, desplegar la versión correcta del código en la máquina apropiada.

Control de Versión. La herramienta debe reconocer las versiones de códigos que se ejecutan en los clientes y servidores, y asegurarse que sean consistentes. También, la herramienta debe ser capaz de controlar un gran número de tipos de objetos incluyendo texto, gráficos, mapas de bits, documentos complejos y objetos únicos, tales como definiciones de pantallas y de informes, archivos de objetos y datos de prueba y resultados. Debe mantener versiones de objetos con niveles arbitrarios de granularidad; por ejemplo, una única definición de datos o una única agrupación de módulos.

[www027]

Crear Código Compilado en el Servidor. La herramienta debe ser capaz de compilar automáticamente código 4GL en el servidor para obtener el máximo performance.

Trabajar con una Gran Variedad de Administradores de Recursos. La herramienta debe adaptarse ella misma a los administradores de recurso que existen en varios servidores de la red; su interacción con los administradores de recurso debería ser negociable en tiempo de ejecución.

Trabajar con una Gran Variedad de Software Intermedios. La herramienta debe adaptar sus comunicaciones Cliente / Servidor al software intermedio existente. Como mínimo la herramienta debería ajustar los temporizadores basándose en si el tráfico se está moviendo en una LAN o WAN.

Soporte Multiusuarios. La herramienta debe permitir que varios diseñadores trabajen en una aplicación simultáneamente. Debe gestionarse los accesos concurrentes a la base de datos por diferentes usuarios, mediante el arbitreo y bloqueos de accesos a nivel de archivo o de registro.

Seguridad. La herramienta debe proporcionar mecanismos para controlar el acceso y las modificaciones a los que contiene. La herramienta debe, al menos, mantener contraseñas y permisos de acceso en distintos niveles para cada usuario. También debe facilitar la realización automática de copias de seguridad y recuperaciones de las mismas, así como el almacenamiento de grupos de información determinados, por ejemplo, por proyecto o aplicaciones.

Repositorio de Librerías Compartidas. Debe proporcionar un mecanismo para compartir las librerías entre distintos realizadores y múltiples herramientas; gestionar y controlar el acceso multiusuario a los datos y bloquear los objetos para evitar que se pierdan modificaciones inadvertidamente cuando se realizan simultáneamente. [www027]

2.12 . ASPECTOS CONSIDERABLES DE CLIENTE / SERVIDOR

Uno de los aspectos que más ha promovido el uso de sistemas Cliente/Servidor, es la existencia de plataformas de hardware cada vez más baratas. Esta constituye a su vez una de las más palpables ventajas de este esquema, la posibilidad de utilizar máquinas considerablemente más baratas que las requeridas por una solución centralizada, basada en sistemas grandes. Además, se pueden utilizar componentes, tanto de hardware como de software, de varios fabricantes, lo cual contribuye considerablemente a la reducción de costos y favorece la flexibilidad en la implantación y actualización de soluciones.

El esquema Cliente/Servidor facilita la integración entre sistemas diferentes y comparte información permitiendo, por ejemplo que las máquinas ya existentes puedan ser utilizadas pero utilizando interfases mas amigables al usuario. De esta manera, se puede integrar PCs con sistemas medianos y grandes, sin necesidad de que todos tengan que utilizar el mismo sistema operacional.

Al favorecer el uso de interfaces gráficas interactivas, los sistemas construidos bajo este esquema tienen mayor interacción intuitiva con el usuario.

Cliente/Servidor presenta la ventaja, con respecto a uno centralizado, de que no es siempre necesario transmitir información gráfica por la red pues esta puede residir en el cliente, lo cual permite aprovechar mejor el ancho de banda de la red.

Una ventaja adicional del uso del esquema Cliente/Servidor es que es más rápido el mantenimiento y el desarrollo de aplicaciones, pues se pueden emplear las herramientas existentes, por ejemplo, los servidores de SQL o las herramientas de más bajo nivel como los sockets o RPC.

La estructura inherentemente modular facilita además la integración de nuevas tecnologías y el crecimiento de la infraestructura computacional, favoreciendo así la escalabilidad de las soluciones.

El esquema Cliente/Servidor contribuye además, a proporcionar, a los diferentes departamentos de una organización, soluciones locales, pero permitiendo la integración de la información relevante a nivel global.

Cuenta con muy escasas herramientas para la administración y ajuste del desempeño de los sistemas.

Es importante que los clientes y los servidores utilicen el mismo mecanismo (por ejemplo sockets o RPC), lo cual implica que se deben tener mecanismos generales que existan en diferentes plataformas.

Además, hay que tener estrategias para el manejo de errores y para mantener la consistencia de los datos. La seguridad de un esquema Cliente/Servidor es otra preocupación importante. Por ejemplo, se deben hacer verificaciones en el cliente y en el servidor. También se puede recurrir a otras técnicas como el encriptamiento.

El desempeño es otro de los aspectos que se deben tener en cuenta en el esquema Cliente/Servidor. Problemas de este estilo pueden presentarse por congestión en la red, dificultad de tráfico de datos, mala seguridad, etc.

Un aspecto directamente relacionado con lo anterior es el de cómo distribuir los datos en la red. En el caso de una organización, por ejemplo, éste puede ser hecho por departamentos, geográficamente, o de otras maneras. Hay que tener en cuenta que en algunos casos, por razones de confiabilidad o eficiencia, se pueden tener datos replicados, y que puede haber actualizaciones simultáneas.

CAPÍTULO III

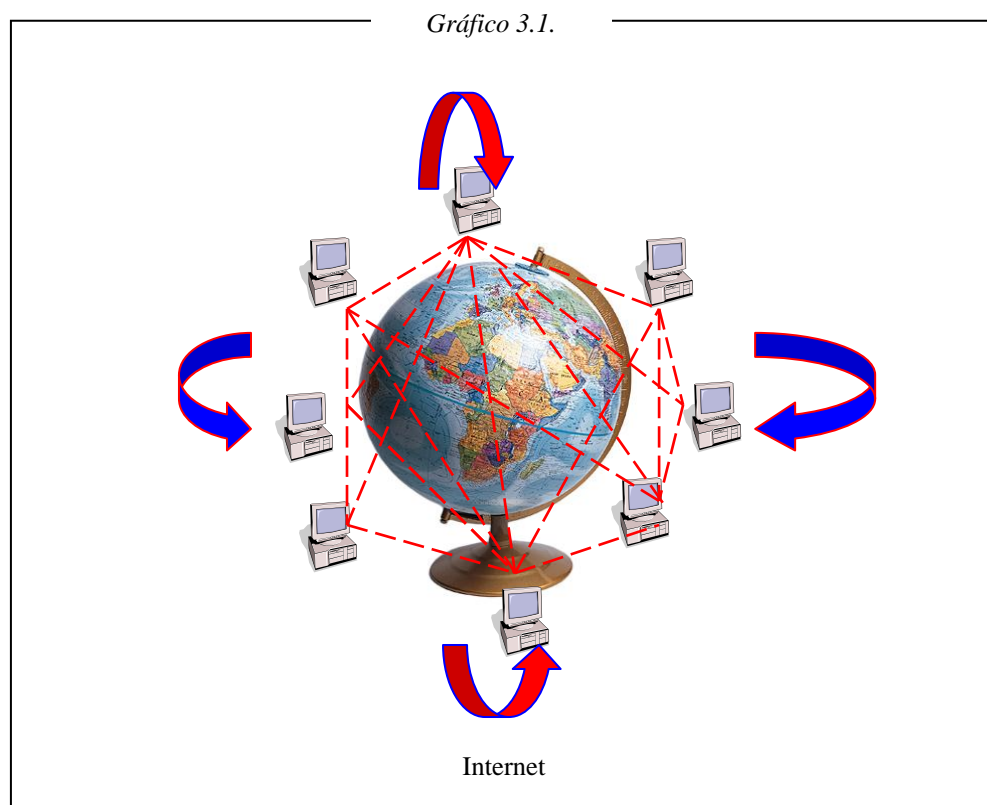
SOLUCIONES INTERNET GENERADAS POR HERRAMIENTAS CASE



- ❖ Generalidades de Internet.
- ❖ Sistema de Nombres de Dominio. (DNS)
- ❖ Direcciones IP.
- ❖ Tres Capas y Aplicaciones WEB.
- ❖ Programación de Aplicaciones WEB.
- ❖ CASE en Soluciones WEB.
- ❖ Estructura de los CASE para el WEB.
- ❖ Utilización de Componentes con CASE.
- ❖ Objetos Generados por CASE en Aplicaciones WEB.

3.1. GENERALIDADES DE INTERNET

Internet es una gran red mundial de redes privadas y públicas de ordenadores que se contactan entre sí, independientemente de la plataforma que utilicen, gracias a un lenguaje estándar de comunicaciones denominado TCP/IP Protocolo de Control de Trasmisión / Protocolo de Internet. (gráfico 3.1)



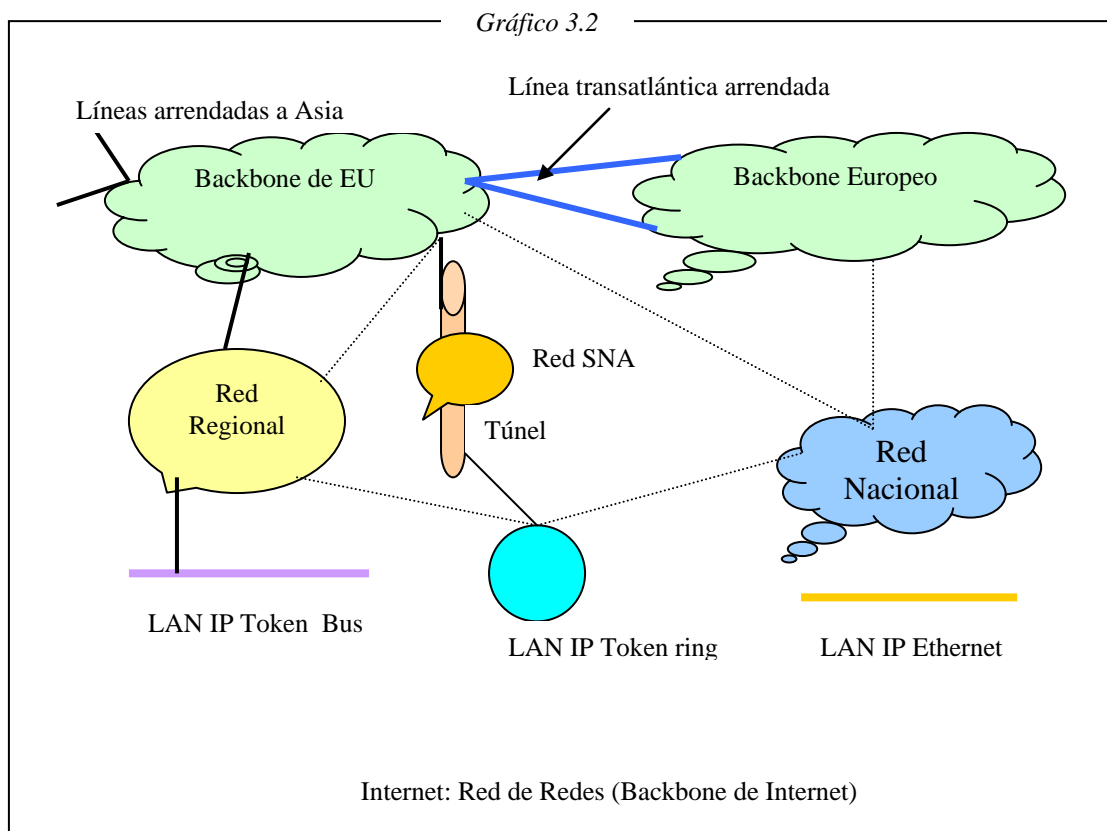
En si, no existe una definición totalmente satisfactoria de lo que es Internet, ya que se la puede concebir en relación con sus protocolos comunes, como una colección de circuitos y rutinas, como un conjunto de recursos compartidos o incluso como una disposición a intercomunicarse, es decir, como una mega red de computadores.

Sin embargo otro de los enfoques es pensar en la red como el medio a través del cual se envía y acumula información, entonces Internet es tanto un conjunto de comunidades como un conjunto de tecnologías.

Técnicamente es una *'Red de Redes de Ordenadores'*. (gráfico 3.2) Se forma por la integración en una única red con cobertura mundial, de multitud de redes de centros estatales, universidades, empresas, etc. Cada uno de estos organismos busca los métodos más adecuados para conectarse a Internet.

Existen diversas organizaciones internacionales encargadas de garantizar el funcionamiento de estos servicios. Se puede decir que Internet es un “*caos organizado*” por lo complicado de su estructura física.

Desde el punto de vista de su función, es un gran proveedor de información y servicios, es la “*Autopista de la Información.*” Determinados ordenadores conectados a Internet ofrecen una gran variedad de datos, a los que se accede con programas específicos. Además, proporciona nuevas formas de comunicación entre personas o grupos de personas, es la autopista de la información por la gran cantidad de información que circula por la red de redes.



Los Sistemas de Información basados en Internet se han hecho populares hace algunos años, como una manera eficaz de proporcionar acceso remoto y amigable a fuentes de información distribuida.

Estos sistemas están implementados combinando tecnologías de software maduras y bien conocidas; en muchos casos involucran dispositivos móviles, como: celulares, palms, pc portátiles, cámaras digitales etc.

Lo interesante de las Aplicaciones WEB es la capacidad de agregar navegación y acceso ubicuo al comportamiento transaccional típico de las aplicaciones de negocios; la navegación y la posibilidad de acceso en todo momento y desde todo lugar (gráfico 3.2) es la atracción principal del software en

Internet. Los sitios de comercio electrónico como amazon.com no son exitosos porque se puede comprar o vender, sino porque se puede encontrar fácilmente lo que se desea desde cualquier lugar del planeta.

Las soluciones web están enfocadas al desarrollo macro de una organización, son las principales artífices de la globalización ya que comparten sus recursos con usuarios del mundo entero. Los conceptos de: comercio electrónico, transacciones electrónicas, e-mail, B2B (negocios sobre negocios), B2C (negocios sobre consumidores), etc; han evolucionado de una forma vertiginosa, de tal manera que las organizaciones mas pequeñas cuentan por lo menos con su propia página web, la marca que no consta en Internet, está sentenciada a desaparecer.

Hoy en día todos los desarrolladores de: lenguajes de programación, bases de datos, herramientas CASE y sistemas operativos elaboran sus productos enfocados al web, toda herramienta tiene ya su módulo para generar aplicaciones www.

Se puede citar algunos ejemplos como: Oracle Portal, Visual Estudio.net de Microsoft, Genexus 7.5, Data Directory de Informix, etc. Si el producto no tiene facilidades considerables para aplicaciones Internet, no puede competir con los similares de su especie. Incluso hasta el hardware como: teclados, mouse, pantallas, equipos portables, etc. Tienen teclas específicas para acceder al web de una manera más rápida. [LIB005] [LIB008] [www017]

3.1.1 HISTORIA DE INTERNET

Aunque se pueda pensar que Internet es algo que ha surgido en los últimos tiempos, no es así: Internet ya lleva unas cuantas décadas, en funcionamiento.

Los inicios de Internet se remontan a los años 60. En plena Guerra Fría, Estados Unidos crea una red exclusivamente militar, con el objetivo de que en el hipotético caso de un ataque Ruso, se pudiera tener acceso a la información militar desde cualquier punto del país, de tal manera que si un nodo fuera destruido, existieran varios caminos de comunicación; esta red se creó en 1969 y se llamó ARPANET.

En principio la red contaba con 4 ordenadores, distribuidos entre distintas universidades del país: la Universidad de California en Los Ángeles, la Universidad de California en Santa Bárbara, el Instituto de Investigación de Stanford y la Universidad de Utah. En un par de años otras instituciones educativas y de investigación se unieron a la red.

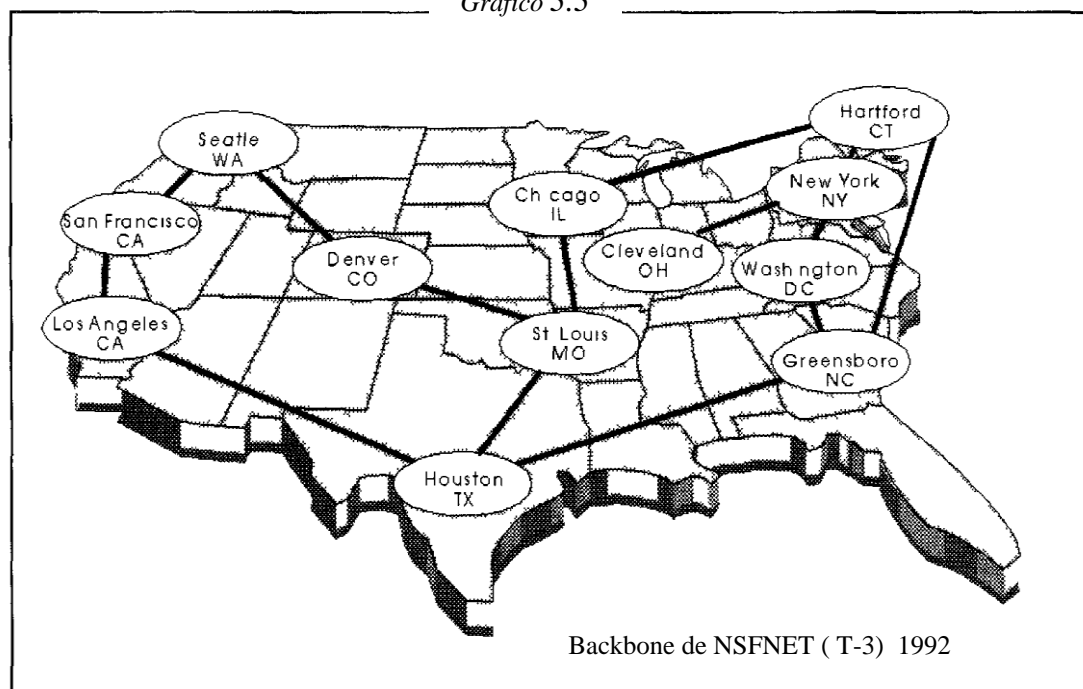
Dos años después, ya contaba con 40 ordenadores conectados; tanto fue el crecimiento de la red que su sistema de comunicación se quedó obsoleto. Entonces dos investigadores crearon el Protocolo TCP/IP, que se convirtió en el estándar de comunicaciones dentro de las redes informáticas.

ARPANET siguió creciendo y abriéndose al mundo, y cualquier persona con fines académicos o de investigación podía tener acceso a la red.

Las funciones militares se desligaron de ARPANET y fueron a parar a MILNET (Red Militar), una nueva red creada por los Estados Unidos. La NSF (National Science Foundation) crea su propia red informática llamada NSFNET, que más tarde absorbe a ARPANET, creando así una gran red con propósitos científicos y académicos.

El desarrollo de las redes fue abismal y se crean nuevas redes de libre acceso que más tarde se unen a NSFNET formando el embrión de lo que hoy se lo conoce como INTERNET. El desarrollo de NSFNET fue tal que hacia el año 1990 ya contaba con alrededor de 100.000 servidores, el gráfico 3.3 muestra uno de los backbones de NSFNET. [LIB006]

Gráfico 3.3



El CERN (Centro Europeo de Investigación de Partículas) crea las Páginas Web, con el objetivo de comunicarse con otros científicos europeos. En 1993 un estudiante norteamericano escribió el código del primer explorador web, el Mosaic que se distribuía de forma gratuita por la red y permitía tener acceso a gráficos y documentos de texto dentro de Internet. Esto supuso una auténtica revolución, y a partir de ese momento Internet no ha parado de crecer. En el año 1996 existían cerca de 90.000 sitios web. [www021] [www022] [www023]

3.1.2. INTERNET EN CIFRAS

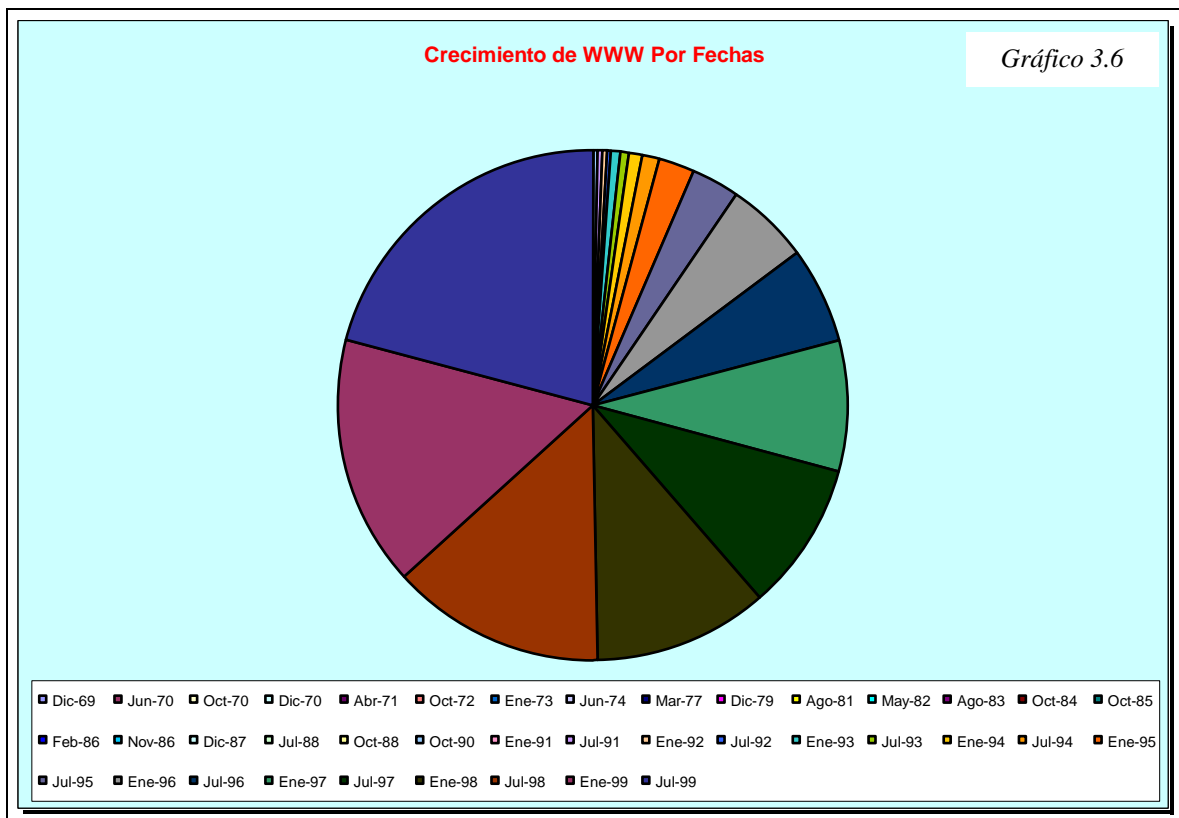
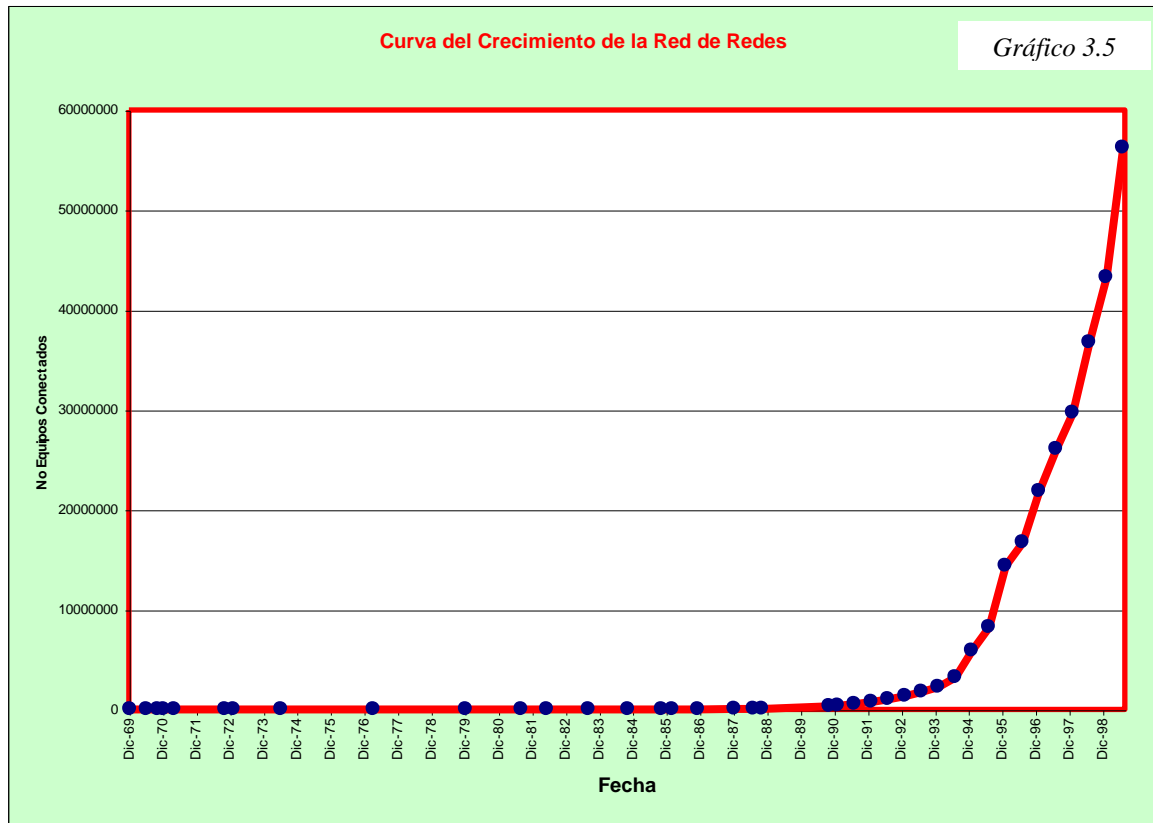
La tabla 3.1 y los gráficos 3.4 – 3.5 muestran el crecimiento de la red www, según el año y los equipos conectados a Internet. [www024] [www025]

Tabla 3.1

FECHA	EQUIPOS CONECTADOS	FECHA	EQUIPOS CONECTADOS
Dic-69	4	Oct-90	313.000
Jun-70	9	Ene-91	376.000
Oct-70	11	Jul-91	535.000
Dic-70	13	Ene-92	727.000
Abr-71	23	Jul-92	992.000
Oct-72	31	Ene-93	1'313.000
Ene-73	35	Jul-93	1'776.000
Jun-74	62	Ene-94	2'217.000
Mar-77	111	Jul-94	3'212.000
Dic-79	188	Ene-95	5'846.000
Ago-81	213	Jul-95	8'200.000
May-82	235	Ene-96	14'352.000
Ago-83	562	Jul-96	16'729.000
Oct-84	1.024	Ene-97	21'819.000
Oct-85	1.961	Jul-97	26'053.000
Feb-86	2.308	Ene-98	29'670.000
Nov-86	5.089	Jul-98	36'739.000
Dic-87	28.174	Ene-99	43'230.000
Jul-88	33.000	Jul-99	56'218.000
Oct-88	56.000		

Equipos Conectados a Internet, Según la Fecha

En estos datos se puede observar el crecimiento desorbitante de www; en 1969 existían 4 computadoras conectadas y en 1999 se conectan más de 56 millones. Es un ejemplo objetivo, claro y que viene de fuentes oficiales de la importancia de las aplicaciones web.



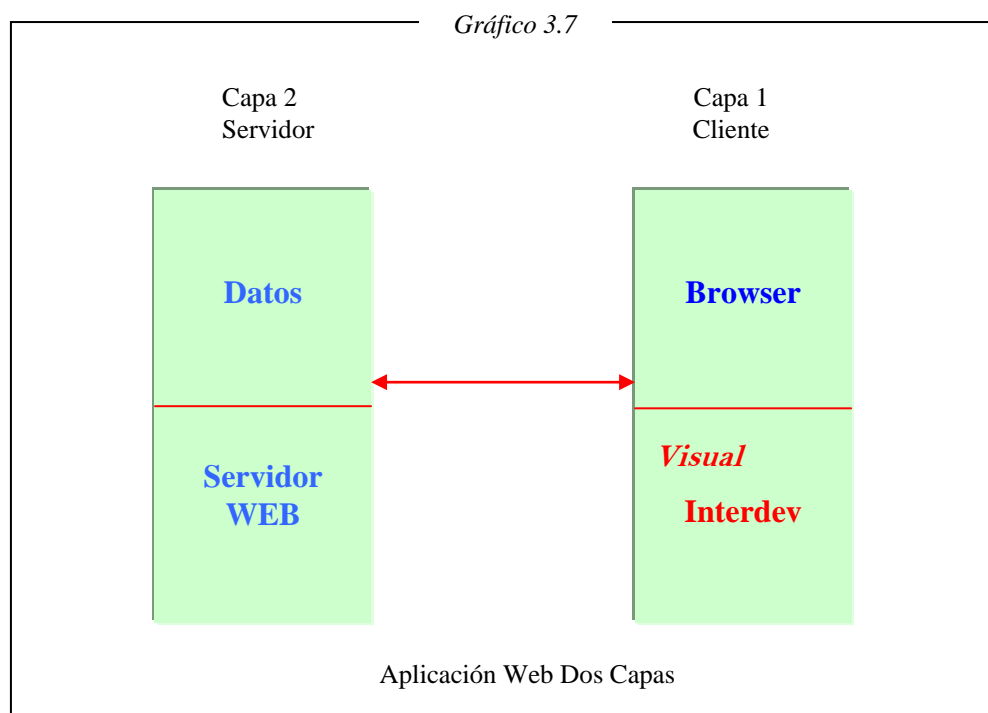
3.2. TRES CAPAS Y APLICACIONES WEB

3.2.1. GENERALIDADES

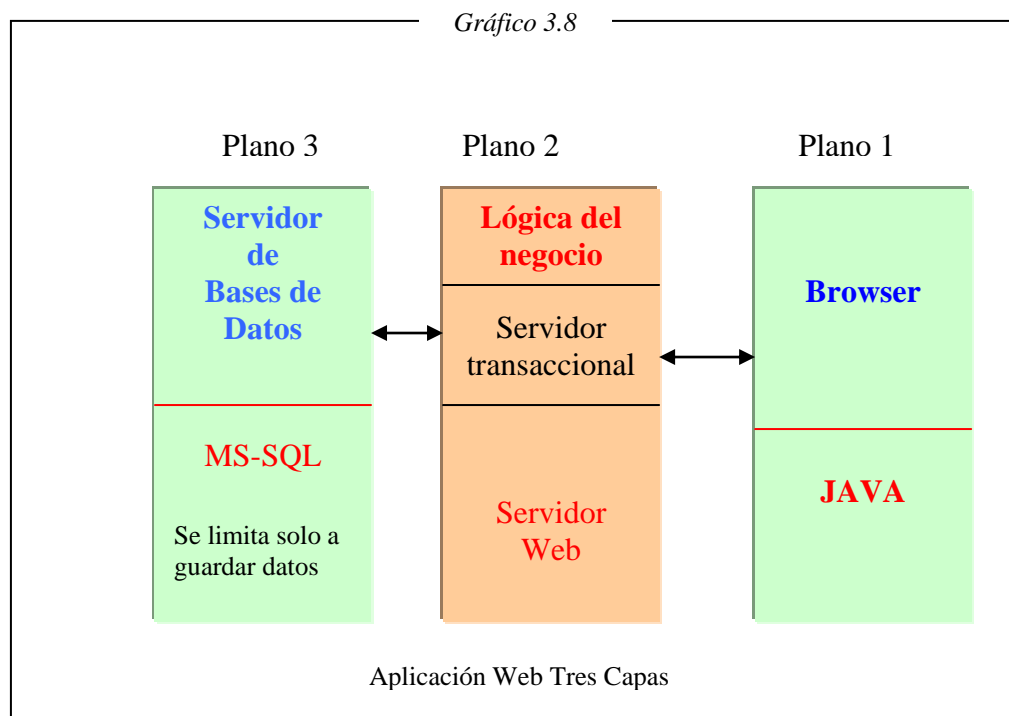
Con el apareamiento de las arquitecturas multicapa y en especial de tres capas, se genera una confusión de tipo técnico, por el enfoque que algunos diseñadores le dieron a las aplicaciones web.

El kit de esta confusión se da porque a los sistemas tres capas se los vincula directamente con los sistemas web. Al aceptar estos enfoques se creería que toda aplicación web es una aplicación tres capas, los textos incluso representan a tres capas en su mayoría con aplicaciones web.

Esto no es así, una aplicación web puede ser: en dos capas o multicapa, si a esta se suman servidores: transaccionales, de correo, noticias, etc. Es de reconocer que tres capas es una de las mejores arquitecturas para este tipo de soluciones, pero también se debe dar cuenta que no es la única arquitectura disponible. Los gráficos 3.7 y 3.8 muestran aplicaciones www en dos y tres capas.



Como se puede observar en el gráfico, las aplicaciones web, si se pueden generar en dos planos, independientemente de las plataformas, bases de datos y browsers utilizados.



La implementación de la aplicación en forma distribuida, consiste en dividir la aplicación en tres niveles lógicos: La capa de interfaz de usuario, solo se dedica a manipular los controles y sus eventos, o sea solo cubre la parte de la aplicación que brinda la interfaz al usuario, manipula los controles y sus eventos, utiliza los objetos creados con las clases de la segunda capa, de esta manera, el sistema ni siquiera se entera que esta accediendo a una base de datos.

La capa donde se encuentran las reglas o la lógica del negocio, encapsula todo el código específico de la aplicación, es decir los procesos particulares del problema que se quiere resolver.

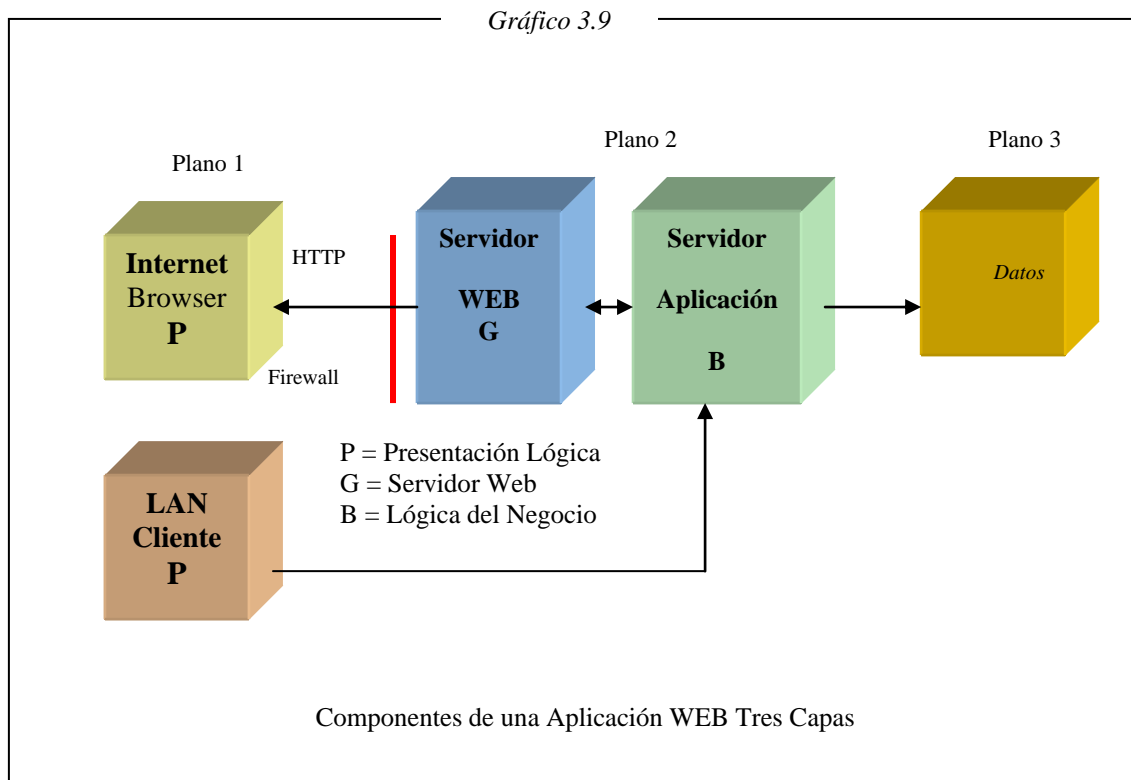
En este segundo plano, se inserta un nuevo componente que es el Servidor WEB que permite publicar y guardar las página en formato de Internet.

La capa de acceso a la base de datos, es genérica para cualquier sistema y consta de clases para establecer conexión con los datos. [www031]

NOTA: “En la mayoría de textos acerca de Aplicaciones WEB Tres Capas, aparece el Servidor WEB en el segundo plano de esta arquitectura, o sea en la parte de la lógica del negocio, pero como el Servicio WEB es otro objeto de distribución de aplicaciones, habría que estudiar si este tipo de arquitectura se le debiera considerar como una aplicación de cuatro capas; donde estén: el servidor de base de datos, el servidor transaccional o lógica del negocio, el servidor web y el cliente”

El Autor.

3.2.2. COMPONENTES DE UNA APLICACIÓN WEB TRES CAPAS



3.2.3. PROGRAMACIÓN DE APLICACIONES WEB.

El programador tal vez se haya preguntado en alguna ocasión, por qué no es posible colocar su ejecutable en el servidor y que los clientes vean las ventanas del programa desde sus ordenadores conectados a Internet.

La respuesta es que la comunicación entre ordenadores no se produce de forma mágica, sino que debe estar basada sobre determinados protocolos. HTTP es el protocolo de la Web, y por tanto en la salida del ejecutable no se deben ser ventanas de Windows sino código HTML.

Tal vez se pregunte por qué usar HTTP en lugar de un protocolo que permita enviar ventanas completas tipo Windows. La respuesta es que los procesos que requieren transmisión por la Red son órdenes de magnitud más lentos que los que se producen en el interior de un ordenador. Es decir, hay que programar de forma que se minimice el tránsito de datos por la Red.

La programación para las redes TCP/IP consiste en que el programa se divide en dos componentes: uno en el ordenador cliente y otro en el ordenador servidor, y si se tiene aplicaciones tres capas, se divide también en el ordenador de lleva la lógica del negocio. La parte cliente se encarga de la interfaz y de la interacción con el usuario, y la parte servidor de obtener y manipular los datos.

La comunicación entre ambos se produce mediante sockets (conexiones TCP) y eventualmente mediante protocolos propios desarrollados para tal efecto.

La programación cliente/ servidor tradicional, como se la explicada en el capítulo anterior, tiene su grado de complejidad, sin embargo con el advenimiento de la World Wide Web se abre una nueva posibilidad: las aplicaciones para la Web. Se trata de una especialización y concreción de las aplicaciones cliente / servidor, donde tanto el cliente o browser, como el servidor (el servidor web), y el protocolo mediante el que se comunican (el HTTP) son estándar, y no han de ser creados por el desarrollador.

La parte del cliente de las aplicaciones web está formada por el código HTML que forma la página web, con opción a código ejecutable mediante los lenguajes de scripting de los navegadores (JavaScript, VB Script, etc) o mediante pequeños programas (applets) en Java.

La parte del servidor está formada por un programa o script que es ejecutado por el servidor web, y cuya salida se envía al navegador del cliente. Tradicionalmente a este programa o script que es ejecutado por el servidor web se le denomina CGI (Common Gateway Interface, Acceso Común a la Interfaz)

Con la entrada en 1995 de Microsoft en el mundo Internet y la salida al mercado de su servidor web (Internet Information Server) se abrió un nuevo campo para las aplicaciones web: el **ISAPI** (Internet Server Application Program Interface, Interfaz de Programación de Aplicaciones del Internet Server, es decir un conjunto de funciones que el servidor web pone a disposición de los desarrolladores de bibliotecas.

Con el ISAPI los desarrolladores pueden crear **DLL** (bibliotecas de funciones) con funciones que son invocadas para determinados archivos. Los filtros ISAPI permiten al desarrollador escribir código que se ejecuta por el servidor web cuando el cliente solicita un archivo con una determinada extensión. Todo el sistema ASP no es más que una DLL del tipo ISAPI, que es invocada automáticamente para los archivos cuya extensión sea **.asp**. La biblioteca ASP preprocesa el archivo ASP interpretando su código como un script a ejecutar en el servidor. Sin embargo ella no interpreta directamente el código, sino que en función del lenguaje en el que está escrito, invoca a otra DLL que se encarga de ejecutar el script. Después recoge la salida y se la envía al servidor web, de igual manera funciona PHP para linux. Las Herramientas CASE actuales como Genexus 7.5 crean sus aplicaciones basándose en módulos *.asp.

3.3. HERRAMIENTAS CASE EN SOLUCIONES WEB

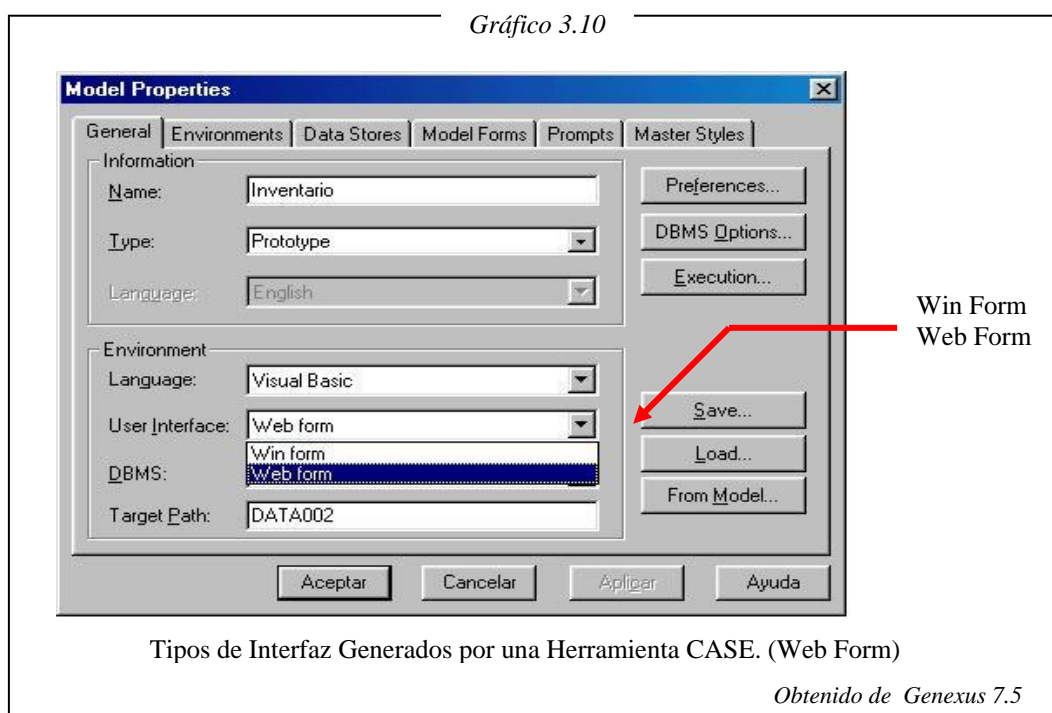
El movimiento hacia las arquitecturas distribuidas nunca ha sido tan fuerte como hoy, esta evolución ha sido impulsada por las aplicaciones web y cimentada por nuevas y poderosas herramientas que ayudan al desarrollo completo de aplicaciones www, como son las Herramientas CASE generadoras de aplicaciones Internet.

El crecimiento vertiginoso de las Herramientas CASE, ha llevado a que estas puedan generar aplicaciones completamente terminadas para ambientes web; con estas herramientas el desarrollador elabora sus sistemas Internet desde un ambiente netamente gráfico de diseño, el cual genera todos los procesos correspondientes al ciclo de vida de una aplicación, tal es el caso de Gexus 7.5.

Por lo general las Herramientas CASE se basan primordialmente en el diseño de la realidad, o sea, en el diseño de las pantallas que desea ver el usuario en Internet, técnicamente se las conoce como: páginas web y hoy en día como transacciones web.

Una vez diseñadas las transacciones web las Herramientas CASE se encarga de crear automáticamente: las páginas web, los hipervínculos, la base de datos, integridad referencial, normalización, prototipos, documentación, etc, y en algunos casos las consultas de dicha transacción, sin necesidad de que el programador tenga que escribir código HTML para cada transacción realizada.

El gráfico 3.10 indica cómo una Herramienta CASE (Genexus 7.5), es capaz de generar aplicaciones web, con el simple cambio de las propiedades de la interfaz de usuario.

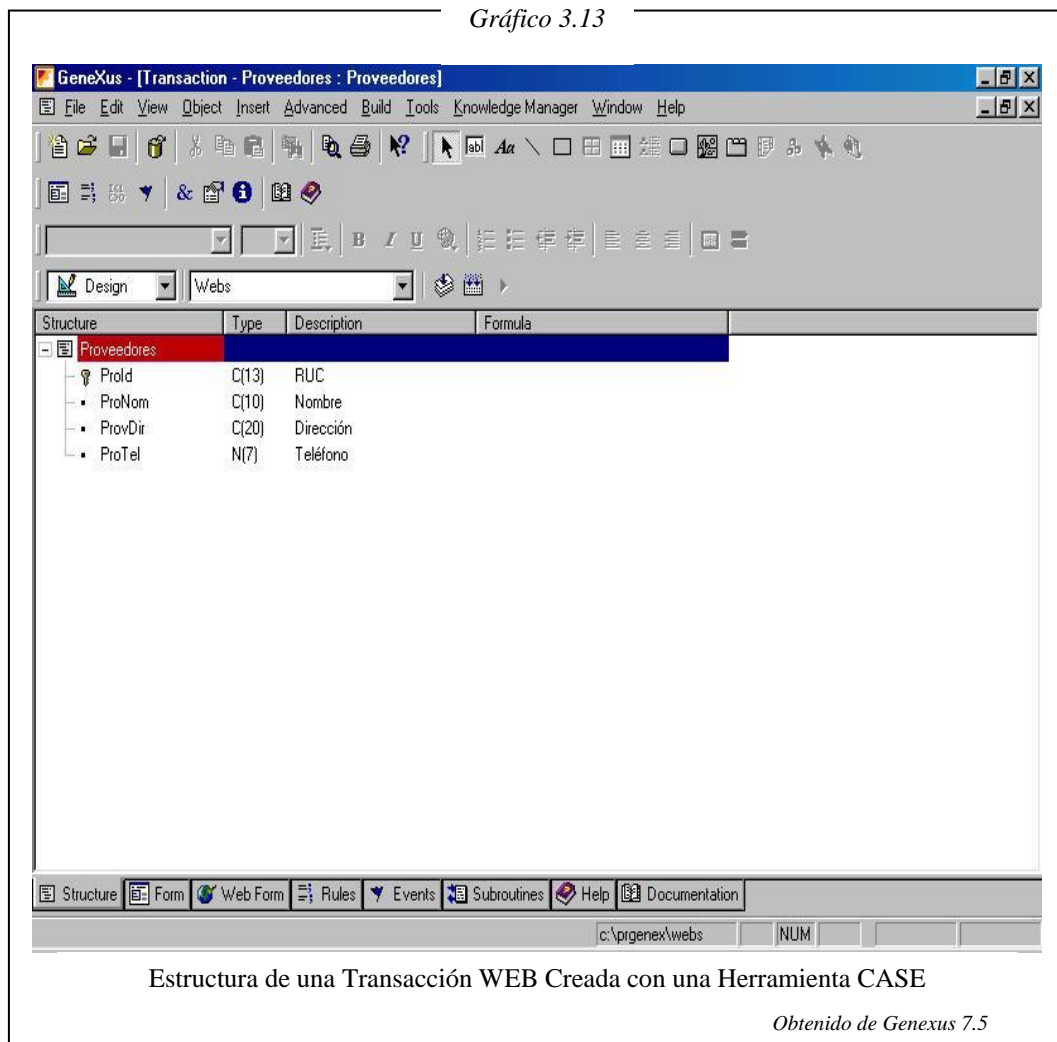


Como se puede observar en este gráfico, la Herramienta CASE le solicita al programador qué tipo de interfaz desea: con ventanas de tipo Windows o Páginas Web y con un este simple cambio de propiedades la herramienta genera soluciones web de la misma forma en la que genera soluciones windows.

Para tener una idea más clara de cómo el una Herramienta CASE, genera aplicaciones web automáticamente, se realizará una transacción web de nombre: **PROVEEDORES** y se valorará su facilidad.

Primeramente se debe realizar la estructura de la transacción (gráfico 3.11), esta tiene: ruc, nombre, dirección y teléfono. Esta estructura será la columna vertebral de la transacción, con esta simple especificación, la Herramienta CASE deberá generar automáticamente: las pantallas, la base de datos, los reportes y en este caso las páginas web.

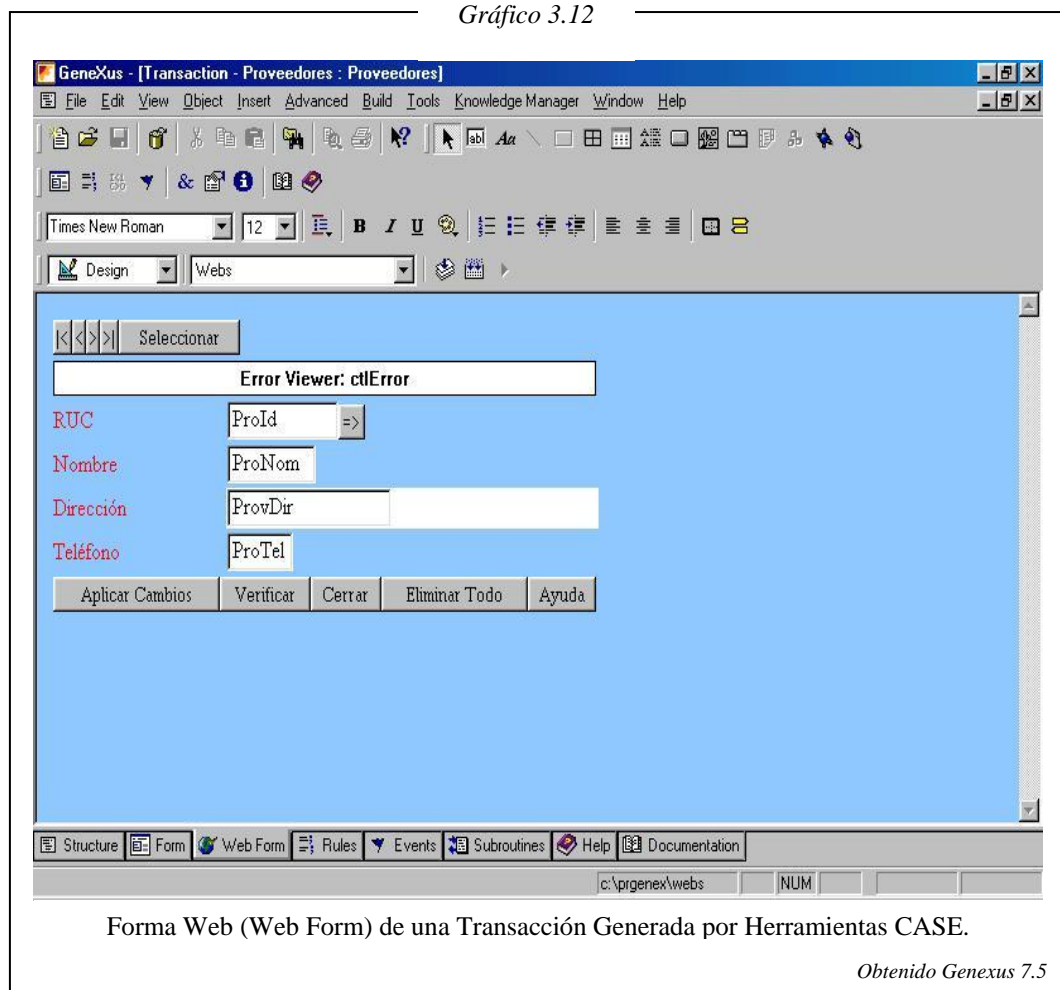
Gráfico 3.13



Al diseñar la estructura de la transacción se está diseñando la realidad, porque en esta parte se ponen todos los datos que el cliente quiere ver en su aplicación, a diferencia de la programación convencional que primero tiene que hacerse un análisis minucioso de los requerimientos mas no del diseño en si, dando como resultado no un diseño prototipo sino un modelo de datos.

El gráfico 3.12 muestra el formato web de la transacción realizada, en esta parte la transacción puede ser modificada a gusto del desarrollador, porque todavía se encuentra en la etapa de diseño y no en prototipo o producción.

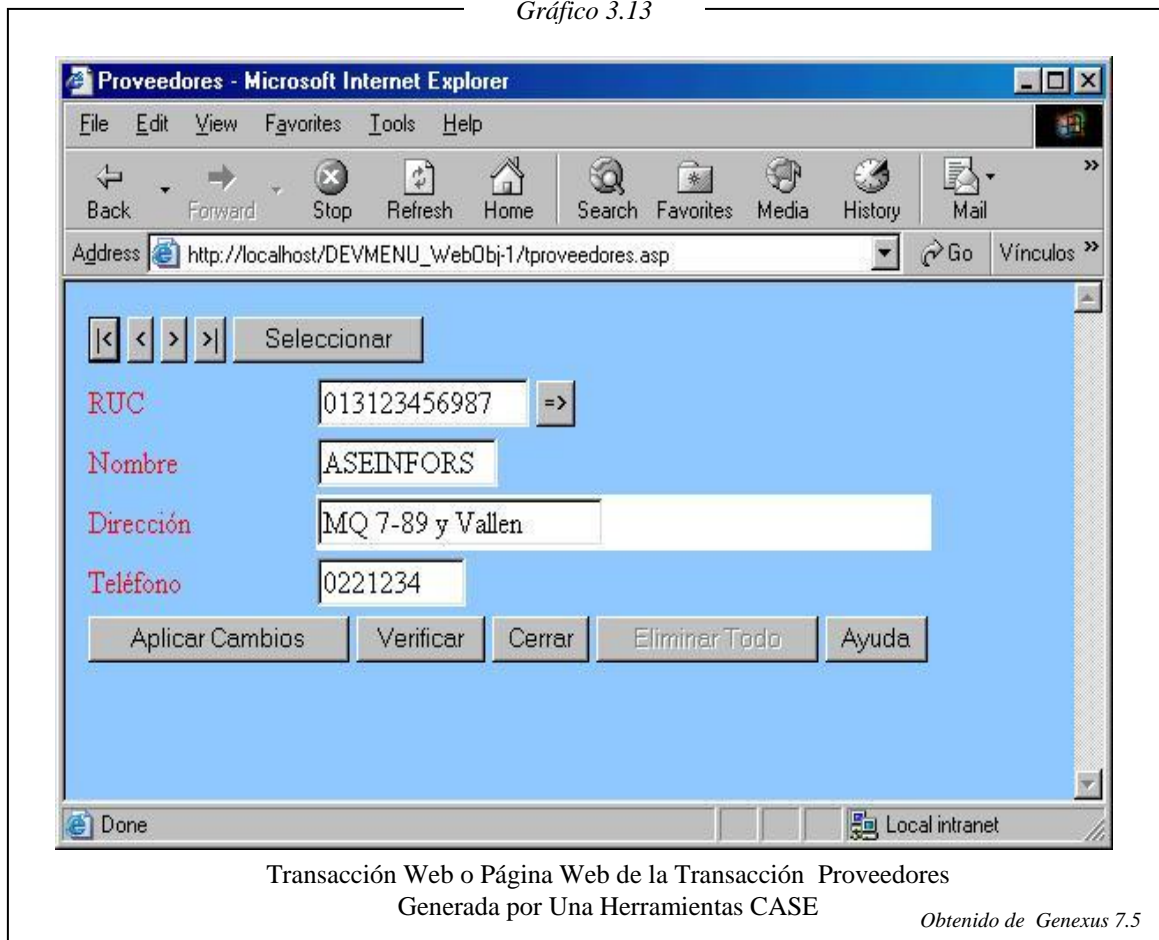
Gráfico 3.12



Cuando el programador ya está decidido a generar su página web con los cambios o especificaciones personales, entonces es la ora de cambiarse del modo de diseño hacia prototipo o producción, se recomienda pasar primeramente a modo prototipo para poder ver a la aplicación en forma real y realizar alguno que otro cambio que no se lo tomo en cuenta.

En estos ambientes: prototipo o producción, la herramienta genera las bases de datos, integridad referencial, diagramas de flujo, etc; aquí ya se puede ejecutar la aplicación; El gráfico 3.13 muestra la página web de la transacción generada por Genexus.

Gráfico 3.13



Los requerimientos de software necesarios para que esta transacción funcione son:

CASE: Genexus 7.5

Lenguaje de Programación: Visual Basic 6.0

Base de Datos: Acces 2000

Sistema Operativo: Windows 98

Servidor Web: Microsoft Personal Web.

A más de la transacción web, la herramienta genera automáticamente algunos menús propios o nativos de la herramienta, como también paneles de trabajo a nivel de consultas, que sirven para seleccionar datos de la transacción, los cuales son llamados por teclas especiales o por viñetas cuando los campos son vinculados a diferentes tablas de la base de datos generada, dicho en otras palabras cuando los campos pertenecen a claves foráneas.

A continuación se mostrará el código html perteneciente a la transacción, cabe recalcar que este código fue creado y generado automáticamente por la Herramienta CASE.

Código HTML Generado por la Herramienta CASE (Genexus 7.5)

```

<html>
<head>
<meta name="Generator" content="GeneXus Visual Basic">
<meta name="Version" content="7_5_2-011">
<meta name="DESCRIPTION" content="Proveedores">
<title>
Proveedores</title>
<link rel="stylesheet" type="text/css" href="styles.css"></head>
<body text="#ff0000" bgcolor=#91C8FF onchange="form_onchange(false, null,'spa');">
<form id="MAINFORM" name="MAINFORM" method="POST" ACTION="tproveedores.asp">

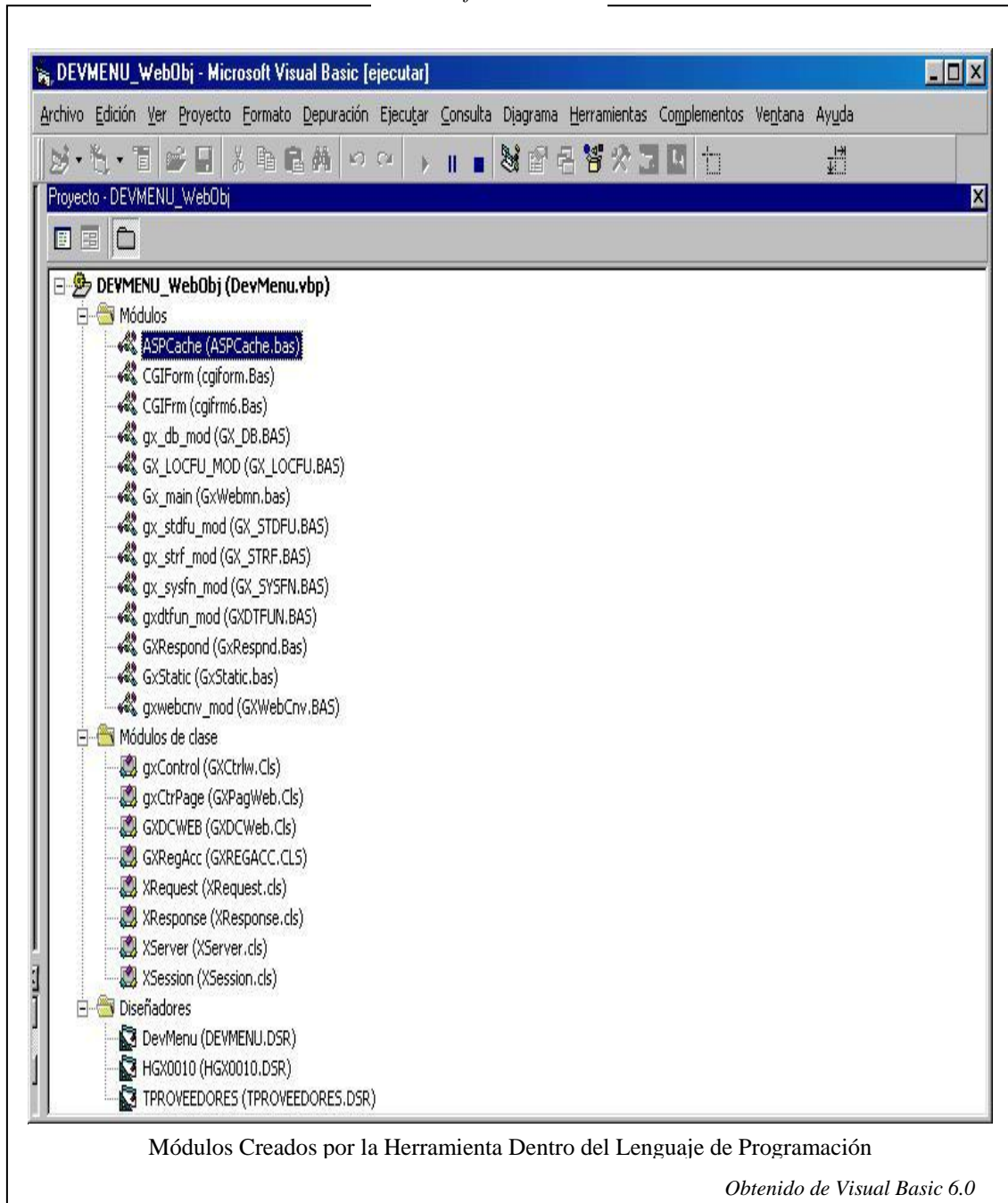
<input type=hidden name="_EventName" value=""><TABLE>
<TBODY>
<TR>
<TR>
<TD colspan=2><input type=SUBMIT name="BTN_FIRST" value="|"
onclick="document.forms[0]._EventName.value='FIRST';" >
<input type=SUBMIT name="BTN_PREVIOUS" value="<"
onclick="document.forms[0]._EventName.value='PREVIOUS';" >
<input type=SUBMIT name="BTN_NEXT" value=">"
onclick="document.forms[0]._EventName.value='NEXT';" >
<input type=SUBMIT name="BTN_LAST" value=">|"
onclick="document.forms[0]._EventName.value='LAST';" >
<input type=BUTTON name="BTN_SELECT" value="Seleccionar"
onclick="HGx0010(document.forms[0].PROID);return false;" ></TD></TR>
<TR>
<TD colspan=2></TD></TR>
<TR>
<TD>RUC</TD>
<TD><input type="text" name="PROID" value="" size="13" maxlength="13" CLASS='S2'
onFocus="this.select()" onchange="form_onchange(false, null,'spa');">
<input type=SUBMIT name="BTN_GET" value="=>"
onclick="document.forms[0]._EventName.value='GET';" ></TD></TR>
<TR>
<TD>Nombre</TD>
<TD><input type="text" name="PRONOM" value="" size="10" maxlength="10" CLASS='S2'
onFocus="this.select()" onchange="form_onchange(false, null,'spa');"></TD></TR>
<TR>
<TD>Dirección</TD>
<TD bordercolor=#c0ffff bgcolor=#ffffff><input type="text" name="PROVDIR" value="" size="20"
maxlength="20" CLASS='S2' onFocus="this.select()" onchange="form_onchange(false,
null,'spa');"></TD></TR>
<TR>
<TD>Teléfono</TD>
<TD><input type="text" name="PROTEL" value="0" size="7" maxlength="7" CLASS='S2'
onFocus="this.select()" onBlur="valid_integer( this)" onchange="form_onchange(false,
null,'spa');"></TD></TR>
<TR>
<TD colspan=2><input type=SUBMIT name="BTN_ENTER" value="Aplicar Cambios"
onclick="document.forms[0]._EventName.value='ENTER';" >
<input type=SUBMIT name="BTN_CHECK" value="Verificar"
onclick="document.forms[0]._EventName.value='CHECK';" >
<input type=SUBMIT name="BTN_CANCEL" value="Cerrar" onclick="GX_js_close();return false;"
>
<input type=SUBMIT name="BTN_DELETE" value="Eliminar Todo" DISABLED >
<input type=BUTTON name="BTN_HELP" value="Ayuda" onclick="GX_help(",
'HLP_TProveedores.htm');return false;" ></TD></TR></TBODY></TABLE>
<input type=hidden name="ZB1004ProId" value="">

```

```
<input type=hidden name="ZB2001ProNo" value="">
<input type=hidden name="ZB3002ProvD" value="">
<input type=hidden name="ZB4003ProTe" value="0">
<input type=hidden name="IsConfirmed" value="0">
<input type=hidden name="Mode" value="INS">
<input type=hidden name="sCallerURL" value="http://localhost/DEVMENU_WebObj-1/DevMenu.ASP"></form>
<script language="JavaScript" type="text/javascript">
function valid_integer( Elem)
{
  var gx_IntRegExp = /^[ ]*([+]?[0-9]+)?[ ]*$/;
  if (! gx_IntRegExp.test( Elem.value))
    { alert("El valor no representa un número correcto."); Elem.focus();}
}
var GXPARAMETERS = new Array();
var wHGx0010=null;
var GXISGET=false;
if (document.MAINFORM.PROID != null && document.MAINFORM.PROID.type != "hidden")
document.MAINFORM.PROID.focus();
function HGx0010(P0,isMod)
{
  GXPARAMETERS[0] = P0;
  if (wHGx0010== null || wHGx0010.closed)
  {
    GXISGET = true;
    sParms = "?";
    sParms = sParms + escape( typeof(P0) == 'string'?P0:P0.value);
    wHGx0010=open("http://localhost/DEVMENU_WebObj-1/hgx0010.asp" + sParms, "",
'toolbar=no,location=no,directories=no,status=yes,menubar=no,scrollbars=yes,resizable=yes,copyhistor
y=no');form_onchange(true, null,'spa');
  }
  wHGx0010.focus();
  if (isMod != null) isMod.value = 1;
}
function GX_js_close()
{
  if (document.forms[0].sCallerURL.value != "")
  {
    location.assign(document.forms[0].sCallerURL.value);
  }
  else
  {
    self.close();
  }
}
function ClearNonKeyFields()
{
  var obj = document.forms[0];
  obj.PRONOM.value = "";
  obj.PROVDIR.value = "";
  obj.PROTEL.value = "0";
}
</script>
<script language="JavaScript" src="form_onchange.js"></script>
<script language="JavaScript" src="std_messages.js"></script>
<script language="JavaScript" src="gx_help.js"></script>
</body>
</html>
```

El gráfico 3.14 indica los módulos creados por la Herramienta CASE, dentro del lenguaje de programación (Visual Basic 6.0)

Gráfico 3.14

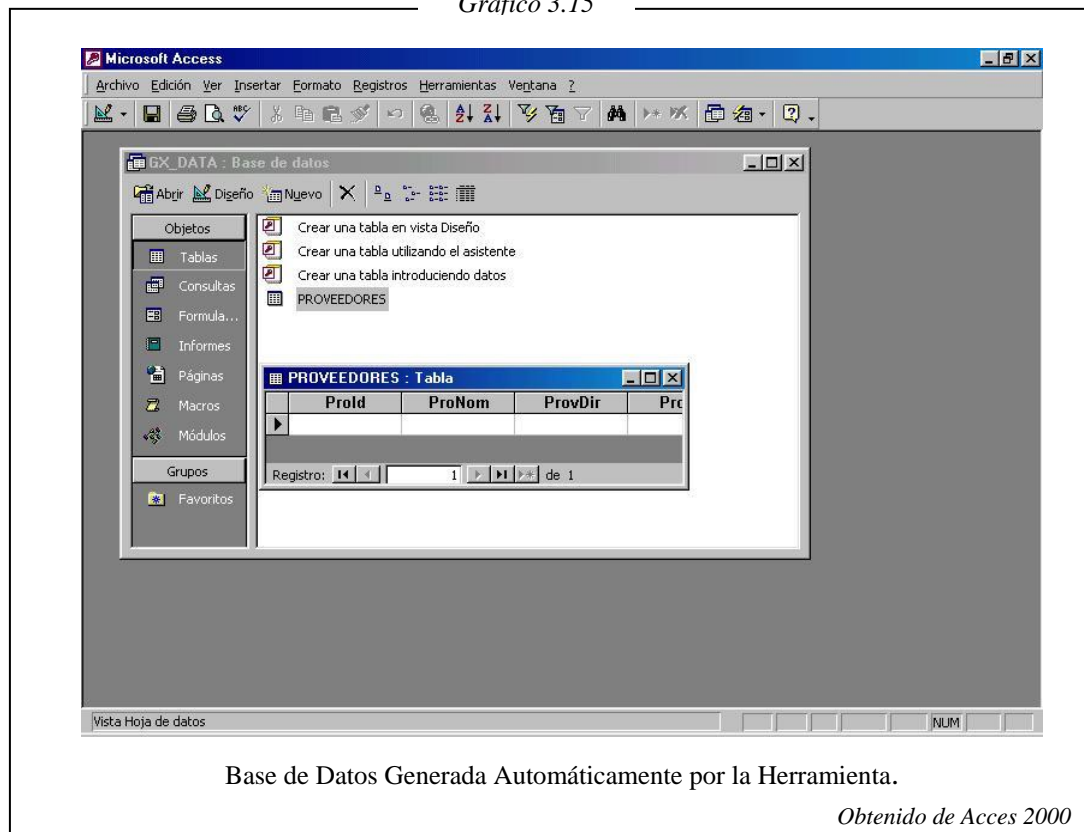


Como se puede dar cuenta esta Herramienta CASE *workbench*, genera automáticamente todo el ciclo de vida de la aplicación, incluso genera a través de módulos ASP.

El gráfico 3.15 muestra a la base de datos: creada, corregida y normalizada en tercera forma normal, de una forma automática desde el diseño de la transacción.

De esta manera el desarrollador ya no se centra a realizar los procesos estructurales automatizables, empezando desde el diseño mismo de la base de datos, pruebas de normalización, integridad referencial, etc sino que deja que la herramienta se encargue de eso, y el se centra netamente en el diseño de la aplicación.

Gráfico 3.15



Ahora bien, una vez realizado un ejemplo de cómo una Herramienta CASE genera soluciones web simplemente especificando el diseño, se puede dar cuenta de qué tan poderosas son estas herramientas y de cuánto le ayudan al desarrollador a realizar sus aplicaciones.

Lo importante de las Herramientas CASE para el web y para todo tipo de aplicación, es que siguen una estricta cultura de programación, generan los códigos, basados en estándares de programación que en la programación común algunas veces se los pasa desapercibidos.

La mayoría de Herramientas CASE para el web, generan código para JAVA, C# y Visual Interdeb, que son los lenguajes más comunes del web. Java y C# son las mejores opciones para las aplicaciones Internet ya que tienen mejores propiedades para el desarrollo de soluciones utilizando componentes.

3.4. ESTRUCTURA DE LAS HERRAMIENTAS CASE PARA EL WEB

Para empezar a analizar la estructura de las Herramientas CASE generadores de aplicaciones distribuidas, se tomará como patrón a JAVA, ya que es el lenguaje de programación pionero y más versátil para la realización de aplicaciones distribuidas.

JAVA es el estándar de la mayoría de plataformas de software, sin menospreciar a otras herramientas como la tecnología .NET de Microsoft y otras herramientas existentes en el mercado. Este estudio se va a realizar tomando como referencia a JAVA interactuando con la Herramienta CASE Genexus 7.5.

3.4.1. GENERALIDADES DE JAVA

JAVA es un lenguaje creado por Sun Microsystems que está enfocado al desarrollo de aplicaciones web. Desde el punto de vista de los usuarios de las Herramienta CASE sus características mas importantes son las siguientes:

INTERPRETADO. El código Java se escribe en fuentes con extensión *.java y se compila a clases con extensión *.class. En tiempo de ejecución se interpretan y ejecutan los *.class.

PORTABLE. Para ejecutar una aplicación java es necesario tener una “máquina virtual” que interprete y ejecute el código java. Estas máquinas virtuales son las que traducen las instrucciones de los *.class a las instrucciones nativas del sistema operativo. Existen máquinas virtuales para prácticamente todos los sistemas operativos.

Es posible realizar llamadas a código nativo de cada plataforma, lo que permite aprovechar las ventajas específicas de cada sistema operativo, pero le quita portabilidad.

ORIENTADO A INTERNET. Las aplicaciones Java pueden ejecutarse de modo que el código se va descargando de un servidor Web a medida que es necesario ejecutarlo.

Esto tiene como ventaja que no se descarga mas código que el necesario, y que la distribución de las aplicaciones deja de ser un problema, ya que cambiando la aplicación en el servidor, se actualizan para todos los clientes de forma automática.

Como contrapartida, cada *.class se transfiere en una operación de descarga independiente (en un requerimiento HTTP distinto), lo que implica una carga agregada importante. Para evitar esto se pueden instalar automáticamente en el cliente las clases necesarias para ejecutar la aplicación.

Por otro lado, java, tiene en sus bibliotecas estándar, clases para utilizar TCP-IP de una forma sencilla, lo que simplifica la programación de aplicaciones que utilicen Internet como medio de comunicación, y clases para permitir la ejecución distribuida de las aplicaciones.

SEGURO. Las aplicaciones java, que se descargan de Internet, se ejecutan en una “caja” que no les permite realizar diversas operaciones, como por ejemplo leer y escribir en el disco duro del usuario. Esto da a los usuarios de aplicaciones java la tranquilidad de que su máquina no corre ningún peligro

3.4.2. EL GENERADOR JAVA CON RESPECTO A GENEXUS

El generador JAVA, solo generará aplicaciones Cliente / Servidor, no habrá una versión que genere programas con acceso a base de datos local como Access o DBFs.

Las aplicaciones pueden generarse en 2 capas utilizando JDBC (el equivalente java a ODBC) o en 3 capas utilizando CORBA, DCOM o RMI como protocolo de comunicación entre las capas.

El generador java se apoya en las GXDB++ de Genexus, para realizar el acceso a la base de datos. Las GXDB++ son un conjunto de clases java, generadas en tiempo de creación y reorganización, que encapsulan los accesos a la base de datos y luego son utilizadas por los programas generados.

Las aplicaciones java, pueden ejecutarse de dos formas. Como “applets” o como “aplicaciones”. En el primer caso se ejecutan dentro de un web browser, los programas compilados java residen en el web server y son bajados al cliente cuando este los necesita ejecutar, o bien se pueden instalar en el cliente utilizando un utilitario que se provee con el generador llamado Deployment Wizard. En el segundo caso, se ejecuta como una aplicación independiente, y los programas compilados están en el disco del cliente, o de algún archivo servidor al que pueda acceder. [www010] [www013]

Desde la versión 7.5 de Genexus se puede utilizar la tecnología *.net con su herramienta ADO.net que es la metodología de punta desarrollada por Microsoft.

3.4.3. ARQUITECTURA DE LAS HERRAMIENTAS CASE EN APLICACIONES DISTRIBUIDAS

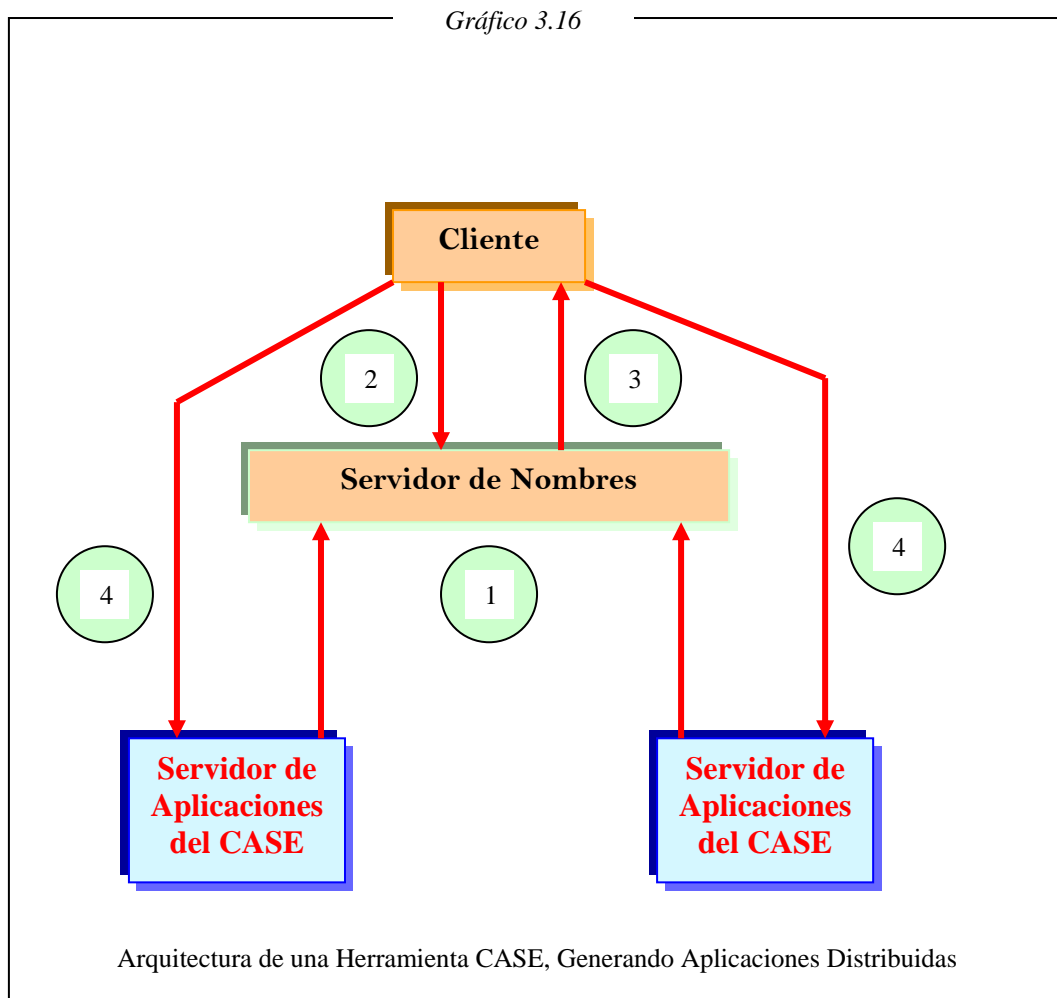
Las aplicaciones generadas con Herramientas CASE pueden ejecutarse en un esquema de múltiples servidores. Entre todos los servidores hay uno que tiene que asumir el rol de 'Servidor de Nombres de Dominios' (DNS).

Cada vez que un servidor de aplicaciones se levanta, se conecta con el servidor de nombres, el cual informa que está levantado, y le dice cual es su nombre. El nombre del servidor de aplicaciones es el que se especifica en el Location.

Cuando un cliente necesita conectarse con un servidor de aplicaciones que está en un Location dado, le pide al servidor de nombres una referencia a dicho servidor de aplicaciones, y luego se comunica directamente con éste.

Esta arquitectura permite que en el diseño de la aplicación la única dirección IP física sea la del servidor de nombres. Solo cambiando esta propiedad puede modificarse el ambiente en que ejecuta la aplicación.

A continuación se esquematiza el funcionamiento de una aplicación distribuida. Cada recuadro puede ejecutarse en un host distinto, o en el mismo. (gráfico 3.16)



Se levanta el servidor de nombres en un host.

Se levantan los servidores de aplicaciones de la Herramienta CASE, en este caso el de GENEXUS, en los hosts donde se vayan a ejecutar aplicaciones remotas. El servidor de nombres puede estar en el mismo host que un servidor de aplicaciones de la herramienta, pero solo debe ejecutarse un servidor de nombres.

Los servidores de aplicaciones de la Herramienta CASE, se registran con el servidor de nombres. Esto es lo que se indica en el nivel (1).

El cliente se ejecuta, y por cada Location que vaya necesitando acceder, le pedirá al servidor de nombres una referencia al servidor de aplicaciones de la Herramienta CASE correspondiente. Esto es lo que indica el número (2). El número (3) indica la referencia al servidor de aplicaciones que devuelve el servidor de nombres.

El cliente le solicita al servidor de aplicaciones referencias a los objetos que debe ejecutar remotamente, y los ejecuta. Esto es lo que indica el número (4).

3.5. UTILIZACIÓN DE COMPONENTES CON GENEXUS

3.5.1. UTILIZACIÓN DE CORBA

CORBA: Common Object Request Broker Architecture, es una arquitectura que permite la ejecución distribuida de aplicaciones.

Es un estándar definido por una institución llamada Object Management Group, que está compuesta por varios fabricantes de software. Es una arquitectura multiplataforma, que permite la integración de módulos escritos en diferentes lenguajes.

Existen muchos productos en el mercado que permiten la utilización de CORBA con java. Actualmente se ha probado el código generado utilizando dos productos:

VisiBroker for java 3.4 o posterior, de Inprise Corp.

JDK 1.2.x o posterior, de Sun Microsystems

Para levantar los procesos necesarios en el servidor, se provee un archivo *.bat que lo hace para las dos implementaciones de CORBA soportadas. El bat se llama ORBSRV.BAT, se le debe pasar como primer parámetro "vb" si se quiere usar VisiBroker o "jdk12" si se quiere utilizar el JDK 1.2 y como segundo parámetro el nombre del archivo de configuración (server.cfg) que se desea utilizar. El orbsrv.bat contiene al principio algunos SETs que se deben modificar para reflejar el lugar donde están instalados los productos CORBA:

En el caso de VisiBroker, el bat levanta 3 procesos. En primer lugar un ejecutable llamado OSAGENT, que debe ejecutarse en alguno de los servidores de aplicaciones o de nombres. En segundo lugar, el servidor de nombres, y en tercer lugar el servidor de aplicaciones de la Herramienta CASE.

Para el JDK 1.2/1.3 no es necesario el primer ejecutable, por lo que solo se levanta el servidor de nombres (tnameserv.exe del subdirectorio BIN el directorio donde se instaló el JDK) y el servidor de aplicaciones la herramienta.

Para levantar el servidor de aplicaciones GeneXus se utiliza otro .bat llamado gxappser.bat, que debe estar en el mismo directorio que el orbsrv.bat

Si se quiere levantar manualmente sin utilizar los .BAT, para ejecutar con el JDK 1.3, se deben ejecutar los siguientes comandos en el servidor y luego ejecutar el cliente:

```
<path to java>\bin\tnameserv.exe  
<path to java>\bin\java.com.genexus.ApplicationServer <server.cfg>
```

Es necesario tener en el classpath del servidor las clases GeneXus (gxclassr.zip) y los drivers JDBC que se deseen utilizar.

Classpath. En caso de utilizar VisiBroker hay que agregar al Classpath de las preferencias de ejecución los archivos. vbjapp.jar, vbjcosnm.jar y vbjorb.jar, que se encuentran en el subdirectorio LIB del directorio en donde se instaló el producto.

3.5.2 UTILIZACION DE RMI.

Para utilizar RMI no es necesario instalar ningún producto adicional, dado que está incluido en las clases java estándar. Si se quiere ejecutar con el SDK de Microsoft, es necesario utilizar el RMI de Sun, dado que Microsoft no distribuye RMI con su SDK.

Un detalle muy importante, es que para que el RMI funcione correctamente el cliente debe ser capaz de resolver el nombre del servidor a partir de su IP y viceversa, o sea, si se pone en la preference Name Server = 192.168.0.1, en el cliente debe poder resolverse que esa dirección corresponde al servidor, por ejemplo, "myserver.com". Esta resolución de nombres la debería resolver el DNS, pero si no fuera así hay que agregarlo en el archivo HOSTS del directorio \WINDOWS en Windows 95/98 o en del directorio \WINNT\SYSTEM32\DRIVERS\ETC en Windows NT.

Para verificar que los nombres del DNS estén correctos se pueden hacer pruebas con el comando 'ping', así:

```
ping <IP del Servidor>  
ping <Nombre del Servidor>  
ping -a <IP del servidor>
```

Con estos comandos, se deben devolver valores consistentes o sea siempre debe mostrar el mismo 'nombre' de servidor.

Con SDK de Microsoft no se distribuyen las clases necesarias para ejecutar aplicaciones con RMI. Para obtener las clases que soportan RMI con la maquina virtual de Microsoft, existen dos alternativas, una que provee Microsoft a manera de librerías y otra de IBM.

Ninguna de estas dos alternativas permite instalar RMI en Internet Explorer de forma automática, requieren de una instalación manual. Si se desea que las aplicaciones en Internet usen RMI y lo hagan con Internet Explorer hay que crear un archivo .CAB con una firma digital, con todos los permisos de ejecución dentro del Internet Explorer. Este archivo puede ser instalado en el IE.

3.5.3. UTILIZACIÓN DE DCOM

DCOM solo se puede utilizar con la máquina virtual de Microsoft. En el subdirectorio GXJAVA del directorio donde está instalado GeneXus se distribuye una DLL llamada "GXDCOMJ.DLL". Esa DLL se registra automáticamente cuando se instalan las clases de GeneXus al ejecutar la aplicación como Applet en el Internet Explorer.

Si se quiere ejecutar como Application, debe registrarse manualmente con el comando REGSVR32 gxdcomj.dll tanto en el cliente como en el servidor. Adicionalmente, es necesario ejecutar el utilitario en el servidor NT "DCOMCNFG" y en la sección "Default Properties" poner "Default Authentication Level = None".

También, cuando se instalan automáticamente las clases de la herramienta, al ejecutar como applet, se modifica un seteo del registro del cliente que cambia el 'Default Autenticación Level' a 'None'.

Si es la primera vez que se instalan las clases de la herramienta, para que el cambio tenga efecto es necesario que el cliente reinicie su equipo, de otro modo no podrá conectarse al servidor de aplicaciones, salvo que esté conectado al mismo dominio, o que exista en el dominio del servidor una combinación usuario / password idéntica a la del cliente. Esto es, si el cliente está conectado a un dominio NT1, y su usuario / password son Genexus / Genexus, debe existir en el dominio del servidor un usuario Genexus con password Genexus. [www010] [www013]

3.6. PRINCIPALES OBJETOS GENERADOS POR HERRAMIENTAS CASE EN APLICACIONES WEB

Los Objetos Web se utilizan para desarrollar aplicaciones para Internet desde una Herramienta CASE; el producto final es generar código HTML, el código que los navegadores interpretan pudiendo interactuar con la base de datos, permitiendo de esta forma la generación de páginas web dinámicas y estáticas.

Algunos de los objetos web generados por Herramientas CASE son. transacciones WEB, paneles de trabajo WEB, en algunos casos reportes web, menús web, etc.

3.5.1. TRANSACCIONES WEB

Las Transacciones Web no son un nuevo tipo de objeto, sino una forma de interfaz más para las transacciones que permiten su ejecución en navegadores.

Las Transacciones Web facilitan el diseño de aplicaciones Web porque permiten resolver el ingreso de datos realizando automáticamente todos los controles de integridad referencial necesarios. Para ejecutarlas, sólo se requiere un navegador instalado en el cliente.

3.5.2. WEB PANELS

Se puede decir que los objetos web panel y work panel son similares ya que ambos permiten definir consultas interactivas a la base de datos.

Son objetos muy flexibles que se prestan para múltiples usos, cuya programación está dirigida por eventos. De todos modos, hay algunas diferencias entre ellos, causadas principalmente por el esquema de trabajo en Internet. Una particularidad fundamental que diferencia a estos dos objetos es que en los web panels, en tiempo de ejecución, el resultado de la consulta se formatea en HTML para presentarse en un navegador.

Existe un API especificada por Sun Microsystems llamada 'Java Servlet API' que define la forma para ejecutar lo equivalente a los web panels de GeneXus. Esta API define una arquitectura moderna y eficiente, que presenta algunas ventajas importantes con respecto a la ejecución con CGI que se utiliza en los web panels generados con Visual Basic, RPG o C/SQL. Las ventajas son básicamente las siguientes:

- Con CGI cada vez que se ejecuta un web panel se levanta un proceso en el servidor, se ejecuta, y se baja el proceso. Esto impone una carga considerable al servidor, en particular a aquellos en los que levantar un proceso tiene un costo alto. Los Servlets Java se cargan la primera vez que se ejecutan, y luego no se vuelven a cargar.
- Los web panels con CGI deben conectarse a la base de datos cada vez que se ejecutan. Los Servlets pueden mantener conexiones abiertas y utilizarlas a medida que las van necesitando
- Los Servlets ejecutan dentro de motores de Servlets que permiten administrar eficientemente el manejo de los recursos del servidor.
- Se pueden ejecutar en cualquier sistema operativo para el que haya disponible una máquina virtual Java y un servidor web que permita ejecutar Servlets (NT, Win95, AS/400, AIX, Linux, Macintosh, etc.).
- El soporte de Java para Servlets es bastante sofisticado, lo que hace sencillo agregar nueva funcionalidad a los web panels, como manejo de sesiones, compresión de páginas HTML en tiempo real, etc.

3.5.3. SMART STATIC PANELS.

El objetivo es posibilitar la generación inteligente de páginas estáticas desde la Herramienta CASE.

Como SSP se entiende a la ejecución de un web panel dinámico para una instancia de los datos que recibe como parámetro, es decir archivos de texto conteniendo HTML con la información obtenida a partir de la base de datos.

Estos SSP pueden convivir con los web panels dinámicos, tanto pudiendo llamarlos como pudiendo ser llamados. Esto será muy útil en sitios de información (portales) donde gran parte de la información, después de generada será estática. Los generadores Java y C/SQL permiten la generación de este tipo de páginas.

La finalidad es permitir desde la herramienta la generación de páginas estáticas de una forma inteligente, es decir consultando dinámicamente a la base de datos, ya que tienen las siguientes ventajas.

- Se comportan con mejor performance que los dinámicos
- Recargan menos al servidor en tiempo de ejecución.

Es recomendable utilizarlos en los siguientes casos: Si se tiene páginas cuyo contenido varía en la base de datos con una periodicidad conocida. Si la aplicación puede soportar mostrar alguna información no on-line sino que se actualice hacia el web con cierta periodicidad.

Cuando el SSP se invoca desde el browser se comporta como un web panel normal, resolviendo automáticamente para cada link si el web panel llamado es estático o dinámico.

3.5.4. COMPRESIÓN DE PÁGINAS WEB.

Los web panels en Java pueden enviar la página HTML comprimida, de modo que sea descomprimida en tiempo real por el navegador. La compresión se realiza solo si el navegador indica que es capaz de descomprimirlo.

Esta opción puede deshabilitarse con la preference 'Auto compress web pages', dado que es posible que algunos navegadores no reporten correctamente su capacidad para descomprimir las páginas, y no puedan desplegar correctamente a las mismas. [LIB009]

CAPÍTULO IV

GENEXUS DESARROLLO DE APLICACIONES



- ❖ INTRODUCCIÓN A GENEXUS.
- ❖ DEFICIÓN DE REQUERIMIENTOS, PARA SOLUCIONES GENEXUS.
- ❖ DISEÑO DE TRANSACCIONES.
- ❖ DISEÑO DE REPORTE.
- ❖ DISEÑO DE PROCEDIMIENTOS.
- ❖ DISEÑO DE PANELES DE TRABAJO.
- ❖ GRAFICACIÓN DE DATOS.
- ❖ DISEÑO DE MENÚ.
- ❖ DISEÑO Y PUBLICACIÓN DE OBJETOS WEB.

4.1. INTRODUCCIÓN A GENEXUS

GENEXUS es una Herramienta CASE que cubre todo el ciclo de vida en el desarrollo de soluciones informáticas, en otras palabras esta dentro de las herramientas tipo WORKBENCH.

La desarrolló un grupo de investigadores Uruguayos, y está patentado por la empresa Uruguaya ARTECH, su distribuidor en Ecuador es GMS (Grupo Microsistemas).

El objetivo de Genexus, es ayudar a los desarrolladores a elaborar sus aplicaciones en el menor tiempo y con la mejor calidad posible, omitiendo ciertos pasos que son automatizables, y concentrándose en las partes verdaderamente complejas como son el análisis y diseño.

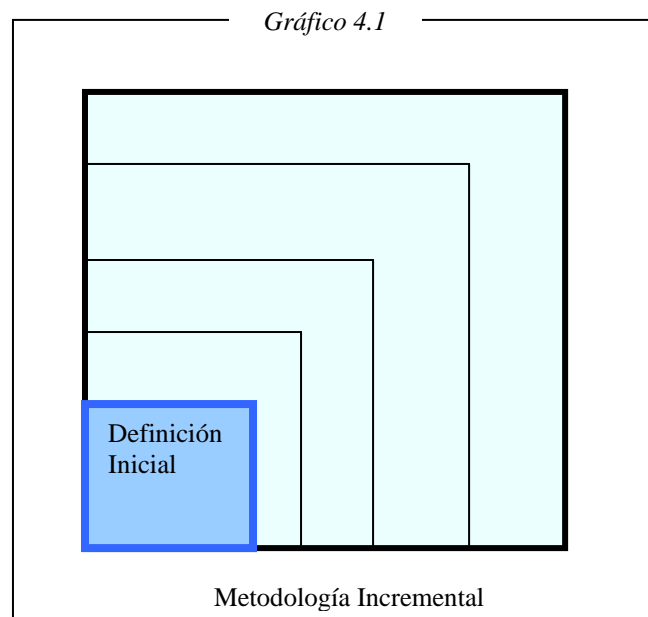
Desde el punto de vista teórico, Genexus es más una metodología para desarrollar aplicaciones que una herramienta de software. Está relacionado con las metodologías tradicionales, pero aporta grandes y novedosos cambios a las mismas.

Al comenzar las aplicaciones por el desarrollo de la interfase de usuario, revoluciona el mundo de la metodología tradicional, ya que el programador no tiene que preocuparse por la implementación física del sistema como: creación y normalización de las bases de datos, generación de programas, generación de diagramas, generación de documentación, etc.

El punto de partida de Genexus, es “*DESCRIBIR LAS VISIONES DE LOS USUARIOS O DISEÑO DE LA REALIDAD*”; a partir de este modelo mediante una “síntesis de visiones” se construye el soporte computacional que es el eje de la aplicación

En la mayoría de metodologías tradicionales como: análisis estructurado, ingeniería de la información, análisis esencial, etc. Se separan el análisis de datos, de los procesos y programas; Genexus hace un análisis conceptual unificado que se lo conoce como “metodología incremental”.

Cuando se comenzaron a utilizar verdaderos modelos corporativos que normalmente poseen cientos de tablas, se confirmó que no debía perderse más tiempo buscando algo que no existe: *las bases de datos estables*. Luego de varias investigaciones, se desarrollo una teoría la cual fue crear una herramienta que pudiera posibilita el desarrollo incremental y permitir la convivencia con las bases de datos reales, el gráfico 4.1 explica el crecimiento del modelo incremental.



La construcción automática de la base de datos y programas ejecutables a partir de una única especificación, permitirá a Genexus utilizar la metodología incremental; este proceso se realiza mediante “*aproximaciones sucesivas*”.

En cada momento se desarrolla la aplicación con el conocimiento que se tiene, y luego cuando se pasa a tener: más, menos o nuevo conocimiento; Genexus se ocupará de hacer automáticamente todas las nuevas adaptaciones en las bases de datos y programas.

Utilizando Genexus la tarea básica del analista es la “*descripción de la realidad*”. Solo el hombre podría realizar esta tarea, ya que solo él puede entender los problemas del usuario. Desde luego que esto modifica la actividad del analista e incluso su perfil supuestamente óptimo ya que lo transforma en un *Business Analyst* o analista del negocio.

Ahora trabaja en alto nivel, discutiendo problemas con el usuario y probando con el las especificaciones a nivel de prototipo, en vez de desarrollar su actividad en tareas de bajo nivel como: diseñar archivos, normalizar, diseñar programas, programar, buscar y eliminar errores de programación, etc.

Genexus es una herramienta que genera código para: Java, C#, Cobol, Visual Basic, Fox, Visual Foxpro, RPG, C, Visual C++, Visual Basic .net, entre otras; y para las bases de datos: Oracle, Informix, Acces, UDB, DB/2 y MS-SQL Server. [LIB001]

4.1.1. ¿CÓMO CONSEGUIR GENEXUS?

Genexus es una herramienta de carácter comercial, por esta razón tiene su costo que relativamente es considerable en las diferentes versiones individuales o corporativas. Estas versiones se las puede comprar por medio de cualquier sitio autorizado por Artech, en Ecuador es GMS. La versión comercial más actual es: GENEXUS 7.5 además se está probando una versión OLIMAR BETA3, que es la antesala de Genexus 8.0 que genera aplicaciones netamente para el web.

Anteriormente existía la versión RELEASE de Genexus 7.0; era una versión de tipo académico que se distribuía a los centros de estudios superiores, por medio de convenios vi direccionales con Artech. Esta versión era muy útil ya que no existían restricciones de generación, se podía desarrollar un número ilimitado de los objetos Genexus. Solo había el inconveniente que el tiempo de duración de la licencia duraba hasta que concluía el convenio.

Hoy en día existe una versión gratis de Genexus 7.5 que se denomina: “*GENEXUS 7.5 TRIAL VERSIÓN*”, esta versión de la herramienta es la que se va a utilizar en este capítulo; se puede bajar y activar desde internet y es de libre uso para todo público.

En las versiones comerciales de Genexus, cada producto o generado tiene su costo y su licencia que se las adquiere de acuerdo a las necesidades del analista.

En la versión 7.5 trial, se adoptó un tipo de licencia centralizada la cual con una sola clave se activan todos los generadores de Genexus permitidos por esta versión.

Cuando alguien obtiene el producto, ya sea trial o comercial, primeramente debe instalarlo y este programa ya instalado le dará un Site Code, el cual deberá ser enviado vía internet o telefónico, a Artech, para que le devuelvan un Site Key que le servirá para activar la versión.

4.1.2. LIMITACIONES, REQUERIMIENTOS E INSTALACIÓN DE GENEXUS 7.5 TRIAL VERSIÓN

GeneXus 7.5 Trial Versión fue creada para quienes desean evaluar el producto, como para docentes y estudiantes de universidades y centros de educación superior y para los alumnos y maestros de los cursos Genexus dictados por Artech.

Es una versión TRIAL UNLIMITED y como tal, tiene algunas limitaciones, que son:

4.1.2.1 LIMITACIONES DE GENEXUS 7.5 TRIAL VERSION

- No se puede abrir bases de conocimiento generadas por: genexus estándar a genexus trial y viceversa.
- Transacciones 20.
- Work Panels 50 incluidos los de genexus.
- Web Panel 50 incluidos los de genexus.
- Procedimientos 20.
- Reportes 20.
- La versión Trial no tiene soporte.
- Tiene soporte solo para problemas de instalación. [www009]

El cuadro 4.1 muestra los productos de Genexus 7.5 Trial Versión.

Cuadro 4.1

PRODUCTO	DESCRIPCIÓN
Modelador	Ambiente de Desarrollo
Generador VB	Para los DBMS soportados por genexus
Generador C#	Para los DBMS soportados por genexus
Generador Java	Para los DBMS soportados por genexus

Generadores de Genexus 7.5 Trial Versión

Las bases de datos soportadas son: SQL Server, Informix, Oracle, Acees, DB2/400, DB2 UDB.

4.1.2.2. REQUERIMIENTOS DE HARDWARE Y SOFTWARE PARA GENEXUS 7.5 TRIAL VERSIÓN.

- PC, Pentium III o superior.
- 32 Mb de ram, se recomienda 128 mb.
- 50 Mb de disco duro y espacio disponible para las generaciones.
- Monitor VGA o superior.
- Windows 98 o superior, NT con SP6.
- Internet Explorer 6.0.
- Software de cada programa a generar.

4.1.2.3. INSTALACIÓN

Existen dos maneras de instalar Genexus 7.5 Trial Versión: puede instalar el producto directamente desde internet o puede instalar bajando los archivos de instalación al disco duro desde el web.

Para que Ud. Pueda bajar los archivos o instalar directamente, debe ser primero un usuario de la comunidad Genexus, Y esto lo realiza loguiandose en la siguiente página web:

<http://www.gxtechnical.com/main/hgxtrialuser.aspx?2,4,228>

A este proceso se le llama registrarse on-line o registrarse en línea, aquí se abrirá un cuadro donde tendrá que contestar algunas preguntas. (gráfico 4.2)

Gráfico 4.2

Usuarios Registrados

Usuarios No Registrados

* Datos Requeridos

Usuario:

Clave:

Login

Información Personal

*Apellido: GUERRA BASTIDAS

*Nombre: JUAN FRANCISCO

*E-Mail: panchos@yahoo.com

*Empresa: UTN

*País: ECUADOR

*Usuario: juan

Página Web para Registrarse como Usuario Genexus

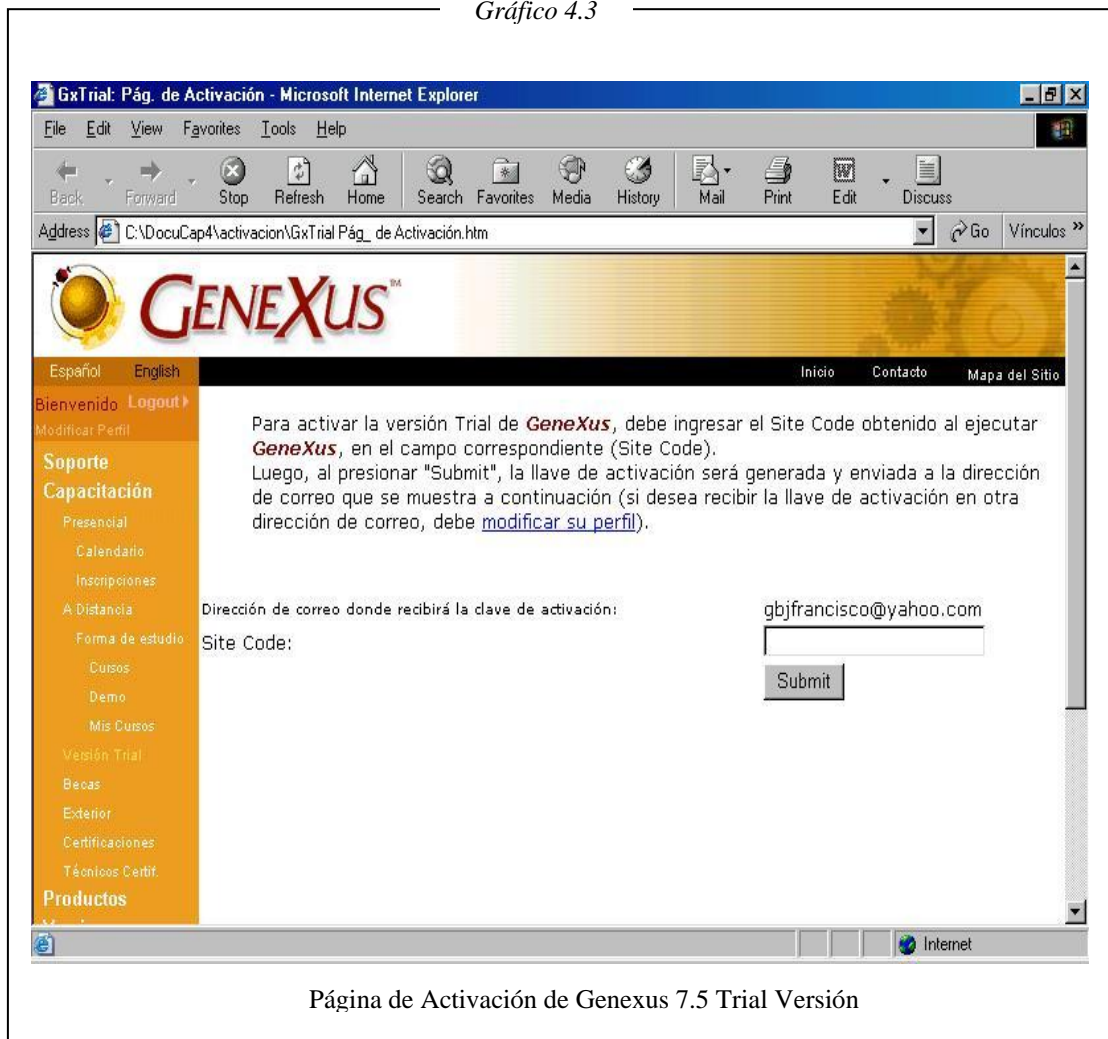
Entonces el nuevo usuario contestará las preguntas personales como: nombres, empresa, nombre de usuario, password, etc. Se solicita la dirección de un correo electrónico, ya que a esa dirección Artech envía el site key de activación de Genexus.

Registrado como usuario Genexus, puede bajar todas las cosas free de la herramienta, en este caso los instaladores de Genexus 7.5 Trial Versión, desde la página:

<http://www.gxtechnical.com/main/hgxuser.aspx?2,4,286>

Una vez instalado Genexus en su PC, la herramienta emite un site código de la siguiente manera: SITE CODE: 44205 12264 63626 40509 y pedirá un SITE KEY del mismo formato, el cual se lo obtiene enviando el site code a la página web de activación y este envía el key code directamente a su dirección de correo electrónico que puso en la página de registro. (gráfico 4.3)

Gráfico 4.3



4.1.3. OBJETOS BÁSICOS GENERADOS POR GENEXUS

TRANSACCIONES. Las transacciones permiten definir los objetos de la realidad. La mayor parte de las transacciones pueden ser identificadas prestando atención a las entidades que el usuario menciona, como: clientes, proveedores, productos, etc. A partir de las transacciones, Genexus infiere el diseño de la base de datos.

PROCEDIMIENTOS. Son los procesos no interactivos de actualización de la base de datos; son conocidos como procesos batch.

REPORTES. Son aquellos que recuperan información a partir de los datos almacenados y no los alteran. A los reportes generalmente se los conoce como listados.

PANELES DE TRABAJO Y WEB PANELS. Permiten definir consultas interactivas a las bases de datos, ya sea en modo windows o en modo web.

MENUES. Son objetos organizadores del resto de objetos Genexus, los cuales sirven para realizar listas desplegables de selección de objetos. [LIB001] [LIB002]

4.2. DEFINICIÓN DE REQUERIMIENTOS PARA SOLUCIONES GENEXUS

Para poder entender mejor el funcionamiento y el diseño de los diferentes objetos Genexus, lo mejor es realizar una aplicación real. En este caso se desarrollará un módulo o una simplificación de un sistema que servirá como referencia en todo este capítulo; y es el siguiente:

“SISTEMA DE CONTROL DE PEDIDOS DE COMPRAS, PARA UNA EMPRESA PROVEEDORA DE ACCESORIOS PETROLEROS ”

Esta mini aplicación explicativa tendrá la facultad de facilitar a los encargados de las compras, tener un control automatizado, sobre los accesorios que provee dicha empresa.

Ellos se encargarán de mantener el stock adecuado y de solicitar los pedidos necesarios para que la empresa marche de la mejor manera y no le falten productos para no quedar mal con sus clientes potenciales.

El personal de compras estará íntimamente relacionado con los proveedores, tendrá toda la información acerca de ellos y los productos que tiene. Se encargará de hacer los pedidos, con consultas previamente al stock en bodega. Cada pedido alimentará el stock de cada producto.

Con esta introducción se empezará el diseño de aplicaciones con Genexus.

4.3. DISEÑO DE TRANSACCIONES

El desarrollo de aplicaciones comienza con la definición de las transacciones, que son el diseño de la realidad. Como el modelo de Genexus es incremental, no es necesario en esta parte definir a todas las transacciones que van a ir en el sistema, sino se debe definir a las más relevantes con su estructura más conveniente.

Para diseñar de mejor manera las transacciones, primero se debe analizar cuales son los objetos de la realidad, ya sean visibles o imaginarios y cuál será su comportamiento.

Se empezará por la *DESCRIPCIÓN DEL SISTEMA*, que es una lluvia de ideas sobre las posibles transacciones que tendrá la aplicación y de qué forma quisiera visualizarlas en pantalla; A breves rasgos el sistema en sus primeros pasos tendría las siguientes transacciones:

- Productos
- Categorías
- Proveedores
- Vendedores
- Pedidos

4.3.1. OBJETOS DE LAS TRANSACCIONES

Las transacciones tienen otros objetos que también pertenecen a la realidad y se los debe definir en esta etapa del desarrollo de aplicaciones:

Estructura. De qué atributos o campos está compuesta la transacción, y qué relación tienen entre si.

Interface o Form GUI-Windows. Cada transacción contiene un form GUI-Windows (Graphical User Interface Windows) mediante el cual se realizarán las altas, bajas y modificaciones en ambiente Windows. Es la parte donde se realizan las pantallas que quiere ver el usuario final.

Form Web. Es la parte donde se realizan las pantallas que quiere ver el usuario final; pero en modo web.

Reglas. Con las reglas se define el comportamiento particular de las transacciones. Por ejemplo: cuáles son los valores por defecto de los atributos, a que atributo se le debe sumar o restar un stock, qué mensaje debe aparecer si el usuario inserto un dato fuera de rango, etc. En las reglas se define grán parte de la lógica del negocio.

Eventos. Las transacciones soportan la programación orientada a eventos. Este tipo de programación permite el almacenamiento de código ocioso el cual es activado luego de ciertas acciones provocadas por el usuario o por el sistema.

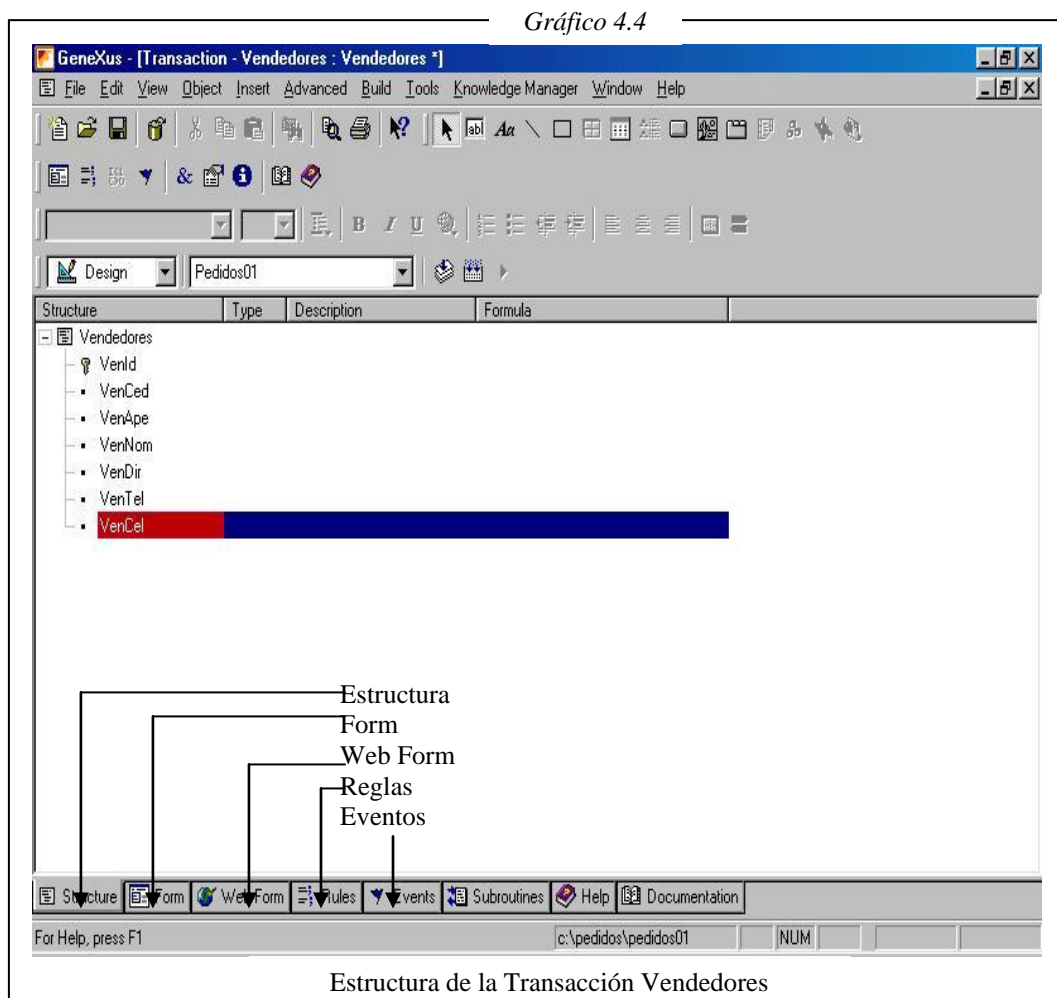
Subrutinas. Son rutinas locales a la transacción que se ejecutan al ser invocadas si se cumple alguna función específica.

Propiedades. Características que se pueden configurar para definir el comportamiento general de la transacción.

Form Classes. Es el tipo de imagen que el usuario quiere ver en su aplicación; ya sea en modo texto para ambientes IBM (AS/400) o tipo Windows con su presentación nativa o web. [LIB001] [LIB002]

4.3.2. ESTRUCTURA DE UNA TRANSACCIÓN

Como se dijo anteriormente, en una transacción se define los atributos o capos y cómo están relacionados unos de otros. Se empezará diseñando la transacción Vendedores. (gráfico 4.4)



VenId	Id del Vendedor
VenCed	Cédula del Vendedor
VenApe	Apellidos del Vendedor
VenNom	Nombres del Vendedor
VenDir	Dirección del Vendedor
VenTel	Teléfono de Vendedor
VenCel	Celular del Vendedor

El campo VenId cuya descripción es Vendedor ID representa al código del vendedor; para esto se debe definir el tipo de campo o atributo de la siguiente manera. (gráfico 4.5)



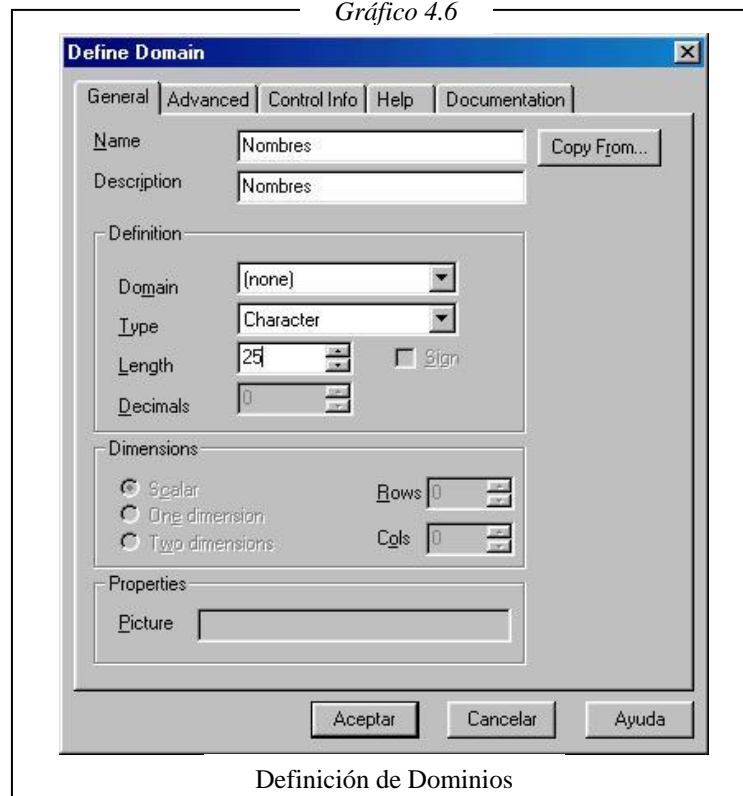
Name: Es el nombre del atributo (VenId).

Descripción: Como se va a identificar al atributo en la realidad (ID del Vendedor).

Type: Es el tipo de datos (character de 3).

Domain: es un tipo estándar de atributo, por ejemplo para nombres y apellidos se usará un character de 25, cuyo nombre de dominio es nombres. (gráfico 4.6)

Gráfico 4.6



Los dominios se utilizan cuando en una base de datos, ciertos atributos comparten definiciones similares; esto no quiere decir que establecen una relación directa entre ellos, en su definición constan los siguientes propiedades:

Length: largo del atributo.

Decimals: Número de posiciones decimales.

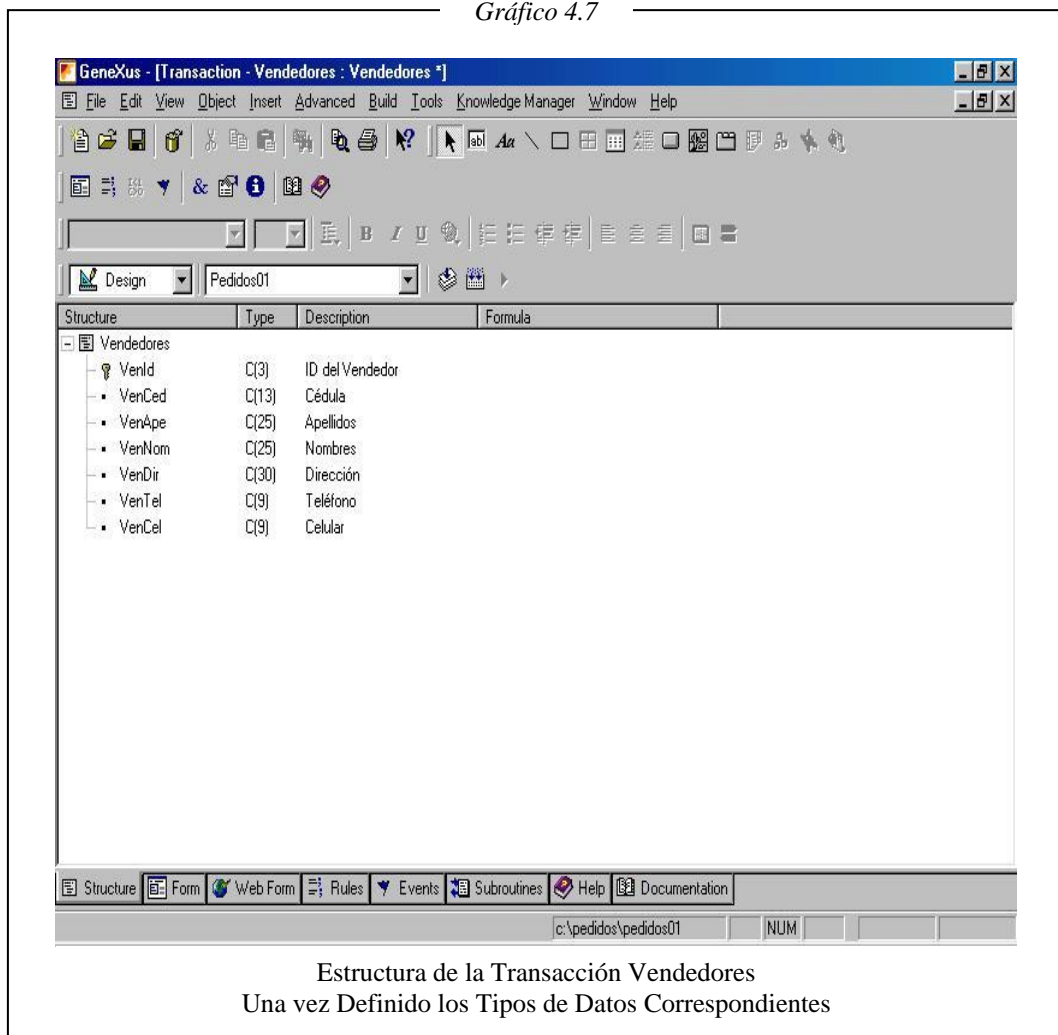
Picture: Formato del campo que permite una visualización en la entrada y salida. Por ejemplo: en campos carácter @! significa que todos los caracteres se aceptan y despliegan automáticamente en mayúsculas.

Value Range: Rango de valores válidos para el atributo. Por ejemplo: La siguiente definición: 1:20 30: significa que el valor debe estar entre 1 y 20 o ser mayor que 30. 1 2 3 4 significa que el valor debe ser 1, 2, 3 ó 4. 'S' 'N' significa que el valor debe ser 'S' o 'N'. [LIB002]

ATRIBUTOS CLAVE: Son los atributos que en las bases de datos se los considera **CLAVES PRIMARIAS**, son únicos y sirven para identificar a la transacción, en vendedores el atributo clave es VenId y se lo reconoce porque esta marcado con una llave.

La estructura de la transacción vendedores, ya con las descripciones, nombres, tipos, dominios, de sus atributos, quedaría de la siguiente manera. (gráfico 4.7)

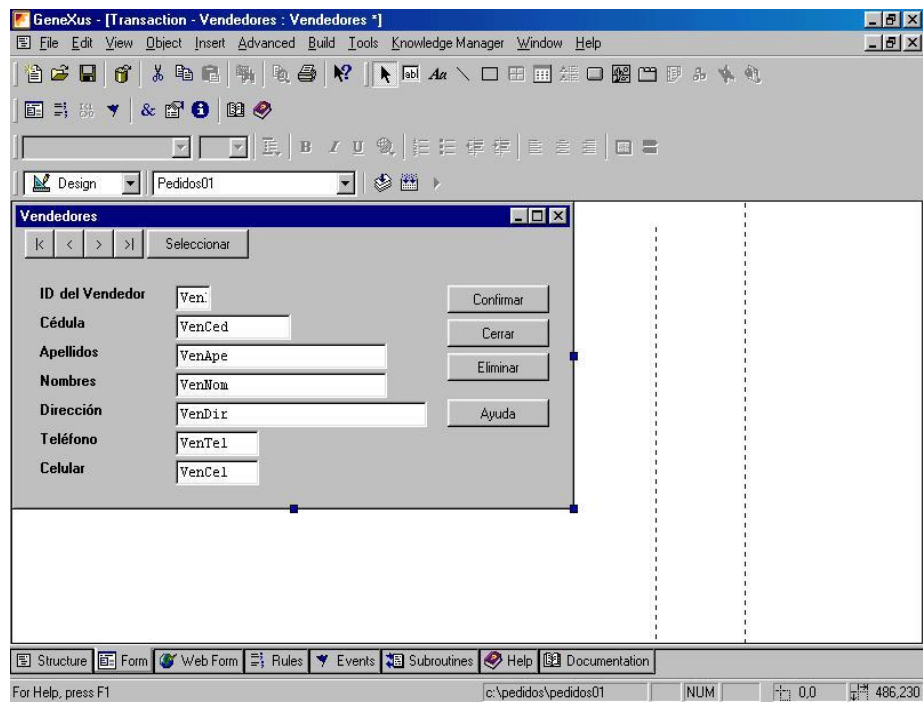
Gráfico 4.7



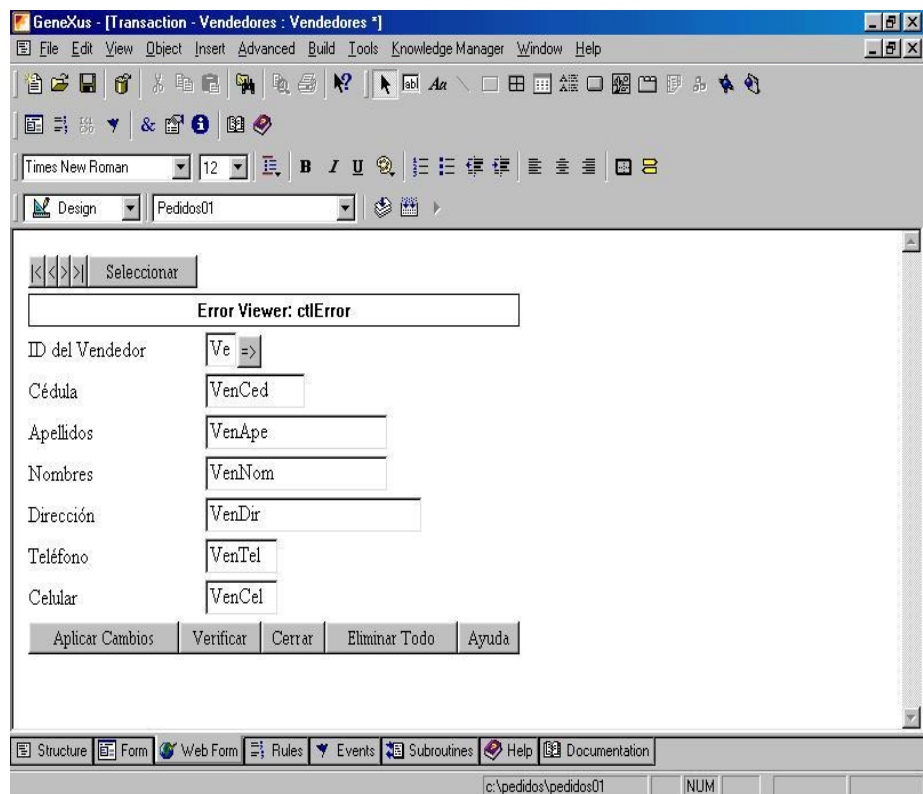
Terminada la definición de datos Genexus genera automáticamente las pantallas o formularios que se observarán en la realidad. Como esta generación es automática el programador no tiene que hacer absolutamente nada para poder visualizar sus aplicaciones, simplemente debe acabar la definición de datos.

En el gráfico 4.8 se observa las pantallas de la transacción vendedores en modo Windows y en modo Web.

Gráfico 4.8



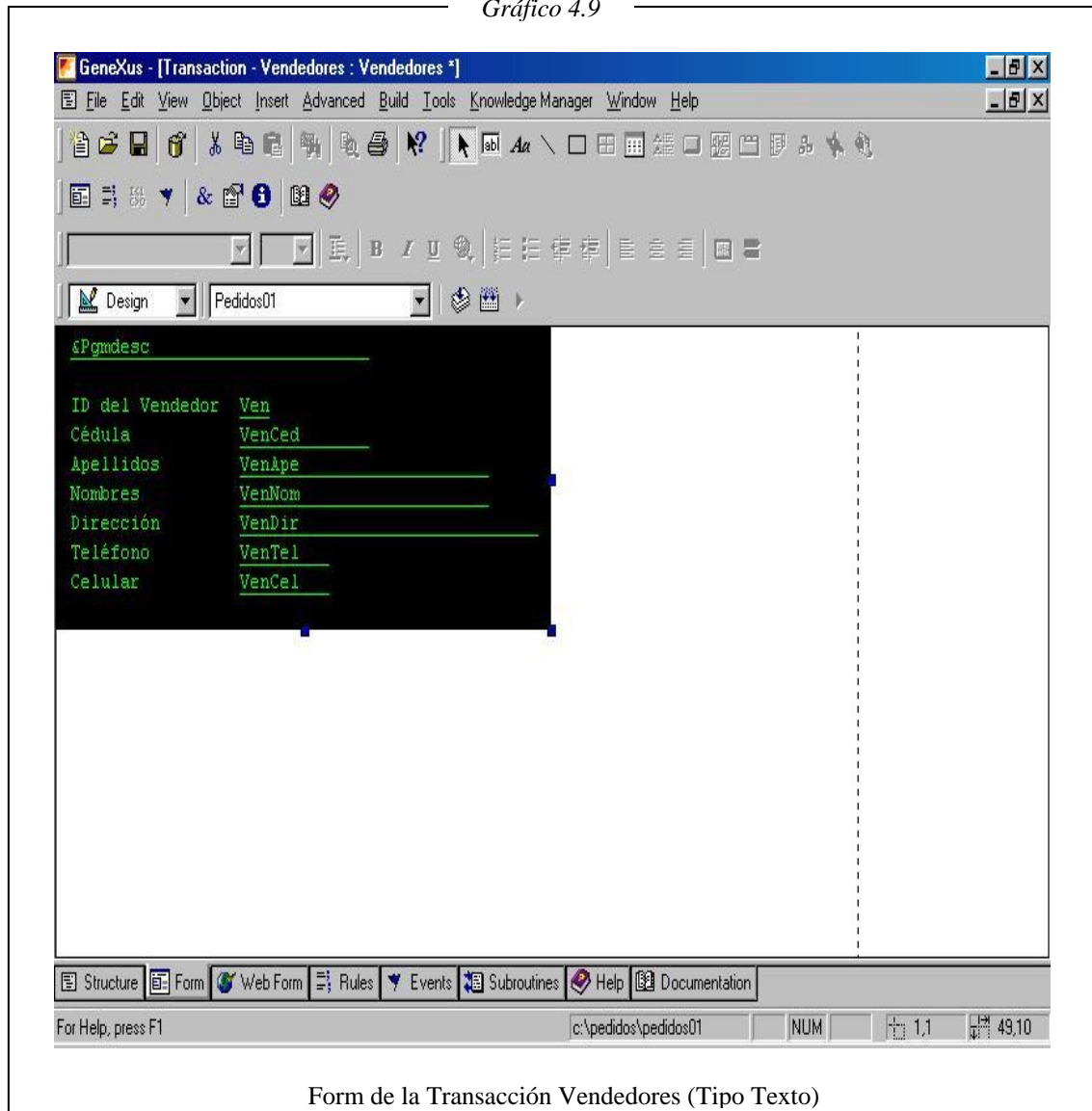
Form de la Transacción Vendedores (Tipo Windows)



Form de la Transacción Vendedores (Tipo Web - Web Form)

Si se cambia las propiedades de las transacciones; específicamente los form class, se puede ajustar a las transacciones a un modo tipo texto, el cual se lo utilizaría en ambientes AS/400. (gráfico 4.9)

Gráfico 4.9



Form de la Transacción Vendedores (Tipo Texto)

4.3.3. RELACIÓN ENTRE LA ESTRUCTURA Y EL MODELO DE DATOS.

Genexus utiliza la estructura de la transacción y captura el conocimiento necesario, para luego automáticamente definir cuál es el modelo de datos que se necesita y con ello crear la base de datos; en la estructura anterior Genexus infiere de la siguiente manera:

No existen dos Vendedores con el mismo código (VenId).

Para cada VenId existe solo un valor de VenCed, VenApe, VenNom, VenDir, VenTel y VenCel. (Normalización: 3ra forma normal)

Con esta información va construyendo el modelo de datos. En este caso a la transacción Vendedores se le asociará la Tabla: Vendedores que quedaría de la siguiente manera. (tabla 4.1)

Tabla 4.1

VENEDORES
VenId*
VenCed
VenNom
VenApel
VenDir
VenTel
VenCel

Tabla Vendedores
Asociada a la Transacción Vendedores

El *ÍNDICE* por defecto está dado por la clave primaria (VenId) pero luego puede ser cambiado por el desarrollador. Como se dijo que la tabla vendedores esta asociada a la transacción vendedores, si: se modifica, se elimina o se altera algo en la transacción vendedores, automáticamente: se modificará, se eliminará o se alterará la tabla vendedores.

4.3.4 NOMENCLATURA GENEXUS. (GIK)

Artech a definido un estándar para la nomenclatura de sus objetos, que se la denomina GIK Genexus Incremental Knowledge Base (Base de Conocimiento Incremental de Genexus), esta nomenclatura es utilizada estrictamente por toda la comunidad de usuarios Genexus, ya que de aquí Genexus saca el conocimiento para la generación de la base de datos (gráfico 4.10, tabla 4.2).

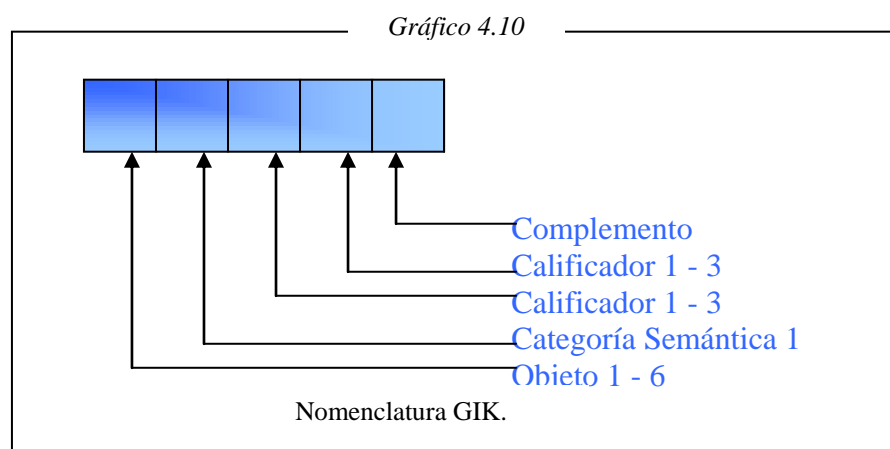


Tabla 4.2

<i>Objeto</i>	<i>Categoría</i>	<i>Calificador</i>
Cli	Cod	
Cli	Nom	
Cli	Fch	
Cli	Fch	Ini
Cli	Fch	Fin
Fac	Nro	
FacDe	Cos	

Ejemplo de Nomenclatura GIK

Esta nomenclatura ayuda a la base de conocimiento a generar de una mejor forma las tablas y es el principio de una cultura de programación nueva; ya que con otras herramientas no importa el nombre que se les de a los atributos de las tablas, con esta nomenclatura el programador se centra a un estándar, el cual es muy útil para reconocer fácilmente los atributos de una determinada transacción.

4.3.5 TIPOS DE CONTROLES GENEXUS

Para explicar los tipos de controles se realizara algunas transacciones con ellos; el gráfico 4.11 muestra la transacción Proveedores utilizando los siguientes controles: para ProTip se utilizó un Combo Box, Para ProAct Check Box y para ArtNac se utilizó un Radio Button. Estos controles son estándar y los generan la mayoría de programas; sus características son:

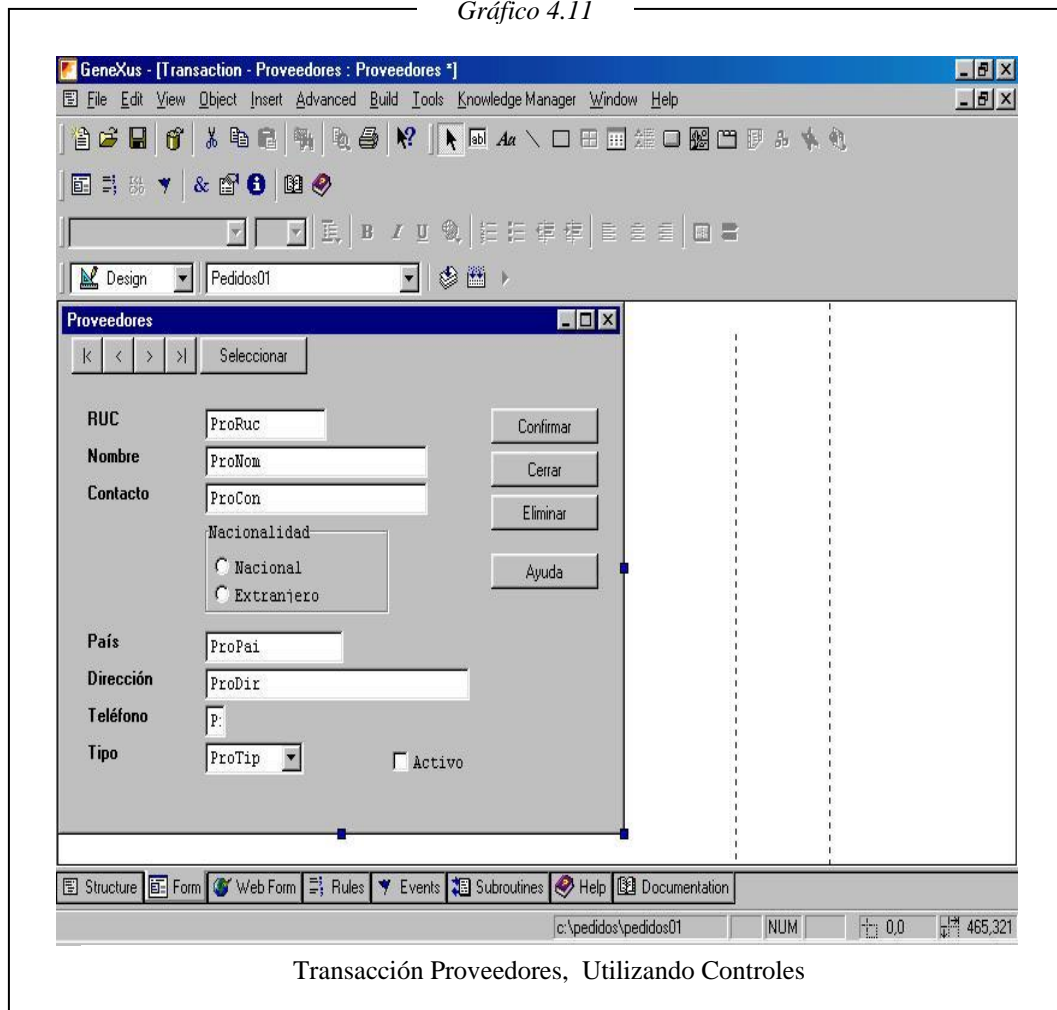
Edit. Normalmente los atributos tienen un rango de valores muy grande, por ejemplo: un nombre, un precio, etc. En estos casos se le permite al usuario entrar el valor del atributo y el sistema se encarga de validarlo.

A estos tipos de controles se los llama 'Edit Box'. Sin embargo, existen atributos que tienen un rango de valores muy pequeño y que pueden ser desplegados de antemano para que el usuario seleccione uno. De esta forma se controla que ingresen solo valores válidos. Estos tipos de controles son:

Check Box. Es usado para aquellos atributos que tienen solo dos valores posibles elegidos por el usuario; existe una única descripción disponible y en caso que este campo esté seleccionado, el valor almacenado será el especificado por el usuario.

Radio Button. Los Radio Button en cambio pueden tener más de dos valores en pantalla, el valor que se almacena es manejado internamente para luego pasar la información a la base de datos; en el ejemplo esta dado por la nacionalidad del proveedor.

Gráfico 4.11



Combo Box. Es generalmente usado para aquellos atributos que tienen un rango grande de valores que hace poco práctico utilizar un Radio Button.

Se despliega un campo de tipo Edit siempre y cuando presione: un botón una viñeta o una tecla específica que se encuentra a la derecha o izquierda del campo seleccionado; no es recomendable utilizar este tipo de control como lista de selección de atributos que puedan ser leídos de una tabla; Para estos casos se usan los Dynamic Combobox.

Dynamic Combobox. Un Dynamic Combobox es un tipo de control similar al Combo Box. La forma de operación es similar, salvo que los valores desplegados no son estáticos, sino que son descripciones leídas de una determinada tabla de la base de datos.

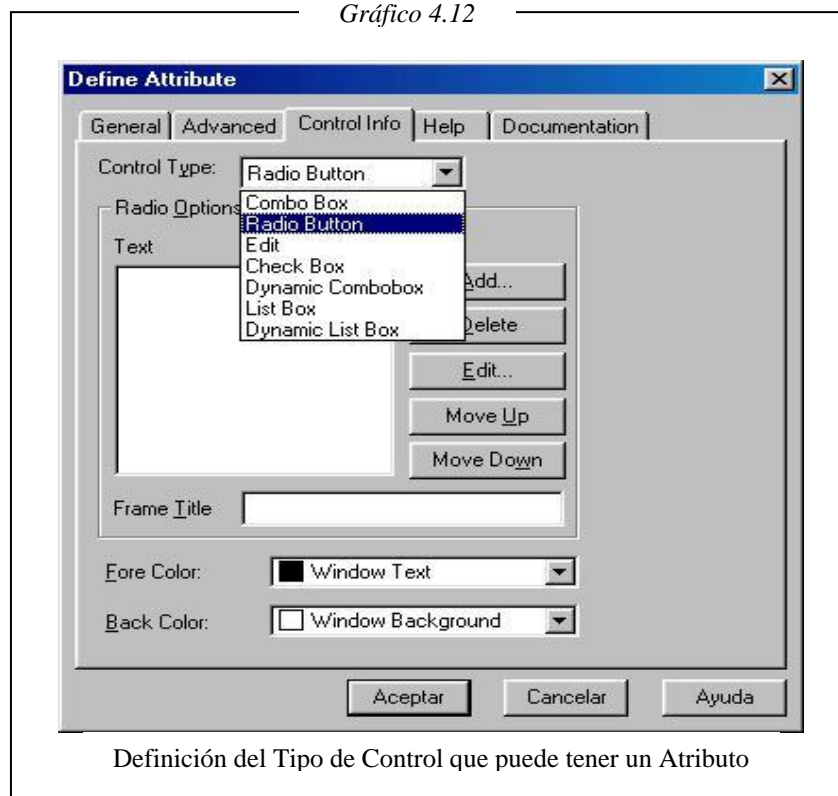
List Box. Este tipo de control tiene asociada una colección de ítems. Cada ítem tiene asociado un par <valor, descripción>. El control da la posibilidad de seleccionar un solo ítem a la vez. El atributo o variable toma el valor en el momento que se selecciona el ítem. La selección se realiza dando clic con el mouse en un ítem o con las flechas del teclado.

El control *List Box* puede verse como un *Combo Box* abierto, es decir, que los valores (items) se muestran desplegados en una lista, siendo la definición y funcionalidad del List Box totalmente análoga a la del *Combo Box*.

Dynamic List Box. Este tipo de control tiene asociada una colección de items. Cada ítem tiene asociado un par <valor, descripción>. La colección de items se carga desde una tabla de la base de datos en tiempo de ejecución. En tiempo de diseño se asocian dos atributos al Dynamic List Box, uno al valor que tendrá el ítem y el otro a la descripción que éste tomará. Ambos atributos deben pertenecer a la misma tabla. En tiempo de especificación se determina la tabla desde la cual se traerán los valores y las descripciones.

El programador podrá definir este tipo de controles en la definición de los atributos, en control de información y control de tipo, esto se los observa en el gráfico 4.12.

Gráfico 4.12



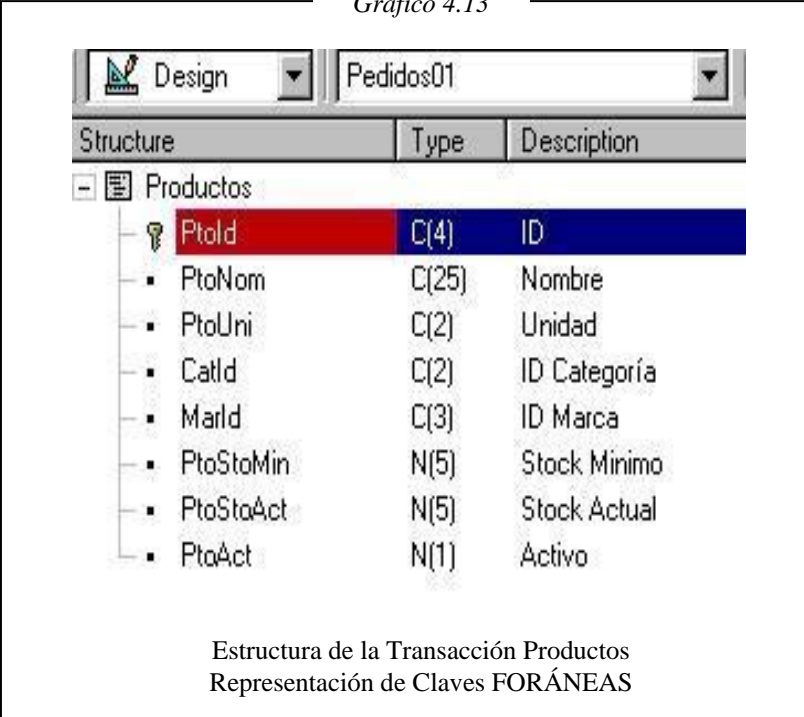
4.3.6 ATRIBUTOS VINCULADOS DE OTRAS TRANSACCIONES E INTEGRIDAD REFERENCIAL

Se va a definir la transacción Productos, para esto ya se han diseñado las transacciones: Categorías y Marcas que están íntimamente vinculadas a Productos. (gráfico 4.13)

En una base de datos común, habría que diseñar la integridad referencial con: Productos, Categorías y Marcas, ya que las claves primarias de categorías y marcas, serían claves foráneas de productos.

Con Genexus, simplemente se agrega el tipo de atributo con su clave primaria de marcas y categorías, en la estructura de la transacción productos. De esta manera Genexus automáticamente crea su propia integridad referencial. (gráfico 4.13)

Gráfico 4.13



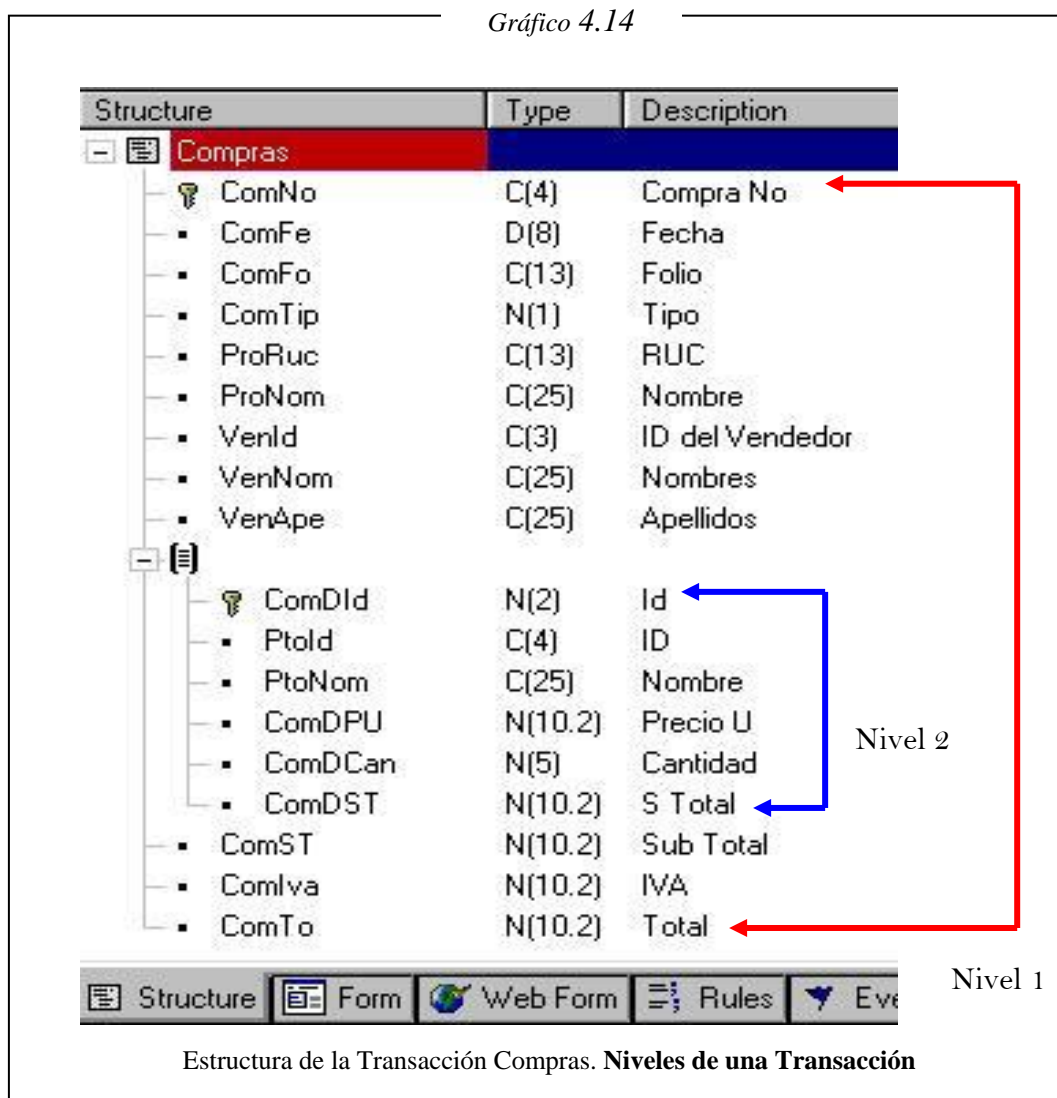
Structure	Type	Description
Productos		
PtoId	C(4)	ID
PtoNom	C(25)	Nombre
PtoUni	C(2)	Unidad
CatId	C(2)	ID Categoría
MarId	C(3)	ID Marca
PtoStoMin	N(5)	Stock Minimo
PtoStoAct	N(5)	Stock Actual
PtoAct	N(1)	Activo

Estructura de la Transacción Productos
Representación de Claves FORÁNEAS

Como se puede apreciar CatId y MarId son las claves primarias de categorías y marcas, y foráneas de productos. Con esta información basta para crear la integridad referencial.

Otra característica a destacar es que cuando se define la estructura de una transacción no se está describiendo la estructura de una tabla y si los datos que se necesitan en la pantalla o en las reglas como: CatNom, no quiere decir que éste se encuentre en la tabla de productos. Se lo incluye en la estructura ya que se quiere que esté presente en la pantalla de productos, en la tabla solo se almacenará CatId.

4.3.7. NIVELES DE UNA TRANSACCIÓN



Como se observa en las transacciones anteriores, estas tienen un solo nivel, o sea las transacciones despliegan información de una sola estructura; para observar los niveles que podría tener una transacción, se realizará la transacción compras. (gráfico 4.14)

En transacción compras se puede ver una nueva situación que es: los niveles de una transacción. Aquí existen dos niveles, se podría decir: el de la tabla compras y el detalle compras, cada nivel se lo define abriendo y cerrando paréntesis.

Cada nivel de una transacción tiene una tabla asociada. La clave primaria de las tablas asociadas a los niveles subordinados, es la concatenación de los identificadores de los niveles superiores. Genexus exactamente generará a las tablas compras y compras1. Compras 1 vendría a ser comúnmente como detalle compras. (tabla 4.3) (gráfico 4.15)

Tabla 4.3

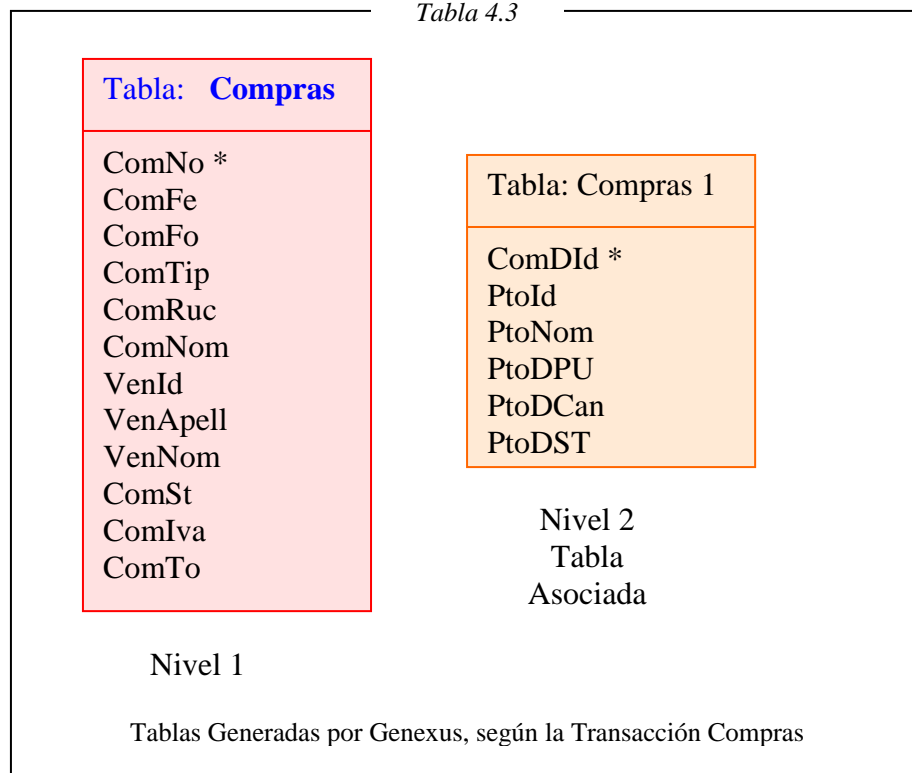


Gráfico 4.15



Tablas Creadas por Genexus Vistas Desde la Base de Datos (Acces 2000)

Desde esta pantalla se observa las tablas creadas automáticamente por Genexus, en especial se muestra compras y compras1, que modelando manualmente serían compras y detalle compras.

La elección, de los identificadores o claves primarias es una de las tareas más importantes que puede simplificar o complicar la implementación del sistema. Por ejemplo compras no admite que estén dos productos de los mismos, en este caso ya no se deberá definir a ComDIId como identificador del segundo nivel de compras, si no a PtoId. Como no se definió a PtoId como identificador Compras1, se deberá hacer un procedimiento para que controle esta situación.

“Cuanto más reglas de la realidad se pueden reflejar en la estructura de los datos, menor será la cantidad de procesos que se deberá implementar, de esta forma se gana simplicidad y flexibilidad” [LIB001]. El gráfico 3.16 indica cómo se verá en la realidad la transacción compras.

Gráfico 3.16

The screenshot shows a software form titled "Compras" with the following fields and controls:

- Design dropdown, Pedidos01 dropdown, and navigation icons at the top.
- Form fields: Compra No (ComN), Fecha (ComFe), Folio (ComFo), Proveedor (ProRuc), Nombre (ProNom), Vendedor (Ven), Apellidos (VenApe), and Nombres (VenNom).
- Radio buttons for "Nacional" and "Importación".
- A table with columns: Id, ID, Nombre, Precio U, Cantidad, S Total.
- Summary fields: Sub Total (ComST), IVA (ComIva), and Total (ComTo).
- Bottom navigation bar: Structure, Form, Web Form, Rules, Events, Subroutines, Help, Documentation.

Id	ID	Nombre	Precio U	Cantidad	S Total
Co:	PtoI	PtoNom	ComDPU	ComDCan	ComDST

Form de la Transacción Compras

4.3.8. TIPOS DE RELACIONES ENTRE OBJETOS

La base para realizar cualquier aplicación es definir bien qué tipo de relación existe entre los diferentes objetos, en especial en las transacciones o en las entidades. Así se podría decir que: Un proveedor tiene muchas compras, entonces sería (relación 1- N) o sea de uno a varios, o que una compra tiene un solo proveedor (relación 1 – N) que viene a ser la relación simétrica de la primera, otra forma son las relaciones M – N, en el caso de las compras y productos que dice que: una compra tiene muchos productos y un producto puede estar en varias compras, en este caso para romper ese varios a varios, Genexus automáticamente crea la tabla Compras1.

Para no tener problemas y para que la herramienta normalice mejor y para que cree integridad referencial de una forma satisfactoria, lo correcto y óptimo es definir bien a los identificadores, de esta forma sabrá cual es la relación existente entre los objetos. Según el tipo de relaciones, Genexus a creado el concepto de tabla base y tabla extendida.

TABLA EXTENDIDA. Dada una tabla de la base de datos, (TABLA BASE) se denomina tabla extendida de la misma, al conjunto de atributos conformados por: Atributos que pertenecen a la tabla y Atributos que tengan una relación N-1 con la tabla extendida determinada hasta el momento.

Los criterios de normalización de diseño de la base de datos, apuntan a minimizar la posibilidad de inconsistencia en los datos. Una base de datos diseñada de esta manera tiene una serie de ventajas importantes, tal es así que hoy en día la normalización de la base de datos es un estándar de diseño universal; también se debe tomar en cuenta algunos inconvenientes.

El inconveniente más notorio es que los datos se encuentran dispersos en muchas tablas, y cuando se quiere hacer consultas muy complejas, se debe consultar a una cantidad importante de tablas.

Un ejemplo de esto es el siguiente: Para listar una consulta de Facturas es necesario consultar a: Facturas, clientes, categorías y productos.

Para simplificar esto Genexus utiliza el concepto de tabla extendida. (gráfico 4.17). La tabla 4.4 aclara cuales con las tablas extendidas de las diferentes tablas bases.

Gráfico 4.17

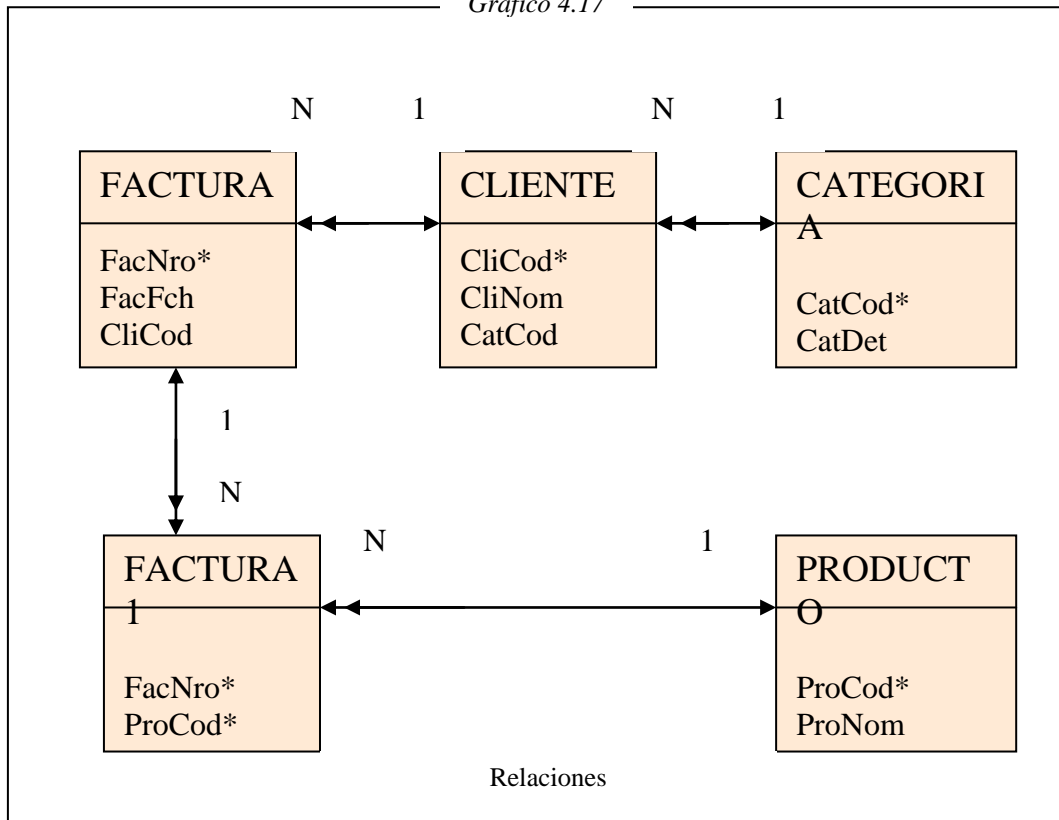


Tabla 4.4

TABLA BASE	TABLA EXTENDIDA
Categoría	Categoría
Cliente	Cliente Categoría
Factura	Factura Cliente Categoría Factura1
Factura1	Factura1 Producto Factura Cliente
Producto	Producto

Tabla Base y Tabla Extendida

4.3.9. FORM DE UNA TRANSACCIÓN (FORMULARIOS O PANTALLAS)

Una transacción puede tener varios forms, ya sea diseñados por el usuario de forma manual, o aceptando los que automáticamente genera Genexus.

El form que genera esta herramienta tanto en modo windows, texto o web, se lo puede modificar a preferencia del usuario, utilizando las herramientas que tiene Genexus, los forms por defecto son los que constan en todos los gráficos anteriores. Cabe recalcar que en cada form se puede también mostrar imágenes.

También se puede usar un style previamente elaborado, el cual servirá como plantilla de los objetos Genexus a crear, el gráfico 4.18 muestra un form por defecto y otro más elaborado.

Gráfico 4.18

Form Generado por Defecto

Form Diseñado por el Programador

4.3.10. ATRIBUTOS Y FÓRMULAS

Un atributo es una FÓRMULA, si su valor se puede calcular a partir del valor de otros atributos. Por ejemplo el subtotal es una fórmula porque su valor es igual a cantidad * precio.

Una fórmula se inserta seleccionando el atributo y haciendo clic con el botón derecho del mouse, saldrá un menú y selecciona fórmula., el gráfico 4.19 muestra la estructura de la transacción compras una vez definidas sus fórmulas.

Gráfico 4.19

Structure	Type	Description	Formula
[-] Compras			
ComNo	C(4)	Compra No	
ComFe	D(8)	Fecha	
ComFo	C(13)	Folio	
ComTip	N(1)	Tipo	
ProRuc	C(13)	RUC	
ProNom	C(25)	Nombre	
VenId	C(3)	ID del Vendedor	
VenApe	C(25)	Apellidos	
VenNom	C(25)	Nombres	
[-] (f)			
ComDId	N(2)	Id	
Ptold	C(4)	ID	
PtoNom	C(25)	Nombre	
ComDPU	N(10.2)	Precio U	
ComDCan	N(5)	Cantidad	
fx ComDST	N(10.2)	S Total	ComDPU * ComDCan
fx ComST	N(10.2)	Sub Total	SUM (ComDST)
fx ComIva	N(10.2)	IVA	ComST * 0.12
fx ComTo	N(10.2)	Total	ComST + ComIva

Estructura de la Transacción Compras
Una vez Definidas sus Fórmulas

Los atributos fórmula están marcados con un (fx) y son lo que normalmente se conoce como campos calculados.

En cada transacción se puede definir qué atributos son fórmulas, esta definición es global, o sea tendrá un impacto en todos los objetos de la aplicación.

Existe mucha similitud entre fórmulas y reglas, incluso la sintaxis es similar, pero la fórmula es global y la regla es local, o sea solo para el objeto Genexus que a sido definida.

FÓRMULAS HORIZONTALES. Son aquellas en las que los atributos involucrados deben pertenecer a la tabla extendida del atributo definido como fórmula; o sea los atributos deben estar en un solo nivel; pueden ser una o varias expresiones condicionales o no condicionales, así:

$$\text{ComDST} = \text{ComDPU} * \text{ComDCan}$$

$$\text{ComIva} = \text{ComST} * 0.12$$

El IVA es igual al subtotal * 0.12 y el subtotalD es igual a precio unitario por la cantidad. Para representar una fórmula con condiciones, se va a suponer que en una transacción ventas existen descuentos.

$$\begin{aligned} \text{Descuento} &= \text{VenTot} * 0.10 \text{ if VenTot} > 500 \text{ .and. VenTot} \leq 1000; \\ &= \text{VenTot} * 0.20 \text{ if VenTot} > 1000; \\ &= 0 \text{ OTHERWISE.} \end{aligned}$$

El descuento será del 10% si la venta total está entre 500 y 1000, será del 20% si la venta total es mayor a 1000 y será cero si la venta es menor a 500

Las expresiones pueden utilizar operadores aritméticos como: (+, -, *, /, ^) y muchas funciones como today () que inserta la fecha actual, cuando los casos son más complicados se puede utilizar subrutinas para que ejecuten las cosas que no pueden hacer las fórmulas.

FÓRMULAS VERTICALES. En este tipo de fórmula los atributos involucrados no se encuentran en la misma tabla extendida que la fórmula. Específicamente las fórmulas verticales son dos: SUM y COUNT.

SUM. Suma un atributo perteneciente a una tabla directamente subordinada. Su sintaxis es:

SUM (Atributo), en la transacción compras se tiene: ComST = SUM (ComDST) que suma los datos del atributo ComDST.

COUNT. Cuenta las filas de una tabla directamente subordinada o sea cuando un atributo está en una tabla con relación (1-N), su sintaxis es COUNT (atributo).

Las fórmulas verticales se resuelven mediante una navegación vertical es por eso su nombre de fórmulas verticales. Dentro de una fórmula vertical no se puede agregar directa o indirectamente expresiones aggregate select y el atributo count no puede ser parte de alguna clave de la transacción. El atributo sum puede ser fórmula de fórmulas, como se lo expresa en el ejemplo anterior.

Un atributo es **FÓRMULA DE FÓRMULA**, cuando su valor es calculado por medio de otros atributos que ya han sido definidos como fórmula, por ejemplo: Et total de una venta se lo calcula de la suma de los subtotales más el porcentaje del IVA sacado del subtotal.

4.3.11. REGLAS

Según los preceptos de las metodologías orientadas a objetos, en cada transacción se debe definir cuál es su estructura y cuál es su comportamiento. En este caso ya se vio el diseño de las estructuras, ahora toca definir cómo será su comportamiento.

Con este preámbulo las *REGLAS* son todos aquellos parámetros que define el analista de sistemas, para definir el comportamiento de los datos y los objetos en las aplicaciones, a continuación se citará algunas reglas disponibles.

Default. Se utiliza para definir los valores por defecto de algunos atributos, por ejemplo:

```
default ( ComFe, today( ) );
```

Añade la fecha actual en el campo seleccionado. El funcionamiento de default varía según el tipo de diálogo, full-screen o campo a campo. En el diálogo full-screen se asigna el valor por defecto si el usuario no digitó nada en el campo. En el diálogo campo a campo primero se asigna el valor por defecto y luego se permite modificarlo. Esto depende de cómo quiera ver el analista.

Error. Es la regla para definir los controles que deben cumplir los datos, por ejemplo si se quiere que la cantidad de cada producto o artículo sea siempre mayor que 4.

```
Error ( 'La Cantidad debe ser mayor a 4 ' )  
if ComDCan <= 4 ;
```

Cuando la cantidad de los productos sea menor o igual a 4 (ComDCan) se desplegara una pantalla que dice “La Cantidad debe ser mayor a 4” y no podrá seguir realizando la transacción hasta no modificar el dato.

Otro ejemplo puede ser si no desea que se eliminen compras, aparecerá una pantalla que diga “Prohibido eliminar las compras.”

```
Error ( 'Prohibido eliminar las compras ' ) if delete (ComId);
```

Asignación. Dentro de las reglas de la transacción se permiten definir asignaciones a atributos y variables. La asignación a atributos implica una actualización en la tabla base del nivel o en alguna de las tablas que pertenecen a la tabla extendida del nivel.

Add y Subtract. Para entender este tipo de reglas, se realizará un ejemplo para manejar el stock de los productos. La transacción de compras debe modificar ese valor porque cada compra nueva aumenta el stock de productos. También existirá alguna otra transacción ventas, que disminuirá el stock, pero en el ejemplo solo se representará con la transacción compras.

La transacción Compras debe actualizar PtoStoAct (Producto: Stock Actual) de la transacción productos cuando la transacción compras asigne más productos en ComDCan (Compras: Cantidad de productos comprados) esta operación se la realiza desde la transacción compras, de la siguiente manera.

```
add(ComDCan, PtoStoAct );
```

NOTA: “Para poder realizar este tipo de operaciones PtoStoAct debe ser un atributo de la Transacción Compras” El Autor.

Se puede decir que SUM redundante es equivalente a la regla ADD. Por ejemplo el ComST se puede definir de dos maneras diferentes:

```
ComST = SUM( PedImp, ComDST) y redundante o  
add( ComDST, ComSt );
```

Es muy común que las fórmulas verticales tengan que estar redundantes para tener buena performance se recomienda el uso de la regla ADD y no de la fórmula SUM. La regla *SUBTRACT* es equivalente pero realiza las operaciones contrarias, es decir en ves de asignar resta.

Serial. Esta regla es útil cuando se quiere asignar automáticamente valores a algún identificador de la transacción, son conocidas como auto series. Por ejemplo se quiere hacer automático el identificador del segundo nivel de la transacción compras. (ComDId)

```
serial(ComDId, ComUltLin, 1);
```

En donde el primer parámetro define cuál es el atributo que se está numerando, el segundo indica cuál es el atributo que tiene el último valor asignado, o sea la última línea y el tercer parámetro el incremento, en este caso es 1.

El atributo ComUltLin (Ultima línea asignada) debe ser incluido en la estructura de la transacción en el nivel de ComNo.

4.3.12 CALL, AFTER Y ORDEN DE DISPARO DE LAS REGLAS.

Call es una de las reglas más importantes de Genexus, que permite llamar a otros objetos como: transacciones, paneles de trabajo, menús, reportes, textos, etc.

En el caso de la regla call, es necesario definir el momento en que se va a disparar el call, por ejemplo, desde la transacción compras se quiere llamar a un reporte que imprima la factura de compras.

CALL ('RIMPREC' , ComNo) if After (trn);

Donde rimprec es el programa llamado (factura) y ComNo es el parámetro que se envía. Al tener after (trn) el call se realiza después de haber realizado todo el pedido, los after existentes son:

Confirm. La regla se dispara después de haber confirmado los datos del nivel pero antes de haber realizado la actualización.

Insert, Update, Delete. Se dispara después de haber insertado, actualizado o eliminado el registro de la tabla base del nivel. Se usa fundamentalmente para llamar a procesos que realizan actualizaciones.

Level. Se dispara después de haber entrado todos los datos de un nivel. Se usa muchas veces para controles de totales. El control recién se realizará cuando se a terminado de ingresar todas las líneas del nivel.

Trn. Se dispara después de haber entrado todos los datos de una transacción. Se usa fundamentalmente para llamar a programas que imprimen los datos. En ambientes con integridad transaccional esta regla se dispara luego que se realiza el commit de la transacción.

4.4. DISEÑO DE REPORTE

Los reportes son procesos no interactivos de extracción de datos a manera de listados o consultas; se los considera interactivos porque el usuario no puede: insertar, eliminar, borrar o actualizar los datos consultados. Pueden ser visualizados en: pantalla o en impresora

4.4.1. DEFINICIÓN DE OBJETOS DE UN REPORTE

INFORMACIÓN. Son los datos generales del reporte, o sea la información de los datos que desea visualizar, como: nombre, id, apellido, teléfono, etc.

LAYOUT. Es la pantalla de presentación para el diseño de los reportes, similar al form de las transacciones.

CONDICIONES. Son las condiciones que deben cumplir los datos para ser impresos o visualizados, por ejemplo solo se quiere visualizar el nombre y el monto de los vendedores en el mes de agosto.

REGLAS Y PROPIEDADES. Definen aspectos generales del Reporte.

AYUDA. Texto de ayuda a los usuarios que usan el reporte.

DOCUMENTACIÓN. Texto técnico para ser utilizado en la documentación del sistema.

4.4.2. LAYOUT

En el Layout o pantalla de diseño de reportes se define simultáneamente la estructura de navegación, donde van los datos que se quieren listar y en qué orden, como también el formato de salida de los datos. Para un mejor entendimiento se muestra el formato o layout por defecto del reporte vendedores (gráfico 4.20).

Aquí se tiene comandos como: header, end, for each y end dor. Que son las partes cabeceras para definir un reporte, tanto en la presentación gráfica como títulos, carátulas, etc, como también en la parte lógica o de comportamiento de los datos.

Los datos que están en este reporte se los puede observar en el gráfico 4.21.

GRÁFICO

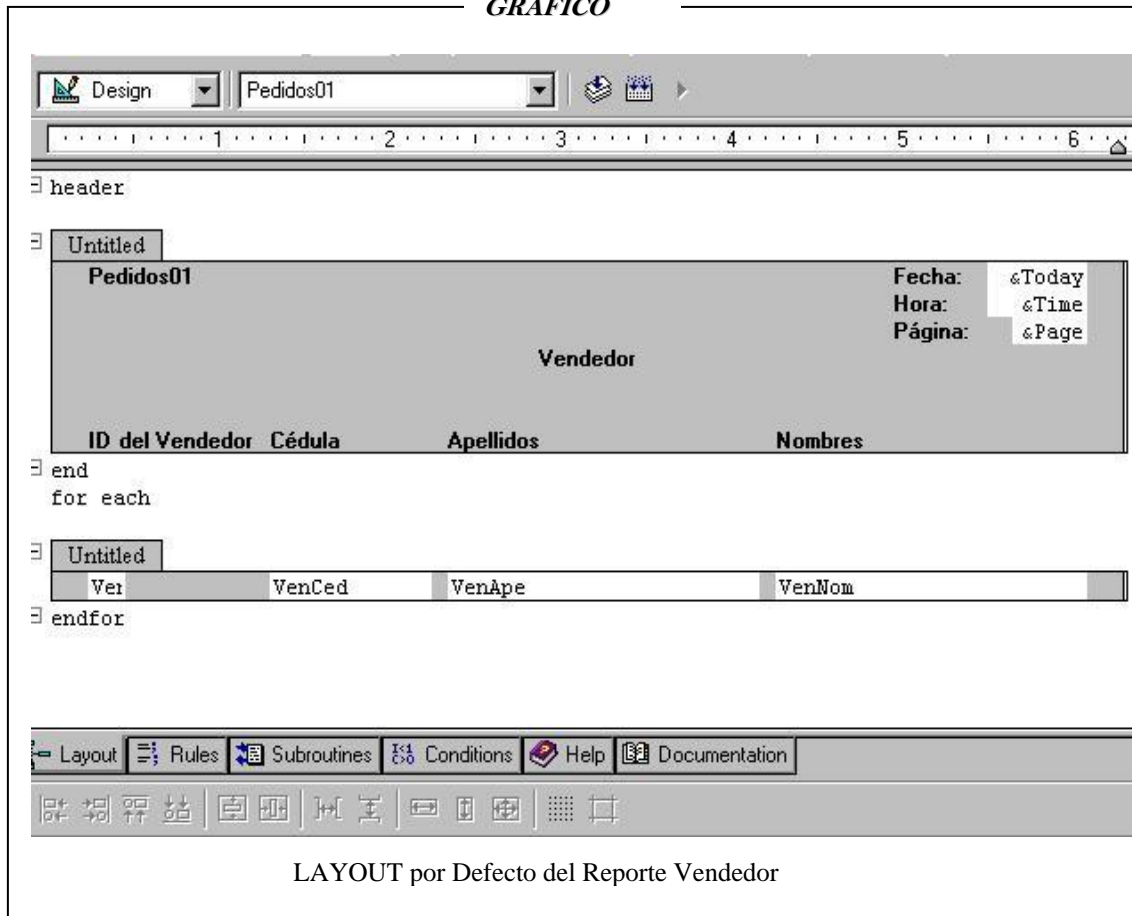
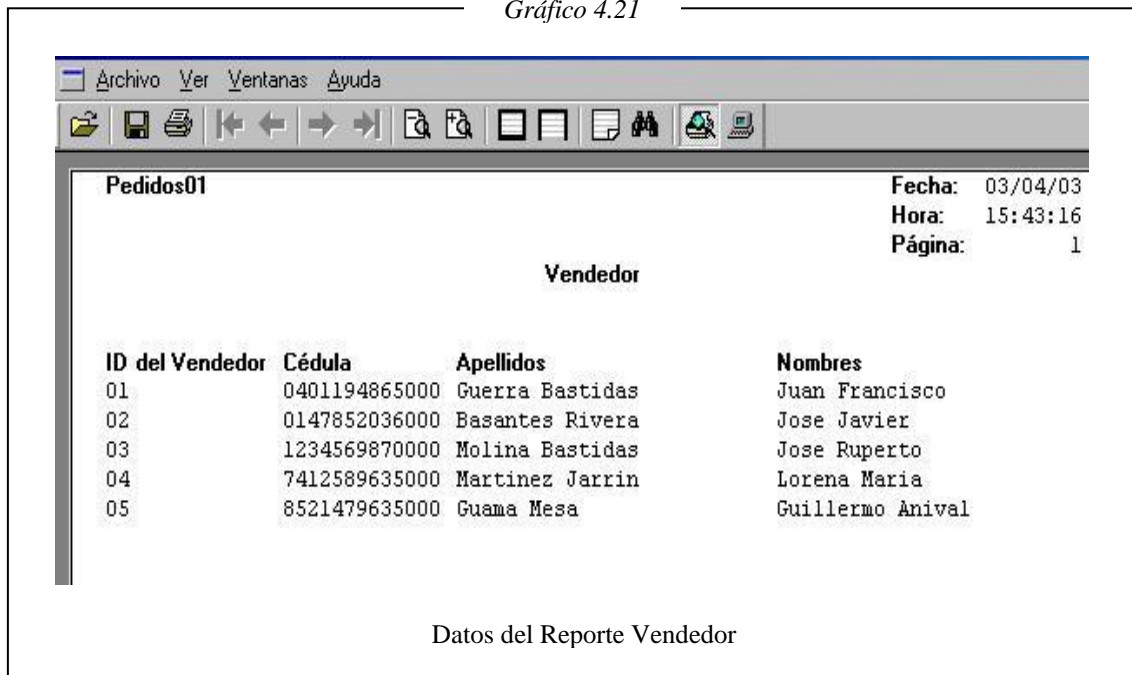


Gráfico 4.21



La definición de reportes se divide en uno o varios bloques. Estos bloques pueden ser: de **código** o de **impresión**. En los bloques de impresión se diseña la salida que posteriormente será impresa y se los denomina Prin Blocks.



Cada bloque de impresión puede tener un conjunto de controles: atributos, variables, textos, etc. Los bloques de impresión o **Print Blocks** tienen asociadas ciertas propiedades que pueden ser configuradas desde un diálogo que se abre al hacer doble-clic sobre el bloque de impresión; aquí se configura: el tipo de papel, el tipo de letra, etc. (gráfico 4.22) en este caso el print block del reporte vendedor es: VENDEDOR01, estos print blocks se utiliza para llamar a los procesos impresión desde cualquier objeto Genexus.

Se puede realizar una transacción factura, luego un reporte facturas, y después un prin block de facturas, de esta manera solo se llama a travez de un Call al Prin Block factura para que sea impreso.

Todos los bloques de impresión que se encuentren entre los comandos **HEADER** y **END** son las líneas que se imprimen en el cabezal de la página. También existe el comando **FOOTER** que permite imprimir un pie de página en cada hoja. El comando **FOR EACH** indica qué datos se desplegaran; del ejemplo anterior se tendría.

```
For each  
VenId VenCed VenApe VenNom  
Endfor
```

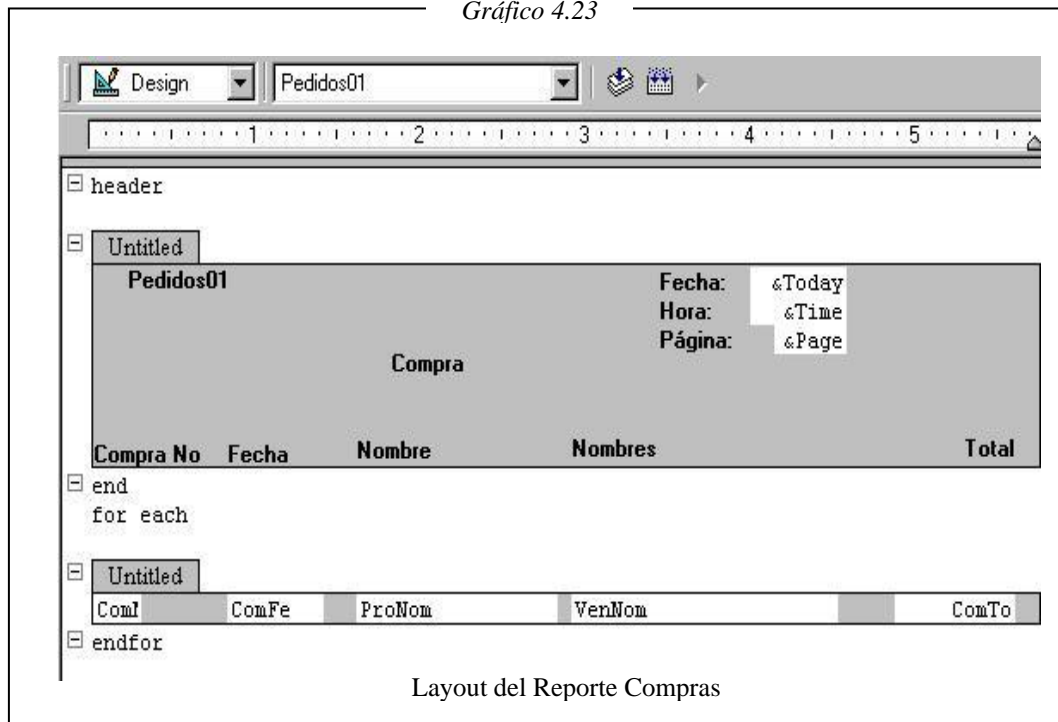
Quiere decir que se imprimirán: el id del vendedor, la cédula, el apellido y el nombre de todos los vendedores existentes.

4.4.3. FOR EACH - ENDFOR

Es uno de los comandos más importante de los reportes, ya que dentro de el se define a toda la estructura del reporte con respecto al acceso a la base de datos.

Con este comando, se leen todos los atributos que Ud. Quiere visualizar desde la base de datos, solamente especificando el atributo que desea, sin necesidad de especificar la tabla de donde provienen los datos, la herramienta automáticamente busca la mejor forma y las tablas de donde sacar los datos; Genexus se encarga de cómo hacerlo, a continuación se realizará un reporte de compras (gráfico 4.23)

Gráfico 4.23

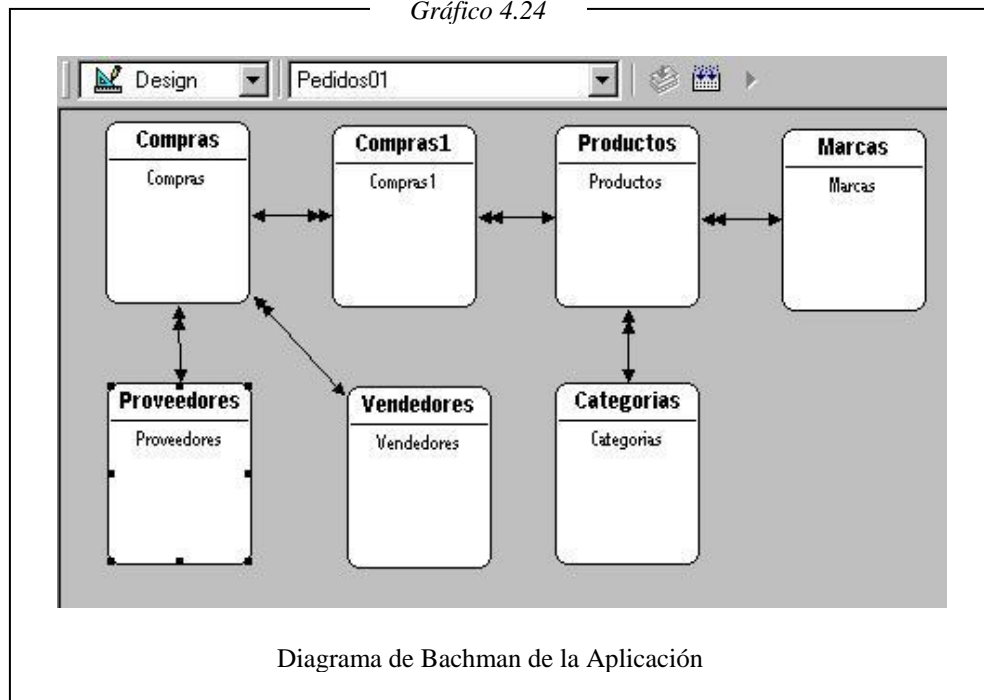


El reporte compras tiene datos de varias tablas: compras, compras1, productos, proveedores y vendedores. En el diagrama de Bachman (gráfico 4.24) se dará cuenta cómo son las relaciones y de donde Genexus saca los datos para publicarlos.

```
For each record in COMPRAS
Find the corresponding ProNom in table PROVEEDORES
Fin the corresponding VenNom in table VENDEDORES
Endfor
```

Cuando se especifica un Reporte Genexus muestra un listado que se llama: Listado de Navegación, donde se indica cuáles son las tablas que se están accediendo.

Gráfico 4.24



En el listado de navegación (gráfico 4.25) la primer tabla representada indica cuál es la tabla base de la tabla extendida y las tablas que figuran debajo de ésta indican las tablas 1-N de la misma.

Una interpretación muy práctica de este diagrama es: para la tabla del primer nivel de indexación se leen muchos registros y para cada tabla del siguiente nivel se lee un solo registro por cada registro leído en la tabla del primer nivel.

- Order ConNo. Es el atributo por el cual se ordena el reporte.
- COMPRAS (ConNo) Tabla que se recorre secuencialmente.
- PROVEEDORES (ProRuc) Tablas donde se accede.
- VENDEDORES (VenId) Tablas donde se accede.

Vertical Formulas. Son las fórmulas verticales que tienen los reportes, en este caso ComST es la fórmula involucrada para ComTo.

Gráfico 4.25

The screenshot displays a configuration window for report navigation. At the top, there is a header 'Levels' and a sub-header 'For Each COMPRAS (Line: 10)'. Below this, the following settings are listed:

- Index: ICOMPRAS
- Order: ComNo
- Navigation filters:
- Start from:
 - FirstRecord
- Loop while:
 - NotEndOfTable

Below these settings, there is a list of tables with their respective primary keys:

- COMPRAS (ComNo)
- PROVEEDORES (ProRuc)
- VENEDORES (VenId)

Further down, under 'Vertical Formulas:', there is a line: **Navigation to evaluate:** ComST

At the bottom, another list of tables is shown:

- COMPRAS (ComNo)
- COMPRAS1 (ComNo)

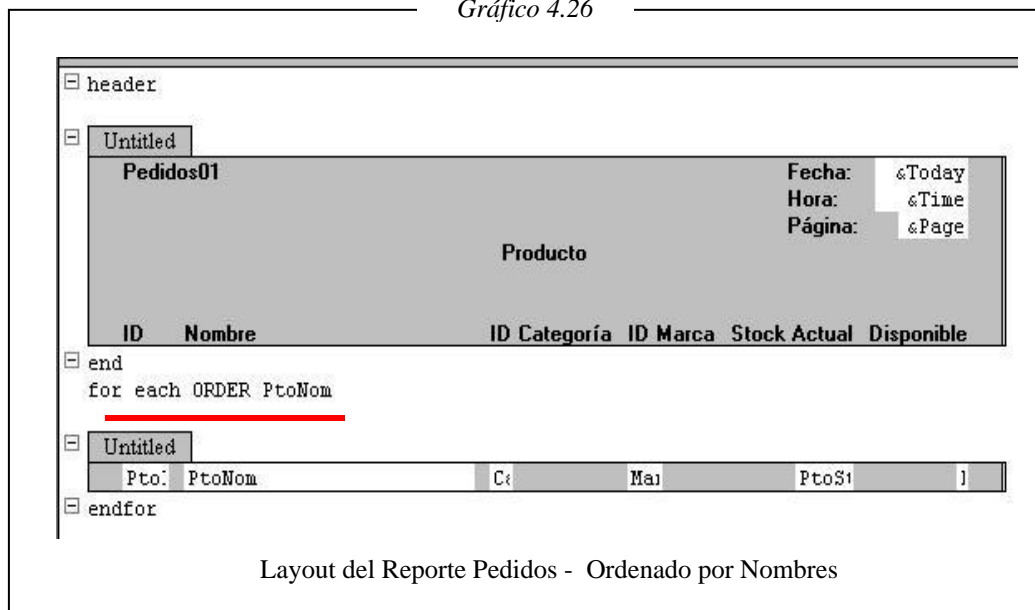
At the very bottom of the window, the text 'Listado de Navegación del Reporte Compras' is displayed.

4.4.4. ORDENAMIENTO DE DATOS ORDER (ÍNDICES)

Los reportes realizados anteriormente fueron elaborados por defecto, sin tomar en cuenta el orden de presentación de los datos o sea de que manera y de acuerdo a que atributo desea ordenar el reporte.

Cuando no se especifica el orden o índice, Genexus automáticamente ordena por medio de la clave primaria de la transacción, como sucedió en los ejemplos anteriores. Para los casos en donde se quiere definir el orden de los datos de forma explícita se utiliza la cláusula ORDER dentro del comando FOR EACH, para indicar este paso se realizará el reporte Productos, ordenado por nombres. (gráfico 4.26)

Gráfico 4.26



En la línea marcada con color rojo se observa: for each ORDER PtoNom Esto significa que este reporte estará ordenado por el nombre del producto, los datos desplegados se observan en el gráfico 4.27.

A continuación se generará otro reporte de productos02, con ORDER de otra tabla extendida CatId o sea se lo ordenará por categorías, también se invocará al nombre de la categoría y la marca, que son atributos que están físicamente aparte de la transacción productos. (gráfico 4.28)

Gráfico 4.27

Pedidos01						Fecha:	04/04/03
						Hora:	12:43:25
						Página:	1
Producto							
ID	Nombre	ID Categoría	ID Marca	Stock Actual	Disponible		
0003	BOMBA CENTRIFUJA RG56	05	01	4	0		
0004	BOMBA CENTRIFUGA RTY48U	05	04	5	0		
0021	BOTAS PETROLERAS PA	08	02	120	0		
0020	BREAQUER 45YH	02	13	10	0		
0013	CARRO WIRE LINE 3000m	12	07	15	0		
0014	CARRO WIRELINE 2000m	12	05	8	0		
0023	CASCOS	08	03	150	0		
0011	CHIVO PERFORACION 2000m	12	06	5	0		
0012	CHIVO WORKOVER	12	05	10	0		
0010	CHIVOS PERFORACION 1000m	12	11	5	0		
0005	COMPRESOR AIRE GR32	04	09	5	0		
0017	CRUDO 24 G	13	03	150	0		
0008	DIAMANTE PERFORACION RT4	09	02	25	0		
0001	GASOLINA SUPER	14	29	30000	0		
0002	GENERADOR TALADRO IDECO56	04	11	4	0		
0024	JAFAS DIA	08	04	520	0		
0025	JAFAS NOCHE	08	04	200	0		
0006	JPI	01	12	4000	0		
0016	LLANTAS CHIVO 18 34X89	11	01	25	0		
0009	LLAVES DE COMBO	10	26	5	0		
0030	MASTIL TALADRO 3000m	02	05	10	0		

Datos del Reporte de Productos – Ordenado por Nombres

En el reporte de productos por categorías, se puede observar el for each ORDER CatId, que es un ordenamiento por medio de la clave foránea de la tabla productos y clave principal de categorías.

De la misma manera se observa a CatNom y MarNom que son los nombres de la categoría y la marca, estos atributos son físicamente distantes de la transacción productos, pero con solo introducirlos en el layout, Genexus automáticamente genera el código para la representación de los datos, incluso el reporte de navegación es muy sencillo. (gráfico 4.29)

Gráfico 4.28

```
header
  Untitled
  Pedidos01
  Fecha: &Today
  Hora: &Time
  Página: &Page
  ID Nombre ID Cate Nombre ID Marca Nombre Stock Actual
end
for each ORDER CatId
  Untitled
  Pto: PtoNom C: CatNom Ma: MarNom Pto$1
endfor
```

Layout de el Reporte Productos 02 - Ordenado por una Clave Foránea CatId

Gráfico 4.29

```
For Each PRODUCTOS (Line: 10)
  Index: IPRODUCTOS2
  Order: CatId
  Navigation filters:
  Start from:
  • FirstRecord
  Loop while:
  • NotEndOfTable
  PRODUCTOS ( PtoId )
    CATEGORIAS ( CatId )
    MARCAS ( MarId )
```

Listado de Navegación de Productos 02

Gráfico 4.30

Pedidos01							Fecha: 05/04/03
<i>REPORTE DE PRODUCTOS POR CATEGORIAS</i>							Hora: 19:13:25
							Página: 1
ID	Nombre	ID Categoría	Nombre	ID Marca	Nombre	Stock Actual	
0006	JPL	01	Aditivos	12	OILWELL	4000	
0007	TRANSMICION IDECO 1000 GT	02	Repuestos	09	CUMMIX	5	
0018	RODAMIENTO BOLAS 34X12GF	02	Repuestos	08	INTERNATIONAL	10	
0019	RODAMIENTO BOLAS 5X47PO	02	Repuestos	07	JOHN DERE	25	
0020	BREAQUER 45YH	02	Repuestos	13	PERKINS	10	
0027	TORRES CHIVO IDECO 1000	02	Repuestos	12	OILWELL	10	
0028	SIGUEÑAL CHIVO IDECO 1000	02	Repuestos	02	HARRISBURG	10	
0030	MASTIL TALADRO 3000m	02	Repuestos	05	TOYOTA	10	
0031	MOTOR PERFORACION ID1000	03	Motores	08	INTERNATIONAL	5	
0032	MOTOR ELECTRICO ID1000	03	Motores	06	MERCEDES	5	
0002	GENERADOR TALADRO IDECO56	04	Generadores	11	IDECO	4	
0005	COMPRESOR AIRE GR32	04	Generadores	09	CUMMIX	5	
0003	BOMBA CENTRIFUJA RG56	05	Bombas I	01	GENRAL	4	
0004	BOMBBA CENTRIFUGA RTY48U	05	Bombas I	04	MISSION	5	
0022	PONCHOS DE AGUA	07	Ropa de Trabajo	03	PEDROLEO	500	
0021	BOTAS PETROLERAS PA	08	Seguridad Per	02	HARRISBURG	120	
0023	CASCOS	08	Seguridad Per	03	PEDROLEO	150	
0024	JAFAS DIA	08	Seguridad Per	04	MISSION	520	
0025	JAFAS NOCHE	08	Seguridad Per	04	MISSION	200	

Datos Desplegados por Productos 02

4.4.5. FOR EACH CON CONDICIONES WHERE

La condición where dentro de un for each, se la utiliza de una manera similar como en sql puro. Esta condición es necesaria para filtrar o restringir los datos, por ejemplo si se quiere obtener un reporte de los productos que liste a los productos cuyo stock mínimo sea igual al stock actual; este reporte se identifica como Producto 03 (gráfico 4.31).

For each

WHERE PtoStoMin = PtoStoAct

Por medio de este for each con condición where se listarán los productos cuyo stock actual sea igual al stock mínimo.

Gráfico 4.31

header

Untitled

Pedidos01

Fecha: &Today
Hora: &Time
Página: &Page

Productos con Stock Bajo

ID	Nombre	Stock Minimo	Stock Actual
----	--------	--------------	--------------

end

for each

WHERE PtoStoMin = PtoStoAct

Untitled

Pto:	PtoNom	PtoS1	PtoS1
------	--------	-------	-------

endfor

Layout del Reporte Productos 03

Productos con Stock Bajo

ID	Nombre	Stock Minimo	Stock Actual
0002	GENERADOR TALADRO IDECO56	4	4
0003	BOMBA CENTRIFUJA RG56	4	4
0004	BONMBA CENTRIFUGA RTY48U	5	5
0005	COMPRESOR AIRE GR32	5	5
0006	JPI	4000	4000
0007	TRANSMICION IDECO 1000 GT	5	5
0009	LLAVES DE COMBO	5	5
0010	CHIVOS PERFORACION 1000m	5	5
0011	CHIVO PERFORACION 2000m	5	5
0018	RODAMIENTO BOLAS 34X12GF	10	10
0020	BREAQUER 45YH	10	10
0027	TORRES CHIVO IDECO 1000	10	10
0030	MASTIL TALADRO 3000m	10	10
0031	MOTOR PERFORACION ID1000	5	5
0032	MOTOR ELECTRICO ID1000	5	5

Datos de Producto 03

Generación de un Reporte con Condiciones: WHERE

En Compra 3 se visualiza la utilización de un for each, tipo AND, que consta de los siguiente:

For each

WHERE ComFe = Today() Fecha de Hoy.

WHERE ProNom = 'DYCMECANIC' Proveedor DYCMECANIC

WHERE ComTo > 1000 La compra total > 1000

Esto quiere decir que liste el reporte de todas las compras del día de hoy, cuyo proveedor sea DYCMECANIC y cuyo valor de la compra total sea mayor que mil.

Para un WHERE con la condicionante OR, no es necesario poner varios WHERE, sino solamente uno con la siguiente estructura. WHERE ComFe = Today **OR** ComFe = 03/04/03. Quiere decir que reporte las compras del día de hoy o del día 03/04/03.

4.4.6 FOR EACH ANIDADOS

Se utilizan para realizar reportes más complejos, donde dentro del reporte pueden ir dos niveles o pueden ir dos tipos de índices. Los reportes anteriores solamente tenían un nivel.

Para entender los for each anidados se realizará un reporte de vendedores que desplegara al vendedor y los pedidos realizados. (gráfico 4.33)

En este reporte se introducen atributos de otras tablas, pero con el concepto de tabla extendida que maneja Genexus es muy fácil de realizarlo, simplemente se añade los atributos deseados en la estructura o layout del reporte.

Cuando se realiza reportes con dos niveles, o con for each anidados, el reporte se lo define a manera de una estructura de la siguiente manera.

VenId

VenNom

VenApel

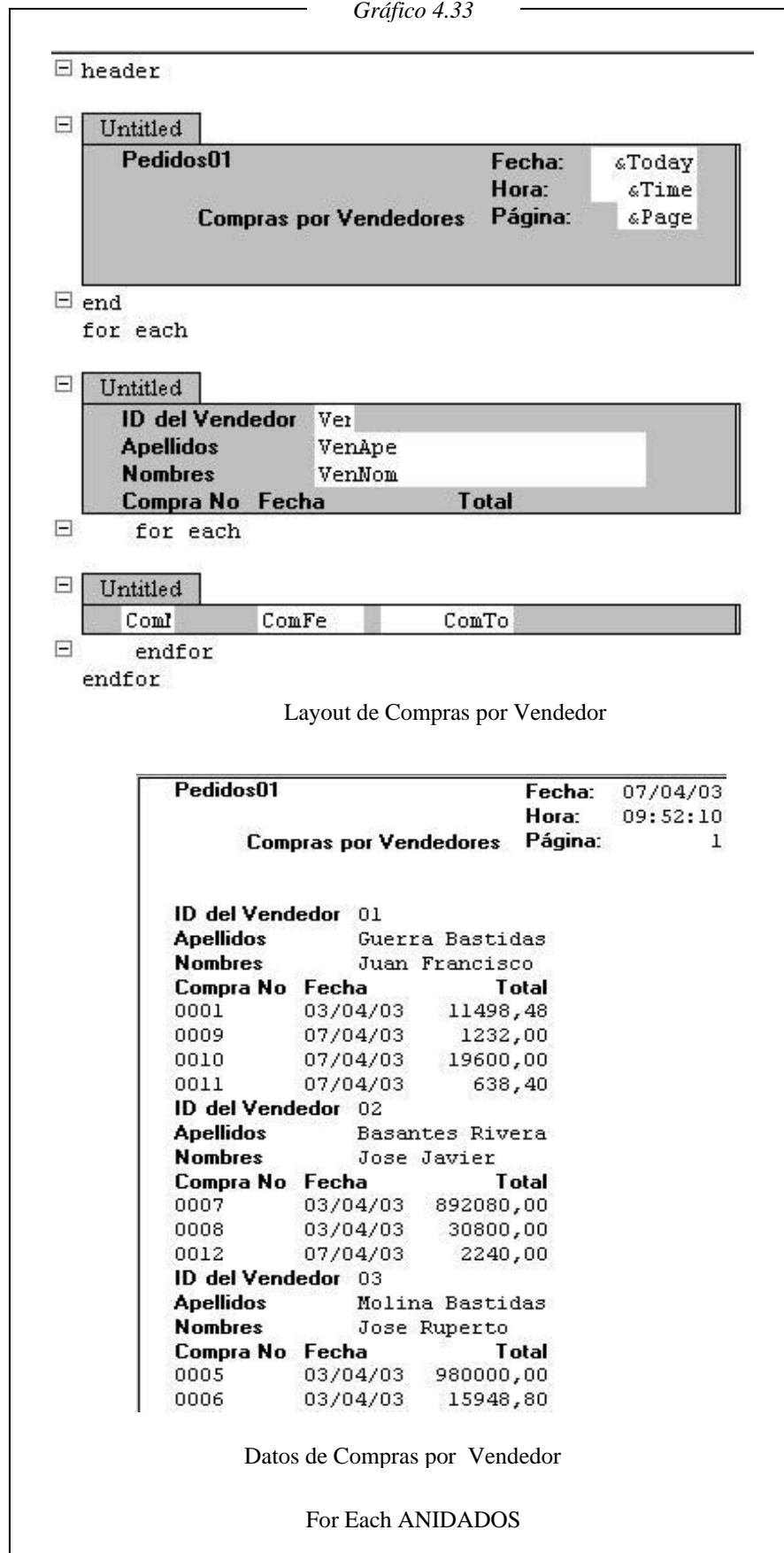
(ComNo

ComFec

ComTo)

No se está definiendo a alterando la estructura de ninguna transacción, simplemente se está definiendo la estructura del reporte; esta no afectará a las transacciones involucradas en el reporte.

Gráfico 4.33



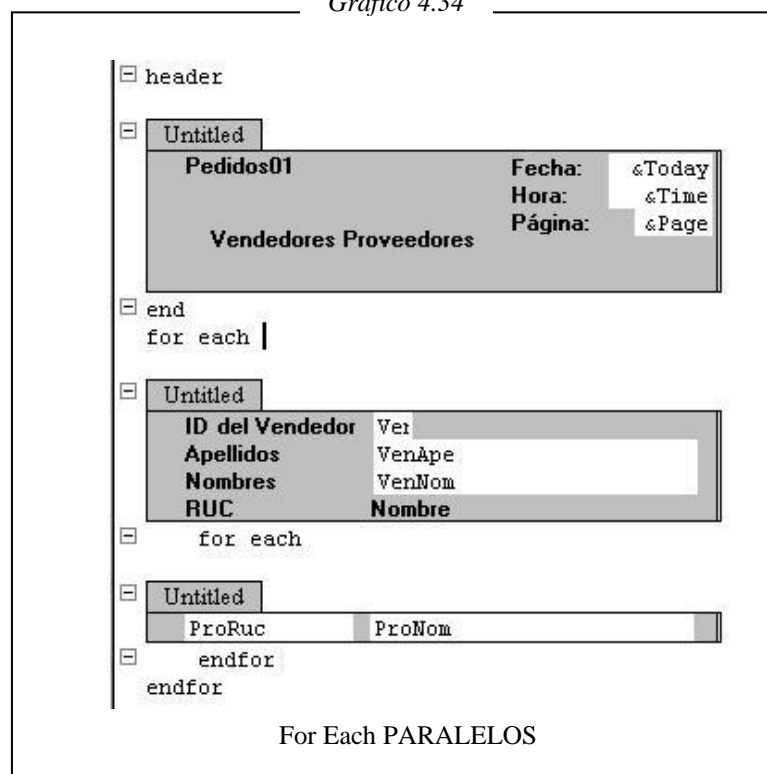
Se nota que existen dos niveles, un nivel para cada for each, de esta manera primero se extrae el Id del vendedor y este a su vez lo valida automáticamente con el id del vendedor de la transacción compras.

Cuando hay dos FOR EACH anidados, Genexus busca cuál es la relación de subordinación existente entre la tabla base del segundo FOR EACH y la tabla extendida del primer FOR EACH, en el caso anterior la relación era que la tabla Compras estaba subordinada a la tabla vendedores por VenId y establece de esta manera la condición que deben cumplir los registros del segundo FOR EACH.

Puede darse el caso de que no exista relación entre ambos FOR EACH, por ejemplo un reporte de vendedores y proveedores, en este caso Genexus hace el producto cartesiano entre los dos for each; a este tipo de for each se les denomina **FOR EACH PARALELOS**.

Para representar for each paralelos se realizará un reporte de proveedores y vendedores, los cuales no tienen ninguna relación entre si (gráfico 4.34).

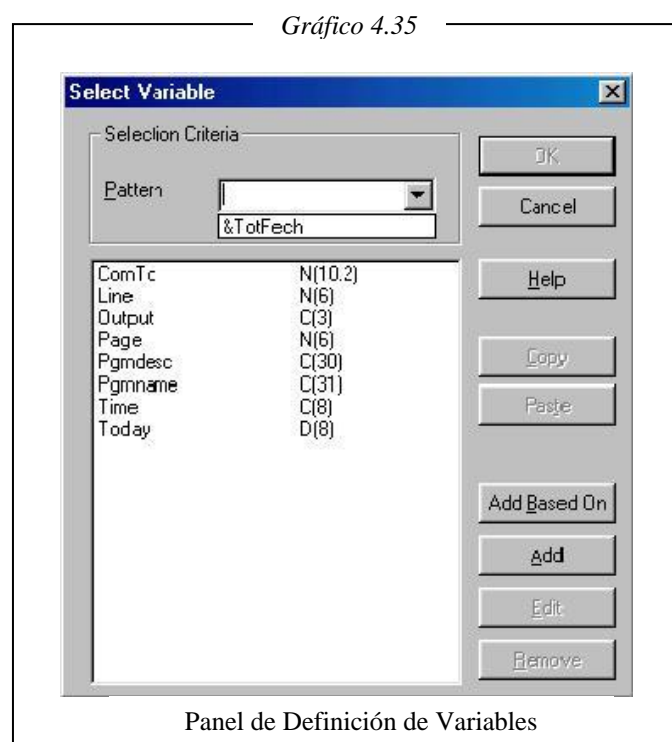
Gráfico 4.34



4.4.7. DEFINICIÓN DE VARIABLES

A veces es necesario utilizar atributos que no están almacenados en la base de datos, estos atributos que no están en la base de datos se los conoce como variables; son de orden local, o sea solo del objeto en las cuales se las define. Las variables están acompañadas de el símbolo (&) y su sintaxis es &nombre.

Genexus tiene algunas variables predefinidas, es decir que ya tienen un significado propio, como: &Today que es una variable con que saca la fecha de hoy. Otro ejemplo es la variable &Page que indica cuál es la página que se está imprimiendo. Las variables se pueden definir también en función de otros atributos, usando la opción de Based On (gráfico 4.35).



Para indicar cómo se insertan variables en un reporte, se realizará el reporte compras por fecha, y a cada registro se le añadirá el total de las compras de cada día, el total de compras será la variable insertada (gráfico 4.36).

Gráfico 4.36

```

end
for each order ComFe
  &SubTot = 0
  Untitled
  Fecha ComFe
  Compra No Nombre Total
  for each order ComNo
    &SubTot = &SubTot + ComTo
  Untitled
  ComI ProNom ComTo
endfor
Total
Total &SubTot
endfor

```

Reporte Generado con Variables. (&SubTot)

La variable (&SubTot) es una variable de tipo precios, esta suma a todos los ComTo (total de cada compra) realizada cada día. Cave recalcar que &SubTot no se almacena en la base de datos, porque es local solo para el reporte.

Este tipo de variables se utiliza para realizar cierres, tanto diarios mensuales anuales, etc, depende del criterio del analista.

Para tener una idea más clara de lo que se quiso realizar se va a desplegar los datos del reporte realizado (gráfico 4.37).

La manera como Genexus realiza reportes, se la puede considerar como un generador automático de consultas.

Gráfico 4.37

Pedidos01			Fecha: 08/04/03
			Hora: 11:14:19
Compras por Fecha			Página: 1
Fecha 03/04/03			
Compra No	Nombre		Total
0001	DYCMECANIC		11498,48
0002	YPF		327600,00
0003	DYCMECANIC		52682,56
0004	LG		2008160,00
0005	REPSOL		980000,00
0006	CASA VACA		15948,80
0007	CERAMIN		892080,00
0008	REPSOL		30800,00
			Total 4318769,84
Fecha 07/04/03			
Compra No	Nombre		Total
0009	DYCMECANIC		1232,00
0010	DYCMECANIC		19600,00
0011	DYCMECANIC		638,40
0012	YPF		2240,00
			Total 23710,40

Listado de Compras por Fecha (Con Variables)

En el diseño de reportes existen también *Comandos de Control*, los comandos de control son IF-ELSE-ENDIF y DO WHILE-ENDDDO.

El comando IF-ELSE-ENDIF es muy utilizado para impresión condicional, es decir, imprimir una línea solo cuando se cumple alguna condición. El comando DO WHILE-ENDDDO es muy usado para imprimir varias líneas, por ejemplo si se quiere imprimir dos líneas del reporte.

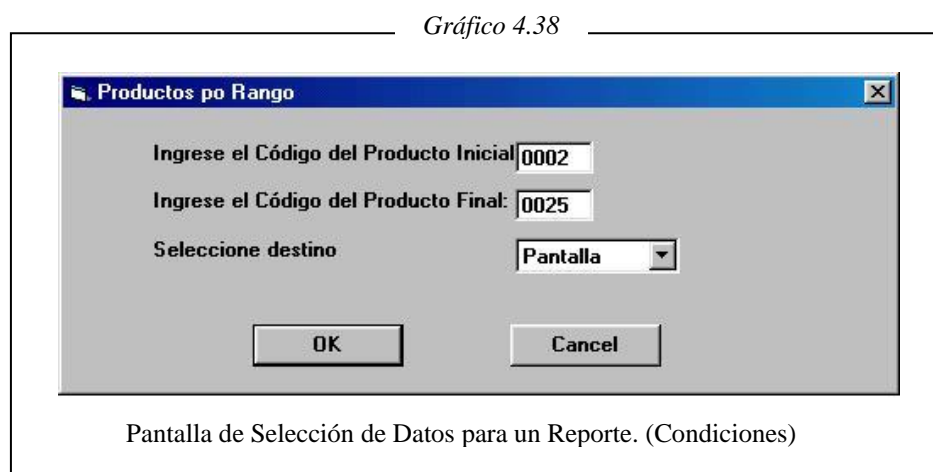
4.4.8. CONDICIONES EN UN REPORTE (CONDITIONS)

En esta parte se definen las condiciones que se quieren imponer a los datos, es equivalente a la cláusula WHERE del comando FOR EACH, incluso tienen la misma sintaxis, con la diferencia que el WHERE se aplica al FOR EACH en el que se encuentran definidas y las condiciones definidas aquí se aplicarán a todo el reporte. En el Listado de Navegación se indica a qué FOR EACH se están aplicando.

Muchas veces se requiere que las condiciones no sean valores fijos, sino que el usuario elija los valores cuando se ejecuta el reporte. Por ejemplo, se quiere listar solo algunos productos que el usuario seleccione antes de ejecutar el reporte productos; esto se hace de la siguiente manera.

```
PtoId >= ask( 'Ingrese el Código del Producto Inicial: ' );  
PtoId <= ask( 'Ingrese el Código del Producto Final : ' );
```

En una vez especificado estas condiciones, aparecerá una pantalla de la siguiente manera (gráfico 4.38).

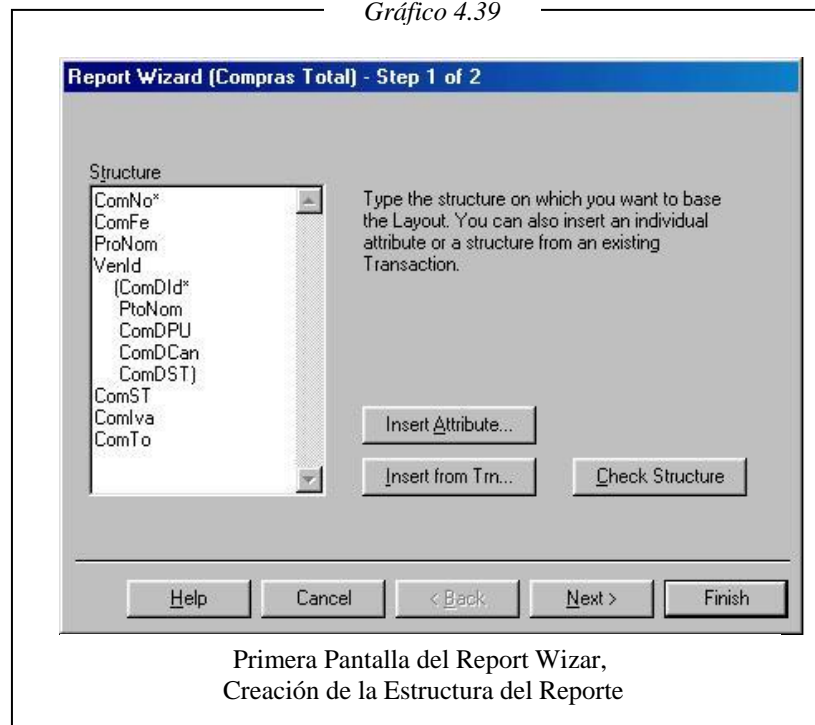


4.4.9. REPORTES GENERADOS CON WIZARD

Los reportes generados por medio de report Wizard, son objetos realizados siguiendo una ayuda propia de Genexus. Es la manera más fácil y práctica de realizar un reporte. El diseño del layout consiste en dos pasos.

Primero, se necesita crear una estructura de datos similar a la estructura de una transacción, algunos atributos estarán marcados con un (*) que no significa que son claves primarias, si no se está especificando el orden de despliegue de los datos. (gráfico 4.39) En este caso se realizará un reporte de compras pero generado con wizard.

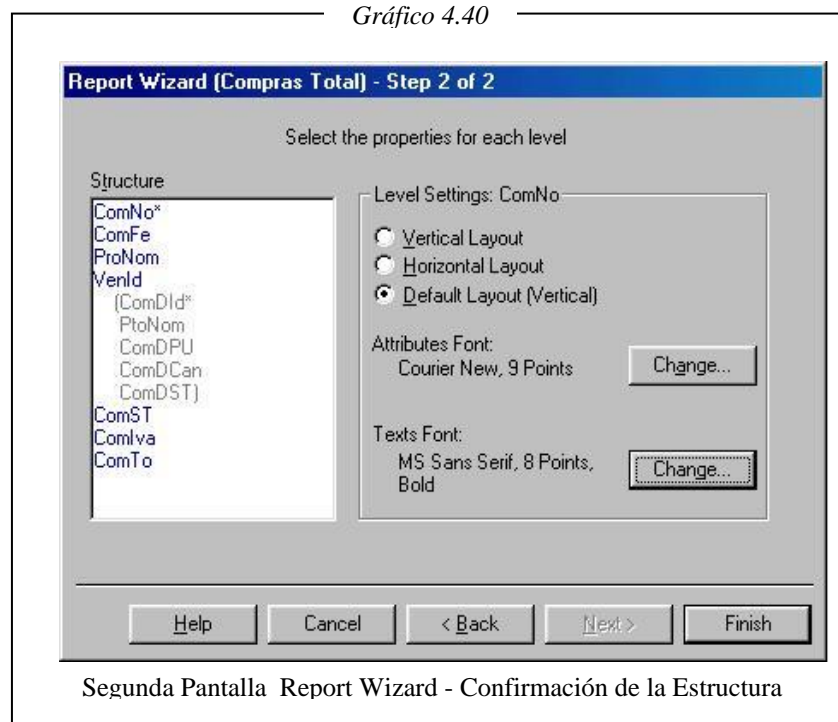
Gráfico 4.39



El primer nivel del reporte se va a ordenar por ComNo y el segundo nivel por ComDId. La estructura de cualquier reporte nunca modifica la estructura de la transacción.

El segundo paso simplemente sirve para modificar la forma del reporte, como: fuente, tipo, tamaño y el despliegue de datos: vertical, horizontal o por defecto. (gráfico 4.40)

Gráfico 4.40



4.5. DISEÑO DE PROCEDIMIENTOS

Los procedimientos son procesos no interactivos de actualización de la base de datos; para ello es necesario conocer algunos comandos, que permiten modificar los datos.

4.5.1. MODIFICACIÓN DE DATOS

La modificación de datos se realiza de forma implícita, ya que no existe un comando específico de modificación como por ejemplo el REWRITE, sino que basta con asignar un valor a un atributo dentro de un For Each para que automáticamente se realice la modificación.

Por ejemplo se realizó un contrato considerable para proveer de suministros petroleros a una compañía internacional, para complacer de una manera eficaz a estos clientes y para no tener problemas de abastecimiento, es necesario aumentar en 5 unidades el stock mínimo de cada producto. Para esto se debe actualizar el campo (PtoStoMin) de la transacción productos; la modificación en se realiza en el ENDFOR siguiente manera (gráfico 4.41).

```
Programa:  
For each  
PtoStoMin = PtoStoMin + (1 * &stok)  
Endfor  
Reglas:  
Parm( &stok );
```

No todos los atributos pueden ser modificados. No se pueden modificar los atributos pertenecientes a la clave primaria de la tabla base del FOR EACH, tampoco se pueden modificar los atributos que pertenecen al índice. (orden) con el que se está accediendo a la tabla base.

En el Listado de Navegación se informa cuáles son las tablas que se están actualizando marcando con un símbolo correspondiente a un lápiz (Grafico 4.42). La tabla a ser modificada es: productos, y el atributo es PtoStoMin.

Gráfico 4.41

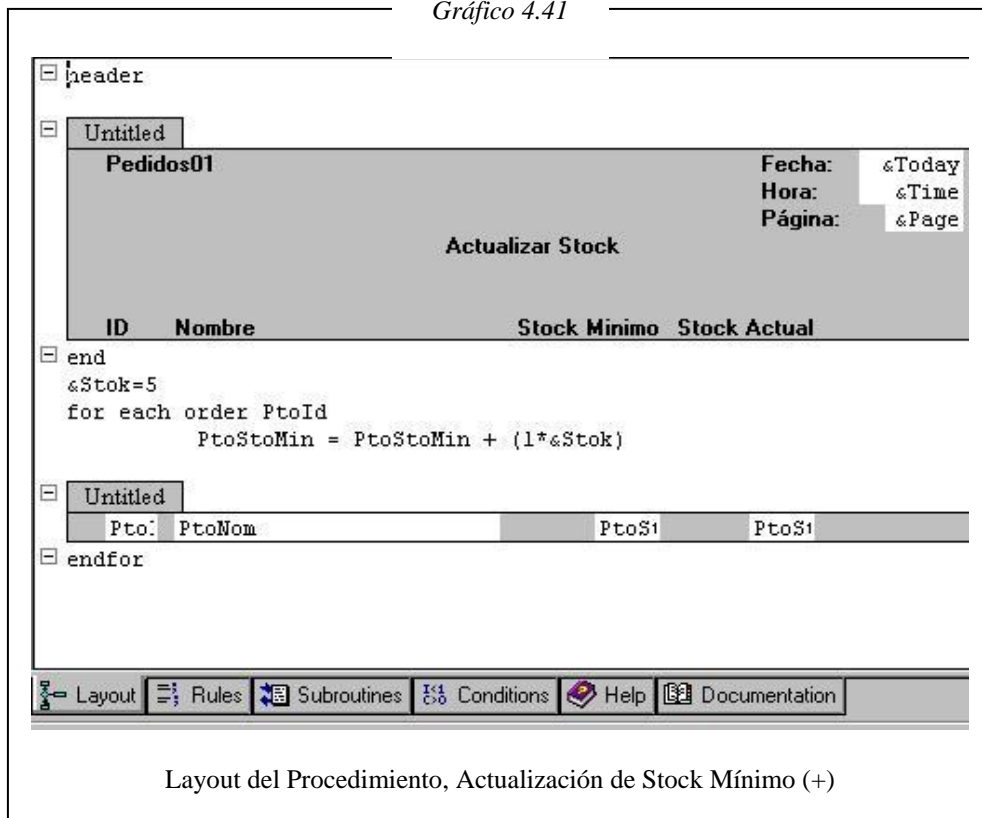


Gráfico 4.42



Si se quiere rebajar el stock mínimo de un rango de productos especificado por el usuario, solamente se cambia a fórmula y se pone condiciones así: (gráfico 4.43)

Gráfico 4.43

```
header
  Pedidos01
  Fecha: &Today
  Hora: &Time
  Página: &Page
  Disminuir el Stock Mínimo
  ID  Nombre  Stock Mínimo  Stock Actual
end
  &Stoc2 = 4
  for each order PtoId
    PtoStoMin = PtoStoMin - (1*&Stoc2)
  endfor
```

Layout del Procedimiento para Actualizar el Stock Mínimo (-)
Con Condiciones

Se está definiendo la variable &Stoc2, la cual tiene un valor de (4), o sea se va a restar 4 unidades del stock mínimo, así: $PtoStoMin = PtoStoMin - (1 * \&Stoc2)$. Las condiciones se las define de la siguiente manera.

```
PtoId >= ask("Ingrese el Producto Inicial:");
PtoId <= ask("Ingrese el Producto Final:");
```

Los procedimientos en la vida real se los observa como cualquier objeto Genexus,; constan en las ventanas por defecto de las aplicaciones. (gráfico 4.44)

Gráfico 4.44

Vista en Forma Real de los Objetos Genexus. (Procedimientos)

4.6. DISEÑO DE PANELES DE TRABAJO (WORK PANELS)

Los paneles de trabajo o work panels sirven para realizar consultas interactivas a la base de datos, son generalmente utilizados por las personas que suelen tomar decisiones en las que están incluidos los datos de la base.

Genexus genera automáticamente work panels para cada nivel de una transacción diseñada, estos se los utiliza como listas de selección de datos cuando existen claves foráneas en la transacción que se diseño y son conocidos como Promps.

La metodología de los work panels está enfocada a la programación de ambientes gráficos dirigidos por eventos. En cada work panel se definen las siguientes partes:

4.6.1 DEFINICION DE OBJETOS DE UN WORK PANEL

INFORMACIÓN: Son los datos generales del panel de trabajo, por ejemplo: nombre, descripción, tipo de letra, fuente, etc.

FORM: Es la pantalla de presentación en la cual va a interactuar el usuario, son similares a las de las transacciones.

EVENTOS: Aquí se define la respuesta a los eventos que pueden ocurrir durante la ejecución del Panel de Trabajo. Por ejemplo, qué respuesta dar cuando el usuario presionó: enter, un botón, etc.

CONDICIONES: Definen el comportamiento de los datos, en si qué condiciones debe cumplir cierto dato para ser desplegado. Son equivalentes a las Condiciones de los Reportes y Procedimientos.

REGLAS: Definen aspectos generales del panel de trabajo, por ejemplo: orden de los datos a mostrar, parámetros, etc.

AYUDA: Texto para la ayuda a los usuarios del panel de trabajo.

DOCUMENTACIÓN: Es el texto técnico para ser utilizado en la documentación del sistema.

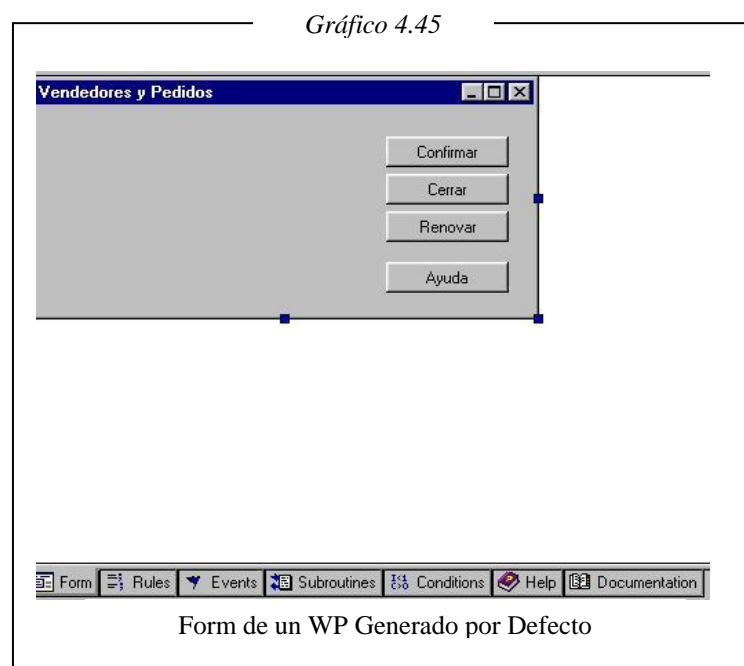
PROPIEDADES: Propiedades generales del panel de trabajo.

4.6.2. PAZOS PARA REALIZAR UN PANEL DE TRABAJO CON WIZARD

Para entender mejor la función que hace un work panel, se realizará un WP de Vendedores, el cual al desplegar su información y señalando un registro de proveedores, se visualizará con algún botón la información de las compras realizadas por el vendedor.

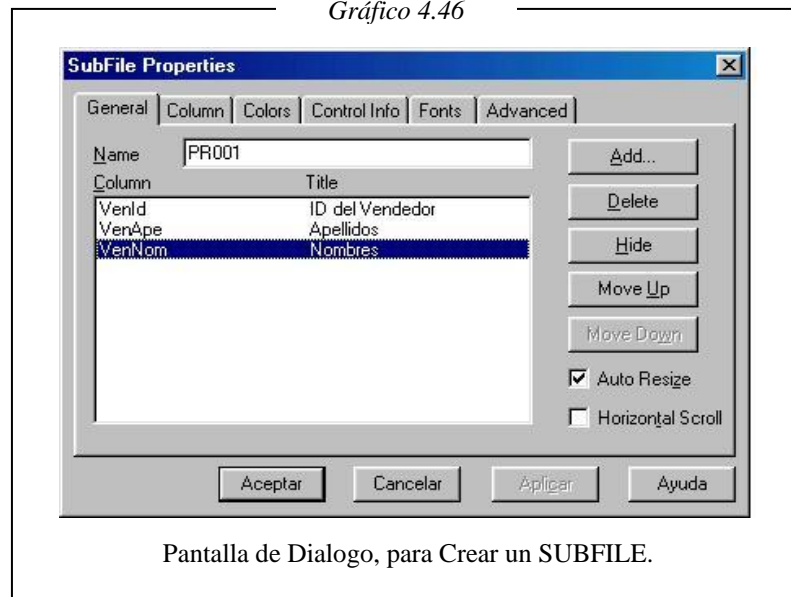
Primeramente se debe insertar un nuevo objeto Genexus (Work Panel), y a continuación aparecerá una pantalla en la que va la información del objeto como: nombre, descripción, se pone siguiente y le saldrá una pantalla donde van los atributos del WP (gráfico 4.45)

Aquí usted puede ingresar los atributos, o simplemente finaliza la elaboración de WP para luego insertar un SUBFILE, (gráfico 4.46) que viene a ser un especie de tabla, donde se desplegarán los datos como una hoja de excel, para desde aquí ser señalados y con alguna otra acción van a ser modificados o simplemente desplegados. El form y los datos generados por este WP con un subfile incluidos se los visualiza en los gráfico 4.47 y 4.48.



En esta pantalla (gráfico 4.45) se inserta los atributos manualmente o se inserta una tabla al estilo subfile.

Gráfico 4.46



Con el diálogo que se realiza para crear un subfile se puede introducir los atributos que se quiere desplegar en la tabla, en este caso son: VenId, VenApel y Ven Nom.

Gráfico 4.47

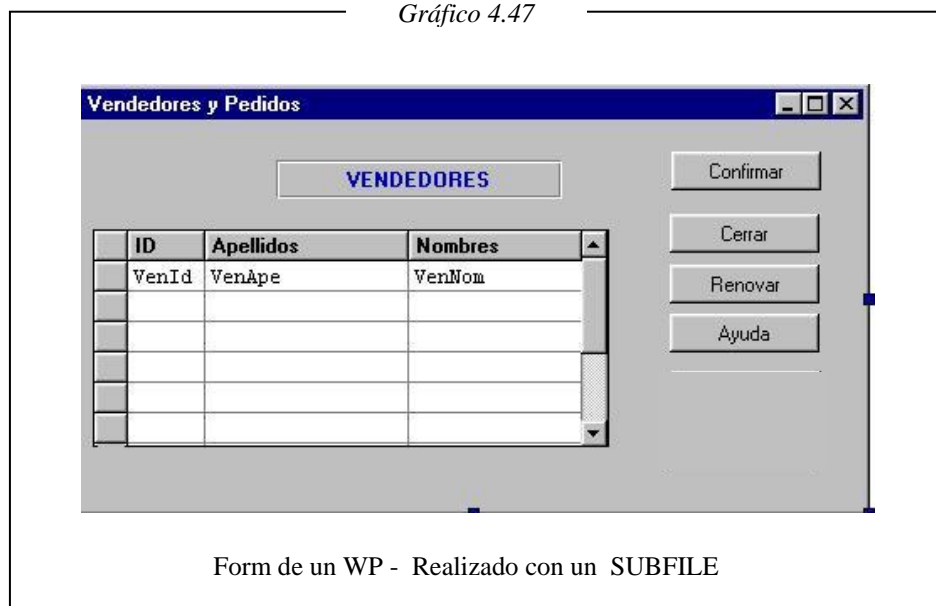
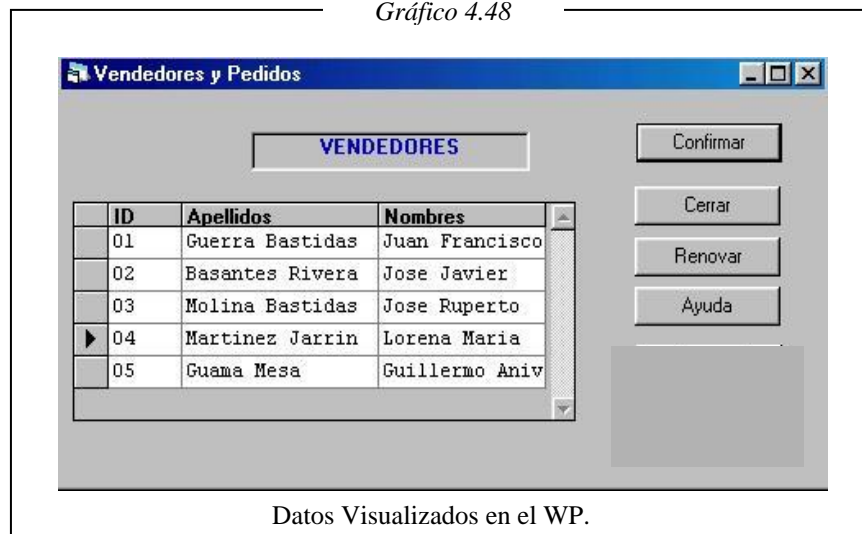


Gráfico 4.48



Como se puede observar a simple vista este WP, pareciera una consulta común y corriente. Para que no se lo vea de esta manera falta insertar algún botón que despliegue toda la información del proveedor, o las compras realizadas por cada vendedor señalando a uno de los vendedores que están desplegados en el work panel.

Para esto se creará dos botones: Compras que desplegara las compras por vendedor llamando al work panel de compras y Personal, que desplegara toda la información de cada proveedor, igualmente llamando a un WP Proveedor.

Primeramente, se creará los dos WP que van a ser llamados por medio de eventos, desde el primer WP creado. (gráfico 4.49 y 4.50)

Gráfico 4.49

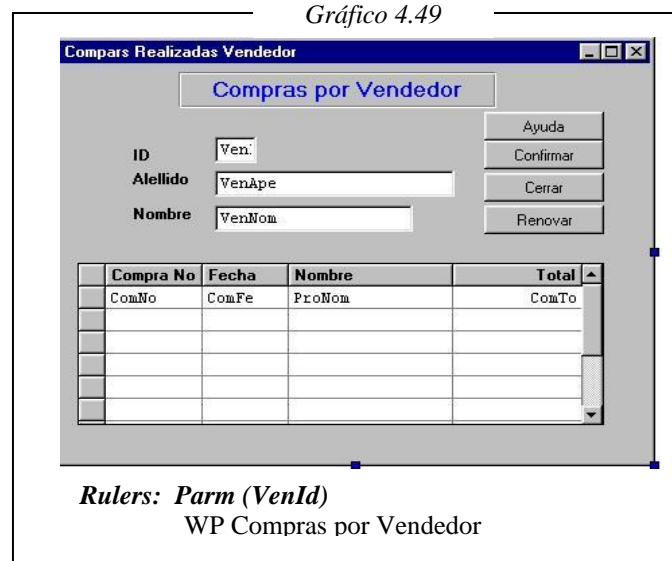


Gráfico 4.50

Datos del Vendedor

ID Ven: Cédula VenCed

Apellidos VenApe Nombres VenNom

Dirección VenDir

Teléfono VenTel VenCel

Confirmar
Cerrar
Renovar
Ayuda

Rulers: Parm (VenId)
WP Datos del Vendedor

Se observará que los dos WP tienen o llevan la regla: Parm(VenId), esto quiere decir que recibirán la clave principal de la transacción vendedores como parámetro, o sea recibirán esa información del vendedor seleccionado en el primer WP.

El primer WP que se realizó, tiene que tener los dos botones adicionales que se mencionó anteriormente: Personal y Compras (gráfico 4.51)

Gráfico 4.51

Vendedores y Pedidos

VENDEDORES

ID	Apellidos	Nombres
01	Guerra Bastidas	Juan Francisco
02	Basantes Rivera	Jose Javier
03	Molina Bastidas	Jose Ruperto
04	Martinez Jarrin	Lorena Maria
05	Guama Mesa	Guillermo Aniv

Confirmar
Cerrar
Renovar
Ayuda
Personal
Compras

EVENTOS:

```

Event 'Personal'
  call(WVendedor, VenId)
EndEvent // 'Personal'

Event 'Compras'
  call(WComVen, VenId)
EndEvent // 'Compras'
    
```

WP. Vendedor - Realizado con EVENTOS.

Cada botón tiene asociado un evento (Enter), los botones asignados a este WP son: Personal y Compras, los cuales llaman a otros WPs que reciben como parámetros en las reglas las claves primarias de la transacción vendedores. El código del evento personal es:

```
Event 'Personal'  
call(WVendedor, VenId)  
EndEvent // 'Personal'
```

Las propiedades del evento son:

Event 'Personal' es el nombre del evento.

Call (Wvendedor, VenId) Está llamando por medio de un Call al WP Vendedor, es por eso que se escribe Wvendedor.

VenId es el parámetro que recibe el WP al seleccionar un registro.

Como son eventos Enter se van a disparar cuando el usuario presione enter..

Para demostrar cómo funciona este ejemplo, se empezará señalando a un vendedor del primer WP. Se marcará al vendedor ID = 01 que es Guerra Bastidas Juan Francisco (gráfico 4.52) luego se seleccionará a las opciones, Personal y Compras, (gráfico 4.53) las cuales despliegan la información y las compras realizadas por el vendedor 01.

Gráfico 4.52

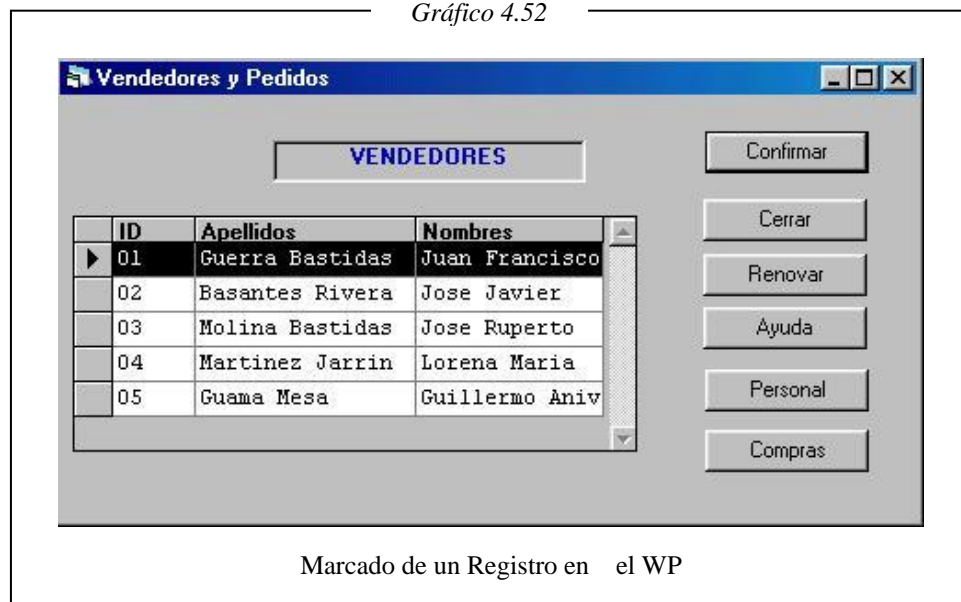


Gráfico 4.53

Datos del Vendedor

ID: 01 Cédula: 0401194865000

Apellidos: Guerra Bastidas Nombres: Juan Francisco

Dirección: Los Ceibos Casa25 Man12

Teléfono: 006643273 098113202

Botones: Confirmar, Cerrar, Renovar, Ayuda

Datos Desplegados del Evento: **PERSONAL**
Datos del Vendedor

Compras Realizadas Vendedor

Compras por Vendedor

ID: 01 Alelido: Guerra Bastidas Nombre: Juan Francisco

Compra No	Fecha	Nombre	Total
0001	03/04/03	DYCMECANIC	11499,04
0009	07/04/03	DYCMECANIC	1232,00
0010	07/04/03	DYCMECANIC	19600,00
0011	07/04/03	DYCMECANIC	638,40
0015	08/04/03	LG	252000,00
0017	12/04/03	LG	514080,00

Botones: Ayuda, Confirmar, Cerrar, Renovar

Datos Desplegados del Evento: **COMPRAS**
Compras por Vendedor

Objetos Desplegados al Presionar Algunos Botones del WP Principal
WP con Eventos

Los WP no solamente pueden llamar a wps, sino a: transacciones, reportes, web panels, etc. En los ejemplos que se realizó anteriormente de un WP se llamó simplemente a otro wp, estos pueden crecer, ya que de vendedores se puede llamar a la transacción vendedores, o de compras o un reporte que envíe como parámetro el ID de la compra y con este liste los productos, etc.

4.6.3 EVENTOS EN LOS PANELES DE TRABAJO.

La forma tradicional de programar la interfaz entre el usuario y el computador, incluidos los lenguajes de cuarta generación 4GL es subordinar la pantalla al programa.

O sea desde el programa se hacen *calls* o llamadas a la pantalla. En Paneles de Trabajo es lo contrario: el programa está subordinado a la pantalla, ya que una pantalla realiza 'calls' al programa para dar respuesta a un evento.

Esta forma de trabajar es conocida como *EVENT DRIVEN* (dirigida por eventos) y es típica de los ambientes GUI (Interfaz de Usuario Gráfica), en donde ya se ha demostrado su productividad, fundamentalmente porque se necesita programar sólo la respuesta a los eventos y no la tarea repetitiva que implica el manejo de toda la interfaz.

Para programar cada uno de los eventos se utiliza el mismo lenguaje que en los Reportes y Procedimientos, aunque no están permitidos ni los comandos relacionados con la impresión (Header, Footer, etc.) ni los de actualización de la base de datos (New, Delete, etc.).

También existen comandos específicos para Paneles de Trabajo, como son el comando refresh, el comando load, etc. La estructura de los eventos se la puede observar en el gráfico 4.54.

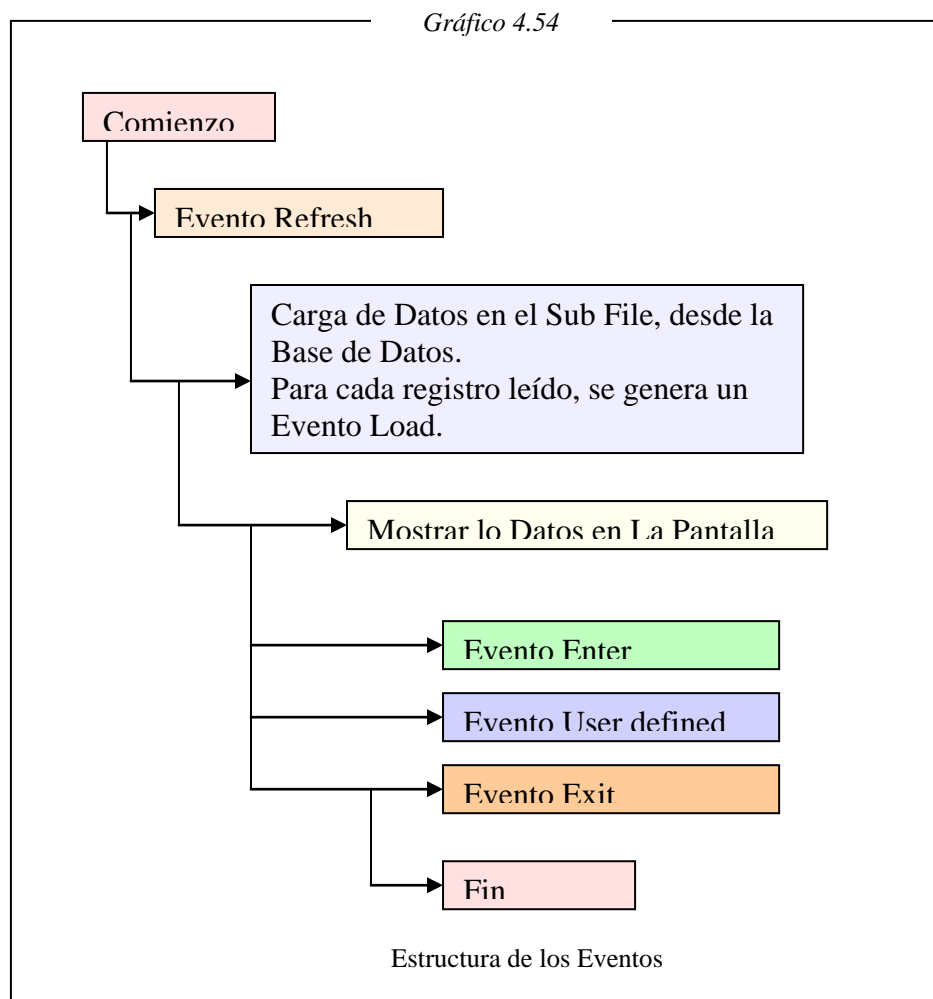
Eventos Definidos por el Usuario. (User Defined)

```
Event 'Insertar'  
Call(Tvendedor, INS, 0)  
Refresh  
EndEvent // 'Insertar'
```

Este es un evento definido por el usuario, para el cuál se debe definir el nombre ('Insertar'). En este evento se llama a la transacción de Vendedores.

Luego de llamar a la transacción TVendedor, por medio de un (CALL) y después se ejecuta el comando Refresh, que indica que se debe cargar nuevamente el SubFile, ya que se ha adicionado un nuevo proveedor.

También se podría haber usado el comando con el parámetro keep (Refresh Keep) que hace que una vez que el comando refresh se ejecute, el cursor quede posicionado en la misma línea del subfile en la que estaba antes de la ejecución.



EVENTO REFRESH. Los datos que se presentan en pantalla son los cargados en el SubFile, que a su vez fueron cargados desde la base de datos. En un ambiente multi-usuario, los datos de una pantalla pueden haber quedado desactualizados si desde otra terminal fueron modificados.

Cuando el usuario desea actualizar los datos, debe dar clic sobre el botón Refresh (“Renovar”), o presionar la tecla F5. El efecto es que se carga nuevamente el SubFile con información actualizada.

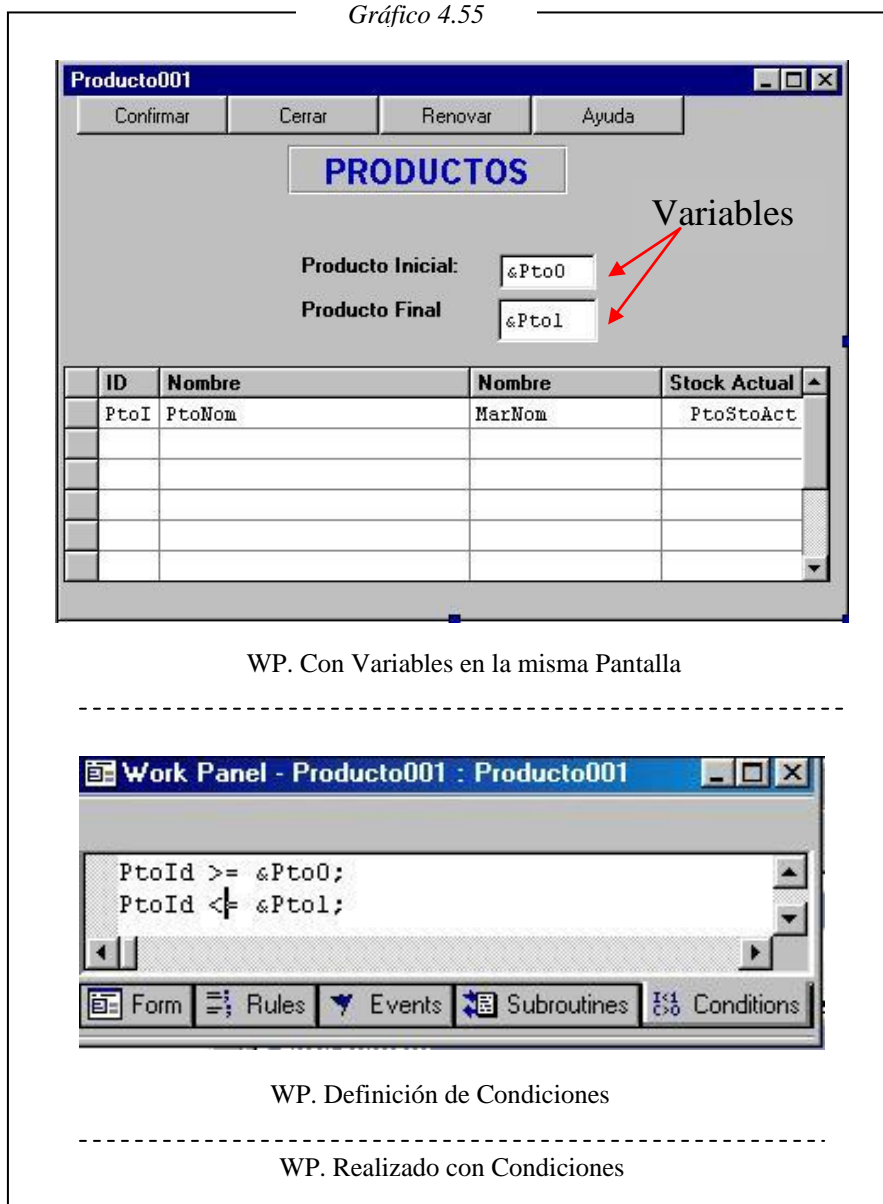
En algunos casos desde otro evento también se necesita cargar nuevamente el SubFile. Por ejemplo, cuando en otro evento se llama a una transacción que actualiza los datos, como sucede con el evento insertar, aquí el SubFile se carga cuando se genera el evento, y refresh se genera cuando:

- Se carga la pantalla, por lo tanto después del evento Start siempre hay un evento Refresh.
- El usuario hizo clic sobre el botón Refresh, o presionó la tecla F5.
- Se ejecutó el comando Refresh en otro evento.
- El usuario modificó el valor de alguna de las variables de la parte fija del subfile, que son utilizadas como condiciones sobre los datos a cargar.

4.6.4 CONDICIONES Y REGLAS DE LOS PANELES DE TRABAJO

Aquí se definen las restricciones de los datos de la misma forma que en los Reportes. Lo interesante de las condiciones en los paneles de trabajo, es que en las mismas pantallas del WP se pueden visualizar las variables de las condiciones, de tal manera que el usuario, cambiando los valores de estas variables, cambia los datos que se muestran en el SubFile sin tener que salir de la pantalla. Por ejemplo, si se quiere visualizar sólo un rango de productos, agrega las variables en la misma pantalla (gráfico 4.55)

Gráfico 4.55



REGLAS:

ORDER: Define cuál es el orden de los datos del SubFile. Es equivalente a la cláusula ORDER del FOR EACH y se aplica de la misma manera. En estos casos se aplica índices temporales, Por ejemplo, si se quiere ver los productos por orden alfabético, se define la regla: Order (PtoNom);

NOACCEPT: A diferencia de las transacciones, en los paneles de trabajo ningún atributo es aceptado, siempre se muestra su valor pero no se permite modificarlo. Por ejemplo, en un WP en el que se desea poner la fecha en el form, se escribe la regla: noaccept(&Fecha);

SEARCH: Cuando el SubFile es demasiado extenso, muchas veces se quiere brindar la posibilidad de que el usuario pueda “posicionarse” en alguna línea determinada del mismo en forma directa, sin tener que avanzar página por página. Por ejemplo, si en el WP se quiere que exista la posibilidad de buscar productos según su nombre, se debe definir la regla: search(PtoNom >= &Nom).

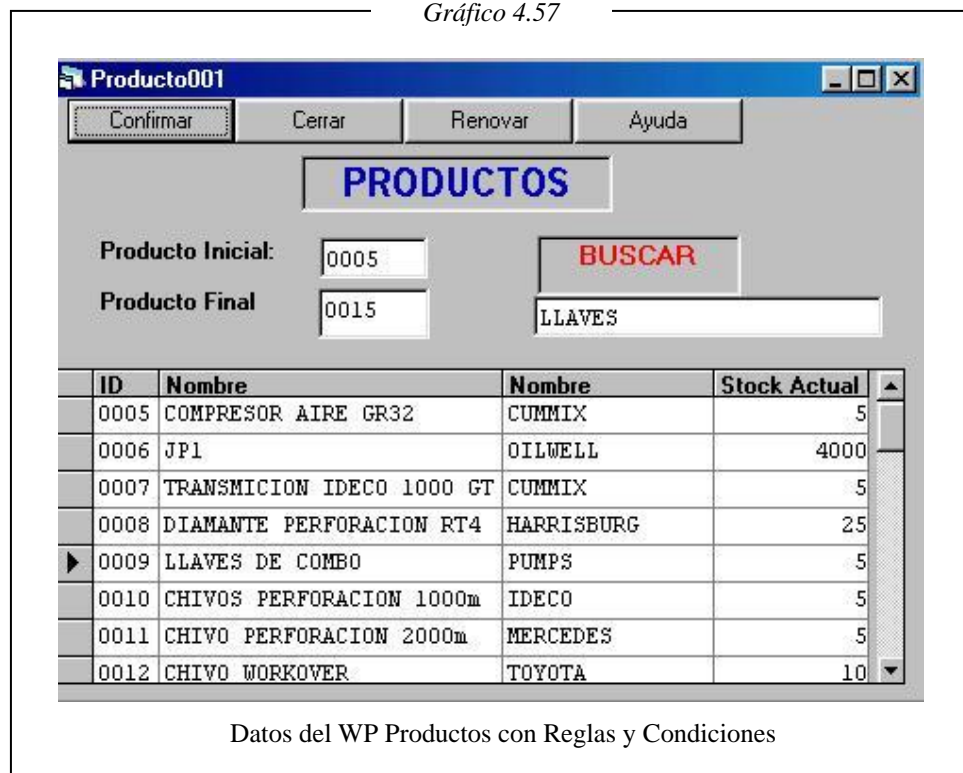
De esta manera cada vez que el usuario digite algo en &Nom, luego de dar enter, el cursor quedará posicionado en el primer proveedor con PrvNom mayor o igual al digitado. Cuando hay muchos nombres la regla es: search(PrvNom LIKE &Nom) (gráfico 4.56)

Gráfico 4.56

ID	Nombre	Nombre	Stock Actual
PtoI	PtoNom	MarNom	PtoStoAct

Form del WP Productos - Regla SEARCH (Buscar)

Gráfico 4.57



El gráfico 4.57 muestra cómo actúan las condiciones y los search. Primero se digita los productos que uno quiere ver, en este caso desde el 0005 hasta el 0015, luego en el cuadro de search (búsqueda) se escribe el nombre de el producto donde se quiere posesionar, pero que este dentro del rango de búsqueda, entonces se digitó (llaves) y el cursor se posesionó en el registro 0009 que tiene como nombre llaves de combo.

4.7. **GRAFICACIÓN DE DATOS**

Los gerentes, directores, o personas encargadas de la toma de decisiones en las empresas, suelen utilizar más a menudo los gráficos de los datos.

Para aprender a graficar un determinado grupo de datos, se realizará un Work Panel en el que consten el Id del vendedor y el total de compras realizadas, este va a tener un evento 'graficar' el cual se encargará de la graficación de los datos.

Primeramente se reestructurará la transacción de vendedores, aumentando un atributo (VenPedTotT) pedidos totales del vendedor; este atributo tiene la fórmula:

$VenPedTot = SUM(ComTo)$ que suma todos los pedidos realizados por cada vendedor (gráfico 4.58) cabe recalcar que al modificar la transacción vendedores, se modificará la base de datos, esto se realiza automáticamente, pero no se borran los datos almacenados anteriormente.

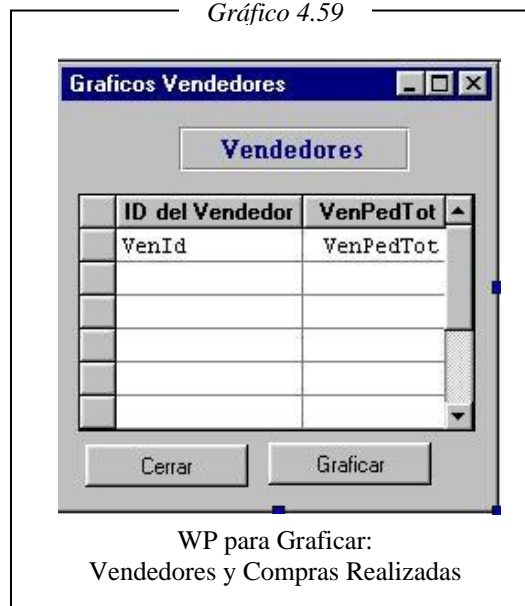
Gráfico 4.58

Structure	Type	Description	Formula
- Vendedores			
• VenId	C(3)	ID del Vendedor	
• VenCed	C(13)	Cédula	
• VenApe	C(25)	Apellidos	
• VenNom	C(25)	Nombres	
• VenDir	C(30)	Dirección	
• VenTel	C(9)	Teléfono	
• VenCel	C(9)	Celular	
<i>fx</i> VenPedTot	N(10.2)	VenPedTot	SUM (ComTo)

Estructura de la Transacción Vendedores. (Modificada)

Una vez realizado este proceso, se creará un WP (GraVen) Graficar Vendedores, el cual consta de un sub file con: VenId y VenPedTot de la transacción vendedores: (gráfico 4.59)

Gráfico 4.59

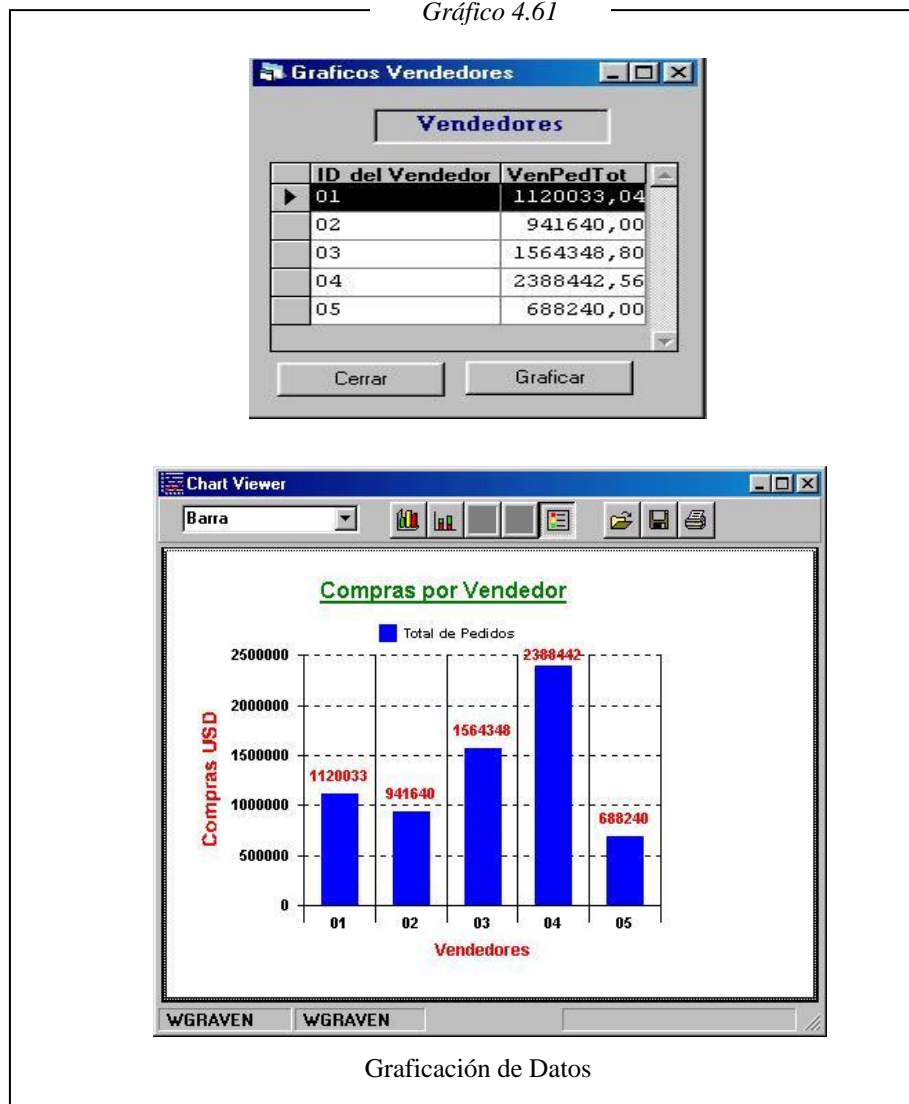


En este WP se observa un botón graficar, el cual es un evento que grafica los datos del mismo WP, la sintaxis y estructura de este evento se la observa en el gráfico 4.60. y los datos graficados en el gráfico 4.61.

Gráfico 4.60



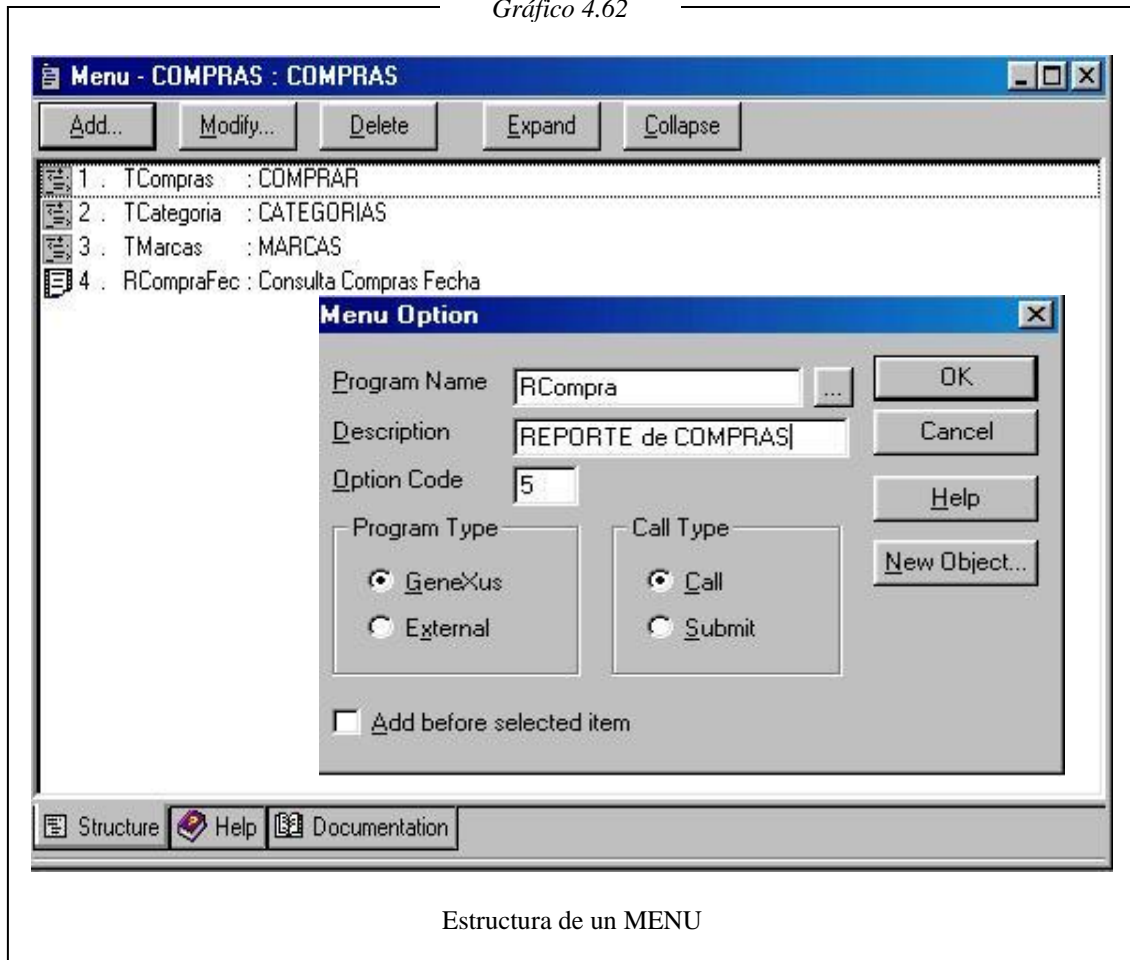
Gráfico 4.61



4.8.DISEÑO DE MENÚS

Los menús son presentaciones personalizadas, las cuales le permiten al usuario escoger de una forma personal, las opciones de ejecución y presentación de los diferentes objetos, lo único que se requiere para elaborar un menú es simplemente agregar los objetos que quiera.

Gráfico 4.62

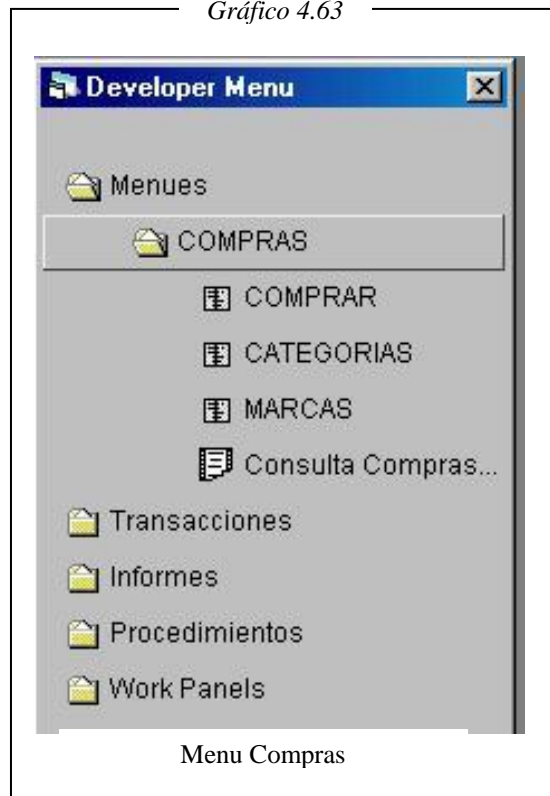


Estructura de un MENU

Para que pueda ver como se diseña un menú, se elaborara un menú de COMPRAS: este contendrá a los siguientes objetos: Tcompras, Tcategorias, Tmarcas, RcompraFec.

Primeramente se debe insertar un nuevo objeto (Menu), Luego de esto se debe agregar los objetos que Ud. Desea con el botón (ADD) y debe poner una descripción de cómo quiere que se vea en pantalla; el nombre del objeto como Tcompras se lo inserta en la opción (PROGRAM NAME) (gráfico 4.62 – 4.63).

Gráfico 4.63



4.9 DISEÑO Y PUBLICACIÓN DE OBJETOS WEB

El diseño de objetos web: transacciones, reportes, web panels, etc. Se lo realiza siguiendo los mismos pasos que los objetos windows. Simplemente se debe especificar desde el principio, que la forma del proyecto sea web.

Primeramente se selecciona: File – New Model y aparece la pantalla para cambiar las formas windows a formas web (gráfico 4.64)

Como a esta presentación web se la realizará de un proyecto windows ya elaborado, aparecerá un pantalla donde pide reorganizar la base de datos. (gráfico 4.65)

Después de este proceso se genera el proyecto y ya se tiene pantallas web. Cabe indicar que: como el proyecto fue realizado exactamente bajo formas windows, al cambiar a formas web solamente se generarán las transacciones y los paneles de trabajo; a los reportes toca cambiar algunas propiedades para que se puedan visualizar vía web.

Gráfico 4.64

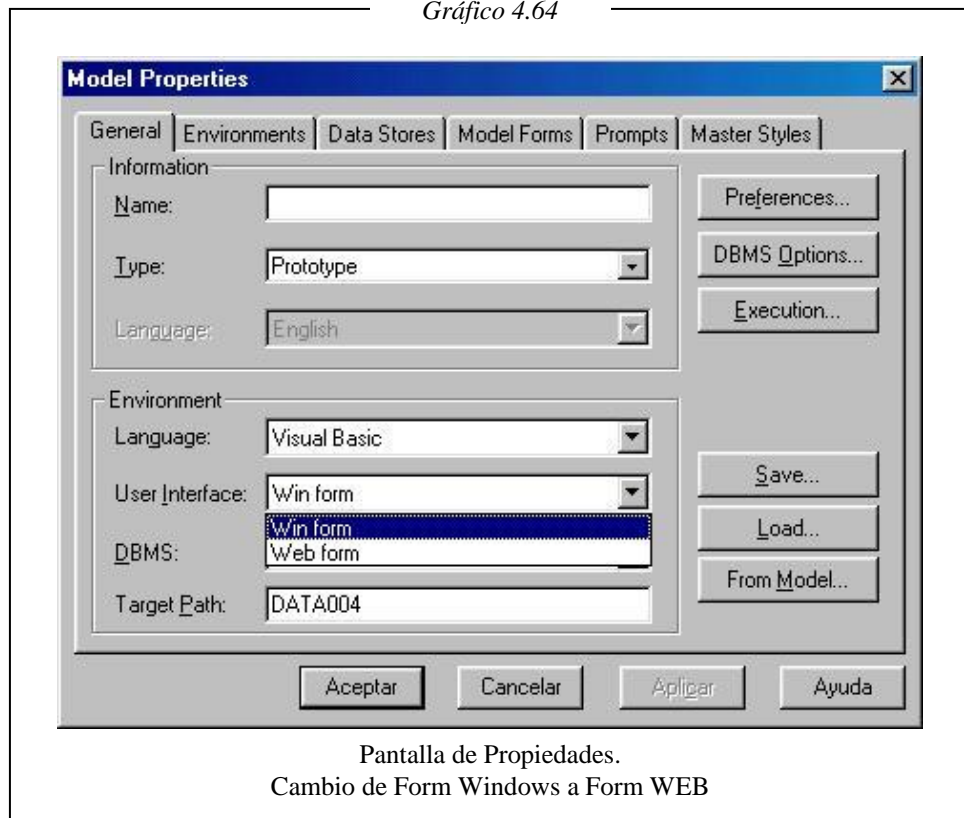


Gráfico 4.65



El gráfico 4.66 muestra la pantalla de la transacción Web productos, desde un ambiente de diseño generado por defecto (Web Form).

Gráfico 4.66

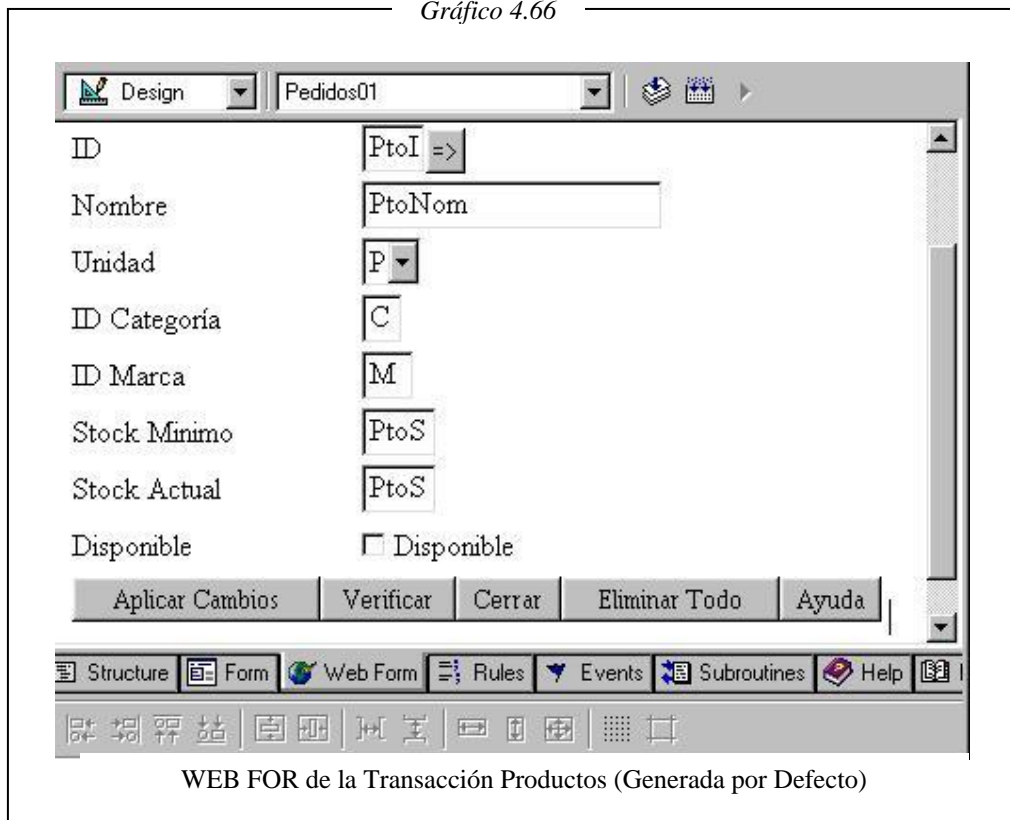
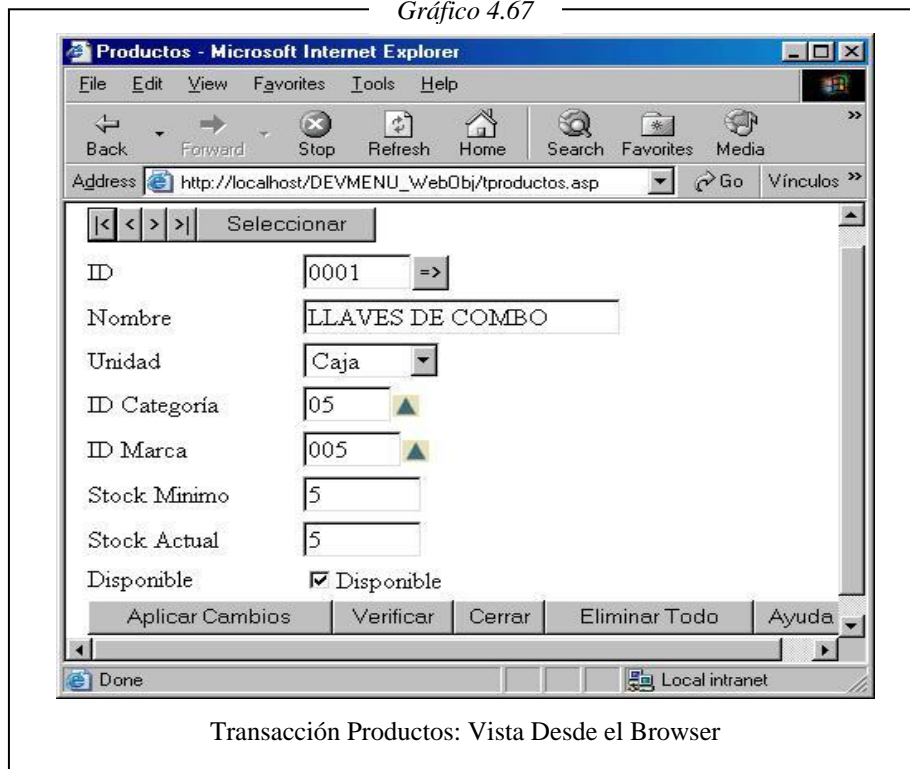


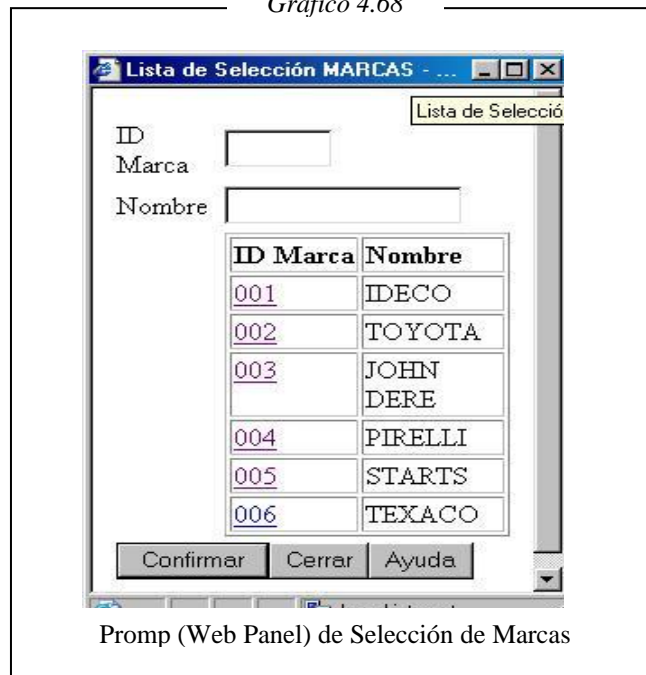
Gráfico 4.67



La estructura de la transacción no cambia, permanece intacta, los prompts generados automáticamente, tampoco cambian solamente pasan a ser objetos web.

En el diseño de transacciones comunes, se debía pulsar la tecla f4 para visualizar los prompts en cambio en el web van acompañados de una flecha al lado izquierdo. En el gráfico 4.67 se observará a la transacción productos desde internet.

Gráfico 4.68

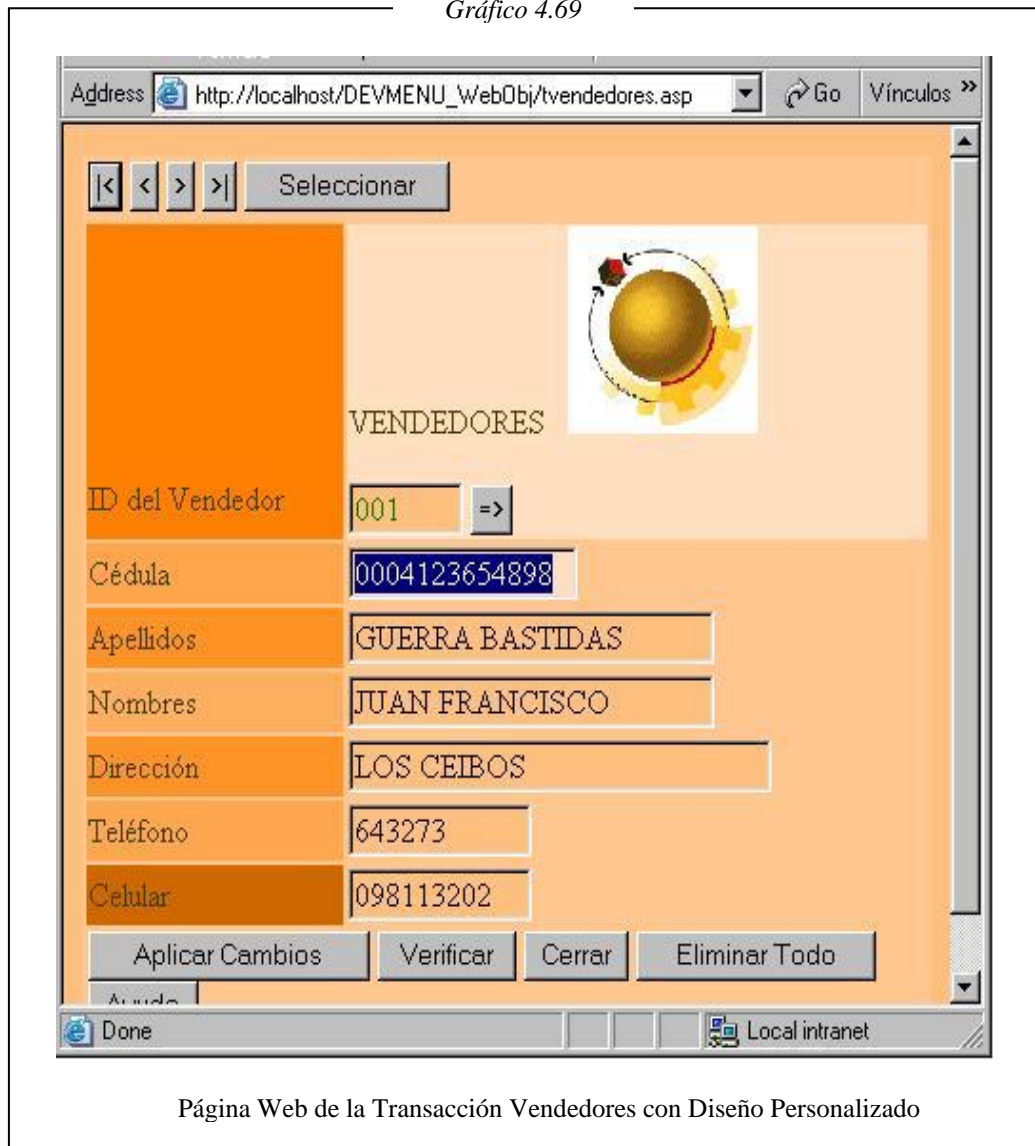


Prompt (Web Panel) de Selección de Marcas

El gráfico 4.68 lista los datos de selección de marcas por medio de un web prompt, este objeto es generado e invocado por la transacción web de productos.

Estas páginas o transacciones web generadas por defecto no son de mucho agrado para el usuario, y siempre se prefiere páginas personalizadas, o con un diseño más agradable, a continuación se mostrará la transacción vendedores, con un diseño personalizado.(gráfico 4.69)

Gráfico 4.69



NOTA:

“Para elaborar este capítulo, se tomó como referencia el siguiente manual:

GENEXUS DISEÑO DE APLICACIONES

Copyright @ARTech Consultores 1998- -2000.

All right reserved, ARTECH

Entregado a los profesores de la Facultad de Ingeniería en Ciencias Aplicada,. de la Universidad Técnica del Norte, en Octubre del año 2001, por motivo de la realización del Curso Básico d Genexus, dictado por Jorge Izquierdo, técnico de GMS.

Esta mini aplicación que sirvió de muestra para explicar el funcionamiento de Genexus esta disponible en el CD – ROM que viene conjuntamente con la tesis; Está ubicada en la carpeta Pedidos que tiene otras subcarpetas: Pedidos 1 - - - - Pedidos 5, cada subcarpeta contiene el contenido de la aplicación elaborado por partes, para que el investigador pueda darse cuenta de los cambios realizados en la aplicación de una forma más objetiva.

Para que Ud. pueda ver el funcionamiento del programa tendrá que instalar en su PC Genexus 7.5 Trial Versión, herramienta que la puede bajar de Internet o del CD – ROM de la tesis, esta herramienta es de libre distribución”

El Autor.

CAPÍTULO V

APLICATIVO DISEÑO DE UN SISTEMA DE CONTROL DE INVENTARIOS Y BODEGAS PARA LA EMPRESA PETROLERA DYGOIL. CIA. LTDA. UTILIZANDO GENEXUS 7.0

Dygoil cía. Ltda.



Consultoría y Servicios Petroleros
Pioneros en Servicios Petroleros en el Ecuador

- ❖ Introducción.
- ❖ Soluciones a Generarse en la Aplicación.
- ❖ Análisis de Necesidades.
- ❖ Diseño de la Aplicación.
- ❖ Implementación de la Aplicación.
- ❖ Prototipo / Producción.
- ❖ Aplicaciones para el WEB.

5.1. INTRODUCCIÓN

El aplicativo a desarrollarse se denomina: *"Diseño de un Sistema de Control de Inventarios y Bodegas para la Empresa Petrolera Dygoil. Cia. Ltda. Utilizando el CASE Genexus 7.0"*. Con la construcción de esta solución informática se pondrá en práctica todas las facetas de la Herramientas CASE investigadas en anteriormente.

El objetivo de este capítulo, no es la documentación del sistema desde el punto de vista de la empresa como: como manuales de usuario, manuales técnicos; este tipo de documentación será entregado a Dygoil cuando la aplicación quede totalmente implantada en la misma.

Mas bien se enfoca hacia las personas que estudian el desarrollo de aplicaciones, el cual muestra una metodología diferente de diseñar soluciones de software como es la utilización de Herramientas CASE en todo el ciclo de vida de un sistema.

La herramienta que se utilizará sigue siendo Genexus, pero por motivos técnicos, se ha realizado un cambio en la versión de la herramienta.

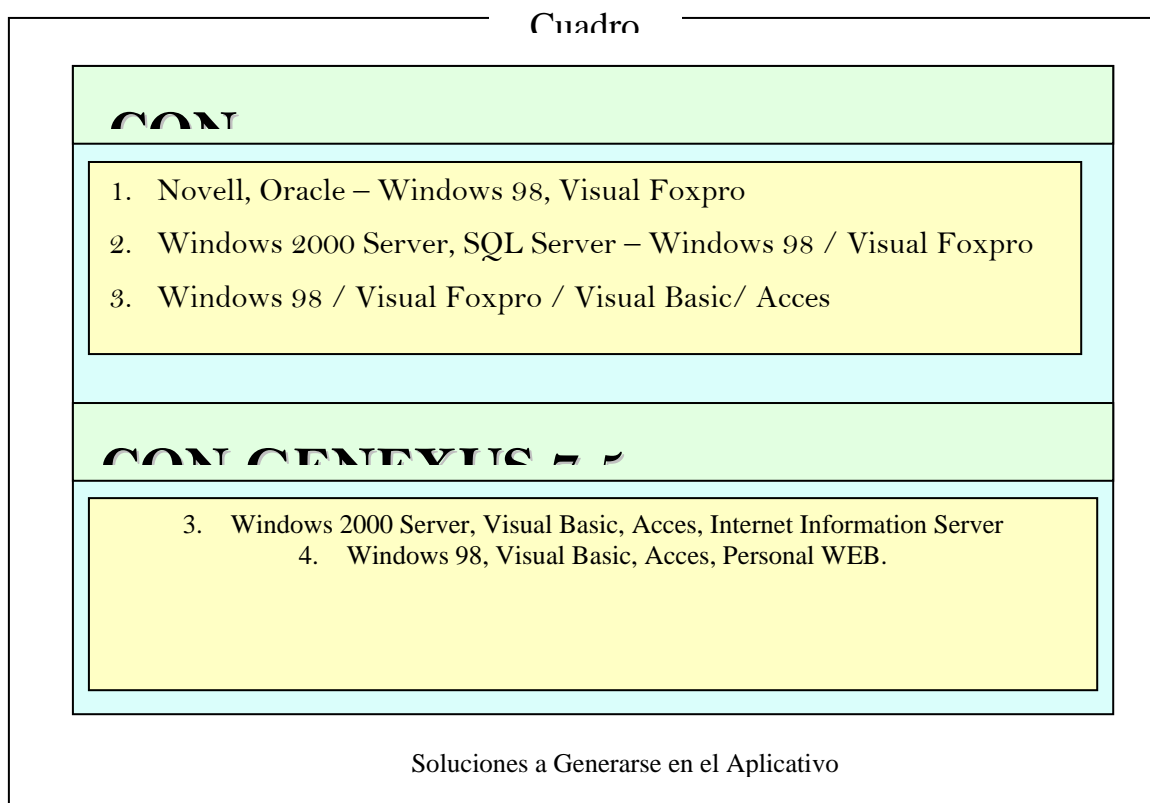
El sistema a diseñarse, es una aplicación de carácter considerable, el cual no podía ser elaborado en su totalidad con Genexus 7.5 Trial Versión, ya que por tener sus limitaciones en el número de objetos a desarrollar, propios de las versiones trial, no podía cubrir todos los objetos requeridos para el óptimo diseño de esta aplicación.

El sistema general se lo realizará con la versión comercial de ***"GENEXUS 7.0"*** y con su generador ***"CLIENT / SERVER VISUAL POXPRO GENERATOR"***.

Para demostrar que las Herramienta CASE si son capaces de generar soluciones multicapa y multiplataforma, se diseñará una sola aplicación con Genexus 7.0, la cual deberá ser generada en diferentes capas y plataformas reduciendo los tiempos y los costos, pero conservando la calidad del software.

También se diseñará una mini aplicación similar con Genexus 7.5 Trial Versión, que simplemente demostrará las facilidades que brinda la herramienta para desarrollar soluciones completamente para el web, ya que Genexus 7.0 no cuenta con estos privilegios en su totalidad, debido a que es una versión anterior.

5.2. SOLUCIONES A GENERARSE EN LA APLICACIÓN.



Esta es la propuesta de soluciones que se generarán en la aplicación, se menciona generalmente ambientes visual foxpro en lo que respecta a cliente / servidor, porque se cuenta solo con esta licencia, si el desarrollador posee las licencias de los otros generadores, puede realizar cualquier arquitectura de solución, dependiendo de la licencia o licencias que tenga, sin necesidad de cambiar o reprogramar los aspectos fundamentales de la aplicación, si es de realizar algún cambio solo tendrá que ver con la conexión o con las propiedades de la base de datos.

Para demostrar los ambientes multicapa se elaborará arquitecturas cliente servidor, los cuales se los puede observar en las dos primeras aplicaciones, que demuestran sistemas en dos capas, la tercera aplicación es una arquitectura nativa donde toda la aplicación y los datos están en visual foxpro. Estas arquitecturas nativas no se las toma como cliente servidor dos capas, pueden variar con acces y visual basic. Por último se hará una demostración de aplicaciones y arquitecturas internet con la versión trial de genexus.

Para demostrar multiplataforma las aplicaciones serán generadas para Novell Netware y para Windows.

**PROPUESTA 1: NOVELL - ORACLE
VISUAL FOX PRO**

WINDOWS 98 -

Cuadro

SSOO de Red	Novell Net Ware 5.1
Base de Datos	ORACLE 8i 8.1.5.0.4
SSOO de Estación	Windows 98
Lenguaje de Programación	Visual Foxpro
Herramienta CASE	<i>GENEXUS 7.0</i>

Propuesta 1: Novell, Oracle – Windows 98, Visual Foxpro

La propuesta número uno es la principal, en base a esto se generarán todas las otras aplicaciones, dicho de otra manera la base de conocimiento se realizará con estas indicaciones.

Como se puede observar es una aplicación multicapa y multiplataforma, realizada en una base de datos grande y considerable, o sea, se está trabajando con un servidor pesado y un cliente liviano, todos los requerimientos y las propiedades están relacionadas con la propuesta número uno.

A la generación de las otras aplicaciones, no se le dará tanto énfasis, ya que una vez generada la base de conocimiento principal solo hay que hacer algunos cambios de propiedades.

**PROPUESTA 2: WINDOWS 2000 SERVER - SQL SERVER WINDOWS 98
- VISUAL FOXPRO**

Cuadro 5.3

SSOO	Windows 2000 Server
Base de Datos	MS – SQL Server
SSOO de Estación	Windows 98
Lenguaje de Programación	Visual Foxpro
Herramienta CASE	<i>GENEXUS 7.0</i>

Propuesta 2: Windows 2000 Server, SQL Server – Windows 98, Visual Foxpro

PROPUESTA 3: WINDOWS 98 – VISUAL FOXPRO

Cuadro 5.4

SSOO	Windows 98
Base de Datos	Visual Fox / Acces
Lenguaje de Programación	Visual Foxpro / Visual Basic
Herramienta CASE	GENEXUS 7.0

Propuesta 3: Windows 98 – Visual FoxPro

La propuesta número dos es una aplicación cliente servidor, dos capas, uniplataforma y la propuesta tres, es una aplicación centralizada o nativa.

Para que el estudiante de estos temas pueda darse cuenta como un case es capaz de generar aplicaciones para varios Front End, se puede interactuar con Acces y Visual Basic, pero en modos centralizados, nuevamente se recalca que es por motivos simplemente de no poseer las licencias.

PROPUESTA 4: WINDOWS 2000 SERVER O WINDOWS 98 VISUAL BASIC - ACCES INTERNET INFORMATION SERVER O PERSONAL WEB

Cuadro 5.5

SSOO	Windows 200 Server o Windows 98
Base de Datos	Acces 2000
Lenguaje de Programación	Visual Basic 6.0
Herramienta CASE	GENEXUS 7.5 Trial Version
Servidor WEB	IIS o Personal Web

Propuesta 4: Demostración de Aplicaciones Internet

En esta propuesta web se generará un demo similar al aplicativo, simplemente para demostrar la generación de soluciones web, la cual tendrá de nuevo, la interacción de dos servidores: internet information server y personal web. Para estas soluciones se trabajará con Genexus 7.5 Trial Versión, porque cubre toda la aplicación Internet.

5.3. ANÁLISIS DE NECESIDADES

DYGOIL. Cia Ltda. es la primera empresa cien por ciento ecuatoriana dedicada a la prestación de servicios petroleros en el Ecuador.

Su nombre se forma por la unión de las letras de los primeros apellidos de sus dos principales accionistas: el Ing. Mauricio Dávalos y el Ing. Cesar Guerra, lo que a llevado a definir: Petroleros, Dávalos y Guerra (DYGOIL. Cia. Ltda.)

Tras un dedicado análisis en el campo de bodegas e inventarios, se llegó a la conclusión que DYGOIL no cuenta con un sistema apropiado para controlar sus productos, tanto en la bodega principal ubicada en Quito, como en las bodegas foráneas ubicadas en distintos sectores de la región Amazónica, es por eso que se decidió diseñar un sistema que soluciones este inconveniente.

5.3.1 FUNCIONAMIENTO DEL SISTEMA

Primeramente se debe crear una transacción donde se tenga almacenados todos los productos, debidamente codificados y ordenados de acuerdo a su código. Esta transacción, se la toma como la bodega principal en la ciudad de Quito.

Para poder ingresar y codificar los productos, se requiere de otras transacciones como: Categorías, Subcategorías y Marcas, las cuales deben ser diseñadas anteriormente, ya que vienen a ser claves foráneas en la transacción de productos y son los parámetros principales para la codificación.

El tipo de producto, como: nacional o importado también es un parámetro llamado desde la transacción Tipo de Producto, otro parámetro es la Unidad de Medida del producto.

A los productos que están en la ciudad de Quito, se les asigna una ubicación, por ejemplo, el producto 04010205 está en la percha de accesorios de taladro, en el bloque 05, parámetro llamado desde la transacción Ubicaciones, creada con anterioridad.

Los productos manejan tres tipos de STOCK, un stock disponible, que es el stock que está disponible en Quito, un stock en ejecución, que es la cantidad del producto asignada a las bodegas foráneas, y un stock total que viene a ser la suma de los dos stocks. El stock varía dependiendo de los ingresos egresos y las bajas.

Para alimentar la transacción productos, se creó una transacción Compras, la cual viene a sumar el stock disponible de productos, actualizando la fecha y el precio de la última compra en la transacción de productos.

La transacción compras está asociada a la transacción Proveedores, Productos, y Tipo de Compra, como también a la del Personal Administrativo de DYGOIL, ya que como son compras de mucho dinero, tiene que autorizar un alto funcionario de la empresa.

Como es normal que algún producto comprado tenga fallas o no cuente con la satisfacción total de la empresa, se creó la transacción Devolución en Compras, que viene a restar al stock disponible de productos.

Una vez alimentada la base de datos principal de productos, se puede realizar las transacciones de bodegas, las cuales vienen a ser una especie de mini transacción de productos. En estas transacciones de bodegas, las cuales constan de: Bodega 01, Bodega 02 y Bodega 03 como transacciones independientes, por motivos técnicos y económicos, tienen similares características que las de los productos, cabe señalar que al ser bodegas independientes si se tiene la posibilidad de incrementar el número de bodegas, ya que se está frente a una tecnología incremental.

Las bodegas tienen asociadas las transacciones: Bodegueros y Productos, en bodegueros se almacenan solo los bodegueros de esa bodega, y en productos solo los productos que se van a asignar a dicha bodega.

Los productos de las respectivas bodegas, también manejan un stock, como el de productos generales, el stock disponible es el que se puede utilizar, pero solo de la seleccionada, el stock en ejecución juega con las órdenes de egreso e ingreso, dentro de la bodega y el stock total es de igual forma la suma de los dos.

Como cada bodega tiene su base de datos de productos, se realizó una transacción que se la conoce como Asignaciones de Bodegas, como son independientes las bodegas, habrá una transacción de asignación para cada bodega. Esta toma como referencia y clave foránea el identificador de la bodega.

Cada producto asignado a determinada bodega hace lo siguiente con el stock: resta del stock disponible y suma al stock en ejecución de la transacción de productos generales y suma stock disponible de los productos de la transacción bodegas, cada transacción realizada donde hay: ingresos o egresos el sistema automáticamente le devolverá un recibo de comprobación. Como es lógico cada bodega tendrá la necesidad de devolver algún producto a Quito, para esto se creó la transacción Devoluciones de Bodegas, que realiza lo contrario que las asignaciones con los stocks.

Se vuelve a recalcar, que cada transacción que tenga que ver con bodegas, se la debe realizar independientemente una de otra, así: si Ud. Realiza una transacción Devoluciones01 para la bodega 01, debe diseñar otra transacción devoluciones02 para la bodega02.

Una vez asignados cierta cantidad de productos a las diferentes bodegas, ya se puede trabajar con cada una de las bodegas foráneas. Se empezará describiendo las órdenes de egreso.

La transacción Orden de Egreso tiene como objetivo registrar los movimientos de los productos de una bodega seleccionada. Para esto el trabajador llega a la bodega y saca los productos que necesita para trabajar, entonces el bodeguero realiza una orden de egreso, la cual consta de número de egreso, fecha, código de la bodega, bodeguero, información del proyecto, supervisión, responsable y productos, los parámetros desde bodeguero hasta aquí son claves foráneas.

Información del proyecto nace en la transacción Proyectos, la cual almacena los datos de los diferentes proyectos suscritos por DYGOIL y las diferentes empresas petroleras, nacionales e internacionales. En esta transacción constan el número de proyecto, el nombre, la empresa con quien suscribió el contrato, el campo donde se está operando, el servicio que está prestando DYGOIL, el personal administrativo responsable y el personal de control responsable, tanto de DYGOIL como de la empresa contratante. Esta transacción es muy importante, ya que las empresas petroleras realizan todas sus transacciones dependiendo del proyecto asignado, y consta en las transacciones de órdenes de egreso ya que de esta manera se puede saber a que proyecto se asignó un determinado producto.

En la transacción órdenes de egreso consta un responsable, el cual es la persona que saca el producto, este tiene la obligación de devolver dicho producto mediante una transacción Orden de Ingreso, la cual recibe como clave foránea el número de orden de egreso, en las dos transacciones de ingresos y egresos el sistema automáticamente emite un recibo de responsabilidad.

Con las ordenes de ingreso y egreso se mueven las cantidades de los stocks, pero solo dentro de la bodega elegida, cuando existe una orden de egreso se suma el stock en ejecución de dicha bodega y se resta el disponible de la misma, y con la orden de ingreso se suma el disponible y se resta el de ejecución.

Como la empresa también posee el servicio de Proveedor de Equipos Petroleros, se realizo también un módulo de ventas, esta transacción afectará a la bodega principal de Quito.

Todo producto tiene su tiempo de vida útil, en especial lo que se refiere a productos petroleros, que tienen que estar cien por ciento utilizables para no detener la producción petrolera, es por eso que se creo dos módulos de bajas de productos, una transacción de Bajas Generales, que afecta a la bodega de Quito, concretamente resta del stock disponible, y otra transacción de Bajas por Bodegas, que afecta a la bodega seleccionada como a la principal.

Para cada clave foránea llamada desde una transacción diferente, se ha generado prompts o consultas de selección, la cual valida los datos posibles que pueden alimentar a dicha transacción.

El comportamiento de las transacciones, el diseño: de work panels, reportes y procedimientos, se los explicara en la parte de diseño.

La información para realizar esta aplicación se la obtuvo, por medio de los: técnicos, personal de servicios y personal administrativos de la empresa DYGOIL. Cia. Ltda., a base de preguntas, criterios y conversaciones sobre lo que está fallando en el control de inventario y bodegas.

5.4. DISEÑO DE LA APLICACIÓN

Para empezar a diseñar la aplicación, se debe primero diseñar la realidad, por medio de la llamada lluvia de ideas, es entonces cuando nacen transacciones como: productos, proveedores, empresas relacionadas, bodegas, compras, ventas, asignaciones, ordenes de ingreso, ordenes de egreso, etc y es cuando el cliente dice: quiero que esta transacción funcione de esta manera, quiero que tenga tales campos, quiero que se visualice algo, etc. Como se está manejando una metodología incremental no es necesario diseñar toda la aplicación en ese momento, de esta forma se empezará diseñando la transacción de productos.

5.4.1 TRANSACCIÓN DE PRODUCTOS

Gráfico 5.1

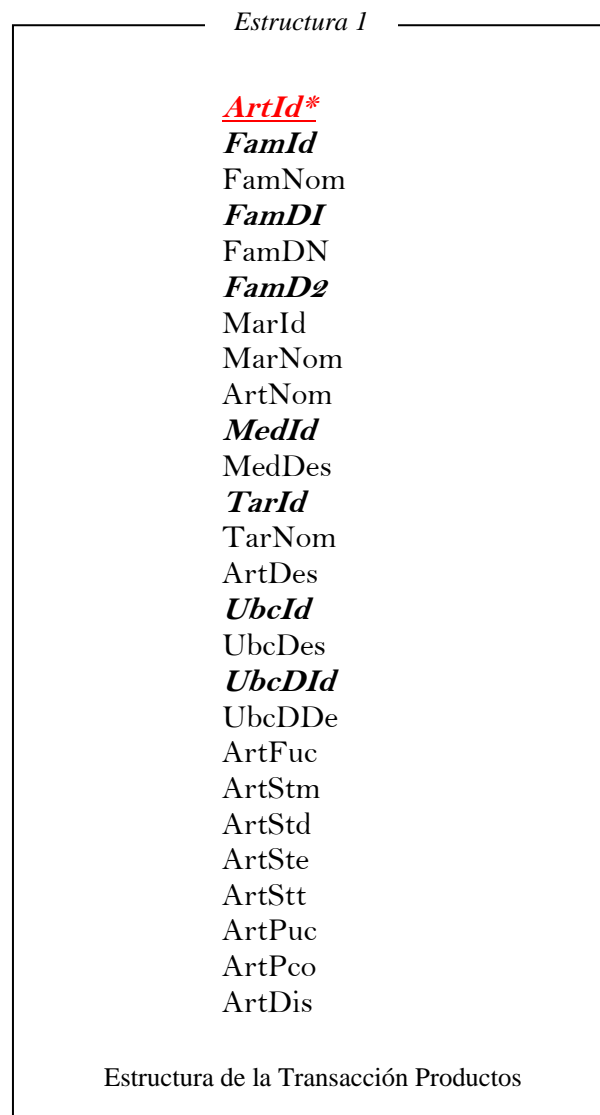
The screenshot shows a window titled "Productos" with a green header bar containing the logo "Dygoil" and the word "PRODUCTOS". Below the header is a navigation bar with buttons: "K", "<", ">", ">|", "Seleccionar", "Confirmar", "Cerrar", "Eliminar", "Ayuda", and "Bodegas". The main area is a form with the following fields and controls:

- Código:** A text box labeled "ArtId".
- Familia:** A dropdown menu labeled "Fa".
- Sub Familia:** A dropdown menu labeled "Fa".
- Marca:** A dropdown menu labeled "Fa" and a text box labeled "Ma".
- FamNom:** A text box.
- FamDN:** A text box.
- MarNom:** A text box.
- Nombre del Producto:** A text box labeled "ArtNom".
- Unidad de Medic:** A dropdown menu labeled "MedI" and a text box labeled "MedDes".
- Tipo de Producto:** A dropdown menu labeled "Ta" and a text box labeled "TarNom".
- Descripción del Producto:** A text box labeled "ArtDes".
- Ubicación:** A dropdown menu labeled "UbcI" and a text box labeled "UbcDes".
- Sub Ubicación:** A dropdown menu labeled "Ub" and a text box labeled "UbcDDe".
- Fecha Ultima Como:** A text box labeled "ArtFuc".
- Precio Ultima Comp:** A text box labeled "ArtPuc".
- Precio Comerc:** A text box labeled "ArtPco".
- Stock Mínim:** A text box labeled "ArtSta".
- Stock Disponib:** A text box labeled "ArtStd" (highlighted in green).
- Stock en Ejecució:** A text box labeled "ArtSte".
- Stock Tot:** A text box labeled "ArtStt".
- Disponib:** A checkbox.

Below the form, the text "Diseño de la Realidad: Transacción Productos" is centered.

Esta es la realidad que desea ver el usuario (gráfico 5.1) el diseñador de la aplicación solo tiene que preocuparse de construir las partes que el cliente quiere ver, con las formas comunes de programación, no se puede diseñar la realidad en los primeros pasos del desarrollo sino que se la ve al final.

Para diseñar la realidad en esta transacción, simplemente se debe de crear una estructura, en la cual van los datos que se quiere visualizar, y es la siguiente: (estructura 1)



Una vez creada esta estructura la Herramienta CASE genera automáticamente la forma, la cual puede ser modificada a gusto del cliente. (gráfico 5.1)

La estructura es la siguiente: ***ArtId**** es la clave principal de la transacción, productos es el nombre descriptivo de esta transacción, ya que el nombre verdadero es **Artic**, que significa Artículos, esta consideración se la tomo para no confundir los atributos, con los de proveedores y productos.

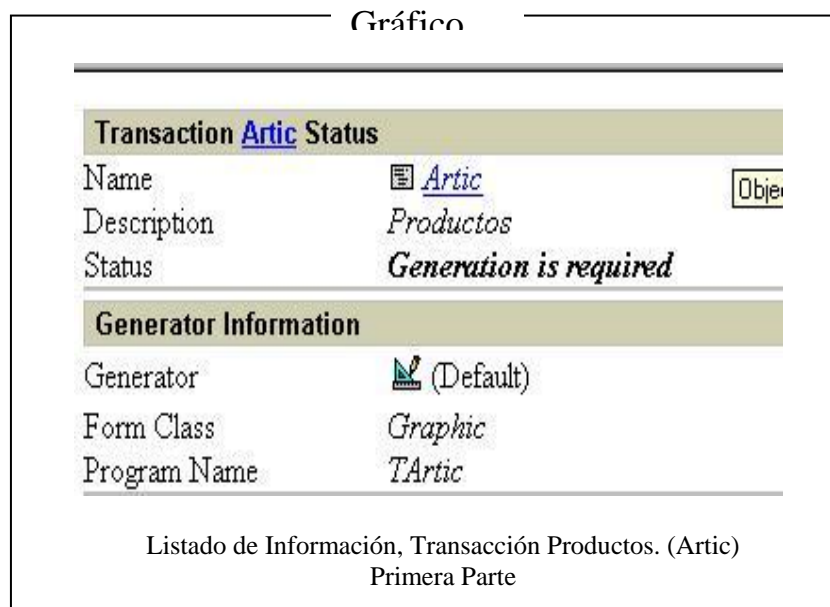
En ***ArtId**** se almacenará el ***Código del Producto***, para poder guardar los datos de esta transacción, se llama a las claves principales de las siguientes transacciones: Familia, Subfamilia, Marca, Unidad de Medida, Tipo de Producto, Ubicación y Sub ubicación. Estas transacciones ya fueron creadas con anterioridad para que puedan ser llamadas por la transacción productos.

Cave mencionar que si Ud. llama a algunos atributos de una transacción externa, en la base de datos solo se guarda la clave primaria de la transacción a la que fue llamada, con esto se esta impidiendo la replicación de datos y ayuda a controlar la normalización de la tabla, que Genexus genera para Tercera Forma Normal.

En estructura 1 se puede visualizar, los atributos que son llamados desde otras tablas, los cuales están remarcados con negrillas y son: FamId y FamDIId los cuales son atributos principales de la transacción familias y subfamilias, FamD2 es el atributo clave de las marcas, se lo considera a FamD2 y no a MarId, ya que la transacción familias está diseñada para leer determinadas marcas dependiendo del tipo de subfamilia que se almacene, cuando el usuario quiere almacenar un nuevo producto, tiene que llenar los datos de FamId y de todos los atributos foráneos, los cuales son fáciles de distinguir porque en ellos aparece una viñeta de selección, la cual llama a un *Prompt*, o lista de selección de los datos almacenados en la transacción foránea a ser llamada, para esto en la pantalla de productos no solo sale por ejemplo FamId, sino FamNom que es el nombre de la familia, esto es solo a manera de lectura y para una mejor visualización dentro de la transacción, ya que en la base de datos solo se almacenará FamId.

También existen atributos fórmula o dicho de otra manera campos calculados, como ArtStt que es el stock total, el cual nace de la suma del stock disponible con el stock en ejecución, este atributo no se almacena en la base de datos.

Una vez generada esta estructura, la herramienta también genera un listado de lo que hizo con la transacción, como este listado es muy amplio se lo graficará por partes. (gráficos 5.2 – 5.4)



Como se puede observar en este gráfico se especifica: el nombre, la descripción, el generador, el nombre del objeto, etc.

En los gráficos siguientes, que son las especificaciones mas concretas de la transacción, (gráfico 5.3) se especificarán detalles como: a qué tablas accede, de que transacciones, etc.

Gráfico

Level ARTIC

```

READ ARTIC
  WHERE
    ARTIC . ArtId = ArtId
  INTO FamId , FamDI , FamD2 , ArtNom , MedId , TarId , ArtDes , UbcId , UbcDI ,
ArtFuc , ArtStm , ArtStd , ArtSte , ArtPuc , ArtDis

READ FAMIL
  WHERE
    FAMIL . FamId = ARTIC . FamId
  INTO FamNom

READ FAMIL1
  WHERE
    FAMIL1 . FamId = ARTIC . FamId
    FAMIL1 . FamDI = ARTIC . FamDI
  INTO FamDN

READ FAMIL2
  WHERE
    FAMIL2 . FamId = ARTIC . FamId
    FAMIL2 . FamDI = ARTIC . FamDI
    FAMIL2 . FamD2 = ARTIC . FamD2
  INTO MarId

READ MARCAS
  WHERE
    MARCAS . MarId = FAMIL2 . MarId
  INTO MarNom

READ MEDIDA
  WHERE
    MEDIDA . MedId = ARTIC . MedId
  INTO MedDes

READ ATIPO
  WHERE
    ATIPO . TarId = ARTIC . TarId
  INTO TarNom

READ UBICA
  WHERE
    UBICA . UbcId = ARTIC . UbcId
  INTO UbcDes

READ UBICA1
  WHERE
    UBICA1 . UbcId = ARTIC . UbcId
    UBICA1 . UbcDI = ARTIC . UbcDI
  INTO UbcDDe

ArtStt = ArtStd+ArtSte
ArtPco = ArtPuc*1.30
    
```









Listado de Información, Transacción Productos. (Artic)
Segunda Parte

Gráfico 5.4

Insert into ARTIC
 Attributes to update : ArtId , FamId , FamDI , FamD2 , ArtNom , MedId , TarId , ArtDes ,
UbcId , UbcDIId , ArtFuc , ArtStm , ArtStd , ArtSte , ArtPuc , ArtDis
 Update on ARTIC
 Attributes to update : FamId , FamDI , FamD2 , ArtNom , MedId , TarId , ArtDes , UbcId ,
UbcDIId , ArtFuc , ArtStm , ArtStd , ArtSte , ArtPuc , ArtDis
 Delete from ARTIC

Referential integrity controls on delete:

- BAGEN1 (ArtId)
- VENTA1 (ArtId)
- ADQUI1 (ArtId)
- BOD012 (ArtId)

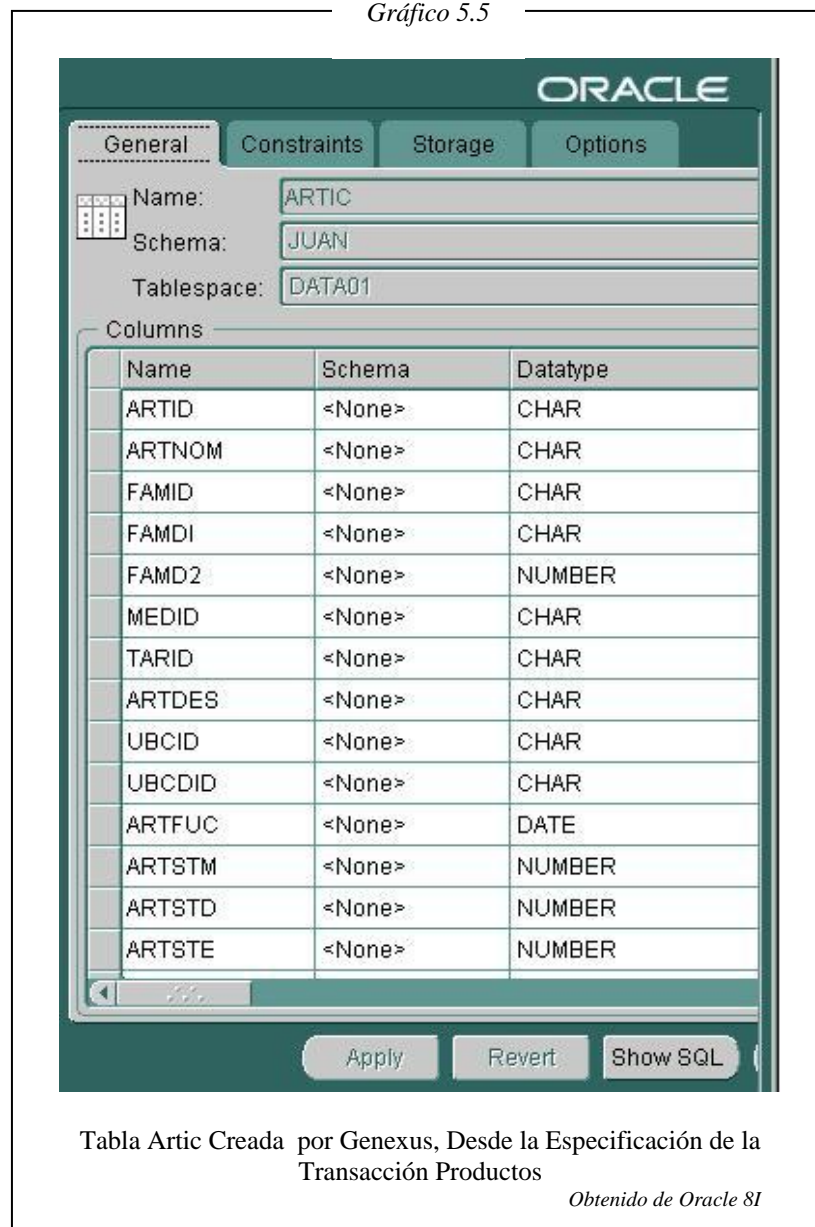
Prompts			
Table	Program	In Parameters	Out Parameters
<u>UBICA1</u>	Gx00K1 	<u>UbcId</u>	<u>UbcDIId</u>
<u>UBICA</u>	Gx00J0 		<u>UbcId</u>
<u>ATIPO</u>	Gx00M0 		<u>TarId</u>
<u>MEDIDA</u>	Gx0040 		<u>MedId</u>
<u>FAMIL2</u>	Gx00B2 	<u>FamId</u> , <u>FamDI</u>	<u>FamD2</u>
<u>FAMIL1</u>	Gx0081 	<u>FamId</u>	<u>FamDI</u>
<u>FAMIL</u>	Gx0070 		<u>FamId</u>
<u>ARTIC</u>	Gx00L0 		<u>ArtId</u>

Listado de Información, Transacción Productos. (Artic)
Tercera Parte

Como se puede observar en estos gráficos, Genexus imprime toda la información de los objetos asociados a la transacción Productos, incluso presenta un listado de los paneles de trabajo o prompts generados, los cuales son listas para seleccionar los datos de las tablas foráneas.

Para esta transacción la herramienta a generado la tabla ARTIC, la cual se la expone a continuación (gráfico 4.5), esta tabla es obtenida desde Oracle si.

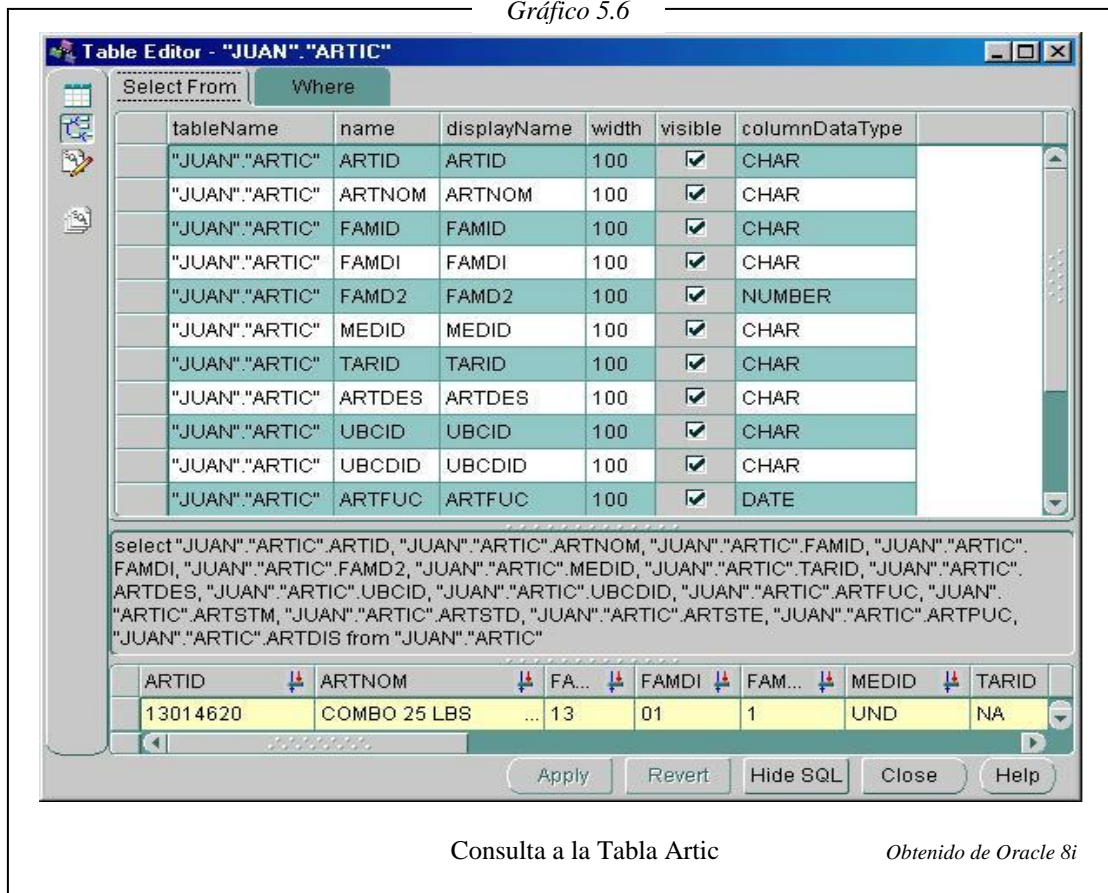
Gráfico 5.5



Como se puede observar solo constan los atributos, ya normalizados automáticamente por el case, los atributos fórmulas o campos calculados no constan en la base de datos, es el caso de ArtStt artículo stock total, que suma el disponible mas el de ejecución, otro campo calculado es el precio comercial, el cual es el precio de compra mas el 30 por ciento.

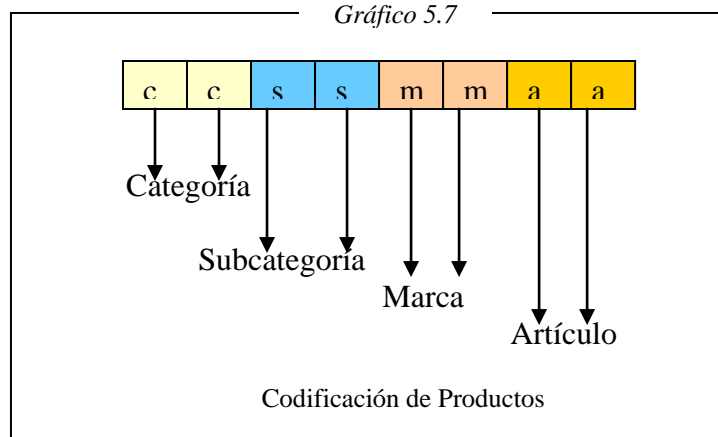
El gráfico 5.6 muestra, otra pantalla de la tabla Artic, capturada desde el Schema manager de Oracle 8i.

Gráfico 5.6



CODIFICACIÓN DE PRODUCTOS. El gráfico 5.6 muestra el siguiente producto: 13014620 Combo de 25 libras y su descripción es la siguiente: 13 categoría, 01 Subcategoría, 46 Marca y 20 Artículo, es un dato tipo carácter de 8, así: ccsmmaa para observar esto de una mejor manera observe el gráfico 5.7.

Gráfico 5.7



Como se puede observar, los dos primeros dígitos corresponden a la categoría, el tercero y cuarto dígito corresponde a la subcategoría, el quinto y sexto a la marca y el séptimo y octavo al detalle de artículo o producto.

Gráfico 5.8



El gráfico 5.8 indica a una lista de selección de productos, la cual es generada automáticamente por la Herramienta CASE, y sus llamadas también son realizadas automáticamente, por eso aparecen viñetas en las transacciones. Este panel de trabajo ya tiene sus reglas de comportamiento, una de ellas es que el usuario puede buscar un producto con la simple inserción de las primeras letras del mismo, así:

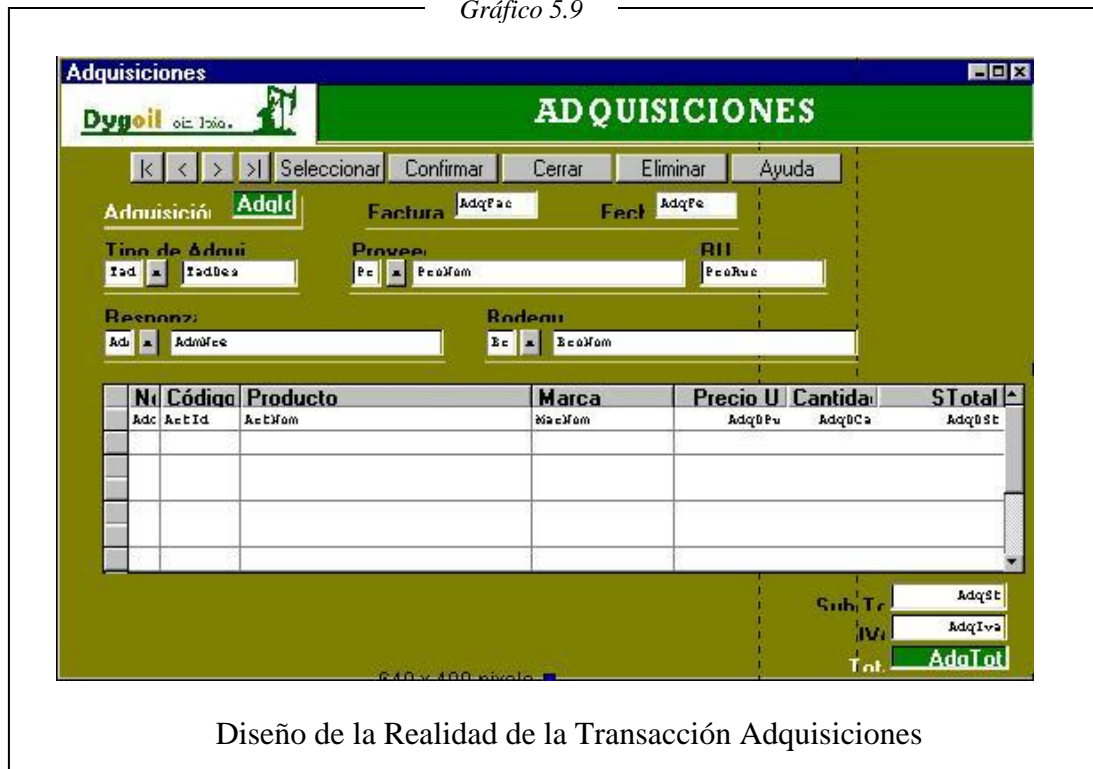
```
order( ArtId );  
search(ArtNom LIKE &C81);
```

Primeramente está ordenado por ArtId, y después se puede realizar la búsqueda (search) por el nombre del artículo.

5.4.2 TRANSACCIÓN ADQUISICIONES.

Siguiendo con la lluvia de ideas, surge la necesidad de adquirir de alguna parte los productos, en este caso sería de comprar los productos a los proveedores, proceso o transacción que se le a denominado adquisiciones.

Gráfico 5.9

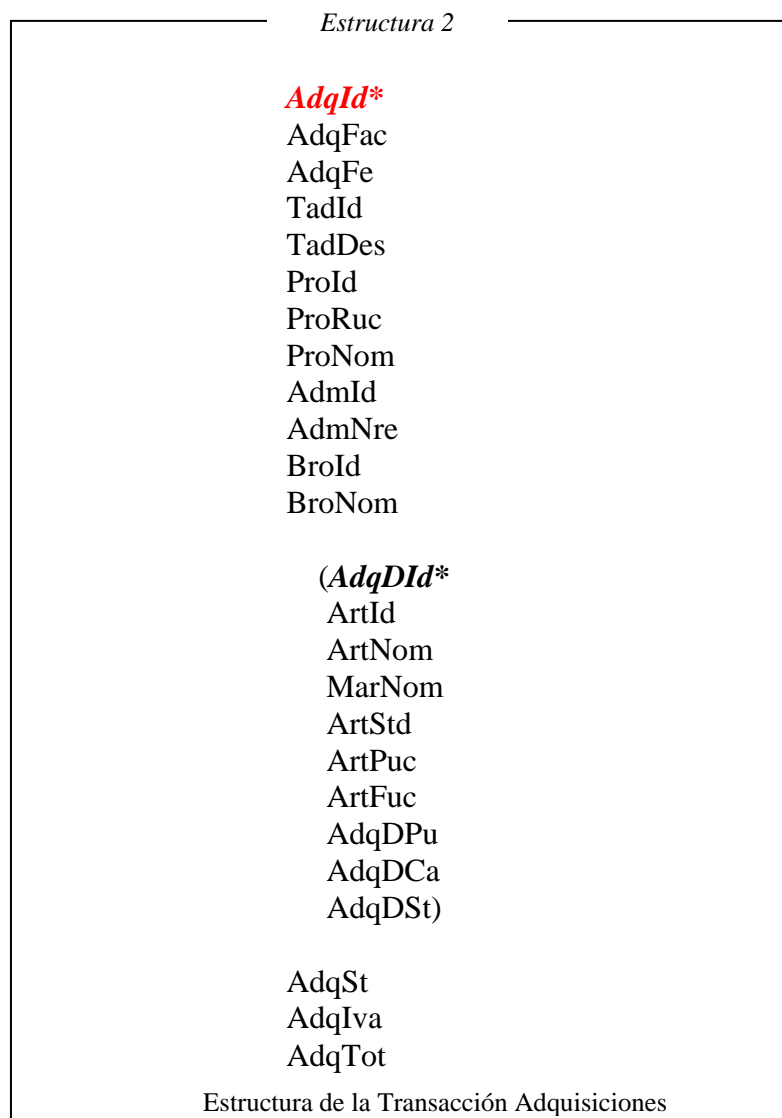


Esta es la realidad, esta es la idea de cómo se quiere ver a la transacción adquisiciones (gráfico 5.9), para que esto pueda plasmarse en la aplicación, simplemente se debe realizar la siguiente estructura: (estructura 2)

Esta transacción se a diseñado para registrar las compras realizadas a los proveedores, cuando DYGOIL compre un producto, automáticamente se debe actualizar el stock general de productos, en este caso debe sumar la cantidad comprada al stock general disponible, de la misma manera tiene que actualizar la fecha y el precio de la última compra en la transacción productos, procesos que se los realiza mediante reglas de comportamiento, y que van a formar parte de los procedimientos almacenados en el cliente.

Una vez realizado un comit a esta transacción, esta llamará a un reporte o factura, el cual se imprimirá automáticamente, especificando todo lo referente a la transacción realizada.

Todos estos pasos se los puede observar por medio del listado de especificaciones de la transacción adquisiciones. (5.10 - 5.12)



La transacción adquisiciones tiene dos niveles, lo que en programación convencional sería adquisiciones y detalle de adquisiciones, para diseñar este tipo de transacciones con niveles, simplemente toca especificar en la transacción, cada nivel encerrado entre paréntesis y automáticamente Genexus genera las tablas relacionadas, en este caso sería: Adqui y Adqui1.

Cave indicar que cada nivel tiene su indentificador o clave principal el cual sirve para llamar a la tabla de la transacción mencionada, la clave principal del primer nivel es: AdqId* y la del segundo nivel es: AdqDId*

En el segundo nivel puede ir como clave principal ArtId, pero se recomienda poner un identificador que no sea identificador principal de otra transacción, ya que al llamar a los listados de selección o al hacer una transacción de devolución, aparecerán todos los artículos, mas no los seleccionados en la adquisición, esta recomendación se la hace para todo tipo de transacción.

Gráfico 5.10

The screenshot displays a software interface with several sections:

- Transaction Adqui Status**:
 - Name: Adqui
 - Description: *Adquisiciones*
 - Status: *Generation is required*
- Generator Information**:
 - Generator: Visual FoxPro
 - Form Class: *Graphic*
 - Program Name: *TAdqui*
- Levels**:
 - Level ADQUI**:
 - SQL Query:

```
READ ADQUI
WHERE
    ADQUI . AdqId = AdqId
INTO AdqFe , AdqFac , TadId , ProId , AdmId , BroId
AdqFe = today()IFinsert;
```
 - VERTICAL FORMULAS: AdqSt
 - $AdqIva = AdqSt * 0.12$
 - $AdqTot = AdqSt + AdqIva$

At the bottom, it specifies: "Especificaciones de la Transacción: Adquisiciones Primera Parte"

En este gráfico ya aparecen algunas reglas como:

$AdqFe = today()$ If Insert, que significa que el campo $AdqFe$ sea igual a la fecha actual.

También aparecen las especificaciones de las fórmulas como: vertical fórmula de $AdqSt$, que suma todos los subtotales de la adquisición.

$AdqIva = AdqSt * 0.12$ el cual está calculando el iva.

$AdqTot = AdqSt + AdqIva$ está sumando el valor del subtotal con el valor del iva.

Gráfico 5.11

```
READ TIPADQ
  WHERE
    TIPADQ . TadId = ADQUI . TadId
  INTO TadDes
READ PROVED
  WHERE
    PROVED . ProId = ADQUI . ProId
  INTO ProRuc , ProNom
READ ADMINI
  WHERE
    ADMINI . AdmId = ADQUI . AdmId
  INTO AdmNre
READ BDGRO
  WHERE
    BDGRO . BroId = ADQUI . BroId
  INTO BroNom
```

Insert into ADQUI

Attributes to update : AdqId , AdqFe , AdqFac , TadId , ProId , AdmId , BroId

Update on ADQUI

Attributes to update : AdqFe , AdqFac , TadId , ProId , AdmId , BroId

Delete from ADQUI

Referential integrity controls on delete:

- DADQU (AdqId)

Level ADQUI1

```
READ ADQUI1
  WHERE
    ADQUI1 . AdqId = AdqId
    ADQUI1 . AdqDIId = AdqDIId
  INTO ArtId , AdqDPu , AdqDCa
```

NoAccept (ArtId) if (update)

NoAccept (ArtStd) if insertORupdate

Especificaciones de la Transacción: Adquisiciones. Segunda Parte

En este gráfico (5.11) se observa como empieza la herramienta a leer la tabla Adqui1, o si se la podría llamar como detalle de aplicaciones, la cual fue creada automáticamente, también indica algunas reglas como las de no accept, que impide modificar los contenidos de los datos de los atributos foráneos, indica también cuales son los campos claves, para un mejor manejo de integridad referencial, indica también a las tablas que accede para formar su transacción, por ejemplo READ Bdgro, está leyendo los datos de la transacción bodeguero, por medio de la tabla Bdgro.

Gráfico 5.12

```








ArtFuc = AdqFeIFinsert;
ArtPuc = AdqDPuIFinsert;
ArtStd = old(ArtStd)-AdqDCaIFdeleteANDlevel(AdqDCa);old(ArtStd)+AdqDCa-
old(AdqDCa)IF(insertORupdateORdelete);
AdqDSt = AdqDPu*AdqDCa
AdqSt = old(AdqSt)+AdqDStIFinsert;old(AdqSt)+AdqDSt-old(AdqDSt)
IFupdate;old(AdqSt)-old(AdqDSt)IFdelete;
AdqIva = AdqSt*0.12
AdqTot = AdqSt+AdqIva

Insert into ADQUI1
  Attributes to update : AdqId , AdqDIId , ArtId , AdqDPu , AdqDCa
Update on ADQUI1
  Attributes to update : ArtId , AdqDPu , AdqDCa
Delete from ADQUI1
Update on ARTIC
  Attributes to update : ArtFuc , ArtPuc , ArtStd

```

After Trn Rules

```
CALL Adq01 AdqId if after(trn)
```

Prompts			
Table	Program	In Parameters	Out Parameters
<u>ARTIC</u>	<u>Gx00L0</u> 		<u>ArtId</u>
<u>ADQUI1</u>	<u>Gx00X1</u> 	<u>AdqId</u>	<u>AdqDIId</u>
<u>BDGRO</u>	<u>Gx00G0</u> 		<u>BroId</u>
<u>ADMINI</u>	<u>Gx00C0</u> 		<u>AdmId</u>
<u>PROVED</u>	<u>Gx00F0</u> 		<u>ProId</u>
<u>TIPADQ</u>	<u>Gx00W0</u> 		<u>TadId</u>
<u>ADQUI</u>	<u>Gx00V0</u> 		<u>AdqId</u>

Especificaciones de la Transacción: Adquisiciones Tercera Parte

Las reglas definidas en la transacción adquisiciones son:

```
AdqFe = today() if Insert; // Fecha Actual.  
Noaccept(ArtId)if(update); // Noaccep ArtId.  
ArtFuc = AdqFe if Insert; // Actualiza la fecha en productos.  
ArtPuc = AdqDPu if insert; // Actualiza el precio de productos.  
add(AdqDCa,ArtStd); // Actualiza stock de productos.  
call(RAdqO1,AdqId)if after(Trn); // Imprime factura.
```

Todas estas reglas son definidas en el diseño de la transacción; la herramienta se encarga de ver la forma como las utiliza, tanto en la aplicación como en la base de datos, una de las fórmulas definidas es la siguiente:

$$AdqST = SUM (AdqDSt)$$

Esta formula permite hacer una suma vertical de todos los subtotales del nivel dos, pero el Genexus la valida y la interpreta de la siguiente manera:

$$AdqSt = old(AdqSt)+AdqDStIFinsert;
old(AdqSt)+AdqDSt-old(AdqDSt)IFupdate; old(AdqSt)-old(AdqDSt)IFdelete;$$

Si Ud. no contara con una Herramienta CASE, tuviera que realizar toda esta fórmula; con Genexus simplemente: $AdqST = SUM (AdqDSt)$

De la misma manera sucede con la actualización del stock disponible en productos, el cual cuando se realiza una adquisición, la cantidad de productos comprada desde adquisiciones, debe sumarse al stock disponible de productos, la regla definida es la siguiente:

$$add(AdqDCa,ArtStd);$$

Esto quiere decir: sumar o agregar la cantidad de productos adquiridos (AdqDCa) en la transacción adquisiciones al stock disponible (ArtStd) de la transacción productos; la herramienta la interpreta de la siguiente manera:

$$ArtStd = old(ArtStd)-AdqDCaIFdeleteANDlevel(AdqDCa);old(ArtStd)+AdqDCa-
old(AdqDCa)IF(insertORupdateORdelete); (gráfico 5.12)$$

Se a documentado de esta manera el capítulo, para que el desarrollador de aplicaciones tenga una concepción bastante clara, sobre cómo ayudan las Herramientas CASE en el desarrollo de aplicaciones de software.

5.4.3 TRANSACCIONES CON TRES NIVELES.

En las lluvias de ideas a alguien se le ocurrió tener transacciones con tres o más niveles, es el caso de la transacción familias, la cual lleva anidada a subfamilias y a marcas (gráfico 5.13), y la transacción proyectos, la cual lleva anidada a personal de supervisión de la empresa contratante y personal de supervisión de DYGOIL, pero el dilema es cómo realizar estas transacciones.

Gráfico 5.13

The screenshot shows a software interface with a green header bar containing the logo 'Dygoil cía. Ltda.' and the title 'FAMILIAS / SUB FAMILIAS'. Below the header is a navigation bar with buttons: 'k', '<', '>', '>|', 'Seleccionar', 'Confirmar', 'Cerrar', 'Eliminar', and 'Ayuda'. The main area is divided into three sections:

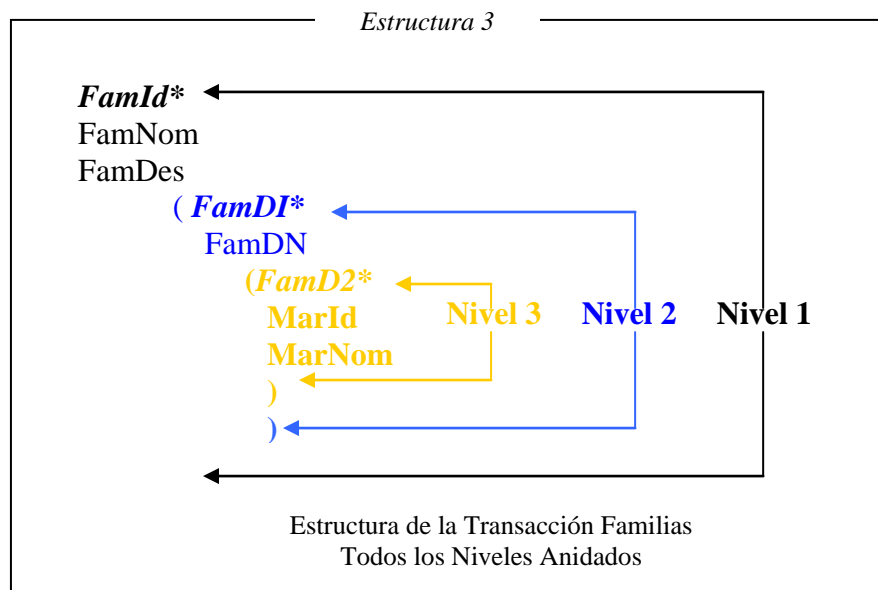
- Familia:** Contains fields for 'Código de la Famil' (with a 'Fa.' icon), 'Nombre de la Famil' (input field 'FamNom'), and 'Detalle' (input field 'FamDes').
- Sub Familia:** Contains fields for 'Código Sub Famil' (with a 'Fa.' icon) and 'Nombre SI' (input field 'FamDN').
- Marcas:** A table with columns 'No', 'Código de la Marc', and 'Nombre de la Marca'. The first row has sub-headers 'Far', 'MarId', and 'MarNom'.

Below the table, the text reads: 'Transacciones con Todos los Niveles Anidados' and 'Diseño de la Realidad de la Transacción Familias'.

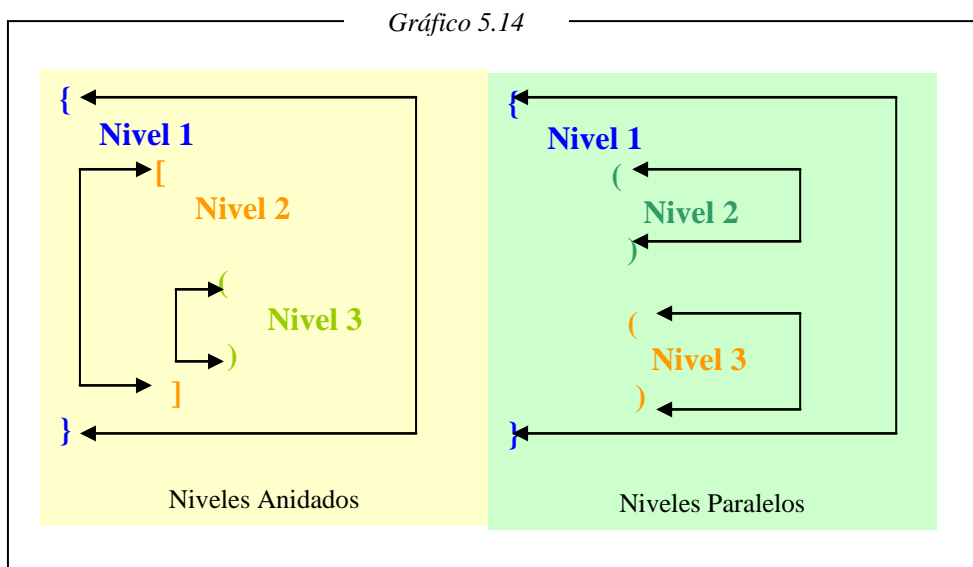
NIVELES ANIDADOS Y NIVELES PARALELOS

El gráfico 5.13 muestra la realidad de cómo quiere verse la transacción familias, a su vez es un ejemplo de diseño de transacciones con todos los niveles anidados. Se señala todos los niveles anidados ya que se quiere demostrar que: el nivel uno tiene relación directa con el nivel dos, el nivel dos tiene relación directa con el nivel tres, esto implica que el nivel tres tiene relación directa con el nivel uno.

Su funcionamiento es el siguiente: en el nivel uno va la parte de las familias, por ejemplo, motores, en el nivel dos van las subfamilias, como gasolina, diesel, etc, y en el nivel tres van las marcas de las subfamilias como kumix, internacional, hino, etc. Esto implica que se puede tener un motor a diesel marca hino, su estructura es la siguiente: (estructura 3)

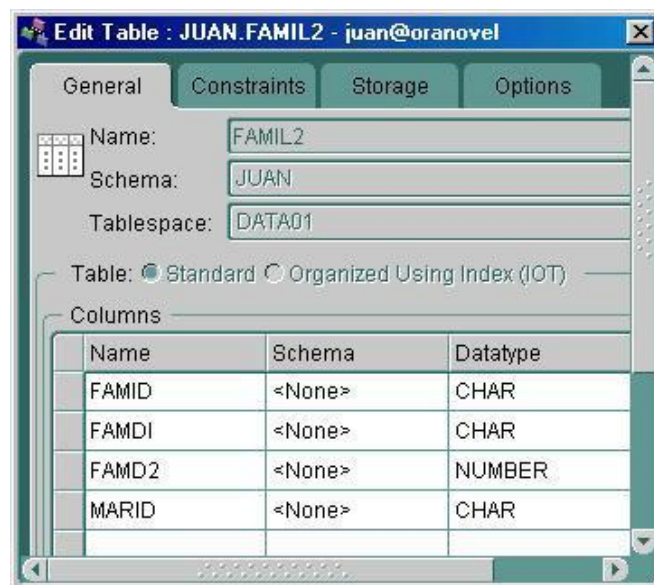
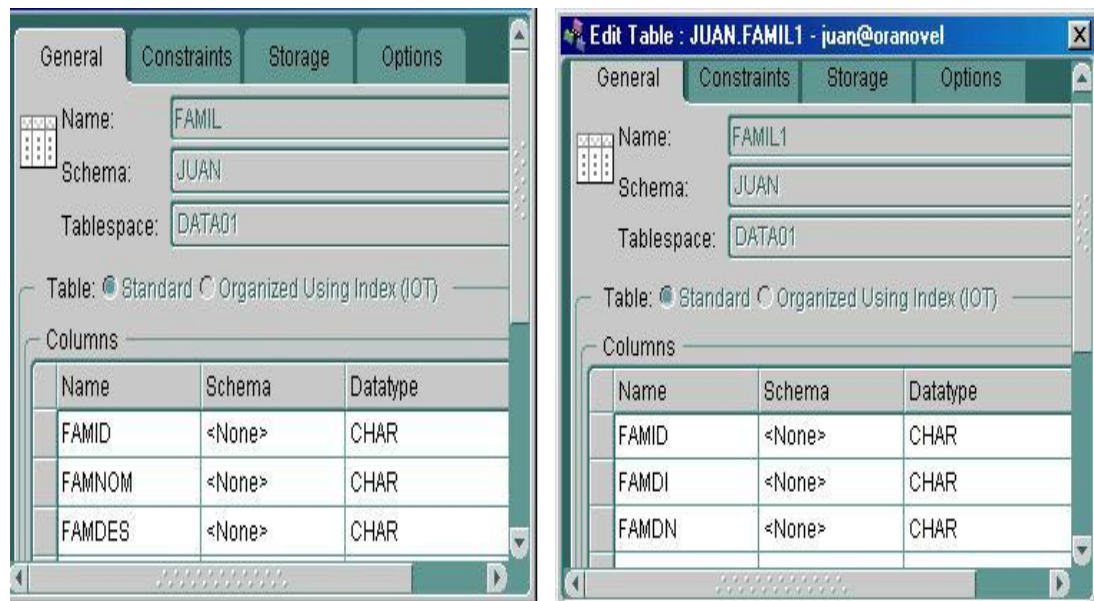


El gráfico indica que todos los datos de todos los niveles son enlazados entre sí, todos los datos dependen mutuamente el uno del otro, lo que no sucede con los niveles paralelos, donde el nivel 1 tiene relación con el nivel 2, el nivel uno tiene relación con el nivel 3 y el nivel 2 con el nivel 3 no tienen ninguna relación, gráficamente sería. (gráfico 5.14)



La transacción familias genera 3 tablas que son: Famil, Famil1 y Famil 2, estas tablas son totalmente formalizadas y controladas su integridad referencial automáticamente; y se pueden observar en el gráfico 5.15.

Gráfico 5.15



Tablas Generadas por la Transacción Familias
Obtenido de Oracle 8 i

Para que Ud. pueda darse cuenta de la diferencia existente entre: niveles anidados y niveles paralelos, se presenta el gráfico 5.16, que es la transacción Proyectos, elaborada con niveles paralelos.

Gráfico 5.16

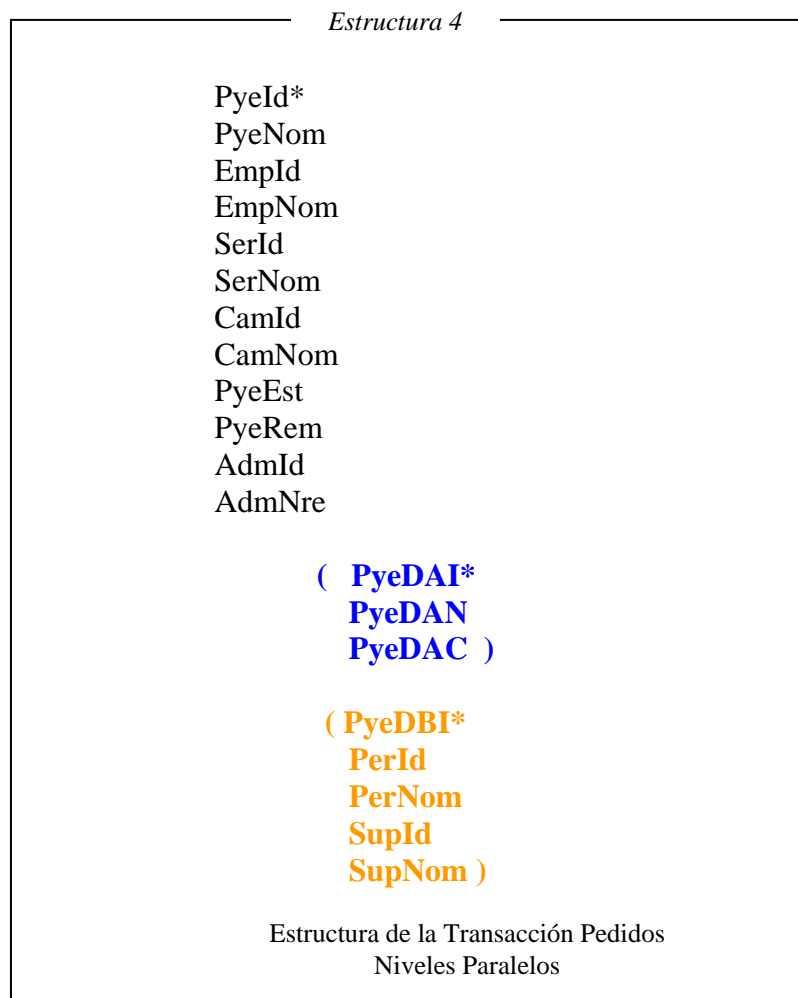


Diseño de la Realidad de la Transacción Proyectos Niveles Paralelos

La transacción proyectos tiene tres niveles, uno que es la información del proyecto, y dos mas a manera de tablas, los cuales son niveles paralelos ya que el dos y el tres no tienen ninguna relación entre si, esto se puede divisar de una mejor manera en la estructura de la transacción, que esta representada en la estructura 4.

Esta transacción funciona de la siguiente manera: Se empezará ingresando el identificador de el proyecto, con su respectivo nombre fecha y estado, este número y nombre lo dan las empresas contratantes como Petroproducción, luego se llamará a unos parámetros foráneos para almacenar los datos del campo de operación y el servicio prestado por DYGOIL, después se ingresarán los responsables de la empresa contratante y de DYGOIL.

En los niveles paralelos, o tablas como se observa en el gráfico, van: el la primera tabla (primer nivel) el personal de supervisión de la empresa contratante y en la segunda tabla (segundo nivel) el personal de supervisión de DYGOIL. [MAN007]

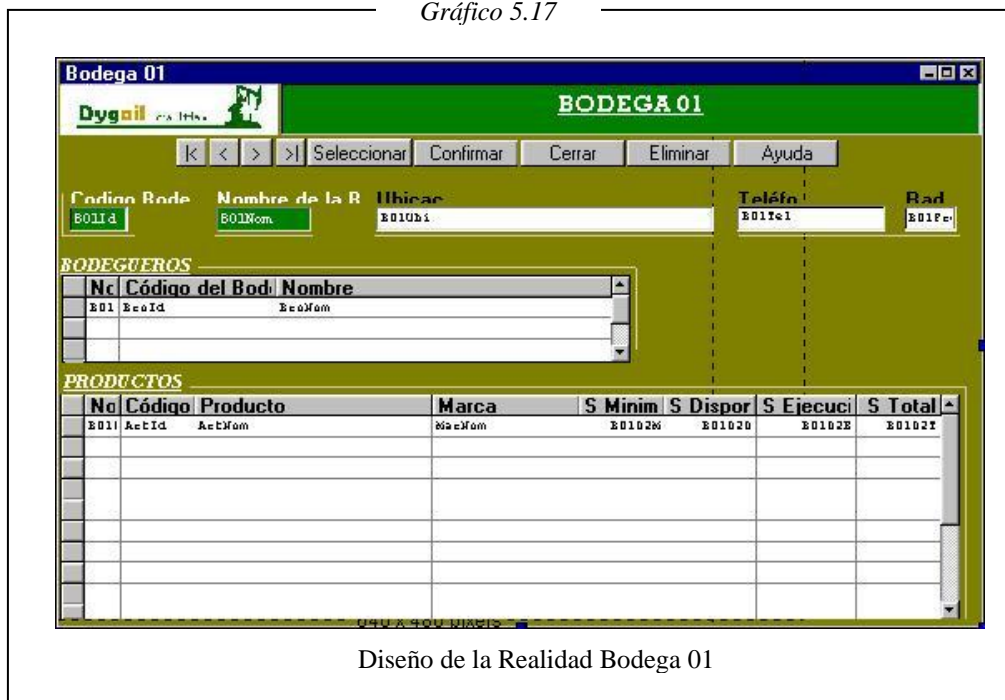


5.4.4. DISEÑO DE BODEGAS

Siguiendo con los diseños de la realidad, surge la necesidad de crear las bodegas, las cuales van a ser diseñadas cada bodega como un módulo diferente, estas tendrán: su código, detalle, descripción, ubicación, también tendrán su nivel de bodegueros, y su módulo de productos, estos dos módulos serán de tipo paralelo.

Para cada producto correspondiente a determinada bodega se le asignará otra clave primaria, la cual sirve solo para la bodega, ya que si se asigna como clave primaria de ese nivel al código del mismo producto, se estaría al frente de la transacción productos, pero en su totalidad, mas no se visualizaría los productos exclusivos de la bodega, se documentará solo el diseño de la Bodega 01, ya que las otras bodegas tienen la misma arquitectura. (gráfico 0.17)

Gráfico 5.17



A más de los datos propios de la bodega y los bodegueros, esta guarda información, sobre todos los productos que en ella existen, cada cual con una clave primaria propia del nivel de cada bodega, de esta forma permitirá observar solo los datos de dicha bodega en los paneles de trabajo llamados desde cualquier parte.

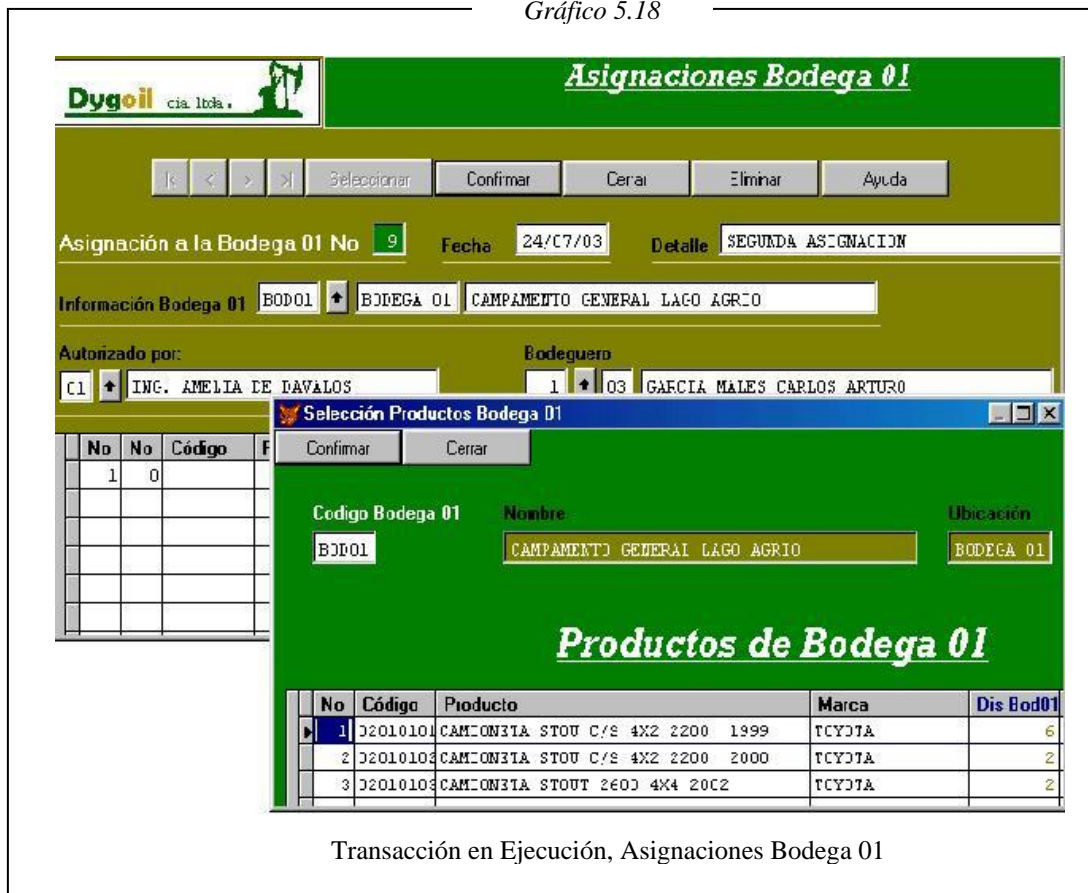
La asignación de productos se la realizará de manera manual, leyendo los productos de la transacción del mismo nombre, esta transacción no afecta en nada al stock, tanto de productos como de bodegas.

La transacción asignaciones de bodegas se encargará de hacer este trabajo, leyendo los productos de bodegas, restando a los de la transacción productos y sumando al stock disponible de bodegas. El gráfico 5.18 hace referencia a este tipo de transacción asignaciones.

5.4.5 ASIGNACIONES A BODEGAS.

En la transacción Asignaciones Bodega 01, se va a asignar un producto el cual está almacenado en bodega 01, es por eso que aparece el panel de trabajo o lista de asignación, selección de productos bodega 01(gráfico 5.19), este panel de trabajo es distinto a los panel de trabajo mostrado en compras o en ventas, ya que ese lee a los productos generales, en cambio este solo lee a los productos de bodega 01.

Gráfico 5.18



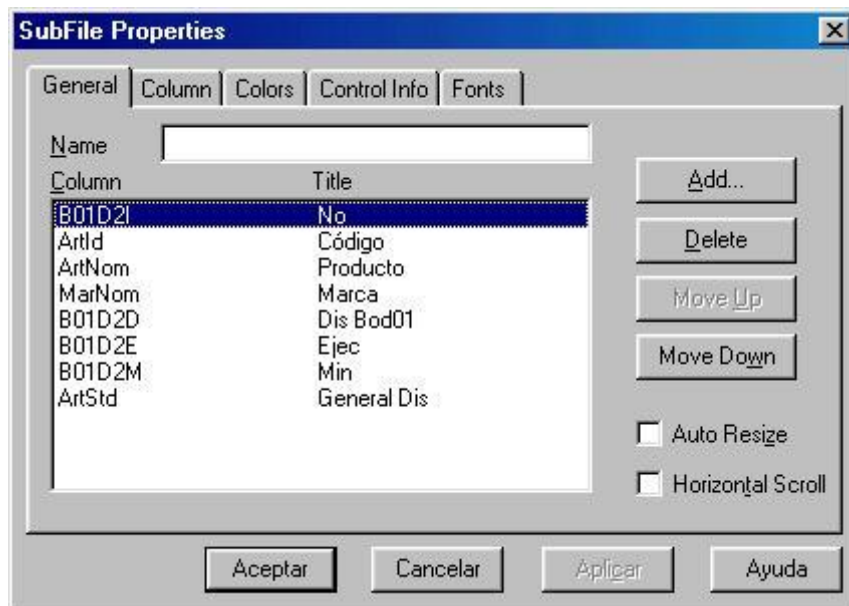
Una vez insertada, la cantidad del producto asignado, esta se suma al stock disponible de bodega 01, y se resta al stock disponible de productos, de la misma forma se suma al stock en ejecución de productos. Con este stock disponible de bodegas se empezara a hacer los movimientos de entrada y salida de los productos, pero dentro de la bodega.

Los movimientos de cada bodega son: órdenes de ingreso, órdenes de egreso, devoluciones y bajas.

Gráfico 5.19



Diseño de la Realidad del Panel de Trabajo: Productos de la Bodega 01

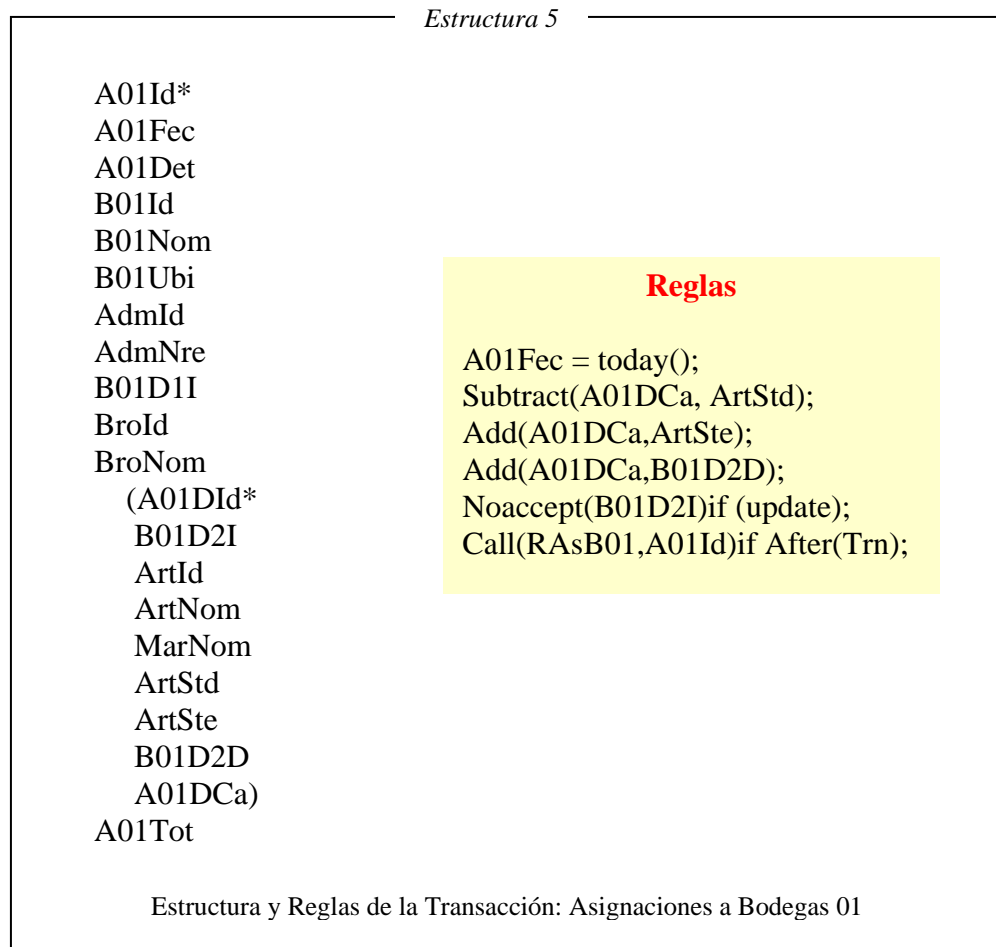


Estructura del Segundo Nivel del Panel: Productos de la Bodega 01

Panel de Trabajo o Lista de Selección de: Productos de la Bodega 01

Se puede observar que en el primer nivel de este panel, se hace un llamado al código de la bodega, clave principal de bodegas, y en el segundo nivel a la clave principal del segundo nivel de bodegas, la cual tiene relacionado a productos, es por eso que solo puede leer los productos asignados a bodega01.

La estructura y las reglas principales de la transacción asignaciones es: (estructura 5)



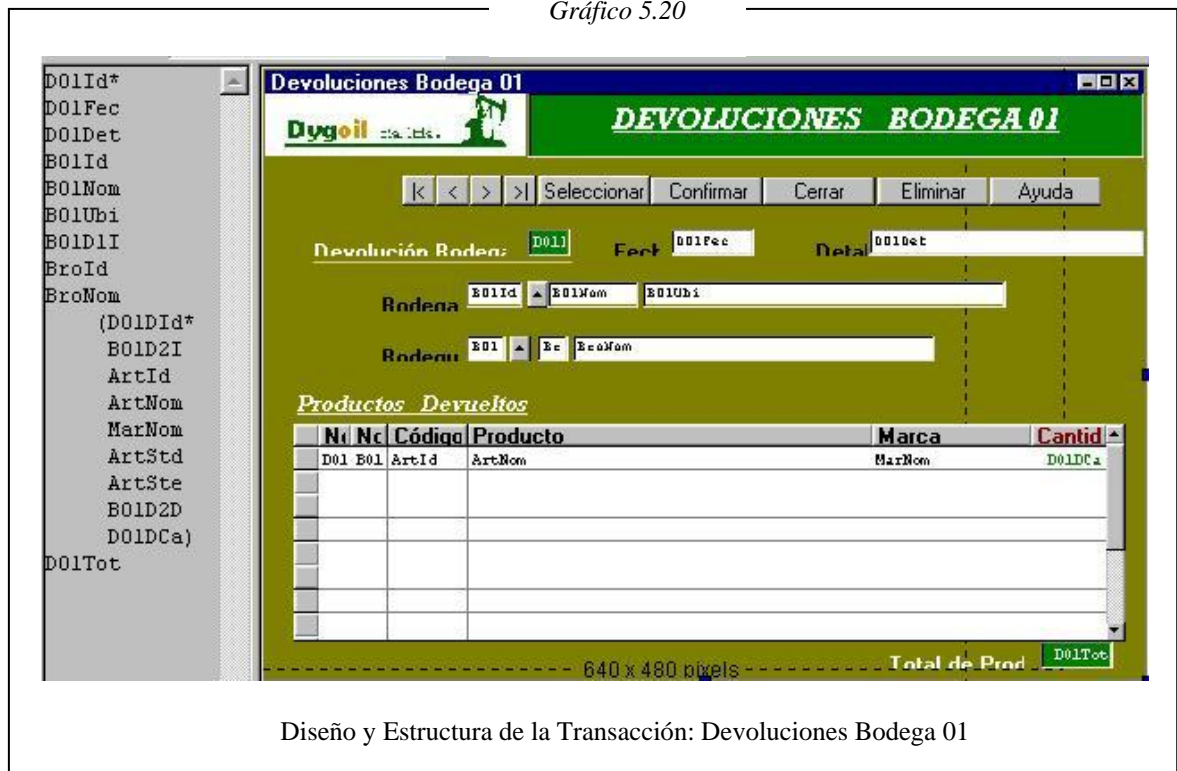
La regla: Subtract(A01DCa, ArtStd); resta la cantidad asignada en asignaciones de bodega 01, al stock disponible de productos, la regla: Add(A01DCa, ArtSte); suma la cantidad asignada por asignaciones de bodega 01 al stock en ejecución de productos y la regla: Add(A01DCa, B01D2D); suma la cantidad asignada al stock disponible de bodegas 01, call hace un llamado a un reporte de impresión cuando la transacción se haya concluido (if aftern (Trn)).

5.4.6 DEVOLUCIONES POR BODEGAS

De igual manera actúan las devoluciones por bodegas, si no que se las realiza de una manera contraria, lo que se sumaba antes ahora se resta y lo que se restaba ahora se suma. Esta transacción nace de una idea, qué pasará con los productos todavía utilizables, designados a cierta bodega, cuando el proyecto haya concluido.

También, qué pasará cuando un producto este inservible, entonces se pensó por medio de los técnicos y los directivos, que se debería devolver a Quito, para que tomen allá una decisión; el diseño más pegado a la realidad de la transacción devoluciones por bodegas es el siguiente: (gráfico 5.20)

Gráfico 5.20



Diseño y Estructura de la Transacción: Devoluciones Bodega 01

Las reglas son las siguientes:

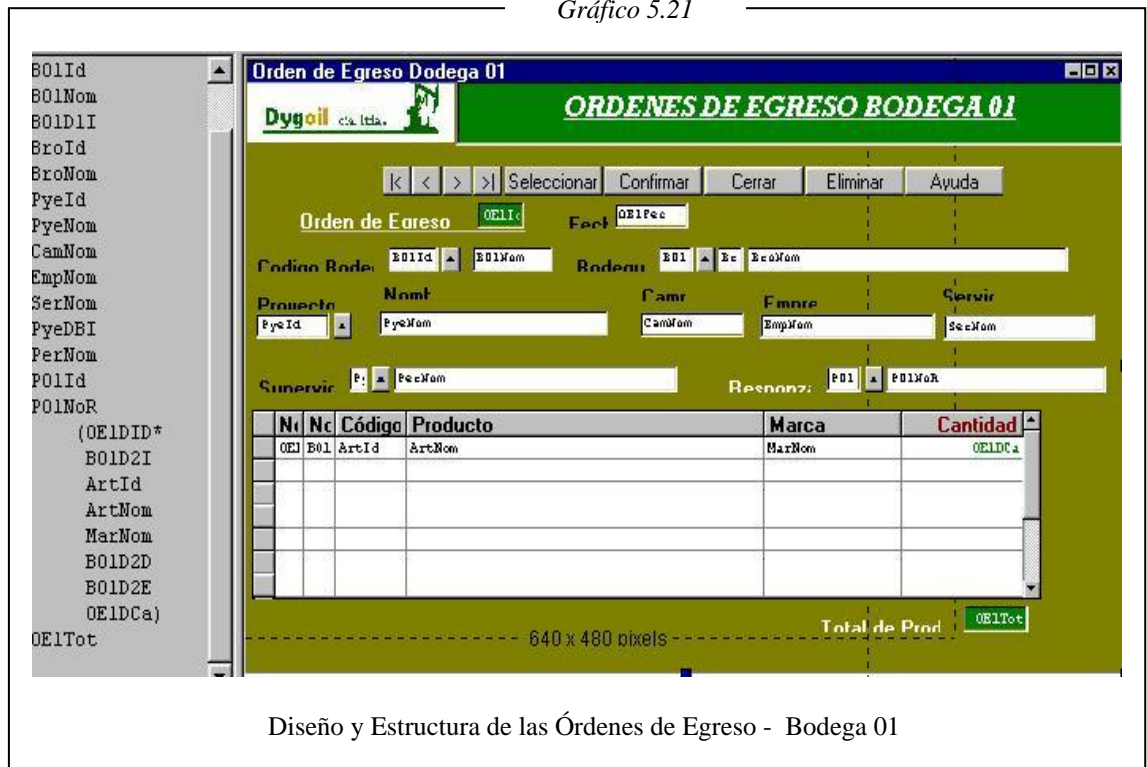
Subtract(D01DCa, B01D2D); resta la cantidad devuelta al stock disponible en bodega 01.
Add(D01DCa, ArtStd); suma la cantidad devuelta al stock disponible de productos.,
Subtract(D01DCa, ArtSte); resta la cantidad devuelta al stock en ejecución de productos.

Todas las transacciones que tienen que ver con bodegas, se generan como módulos de cada bodega, incluso están elaborados en carpetas diferentes, para apuntar a generar solo en la parte donde está la bodega y formar una aplicación y una base de datos distribuidas.

Una vez generado las bodegas conjuntamente con sus productos es la hora de empezar a jugar con ellos mediante órdenes de ingreso y órdenes de egreso de los productos, estos serán realizados, por el personal de cada bodega.

5.4.7 ÓRDENES DE EGRESO E INGRESO

Gráfico 5.21



Diseño y Estructura de las Órdenes de Egreso - Bodega 01

Las órdenes de egreso le afectan directamente a la bodega seleccionada, es por eso que a más de que la transacción lleva su propia clave primaria, está llama a los atributos de la bodega, los cuales tienen productos y stocks, pero de bodega 01, aquí nada tiene que ver el stock de productos general.

En su funcionamiento ya se hace el llamado a los proyectos, ya que se debe tomar en cuenta que si un trabajador, saca determinados materiales o productos, se debe registrar en que proyecto los está utilizando.

Se crea también otra transacción para el personal, pero este personal es el que labora o trabaja con la bodega 01, estos son los responsables de cada material extraído de la bodega. Los responsables a nivel administrativo son: las personas encargadas de la supervisión por parte de DYGOIL pero de cada proyecto.

Cuando un material es sacado a trabajar, se resta el stock disponible de la bodega 01, y se suma al stock en ejecución de la misma bodega, al concluir o al hacer un comit, de la transacción órdenes de egreso, el sistema automáticamente imprime un recibo, el cual es aceptado por el personal de trabajadores responsable, este le servirá para realizar su orden de ingreso.

Las reglas más importantes son:

Subtract(OE1DCa,B01D2D); resta al stock disponible de bodega01

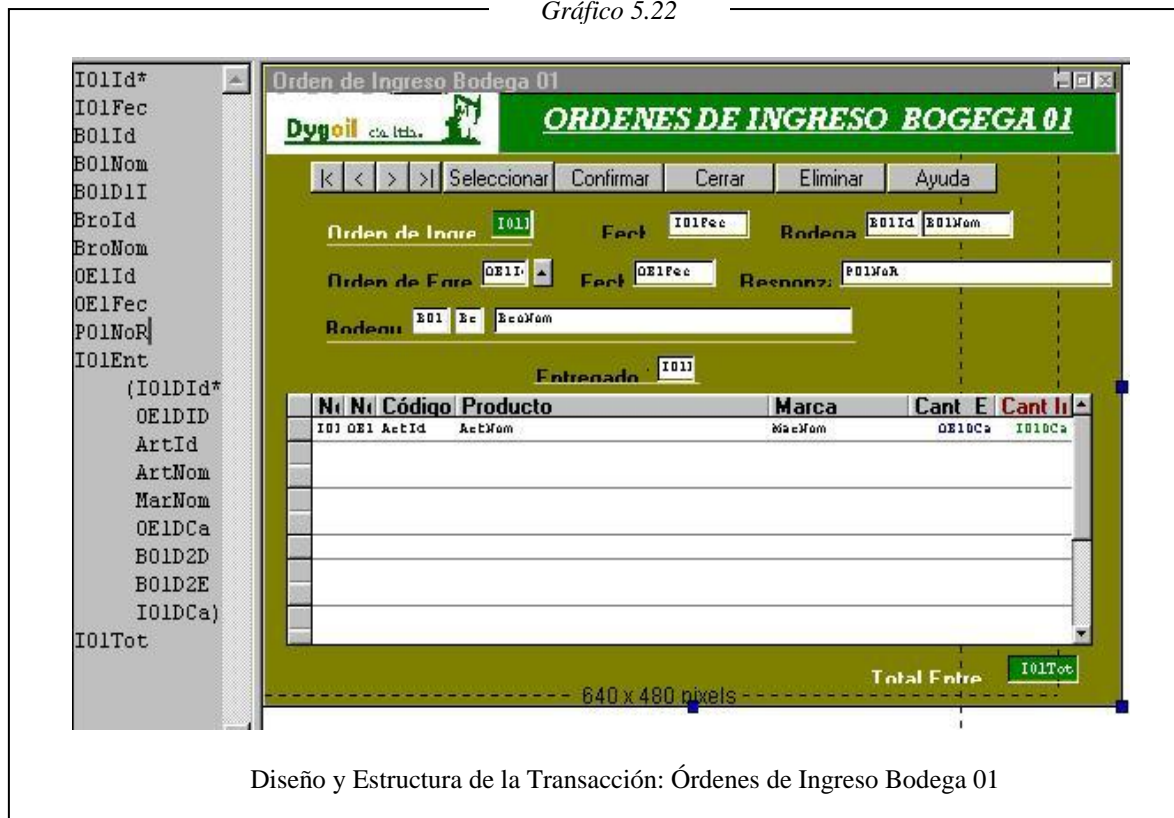
Add(OE1DCa,B01D2E); suma al stock en ejecución de bodega 01.

En las órdenes ingreso sucede lo contrario, el trabajador llevando el número de egreso, entra al sistema realizando otra transacción la cual es : órdenes de ingreso, esta lee todos los movimientos de las órdenes de egreso y en ese momento ingresa todos los productos prestados por medio de la orden de egreso, sus reglas mas importantes son:

ADD(I01DCa, B01D2D); suma al stock disponible de la bodega, los productos entregados.
 Subtract(I01DCa , B01D2E); resta al stock en ejecución de la bodega.

Esta transacción también imprime un recibo para constancia del trabajador que a entregado los productos prestados, su estructura y diseño es el siguiente (gráfico 5.22).

Gráfico 5.22



5.4.8 BAJAS DE PRODUCTOS: GENERALES Y POR BODEGAS

Otra de las ideas que surgieron al diseñar este sistema, fue la baja de productos, tanto desde Quito y afectando a los productos generales, como desde cualquier bodega afectando a los productos generales (gráfico 5.23).

Cuando un producto: ya no sirve, está caduco, a cumplido su vida útil o simplemente está dañado, aparece la necesidad de darle de baja, en este caso la transacción bajas generales, da de baja a un producto, pero desde la bodega principal ubicada en Quito.

Sus principales reglas son: Subtract(BagDCa, ArtStd); la cual quita la cantidad de productos al stock disponible de la base de productos generales.

La transacción bajas por bodegas, quita o da de baja un producto afectando: tanto a los productos generales, como al stock disponible de la bodega así:

Subtract(Ba1DCa,B01D2D); resta la cantidad de productos al stock disponible de bodega 01.

Subtract(Ba1DCa,ArtSte); resta la cantidad de productos al stock en ejecución de productos generales.

Resta de las existencias en ejecución generales, ya que cuando se asignó cierta cantidad de productos desde productos generales, esta cantidad pasa a sumarse al stock en ejecución de los mismos productos generales.

Todas estas transacciones también imprimen automáticamente un recibo para constatación en papel, de los interesados.

Gráfico 5.23



5.4.9 DISEÑO DE LA TRANSACCIÓN VENTAS.

El mundo del petróleo, el momento que menos se piensa surge un problema y se tiene que solucionarlo en el menor tiempo posible, en la gran mayoría de estos casos se requiere importación de repuestos y accesorios, situación que lleva esperar una cierta cantidad de tiempo considerable, es cuando se recurre a la compra de estos accesorios a otras empresas petroleras.

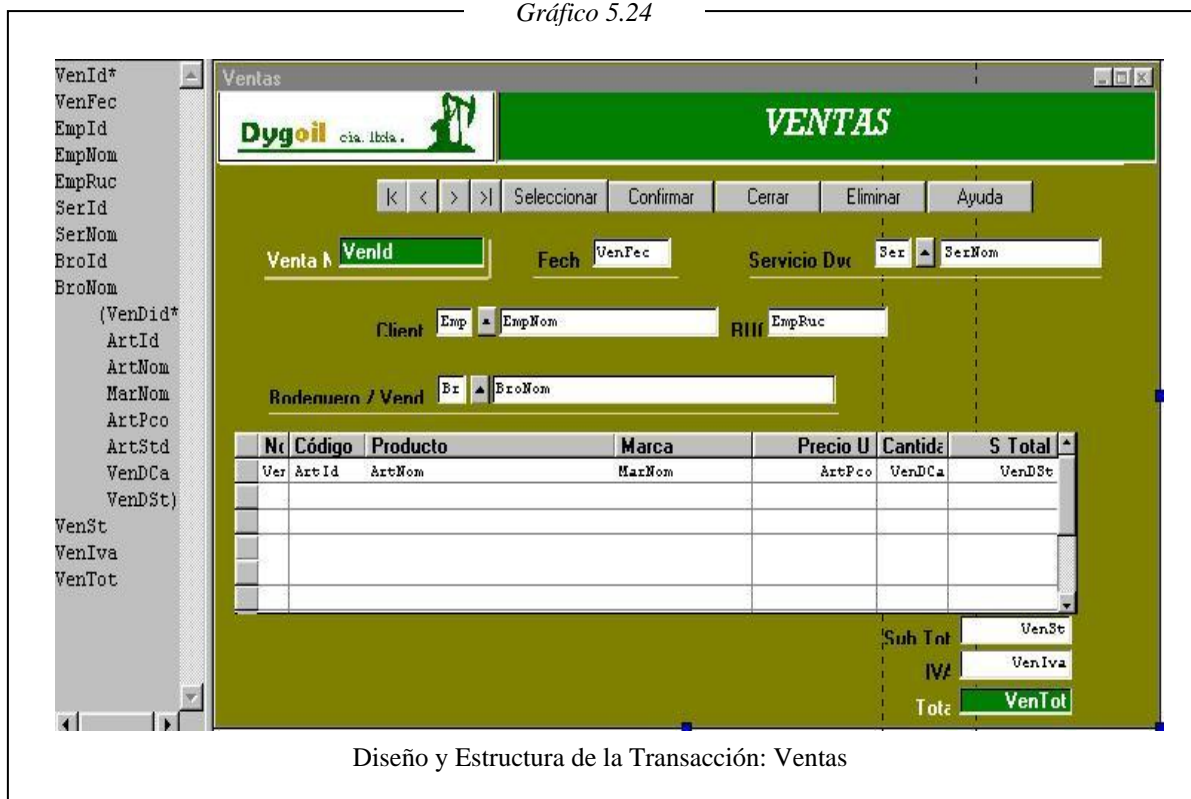
Es por eso que DYGOIL a creado el servicio de: Provisión de Insumos y Accesorios Petroleros, el cual cumple el papel de vendedor de insumos petroleros.

Este módulo de ventas también sirve para la venta de petróleo, servicio también instaurado por esta empresa.

El comportamiento de esta transacción es muy fácil, ya que simplemente resta los productos vendidos del stock disponible de la base de productos generales, obviamente la transacción cuenta con un número de factura, iva y todos los requerimientos que tiene una factura legal; al terminar esta transacción el sistema automáticamente imprime la factura correspondiente.

Sus reglas principales son: Subtract(VenDCa,ArtStd); resta la cantidad vendida del stock disponible y call(RVenta1,VenId)if after(Trn); llama a la factura una vez terminada la transacción, su estructura y diseño están en el gráfico 5.24.

Gráfico 5.24



Es así como se a diseñado y documentado las transacciones y los procesos más importantes de esta aplicación, ya que propiamente se tiene alrededor de 45 transacciones, 76 tablas, y una cantidad considerable de reportes y paneles de trabajo, los cuales se los podrá observar de una forma más objetiva cuando abra el propio aplicativo.

5.5. IMPLEMENTACIÓN DE LA APLICACIÓN

Como se había quedado anteriormente, la aplicación principal está elaborada con:

Novell Netware 5.1, Oracle 8i, Windows 98, Visual Foxpro y Genexus 7.0. La implantación del sistema dará énfasis a las propiedades de Genexus con respecto a esta solución.

Dicho de otra manera, una vez analizados los pasos de diseño solo toca saber cuales son las propiedades a cambiar en la herramienta para que pueda generar una u otra arquitectura.

5.5.1 IMPLANTACIÓN: NOVELL - ORACLE WINDOWS 98 VISUAL FOXPRO.

Requerimientos de Hardware.

- Servidor, Pentium III, de 1.0 Ghz en adelante, con 256 MB de memoria ram como mínimo, disco de 5 Gb en adelante. (Servidor: Novell Oracle)
- PC, Pentium III, de 1.0 Ghz en adelante, con 128 MB de memoria ram como mínimo, disco de 10 Gb en adelante. (Cliente: Genexus Vfoxpro).
- Hab pequeño de 8 puertos para la red, si no es posible, enlazar por medio de cable cruzado.

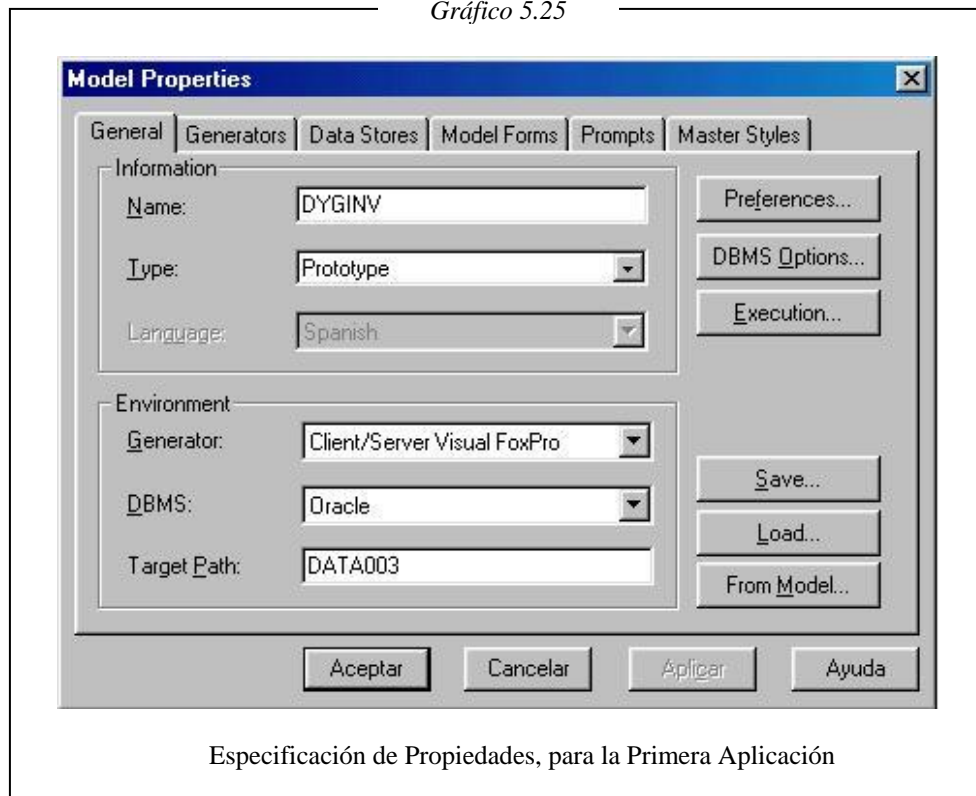
Requerimientos de Software.

- Novell Netware 5.1
- Oracle 8i 8.1.5.0.4.
- Windows 98.
- Microsoft Visual Studio 6.0 Completo.
- Cliente Oracle.
- Cliente Novell
- Internet explorer 6.0
- Drivers ODBC para Oracle.
- Parches y actualizaciones de cada producto.
- Genexus 7.0 generador Visual Foxpro.

Una vez conseguidos estos requerimientos, se va a cambiar las propiedades por defecto que vienen en la herramienta (gráfico 5.25), estas propiedades son de tipo general, las cuales hacen mención al tipo de generador utilizado, al tipo de base de datos, en si son las especificaciones principales de Genexus para generar aplicaciones en diferentes arquitecturas.

Si se quiere cambiar de un modelo a otro se llama a un nuevo modelo y se especifica las propiedades en la pantalla del gráfico 5.25.

Gráfico 5.25



Especificación de Propiedades, para la Primera Aplicación

Aquí se puede observar: el nombre de la aplicación, Name: DYGIVN, el tipo, Type: Prototype, el generador, Client/Server Visual Foxpro, la base de datos o DBMS, Oracle, que son la estructura misma de las propiedades de la aplicación.

Luego de esto se debe insertar las: preferencias (preferences) y las opciones de la base de datos (DBMS options) las cuales son los aspectos fundamentales a cambiar para el funcionamiento de determinada arquitectura.

Las preferencias y las opciones de la base de datos, son el único proceso un poquito tedioso cuando el desarrollador está empezando a manejar Genexus con determinada arquitectura, este es el proceso que requiere de más investigación, ya que son temas con bibliografía restringida, solo para clientes comerciales de la herramienta, es por eso que las versiones triler o relace no tienen ningún tipo de soporte con respecto a estos temas. [MAN001] [MAN003] [MAN004]

5.5.2. DBMS OPTIONS PARA ORACLE

<i>Access Technology Settings</i>	
Access Technology to Set	ODBC
<i>Connection Information</i>	
Data Source Name	Orafox
Database Name	
Connect to Server	At Application startup
Show Connection Dialog	Yes
Additional Connection string attributes	
<i>Creation / Reorganization Information</i>	
Database Schema	Juan
Primary Key Definition	Primary key
Declare Referential Integrity	Yes
Default Tables Storage Area	Data01
Default Indices Storage Area	Index01
Default Temporary Storage Area	
Generate COMMENT ON statement	No
<i>Database Information</i>	
Generate FOR UPDATE clause	Yes
Declare Varchar as varchar2	Yes

5.5.3. PREFERENCIAS

Generator information	
Tables in server	All tables are in the server
Local tables if tables in server=	*
Index type for local tables	Use .IDX indexes *
Reorganize Server Tables	Yes*
General	
Join management	Join tables on the server *
Join Type	Use default for server*
Transactional Integrity	All files *
Isolation Level	Read Committed*
Initialize not referenced attribute	No*
Generate null for nullvalue()	No*
Optimization	
Delete groups	Yes*
Aggregate groups	Yes*
Copy table groups	If no unique index*
Hints	
Fast first rows	Yes*
Transaction's dialog locking sch	Pseudo-conversational (updated tables only)
Confirm	Do not confirm each action
List of remote programs (ODBC)	*
Default remote procedure location	*
AS/400	
Library list	*
OS for AS/400 version	V2R2*
Field exit	Tab*
Cancel button action	Exit form*
Esc key action	Exit level*
Autocenter objects in (0,0)	No*
Show status bar	Depending on object type*
Error if control value invalid	Yes*
End Tab Page Action	Exit tab*
Combo/Listbox Style	Show Descriptions Only*
Show Form	Before Start Event*
Color In Read-Only Fields	Original*
Click to sort in subfiel (workpanel)	Enabled*

<i>PRINTING</i>	
Show Printer Dialog on Reports	Yes*
Text grid size	GeneXus Grid*
<i>REPORT VIEWER</i>	
Report viewer	Yes*
Maximize	No*
Always on top	Yes*
Modal	Yes*
<i>FORMAT</i>	
Decimal separator	Windows settings*
Date Format	English*
Time Format	Language dependent*
Fist year of 20th century	40*
Set Exact	Off*
<i>TEMPORARY FILES</i>	
Temp. Files Environment Variable	*
Subfile Work Area	0*
Subfile Work Area	0*
<i>TYPE CHECKING</i>	
Check Type Errors	Yes*
Functions	Error on non-standard functi
Replication Log File Name	*
<i>OTHER</i>	
Prompt key	F4*
Object menu bar	Yes*
Check indeces existence	No*
Calculator on Rbutton	Yes*
Interface size	Normal*
Main objects build	Include with caller*

5.5.3. OBJETOS GENERADOS EN LA APLICACIÓN.

Una vez insertado estas preferencias, se puede visualizar los objetos generados en esta aplicación, se documentará la aplicación con una sola bodega junto a los objetos generados, posteriormente se documentará con las otras bodegas, para poder especificar los cambios prototipo / producción enfatizando a que no se dañe el sistema por aumento de transacciones. Todavía no se documentará lo que es reportes, ya que solo se esta presentando los objetos generados por defecto, los gráficos 5.26 y 5.27 muestran los objetos generados.

Gráfico 5.26



✓ Abo01	trn : Asignaciones Bodega 01
✓ Admini	trn : Administración Dygoil
✓ Adq01	rpt : Adquisición 01
✓ Adqui	trn : Adquisiciones
✓ Artic	trn : Artículos
✓ AsB01	rpt : Asignaciones Bodega 01
✓ Atipo	trn : Tipo de Artículo
✓ BaGen	trn : Bajas Generales
✓ BagGen	rpt : Bajas Generales
✓ Baj01	trn : Bajas Bodega 01
✓ BB01	rpt : Bajas Bodega 01
✓ Bdgro	trn : Bodegueros
✓ Bod01	trn : Bodega 01
✓ Califi	trn : Calificación
✓ Campos	trn : Campos de Operación
✓ CargAd	trn : Cargos Administrativos
✓ DAdqu	trn : Devoluciones de Adquisiciones
✓ DBod01	rpt : Devoluciones Bodega 01
✓ DeBo1	trn : Devoluciones Bodega 01
✓ DevAd	rpt : Devoluciones de Adquisiciones
✓ DeVen	rpt : Devoluciones de Ventas
✓ DeVen	trn : Devoluciones de Ventas
✓ EGR01	trn : Orden de Egreso Dodega 01
✓ Empres	trn : Empresas Relacionadas
✓ Famil	trn : Familias
✓ Gx0020	wkp : Selection List TIPEMP
✓ Gx0030	wkp : Lista de Selección EMPRES

Objetos Generados por Genexus - con una sola Bodega (Primera Parte)

Gráfico 5.27

✓ Gx0040	wkp	: Selection List MEDIDA
✓ Gx0050	wkp	: Lista de Selección SERVIC
✓ Gx0060	wkp	: Selection List CAMPOS
✓ Gx0070	wkp	: Selection List FAMIL
✓ Gx0081	wkp	: Selection List FAMIL1
✓ Gx0090	wkp	: Selection List MARCAS
✓ Gx00B2	wkp	: Selection List FAMIL2
✓ Gx00C0	wkp	: Lista de Selección ADMINI
✓ Gx00D0	wkp	: Selection List CARGAD
✓ Gx00E0	wkp	: Selection List CALIFI
✓ Gx00F0	wkp	: Lista de Selección PROVID
✓ Gx00G0	wkp	: Lista de Selección BDGRO
✓ Gx00J0	wkp	: Selection List UBICA
✓ Gx00K1	wkp	: Selection List UBICA1
✓ Gx00L0	wkp	: Selection List ARTIC
✓ Gx00M0	wkp	: Selection List ATIPO
✓ Gx00N0	wkp	: Selection List PERSON
✓ Gx00O0	wkp	: Selection List SUPERV
✓ Gx00P0	wkp	: Selection List PROYE
✓ Gx00Q1	wkp	: Selection List PROYE1
✓ Gx00R1	wkp	: Selection List PROYE2
✓ Gx00S0	wkp	: Selection List BOD01
✓ Gx00T1	wkp	: Selection List BOD011
✓ Gx00U1	wkp	: Selection List BOD012
✓ Gx00V0	wkp	: Lista de Selección ADQUI
✓ Gx00W0	wkp	: Lista de Selección TIPADQ
✓ Gx00X1	wkp	: Lista de Selección ADQUI1
✓ Gx00Y0	wkp	: Lista de Selección DADQU
✓ Gx00Z1	wkp	: Selection List DADQU1
✓ Gx0100	wkp	: Lista de Selección VENTA
✓ Gx0111	wkp	: Lista de Selección VENTA1
✓ Gx0120	wkp	: Lista de Selección DEVEN
✓ Gx0131	wkp	: Selection List DEVEN1
✓ Gx0140	wkp	: Selection List ABO01
✓ Gx0151	wkp	: Selection List ABO011
✓ Gx0160	wkp	: Selection List DEBO1
✓ Gx0171	wkp	: Selection List DEBO11
✓ Gx0180	wkp	: Selection List PERBO1
✓ Gx0190	wkp	: Selection List EGR01
✓ Gx01A1	wkp	: Selection List EGR011
✓ Gx01B0	wkp	: Selection List ING01
✓ Gx01C1	wkp	: Selection List ING011
✓ Gx01D0	wkp	: Selection List BAGEN
✓ Gx01E1	wkp	: Selection List BAGEN1
✓ Gx01F0	wkp	: Selection List BAJ01
✓ Gx01G1	wkp	: Selection List BAJ011
✓ ING01	trn	: Orden de Ingreso Bodega 01
✓ Marcas	trn	: Marcas
✓ Medida	trn	: Unidades de Medida
✓ OInB01	rpt	: Orden de Ingreso Bodega 01
✓ OrEgB1	rpt	: Orden de Egreso Bodega 01
✓ PerBo1	trn	: Personal Bodega 01
✓ Person	trn	: Personal
✓ Proved	trn	: Proveedores
✓ Proye	trn	: Proyectos
✓ Servic	trn	: Servicios DYGOIL
✓ Superv	trn	: Supervisión
✓ TipAdq	trn	: Tipo de Adquisición
✓ TipEmp	trn	: Tipos de Empresas
✓ Ubica	trn	: Ubicaciones
✓ Venta	trn	: Ventas
✓ Venta1	rpt	: Factura de Ventas

Objetos Generados Genexus - Con una sola Bodega (Segunda Parte)

En estos gráficos se puede ver cada objeto generado, por Genexus, las transacciones tienen su identificador que es: trn, los reportes rpt, y los work panels: wkp, hasta el momento se tienen 32 transacciones y 50 paneles de trabajo, en este simple ejemplo el programador ya se ahorra de programar las 50 listas de selección, ya que el case los genera automáticamente.

5.5.4. DIAGRAMAS RELACIONALES GENERADOS POR GENEXUS

Genexus genera dos tipos de diagramas relacionales, conocidos de otra forma como: diagramas entidad relación, el primer diagrama se lo puede generar por medio de transacciones, el cual relaciona exactamente a las transacciones de la aplicación y el segundo relaciona a las tablas generadas por medio del diseño de las transacciones.

Todos estos diagramas son generados automáticamente, el desarrollador no tiene que estar viendo cual tabla o cual transacción está asociada una con otra, sino que la herramienta recoge la información guardada en la base de conocimiento perteneciente a la aplicación, y por medio de esta genera todo tipo de relación en un modo gráfico comúnmente conocido como diagramas entidad relación.

Los diagramas transaccionales se los puede considerar como diagramas entidad – relación de entidades y los diagramas de tablas se los considera entidad – relación de tablas.

Los diagramas que se presentan a continuación abarcan no solo a la aplicación con la bodega01 sino que abarcan a la aplicación con las tres bodegas; donde se tiene 76 tablas y 45 transacciones.

5.5.4.1. DIAGRAMA ENTIDAD RELACIÓN POR TRANSACCIONES

5.5.4.2. DIAGRAMA ENTIDAD RELACIÓN POR TABLAS.

5.5.5 BASE DE DATOS VISTA DESDE GENEXUS

Gráfico 5.28

```
Abo01 : Abo01
Abo011 : Abo011
Admini : Admini
Adqui : Adqui
Adqui1 : Adqui1
Artic : Artic
Atipo : Atipo
BaGen : BaGen
BaGen1 : BaGen1
Baj01 : Baj01
Baj011 : Baj011
Bdgro : Bdgro
Bod01 : Bod01
Bod011 : Bod011
Bod012 : Bod012
Califi : Califi
Campos : Campos
CargAd : CargAd
DAdqu : DAdqu
DAdqu1 : DAdqu1
DeBo1 : DeBo1
DeBo11 : DeBo11
DeVen : DeVen
DeVen1 : DeVen1
EGR01 : EGR01
EGR011 : EGR011
Empres : Empres
Famil : Famil
Famil1 : Famil1
Famil2 : Famil2
ING01 : ING01
ING011 : ING011
Marcas : Marcas
Medida : Medida
PerBo1 : PerBo1
Person : Person
Proved : Proved
Proye : Proye
Proye1 : Proye1
Proye2 : Proye2
Servic : Servic
Superv : Superv
TipAdq : TipAdq
TipEmp : TipEmp
Ubica : Ubica
Ubica1 : Ubica1
Venta : Venta
Venta1 : Venta1
```

Base de Datos - Vista Desde GENEXUS

Gráfico 5.29

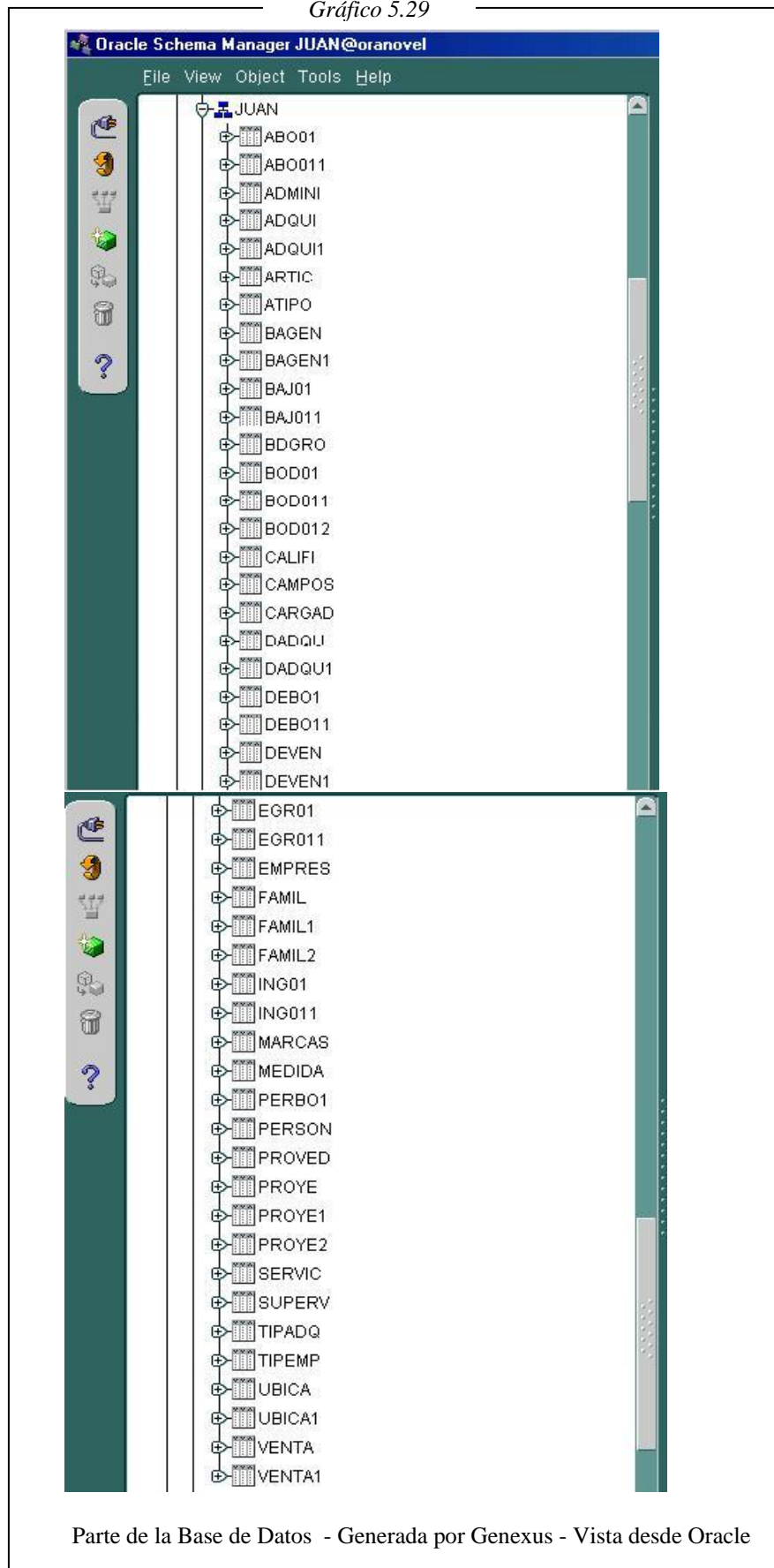
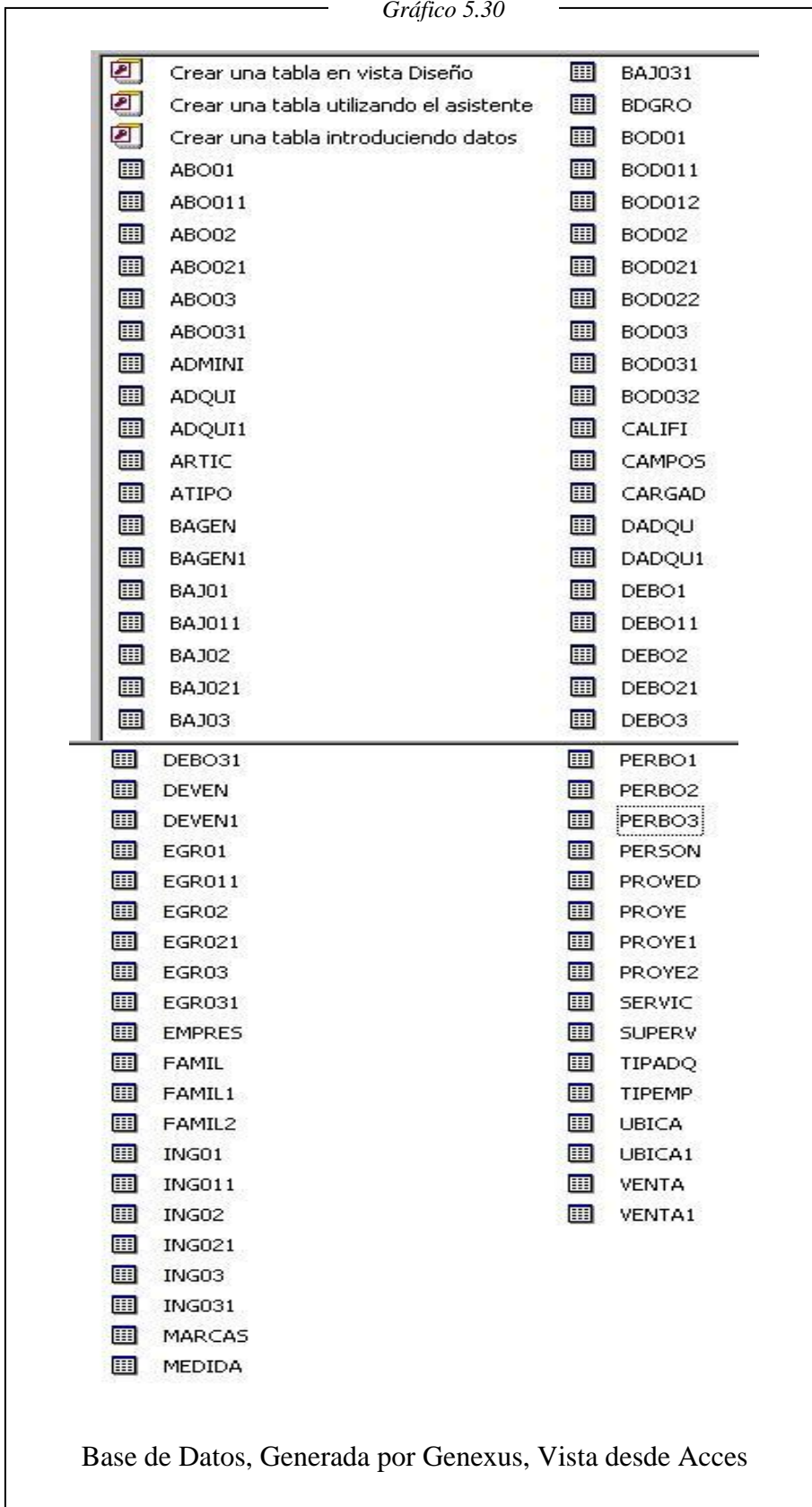


Gráfico 5.30



Se presenta estas tres vistas de la base de datos, para que el desarrollador de aplicaciones observe, que no necesita cambiar absolutamente nada para generar la misma aplicación en diferentes bases de datos, como se puede observar en la base de datos generada por Genexus y aplicada en: oracle, sql y a acces y su programación no cambia absolutamente nada.

Estas son las grandes ayudas que presenta una Herramienta CASE, sin la utilización de las mismas tocaría volver a programar nuevamente gran parte de la aplicación para pasar de una arquitectura a otra.

Las empresas exitosas de hoy en día utilizan Herramientas CASE, porque las ventajas que brindan son palpables a simple vista.

5.6. PROTOTIPO / PRODUCCIÓN

El cambio prototipo producción es un cambio de forma, ya que lo que el sistema desarrollado en prototipo es un replica exacta de la aplicación en producción. Simplemente se debe cambiar las propiedades del modelo de prototipo a producción y la aplicación esta netamente en producción.

El gráfico 5.31 muestra una transacción ya en ejecución, este modo ejecución puede ser visto desde aplicaciones en modo prototipo y transacciones modo producción.

Producción simplemente es el prototipo patentado, no existe ningún problema si su empresa trabaja en prototipo como en producción, claro que es muy recomendable trabajar con producción, para que en prototipo solo se haga las pruebas de los nuevos cambios a la aplicación y cuando estos cambios ya han sido probados y reprobados es la hora de generarlos en producción.

Se habla de prototipos y producción a nivel de aplicaciones corporativas, porque estas aplicaciones no son estables, cada día la empresa tiene nuevos requerimientos, nuevos módulos, nuevas ideas, etc. Y si la aplicación es una solución cerrada, la cual no facilita la generación de prototipos, no facilita la reutilización de sus módulos, etc, simplemente el empresario tiene que volver a comprar una nueva aplicación cada vez que su empresa necesite algo más en sus sistemas.

Cuando el director del proyecto, presenta hacia su jefe una aplicación clara de lo que quiere realizar (prototipo), es mas fácil ver las posibles ventajas del mismo, lo cual le permite tomar decisiones de una forma mas rápida, este prototipo debe ser capaz de utilizar los sistemas adquiridos anteriormente, capaz de que no se pierdan los datos ni se desperdicie el tiempo y los costos asignados a los anteriores sistemas. En si el prototipo es el diseño mismo de la realidad, la imagen de la transacción del gráfico 5.31 es el mismo en prototipo o producción.

Gráfico 5.31

Dygoil cia. ltda. **VENTAS**

Venta No: Fecha: Servicio Dygoil:

Cliente: RUC:

Bodeguero / Vendedor:

No	Código	Producto	Marca	Precio U	Cantidad	S Total
1	03020501	MOTOR 8400 HP I MULTIPUNTO	KUMINS	26000.00	2	52000.00
2	08014603	LLAVES DE COMBO 45 X 60	STANLEY	5850.00	2	11700.00
0				0.00	0	0.00

Sub Total:
 IVA:
 Total:

Transacción en Ejecución - Producción

5.7 APLICACIÓN PARA EL WEB.

Como se dijo anteriormente se desarrollara una aplicación similar demostrativa con Genexus 7.5 Trial Version para ambientes Internet, esta aplicación se la realizará con: Windows 98 con servidor Personal Web o Windows 2000 Server con Internet Information Server, el lenguaje de programación es Visual Basic 6.0.

El requisito principal para que este sistema, generado con genexus 7.5 funciones es internet explorer 6.0 ya que desde la versión 7.5 de Genexus solo se puede generar aplicaciones siempre y cuando cumpla este requisito.

Se demostrará una o dos transacciones, primeramente en modo windows y luego en modo web, para que Ud. vea que no existe ningún cambio en el fondo de las estructuras de estas transacciones.

El gráfico 5.32 muestra la misma transacción productos, que en este caso se le va a nombrar como artículos, generada para modo Windows, y el gráfico 5.33 muestra a su estructura.

Gráfico 5.32

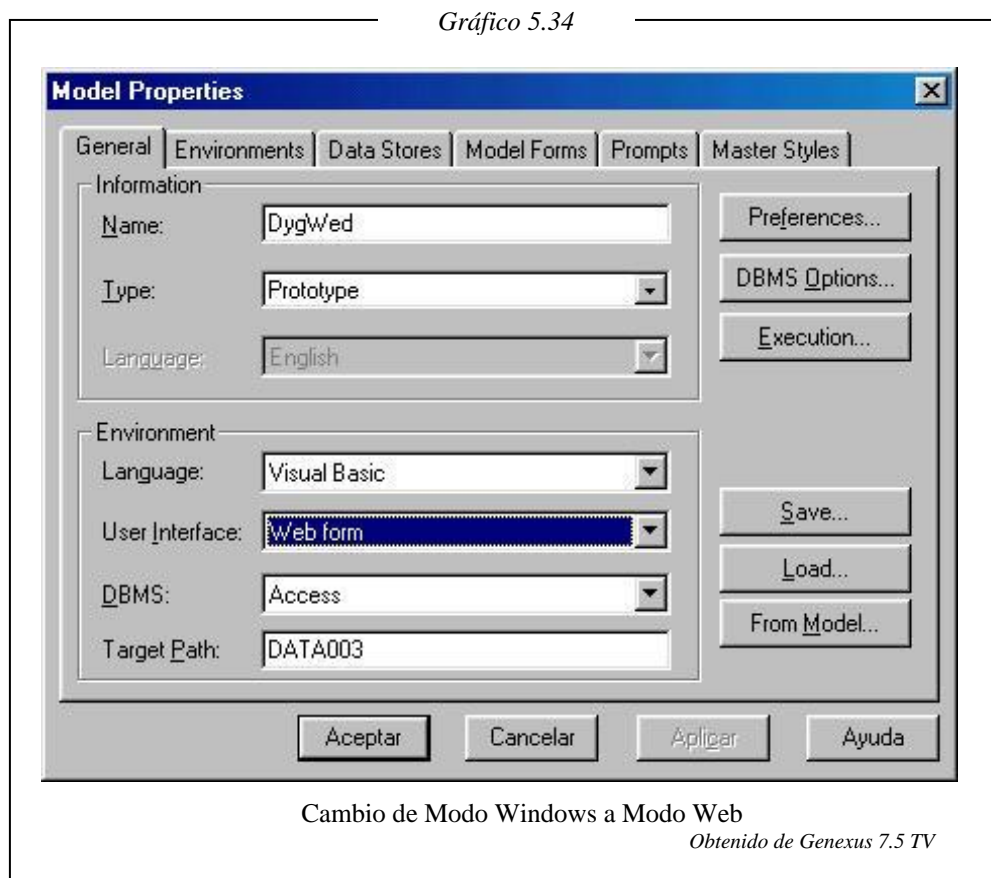
Transacción Artículos (Modo Windows)
Obtenido de Genexus 7.5 TV

Gráfico 5.33

Structure	Type	Description	Formula
ARTICULOS			
ArtId	C(8)	Artículo ID	
CatId	C(2)	Categoría ID	
CatNom	C(25)	Nombre de la Categoría	
CatScId	C(2)	SCald	
CatScNom	C(20)	Nombre SC	
MarId	C(2)	Marca ID	
MarNom	C(15)	Nombre de la Marca	
ArtNom	C(35)	Nombre del Artículo	
ArtDes	L(40)	Descripción	
ArtMed	C(8)	Medida	
ArtSMi	N(5)	Stock M	
ArtSAc	N(5)	Stock A	
ArtSTr	N(5)	Stock Trabajando	
fx ArtTo	N(5)	Stock Total	ArtSAc + ArtSTr
ArtPUC	N(10.2)	Precio Ultima Compra	
ArtFUC	D(8)	Fecha Ultima Compra	
fx ArtPC	N(10.2)	Precio Comercial	ArtPUC * 1.30
ArtCon	N(1)	Disponible	

Estructura de la Transacción Artículos
Obtenido de Genexus 7.5 TV

Para generar soluciones web con genexus 7.5 en adelante, basta con realizar un cambio en las propiedades de la aplicación de modo windows a modo web. (gráfico 5.34).

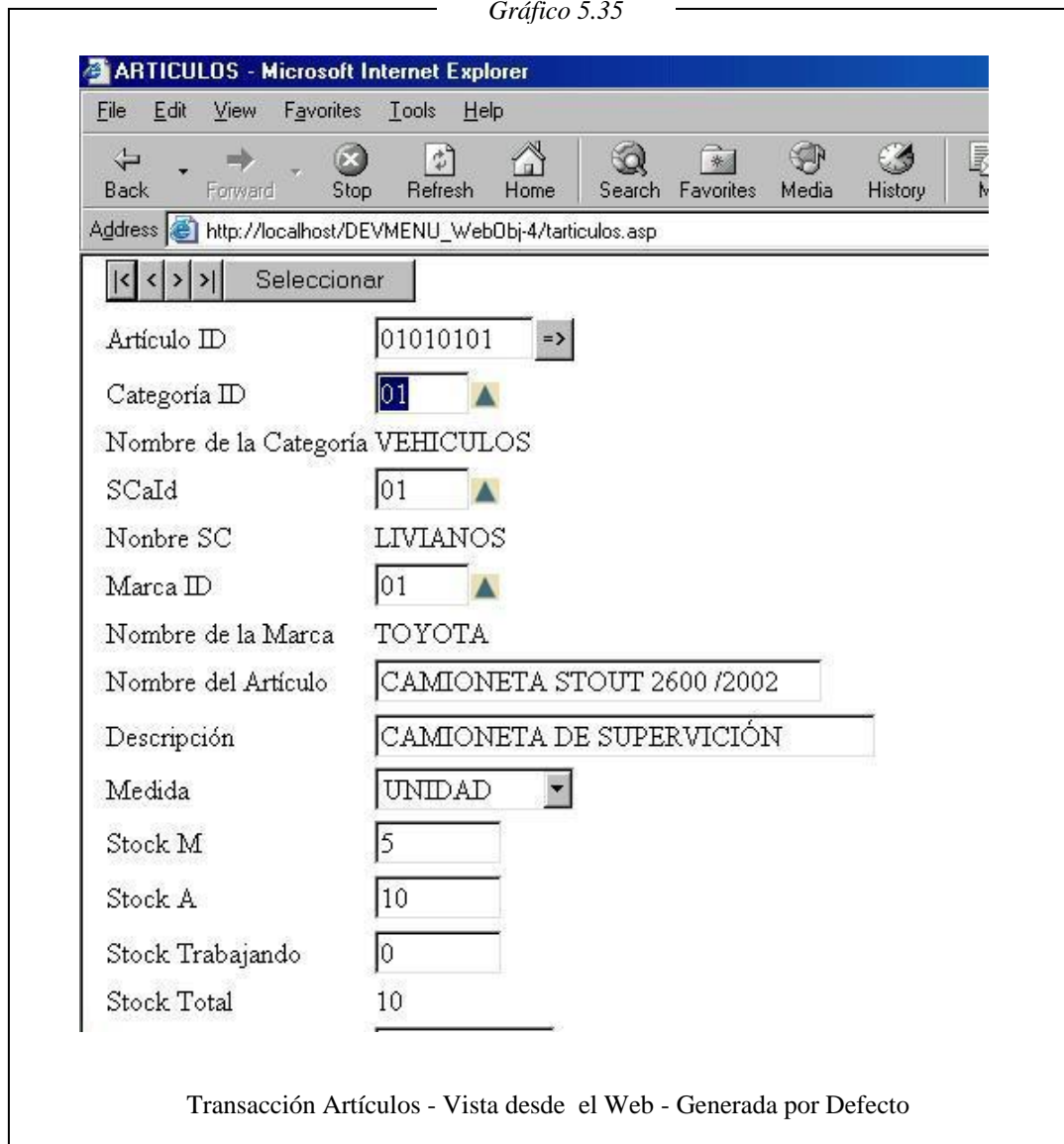


Como se observará en este gráfico, con el simple cambio de User Interface a Web Form en vez de Win Form, la aplicación será generada totalmente en el web.

Incluso sus work panels por defecto se generarán para el web, el gráfico 5.35 muestra cómo Genexus genera transacciones internet, esta transacción es generada por defecto es por eso que sale en un modo no tan presentable.

Las versiones anteriores de Genexus como la 7.0 son capaces de generar objetos web pero solo a manera de consultas, dicho en otras palabras solo generan paneles de trabajo, en cambio con esta versión 7.5 ya se puede hacer alimentación en tiempo real pero desde el Internet, el gráfico 5.35 muestra la misma transacción generada para el web.

Gráfico 5.35



Para generar este tipo de transacciones, Genexus 7.5 utiliza la tecnología de las páginas ASP (Página Activa de Servidor), ya que son páginas con una gran cantidad de contenidos interactivos, porque realizan alimentación de datos en línea por medio de interfaces web.

Se da cuenta Ud. cómo una Herramienta CASE le ahorra tiempo para generar aplicaciones web, desde una simple especificación de una estructura, cuánto tiempo le tomará al programador desarrollar esta página con herramientas puras como: SQL, html y visual basic, el gráfico 5.36 muestra la misma transacción aplicando un poquito de diseño gráfico.

Gráfico 5.36



5.8. ESTRUCTURA GENERAL DEL APLICATIVO

Gráfico 5.37

✓ Abo01	trn : Asignaciones Bodega 01
✓ Abo02	trn : Asignaciones Bodega 02
✓ Abo03	trn : Asignaciones a la Bodega 03
✓ Admini	trn : Administración Dygoil
✓ Adq01	rpt : Adquisición 01
✓ Adqc	rpt : Adquisiciones por Proveedor
✓ AdqFec	rpt : Adquisiciones por Fecha
✓ Adqui	trn : Adquisiciones
✓ Artic	trn : Artículos
✓ AsB01	rpt : Asignaciones Bodega 01
✓ AsB02	rpt : Asignaciones a la Bodega 02
✓ AsB03	rpt : Asignaciones a la Bodega 03
✓ ASG02	rpt : Asignaciones Bodega 02
✓ ASIG01	rpt : Asignaciones Bodega 01
✓ ASIG03	rpt : Asignaciones Bodega 03
✓ Atipo	trn : Tipo de Artículo
✓ B02	mbr : Bodega 02
✓ BaGen	trn : Bajas Generales
✓ BagGen	rpt : Bajas Generales
✓ Baj01	trn : Bajas Bodega 01
✓ Baj02	trn : Bajas Bodega 02
✓ Baj03	trn : Bajas Bodega 03
✓ BB01	rpt : Bajas Bodega 01
✓ BB02	rpt : Bajas de la Bodega 02
✓ BB03	rpt : Bajas de la Bodega 03
✓ Bdgro	trn : Bodegueros
✓ Bod01	trn : Bodega 01
✓ Bod02	trn : Bodega 02
✓ bod02	wkp : Bodega 02
✓ Bod03	trn : Bodega 03
✓ BOD03	wkp : Bodega 03
✓ Bode01	wkp : Bode01
✓ BODE03	mbr : BODEGA 03
✓ Califi	trn : Calificación
✓ Campos	trn : Campos de Operación
✓ CargAd	trn : Cargos Administrativos
✓ DAdqu	trn : Devoluciones de Adquisiciones
✓ DBod01	rpt : Devoluciones Bodega 01
✓ DBod02	rpt : Devoluciones Bodega 02
✓ DBod03	rpt : Devoluciones de la Bodega 03
✓ DeBo1	trn : Devoluciones Bodega 01
✓ DeBo2	trn : Devoluciones Bodega 02
✓ DeBo3	trn : Devoluciones de la Bodega 03
✓ DevAd	rpt : Devoluciones de Adquisiciones
✓ DeVen	rpt : Devoluciones de Ventas
✓ DeVen	trn : Devoluciones de Ventas
✓ EGR01	trn : Orden de Egreso Bodega 01
✓ EGR02	trn : Ordenes de Egreso Bodega 02
✓ EGR03	trn : Ordenes de Egreso Bodega 03
✓ Empres	trn : Empresas Relacionadas
✓ Famil	trn : Familias

Estructura General del Aplicativo – Primera Parte

Gráfico 5.38

✓	Gx0020	wkp	: Selection List TIPEMP
✓	Gx0030	wkp	: Lista de Selección EMPRES
✓	Gx0040	wkp	: Selection List MEDIDA
✓	Gx0050	wkp	: Lista de Selección SERVIC
✓	Gx0060	wkp	: Selection List CAMPOS
✓	Gx0070	wkp	: Selection List FAMIL
✓	Gx0081	wkp	: Selection List FAMIL1
✓	Gx0090	wkp	: Selection List MARCAS
✓	Gx00B2	wkp	: Selection List FAMIL2
✓	Gx00C0	wkp	: Lista de Selección ADMINI
✓	Gx00D0	wkp	: Selection List CARGAD
✓	Gx00E0	wkp	: Selection List CALIFI
✓	Gx00F0	wkp	: Lista de Selección PROVED
✓	Gx00G0	wkp	: Lista de Selección BDGR0
✓	Gx00J0	wkp	: Selection List UBICA
✓	Gx00K1	wkp	: Selection List UBICA1
✓	Gx00L0	wkp	: Selection List ARTIC
✓	Gx00M0	wkp	: Selection List ATIPO
✓	Gx00N0	wkp	: Selection List PERSON
✓	Gx00O0	wkp	: Selection List SUPERV
✓	Gx00P0	wkp	: Selection List PROYE
✓	Gx00Q1	wkp	: Selection List PROYE1
✓	Gx00R1	wkp	: Selection List PROYE2
✓	Gx00S0	wkp	: Selection List BOD01
✓	Gx00T1	wkp	: Selection List BOD011
✓	Gx00U1	wkp	: Selection List BOD012
✓	Gx00V0	wkp	: Lista de Selección ADQUI
✓	Gx00W0	wkp	: Lista de Selección TIPADQ
✓	Gx00X1	wkp	: Lista de Selección ADQUI1
✓	Gx00Y0	wkp	: Lista de Selección DADQU
✓	Gx00Z1	wkp	: Selection List DADQU1
✓	Gx0100	wkp	: Lista de Selección VENTA
✓	Gx0111	wkp	: Lista de Selección VENTA1
✓	Gx0120	wkp	: Lista de Selección DEVEN
✓	Gx0131	wkp	: Selection List DEVEN1
✓	Gx0140	wkp	: Selection List ABO01
✓	Gx0151	wkp	: Selection List ABO011
✓	Gx0160	wkp	: Selection List DEBO1
✓	Gx0171	wkp	: Selection List DEBO11
✓	Gx0180	wkp	: Selection List PERBO1
✓	Gx0190	wkp	: Selection List EGR01
✓	Gx01A1	wkp	: Selection List EGR011
✓	Gx01B0	wkp	: Selection List ING01
✓	Gx01D0	wkp	: Selection List BAGEN
✓	Gx01E1	wkp	: Selection List BAGEN1
✓	Gx01F0	wkp	: Selection List BAJ01
✓	Gx01G1	wkp	: Selection List BAJ011
✓	Gx01H0	wkp	: Selection List BOD02
✓	Gx01I1	wkp	: Selection List BOD021
✓	Gx01J1	wkp	: Selection List BOD022
✓	Gx01K0	wkp	: Selection List BOD03
✓	Gx01L1	wkp	: Selection List BOD031

Estructura General del Aplicativo – Segunda Parte

Gráfico 5.39

✓	Gx01M1	wkp	: Selection List BOD032
✓	Gx01N0	wkp	: Lista de Selección ING01
✓	Gx01O3	wkp	: Selection List ING011
✓	Gx01P0	wkp	: Selection List PERB02
✓	Gx01Q0	wkp	: Selection List PERB03
✓	Gx01R0	wkp	: Selection List ABO02
✓	Gx01S1	wkp	: Selection List ABO021
✓	Gx01T0	wkp	: Selection List ABO03
✓	Gx01U1	wkp	: Selection List ABO031
✓	Gx01X0	wkp	: Selection List DEB02
✓	Gx01Y1	wkp	: Selection List DEB021
✓	Gx01Z0	wkp	: Selection List DEB03
✓	Gx0201	wkp	: Selection List DEB031
✓	Gx0210	wkp	: Selection List BAJ02
✓	Gx0221	wkp	: Selection List BAJ021
✓	Gx0230	wkp	: Selection List BAJ03
✓	Gx0241	wkp	: Selection List BAJ031
✓	Gx0250	wkp	: Selection List EGR02
✓	Gx0261	wkp	: Selection List EGR021
✓	Gx0270	wkp	: Selection List EGR03
✓	Gx0281	wkp	: Selection List EGR031
✓	Gx0290	wkp	: Selection List ING02
✓	Gx02A3	wkp	: Selection List ING021
✓	Gx02B0	wkp	: Selection List ING03
✓	Gx02C3	wkp	: Selection List ING031
✓	ING01	trn	: Orden de Ingreso Bodega 01
✓	ING02	trn	: Ordenes de Ingreso Bodega 02
✓	ING03	trn	: Ordenes de Ingreso Bodega 03
✓	Marcas	trn	: Marcas
✓	MB01	mbr	: Bodega 01
✓	Medida	trn	: Unidades de Medida
✓	MENU	mbr	: MENU
✓	OInB01	rpt	: Orden de Ingreso Bodega 01
✓	OInB02	rpt	: Orden de Ingreso Bodega 02
✓	OInB03	rpt	: Orden de Ingreso Bodega 03
✓	OrEgB1	rpt	: Orden de Egreso Bodega 01
✓	OrEgB2	rpt	: Orden de Egreso Bodega 02
✓	OrEgB3	rpt	: Orden de Egreso Bodega 3
✓	Pbode	wkp	: Productos con Bodegas
✓	PerBo1	trn	: Personal Bodega 01
✓	PerBo2	trn	: Personal Bodega 02
✓	PerBo3	trn	: Personal Bodega 03
✓	Person	trn	: Personal
✓	PrBo02	rpt	: Productos de la Bodega 02
✓	PrBo03	rpt	: Productos de la Bodega 03
✓	PRINCIP	wkp	: DYGOIL
✓	Pro0002	rpt	: Productos por Familias
✓	ProBo01	rpt	: Productos de la Bodega 01
✓	Prod01	rpt	: Catalogo Genera de Productos
✓	Product	wkp	: Productos
✓	Proved	trn	: Proveedores
✓	Prove	trn	: Provetos
✓	Servic	trn	: Servicios DYGOIL
✓	Superv	trn	: Supervisión
✓	TipAdq	trn	: Tipo de Adquisición
✓	TipEmp	trn	: Tipos de Empresas
✓	Ubica	trn	: Ubicaciones
✗	VenFech	rpt	: Ventas por Fecha
✓	Venta	trn	: Ventas
✓	Venta1	rpt	: Factura de Ventas

Estructura General del Aplicativo – Tercera Parte

CAPÍTULO VI

EVALUACIÓN DE UNA HERRAMIENTA CASE



- ❖ Generalidades para Evaluar Herramientas CASE
- ❖ Criterios Preliminares
- ❖ Criterios Internos
- ❖ Criterios Externos
- ❖ Cierre General de la Etapa de Evaluación
- ❖ Interrogantes que se debe Hacer una vez Implantada la Herramienta CASE
- ❖ Herramientas CASE más Conocidas en el Mercado

6.1. GENERALIDADES PARA EVALUAR HERRAMIENTAS CASE

La decisión de adoptar Herramientas CASE como metodologías de desarrollo de software, puede ser la alternativa más adecuada para mejorar la productividad en los ambientes corporativos, pero también puede ser la más arriesgada si no se hace un minucioso estudio de evaluación, ya que todos los aciertos o fracasos van a ser palpados directamente por todas las personas relacionadas con la organización.

La utilización de una Herramienta CASE no queda en la simple *ilusión del usuario*, por sus atractivas y revolucionarias maneras gráficas de diseño, por su simplicidad y consistencia que se adhieren al *principio de asombro mínimo*, aceptando que los sistemas desarrollados con estas herramientas nunca sorprenden.

Para la mayoría de organizaciones de tamaño medio y grande, y para las compañías de desarrollo de software, la problemática no está en la decisión de adquirir una Herramienta CASE, sino en la introducción del CASE en las primeras etapas del ciclo de vida del sistema, esta introducción se lleva a cabo teniendo en cuenta todos los aspectos considerados con el desarrollo de un proyecto informático, en el cuál se añade las características muy particulares de los usuarios y el personal operador del sistema.

Las Herramientas CASE configuran la dimensión de la tecnología en el cuarteto: personas, proceso, producto y tecnología; como tales desempeñan un papel muy importante en el desarrollo de un sistema, no olvide que el objetivo de estas herramientas es: “*Mejorar el Proceso de Desarrollo*”.

La tecnología CASE no puede incrementar la productividad ni la calidad del software, si no se usa correctamente. En muchas organizaciones los programadores no han recibido la formación adecuada y no utilizan la herramienta como metodología de desarrollo.

La tecnología CASE no es sustituto de una buena gestión del proyecto, la actitud de los profesionales, puede afectar beneficiosamente o no al uso de esta tecnología, los proyectos de software quedan a menudo fuera de control porque los directivos no programan una buena campaña de introducción [LIB003] [www034].

6.1.1. CAUSAS DEL FRACASO DE LAS HERRAMIENTAS CASE

Se presentan las causas del fracaso de las Herramientas CASE, para que los desarrolladores de software y los directivos de las empresas, tomen en cuenta ciertos criterios y puedan realizar una comparación adecuada si están o no preparados para introducir esta tecnología:

- Confusión sobre lo que hace realmente cada Herramienta CASE individualmente.
- El uso de Herramientas CASE en problemas para los que no están diseñadas.
- Poner demasiada confianza en las Herramientas CASE, como solución completa a todos los problemas de software.
- Ignorar la importancia de una buena gestión del proyecto.
- No tener buenos: estándares, metodologías y culturas de desarrollo.
- Herramientas CASE pobremente integradas.
- Herramientas pobres en documentación e información.
- Herramientas CASE sin funcionalidad suficiente.
- Falta de claridad sobre el problema de software a resolver.
- Falta de métodos para medir el impacto de la Herramienta CASE en el desarrollo y mantenimiento del software.
- Falta de formación en la metodología de desarrollo de software.
- Indecisión, falta de predisposición a adoptar la metodología CASE.
- Falta de predisposición a cambiar de modo de trabajo en el desarrollo y mantenimiento de software.
- Ver a las Herramientas CASE como una tecnología de máximo riesgo.
- Inexistencia de una planificación para implantar la tecnología CASE.
- Ignorar la implantación de un plan piloto.
- No construir primeramente prototipos.
- Falta de motivación y capacitación constante.

La tecnología CASE es una parte muy importante de la solución de la crisis del software, pero no debe considerarse como la solución total a todos los problemas, el simple hecho de comparar Herramientas CASE es probable que no tenga el efecto deseado en la mejora de la calidad y productividad del software. Una solución total a la crisis del software debe verse a través de múltiples facetas: gestión, personal, herramientas y metodologías [LIB003].

6.1.2 TÉCNICA DE EVALUACIÓN

Esta propuesta diseñada para evaluar Herramientas CASE funciona de la siguiente manera: se formulará una definición conceptual y una definición operacional de cada uno de los indicadores que implican a las Herramientas CASE.

La definición conceptual se basa en una serie de preguntas que le permitirán al evaluador tomar en cuenta una amplia gama de aspectos que deben ser considerados.

En la definición operacional se tomará criterios y subcriterios, sobre temas netamente técnicos, los cuales tendrán una valoración a manera de pesos, éstos serán posteriormente tabulados para sacar un porcentaje y una calificación que determine la calidad de la herramienta evaluada; a los criterios y subcriterios se los puede considerar como métricas de evaluación.

Todos los indicadores a ser evaluados, abarcan directamente a todo el personal relacionado con la empresa: directivos, administrativos, técnicos, usuarios, clientes, proveedores, entre otros. Ya que si alguno de éstos no tiene una buena: información, motivación y predisposición para trabajar con la nueva herramienta puede ser un elemento causante del fracaso del proyecto CASE.

Los diferentes indicadores utilizados para este trabajo, se los evaluará de forma independiente y luego serán tabulados en un de los tres grandes grupos de evaluación; estos son:

- Aspectos Preliminares.
- Aspectos Internos.
- Aspectos Externos.

6.1.3. PESOS ASIGNADOS A CRITERIOS Y SUBCRITERIOS

El valor del peso que puede tomar cada subcriterio, es único y específico, y se encuentra impreso en la parte derecha del subcriterio, esta técnica se implanta para que no haya problemas de ambigüedad o de preferencias hacia alguna herramienta en particular, por ejemplo: ¿La herramienta genera código ejecutable?, tiene una respuesta de SI o NO, entonces se tendría un rango de calificación como uno o cero respectivamente.

Esta sería la manera más imparcial y honesta para evaluar Herramientas CASE, ya que si se coloca un rango de calificación de uno al diez, se deja abierto al criterio del evaluador la calificación de dicho indicador, dando como resultado una calificación no tan transparente. En la tabla 6.1 se observa un ejemplo de criterios, subcriterios y pesos.

Tabla 6.1. Ejemplo de Criterios y Pesos

<i>Criterio</i>	<i>Subcriterio</i>	<i>Peso</i>	<i>Selec</i>	<i>Total</i>
Bases de datos que genera	Oracle	1	X	4
	Informix	1	X	
	Sybase	1		
	Acces	1	X	
	DB/2	1		
	SQL Server	1	X	
Participación en el mercado	Desconocida	1		2
	Poco conocida	2	X	
	Conocida	3		

Como se observa en la tabla 6.1 existen dos tipos de selección de subcriterios, el uno que puede elegir varios subcriterios de cada criterio, el segundo que puede elegir un solo subcriterio de cada criterio; en el primer caso la calificación será la suma de los pesos de los subcriterios elegidos y en el segundo caso será el valor del peso del subcriterio elegido.

6.2. ASPECTOS PRELIMINARES

Estos aspectos engloban a todos los indicadores preliminares para la adopción de una Herramienta CASE, esta parte tiene mucho que ver con la decisión y predisposición de los directivos, la parte técnica también se la toma en cuenta pero de una manera muy subjetiva.

Existen tres indicadores importantes: aspectos de selección y adquisición, requerimientos de software y requerimientos de hardware; su porcentaje total en la evaluación general forma el veinte por ciento de todo el proceso de evaluación.

Para plasmar la metodología de evaluación en con una herramienta real, se evaluará a la Herramienta CASE ***GENEXUS 7.5***. A continuación se presentan los criterios y subcriterios de esta primera etapa de evaluación.

6.2.1. CRITERIOS DE SELECCIÓN Y ADQUISICIÓN

Participación de la herramienta en el mercado local. Quiere decir si la herramienta es conocida en la región donde se la va a aplicar, su importancia radica, en que si la herramienta es muy utilizada en los ambientes locales, se tendrá mayor facilidad de consulta dentro de la región donde usted trabaja, este criterio tiene tres subcriterios a considerar que son:

- Desconocida
- Poco Conocida.
- Conocida

Experiencia previa del grupo seleccionado con la Herramienta CASE. Se refiere a que si los desarrolladores de la herramienta tienen o han tenido algún conocimiento sobre la herramienta, sus literales son:

- Nada
- Poco
- Expertos

Recomendaciones o publicaciones positivas de expertos. Las publicaciones de los expertos son de mucha importancia, ya que ellos son los únicos que han realizado evaluaciones rigurosas de la herramienta, porque se supone que utilizan a la herramienta con frecuencia, sus calificativos serían.

- Nada
- Pocas
- Muchas

Referencias de autores. Las referencias de los autores, sirven para realizar contactos en caso de problemas y expectativas de la herramienta y de sus publicaciones, las valoraciones son las siguientes.

- Nada
- Pocas
- Muchas

Tipo de Herramienta CASE. El tipo de Herramienta CASE depende de la tarea a automatizar, se recomienda una herramienta que genere todo el ciclo de vida del sistema, para que de esta forma, se tenga una única cultura de desarrollo en toda la organización, es por eso que se le da más puntuación al los tipos Workbench.

- Toolkit (Cubre una o varias partes del ciclo de vida)
- Workbench (Cubre todo el ciclo de vida)

Costos. Los costos de las Herramientas CASE por lo general son elevados, pero a mediano plazo, no se convierten en gasto sino en inversión, ya que todo lo que sea para mejorar los procesos de: automatización de la información y de desarrollo de soluciones siempre son bien venidos. Sus calificativos son:

- Elevados
- Medios
- Bajos
- Gratis

Facilidad de preevaluación. La facilidad de preevaluación de la herramienta, es el eje fundamental del vendedor de Herramientas CASE, ya que con ello se puede ver de una forma real si la herramienta funciona o no en la organización donde se la desea adquirir, esto facilita a la rápida toma de decisiones de los directivos, para adquirir o no dicha herramienta.

- Mala
- Buena
- Excelente

Prestigio del distribuidor general. El prestigio que tiene el distribuidor general o dueño de la herramienta, es de mucha importancia, ya que es el primer certificado de garantía para el que desea adquirir la herramienta, nadie podrá dudar del prestigio de los productos IBM.

- Malo
- Bueno
- Excelente

Prestigio del distribuidor local. El prestigio que tiene un distribuidor local, es un poco mas difícil de descubrirlo, ya que en nuestro medio las empresas fantasmas se han convertido en el pan de cada día, si esto llegará a suceder el cliente no pierde la herramienta, sino, pierde las garantías que le brindaba este distribuidor, muchas de las políticas de los distribuidores locales, dañan el prestigio que tiene una herramienta opacando todas las ventajas que presenta la misma.

- Malo
- Bueno
- Excelente

Seriedad del contacto del proveedor. El contacto asignado por el proveedor, es la persona que puede levantar o echar abajo todo el prestigio ganado por los proveedores locales y generales.

Es la persona clave para impulsar los primeros pasos de la implantación de la nueva tecnología, de él depende que la herramienta siga teniendo éxito o fracase.

Muchas ocasiones, el contacto se presenta como la persona más amable del mundo, hasta que logra vender su producto, pero cuando el cliente necesita accesoria o ayuda de la convenida en los contratos de adquisición, desconoce a todo mundo.

Es por eso que se pondrá pesos considerables a este criterio, el comprador debe primero tomar referencias del contacto antes de cerrar el negocio, ya que si por mala suerte cayó con una persona contactante de estas características, los arrepentimientos pueden ser demasiado tarde.

La información del contacto no se la debe sacar de la empresa donde trabaja, sino de los clientes a los cuales ya les vendió la herramienta; si no existe en la empresa otro contacto, cambie de proveedor, no se deje engañar por vendedores y marquetineros de mala fe, asegúrese de escoger una persona que de verdad se preocupe por su organización.

- Pésimo
- Malo
- Bueno
- Excelente

Garantía por incumplimiento y fallas presentadas. Es la garantía que brinda el proveedor por incumplimiento en la entrega del producto y por la no entrega de requerimientos convenidos, como también la responsabilidad por fallas, su calificación es muy objetiva:

- No
- Si

Actualización de la versión. Toda herramienta de software nunca se queda estática, y siempre tiende a crear nuevas versiones, hay que poner muy en claro esta posibilidad, ya que algunas empresas distribuidoras le ofrecen el servicio de actualizaciones como parches o con el cambio de toda la herramienta, otras no, el cliente debe comprar toda la herramienta otra vez, cada que aparece una nueva versión.

- No cubre
- Cubre una parte
- Cubre todo el proceso

Ambientes de Generación: Son los ambientes en los cuales puede generar código, sus calificativos son los siguientes:

- Windows
- Web
- Texto

Arquitecturas Generadas. Es el tipo de arquitectura para las cuales puede generar aplicaciones, tendrán mayor valor las multiplataforma ya que se puede distribuir a la aplicación. [www034]

- Nativa
- Dos capas
- Multicapa

6.2.2. TABLA PARA SELECCIÓN Y ADQUISICIÓN

La tabla 6.2 indica claramente la técnica de evaluación de una Herramienta CASE, tomando como criterio la selección y adquisición, los valores seleccionados están relacionados con la Herramienta GENEXUS 7.5.

Tabla 6.2 Selección y Adquisición

<i>SELECCIÓN Y ADQUISICIÓN</i>				
Criterio	Subcriterio	Peso	Selec	Total
Participación del CASE en el mercado local.	Desconocida	0		1
	Poco conocida	1	X	
	Conocida	2		
Experiencia previa del grupo seleccionado, con el CASE.	Nada	0		1
	Poco	1	X	
	Expertos	2		
Publicaciones positivas de los expertos.	Nada	0		2
	Pocas	1		
	Muchas	2	X	
Referencias de expertos y autores.	Nada	0		1
	Pocas	1	X	
	Muchas	2		
Tipo de CASE	Toolkit	1		4
	Workbench	4	X	
Costos	Elevados	0		1
	Medios	1	X	
	Bajos	2		
	Gratis	3		
Facilidad de preevaluación	Mala	0		1
	Buena	1	X	
	Excelente	2		

Tabla 6.2 (Continuación) *Selección y Adquisición*

<i>SELECCIÓN Y ADQUISICIÓN</i>				
Criterio	Subcriterio	Peso	Selec	Total
Prestigio del distribuidor general	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Prestigio del distribuidor local (Proveedor)	Malo	0	X	0
	Bueno	1		
	Excelente	2		
Seriedad y disponibilidad del contacto	Pésimo	0	X	0
	Malo	1		
	Bueno	2		
	Excelente	4		
Garantía por incumplimiento y fallas posteriores	No tiene	0		1
	Si tiene	1	X	
Actualización de la versión	No cubre	0		1
	Cubre parte	1	X	
	Cubre todo	2		
Ambientes que genera	Texto	1	X	3
	Windows	1	X	
	Web	1	X	
Arquitecturas que genera.	Nativa	1	X	3
	Dos capas	1	X	
	Multicapa	1	X	
Predisposición de los directivos	Mala	0		2
	Buena	2		
	Excelente	4	X	
Predisposición de los técnicos	Mala	0		2
	Buena	2		
	Excelente	4	X	
Calificación Ideal		42	Total	28

6.2.3 CRITERIOS DE HARDWARE Y SOFTWARE DISPONIBLE.

Los criterios de hardware, están relacionados a los requerimientos que necesita la herramienta para su óptimo funcionamiento, relacionándolos con la tecnología actual, y la infraestructura con la que cuenta la organización.

Este criterio influye mucho en lo que es costos, ya que si la herramienta es muy cara y a más de eso requiere propiedades del software que se salen de los parámetros de presupuesto, muchas veces no se puede cubrir estos valores, los criterios son:

Hardware que necesita la herramienta. Este criterio tiene tres subniveles que son: poco, mucho, nada. Esto se refiere al hardware adicional que se debería instalar en la organización una vez instalada la herramienta.

Hardware que dispone la organización. Es el hardware que dispone la organización, con relación a los requisitos de la herramienta, sus niveles serían: compatible, no compatible e intermedio.

Factibilidad de nuevas adquisiciones. Es la predisposición y la factibilidad que tiene la empresa para adquirir nuevos equipos, de acuerdo a los requerimientos de la herramienta, los niveles inferiores son: buenas, malas, excelentes.

6.2.4 TABLA PARA EVALUAR EL HARDWARE DISPONIBLE

Tabla 6.3 *Hardware Disponible*

<i>HARDWARE DISPONIBLE</i>				
Criterio	Subcriterio	Peso	Selec	Total
Hardware que necesita la herramienta.	Mucho	0		1
	Poco	1	X	
	Nada	2		
Hardware disponible en la organización.	No compatible	0		1
	Intermedio	1	X	
	Compatible	2		
Factibilidades de adquisición	Malas	0		2
	Buenas	1		
	Excelentes	2	X	
Calificación Ideal		6	Total	4

6.2.5. TABLA PARA EVALUAR EL SOFTWARE DISPONIBLE.

Tabla 6.4 *Software Disponible*

<i>HARDWARE DISPONIBLE</i>				
Criterio	Subcriterio	Peso	Selec	Total
Software adicional para el funcionamiento de la herramienta.	Mucho	0		1
	Poco	1	X	
	Nada	2		
Software disponible en la organización.	No compatible	0		1
	Intermedio	1	X	
	Compatible	2		
Factibilidades de adquisición	Malas	0		2
	Buenas	1		
	Excelentes	2	X	
Calificación Ideal		6	Total	4

6.2.6. CIERRE DE LA ETAPA DE ASPECTOS PRELIMINARES.

Tabla 6.5 *Cierre de la etapa de adquisición*

<i>Aspectos Preliminares</i>					
Criterio	Resultado Ideal	Resultado Obtenido	Peso Ideal	% Total	Calif
Selección y Adquisición	42	28	60	60%	40
Hardware Disponible	6	4	20	20%	13
Software Disponible	6	4	20	20%	13
Suman					66

La calificación obtenida de la primera etapa de evaluación es 66/100, Como se puede observar se han asignado pesos específicos a cada criterio de evaluación, tomando en cuenta el grado de importancia de estos criterios. La técnica de tabulación se la realiza por medio de regla de tres de la siguiente manera.

$$\begin{array}{l}
 42 \text{ (Resultado ideal)} \quad \text{---} \quad 100 \text{ (Peso ideal)} \\
 28 \text{ (Resultado obtenido)} \quad \text{---} \quad X \text{ (Peso a obtener)} \\
 X = (100 \times 28) / 42 = 40
 \end{array}$$

6.3. ASPECTOS INTERNOS

En esta parte de la evaluación van todos los criterios y subcriterios netamente técnicos que puede brindar la herramienta, no tienen que ver con aspectos foráneos como: proveedores o distribuidores.

6.3.1 FASES DEL CICLO DE VIDA QUE CUBRE

Tabla 6.6 Fases del ciclo de vida que cubre.

<i>FASES DEL CICLO DE VIDA QUE CUBRE</i>				
Criterio	Subcriterio	Peso	Selec	Total
Fases del ciclo de vida que cubre.	Análisis	1	X	12
	Diseño	1	X	
	Generación de código	1	X	
	Construcción	1	X	
	Implantación	1	X	
	Reingeniería	1	X	
	Planificación	1		
	Mantenimiento	1	X	
	Gestión	1		
	Integración	1	X	
	Soporte	1	X	
	Repositorio	1	X	
	Planeamiento	1		
	Ayudas	1	X	
	Documentación	1	X	
	Modelamiento	1	X	
	Simulación	1	X	
	Prototipos	1	X	
Mejora de la calidad	1	X		
Otras	1	X		
Calificación Ideal		20	Total	17

6.3.2 METODOLOGÍAS SOPORTADAS

Tabla 6.7 Metodología Soportada

<u>METODOLOGÍA SOPORTADA</u>				
Criterio	Subcriterio	Peso	Selec	Total
Tipos de Metodologías	Estructurada	1		6
	Orientada a objet	2	X	
	Con objetos	3		
	Con componentes	4	X	
Consistencia de la diagramación con respecto a la metodología	Mala	1		2
	Media	2	X	
	Buena	3		
Tasa de cobertura de la metodología utilizada.	No utilizada	0		2
	Poco utilizada	1		
	Utilizada	2	x	
Calificación Ideal		15	Total	10

6.3.3 USUARIOS CONCURRENTES

Tabla 6.8 Usuarios concurrentes

<u>USUARIOS CONCURRENTES</u>				
Criterio	Subcriterio	Peso	Selec	Total
Número de usuarios concurrentes	Solo uno	1		2
	Pocos	2	X	
	Varios	3		
Grado de satisfacción en el manejo de las concurrencias	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Niveles de Seguridad	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Puede integrar las versiones.	No	0		0
	Si	1	X	
Calificación Ideal		8	Total	7

6.3.4 COMPONENTES

Tabla 6.9 Componentes

<i>COMPONENTES</i>				
Criterio	Subcriterio	Peso	Selec	Total
Es una herramienta integrada	No	0		1
	Si	1	X	
Tiene facilidad para diagramación	No	0		1
	Si	1		
Diagramas genera	Flujo de datos	1	X	3
	Modelo de datos	1	X	
	Estructura de árbol	1	X	
Es generador de pantallas	No	0		1
	Si	1	X	
Tipo de pantallas que genera	Texto	1	X	3
	Windows	1	X	
	Web	1	X	
Tiene un repositorio común	No	1		4
	Si	4	X	
Tiene facilidad para documentación	No	0		1
	Si	1	X	
Es generador de código	Visual estudio	1	X	4
	Visual .net	1	X	
	Java	1	X	
	Delphi	1		
	Ambientes c++	1		
	Ambientes IBM	1	X	
	Otros lenguajes	1	X	
	Otros ambientes	1		
Realiza simulación por prototipos	No	0		1
	Si	1	X	
Calificación Ideal		23	Total	20

6.3.5 CONTROL DE PROYECTOS

Tabla 6.10 Control de proyectos

<i>CONTROL DE PROYECTOS</i>				
Criterio	Subcriterio	Peso	Selec	Total
Facilidad del control de tareas.	No	0		1
	Si	1	X	
Facilidad para el control de recursos y costos	No	0		1
	Si	1	X	
Facilidad de control del tiempo	No	0		1
	Si	1	X	
Calificación Ideal		3	Total	3

6.3.6 PLATAFORMA DE SOFTWARE REQUERIDA

Tabla 6.11 Plataforma de software requerida

<i>PLATAFORMA DE SOFTWARE REQUERIDA</i>				
Criterio	Subcriterio	Peso	Selec	Total
Plataformas soportadas por la herramienta (Cliente o Servidor)	Windows	1	X	4
	Unix	1	X	
	AS/400	1	X	
	Linux	1	X	
	Otras	1		
Portabilidad de diferentes plataformas	no	0		1
	Si	1	X	
Ejecutables para distintas plataformas	No	0		1
	Si	1	X	
Calificación Ideal		7	Total	6

[LIB003] [www0033]

6.3.7 MANEJADOR DE BASES DE DATOS

Tabla 6.12 *Manejador de bases de datos*

<i>MANEJADOR DE BASES DE DATOS</i>				
Criterio	Subcriterio	Peso	Selec	Total
Bases de Datos en las que genera.	Acces / Fp	1	X	6
	Oracle	1	X	
	Informix	1	X	
	DB /2	1	X	
	UDB	1	X	
	Sybase	1		
	SQL Server	1	X	
	Posgres	1		
	My Sql	1		
	Otras	1		
Objetos que genera.	Tablas	1	X	8
	Consultas	1	X	
	Reportes	1	X	
	Desencadenadores	1	X	
	Reglas	1	X	
	Integridad Referencial	1	X	
	Normalización	1	X	
	Seguridad	1	X	
Dificultad para la manipulación de datos	Muy difícil	1		2
	Poco difícil	2	X	
	Fácil	3		
Ubicación de los procesos almacenados	Cliente	1	X	2
	Base de Datos	1		
	S transaccional	1	X	
Calificación Ideal		24	Total	19

[LIB003] [www0033]

6.3.8 FACILIDADES DE USO

Tabla 6.13 *Facilidades de uso*

<i>FACILIDADES DE USO</i>				
Criterio	Subcriterio	Peso	Selec	Total
Grado de claridad de los mensajes	Inapreciable	0		2
	No muy claro	1		
	Claro	2	X	
Bondad del diseño visual	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Localización rápida de opciones	Difícil	0		2
	Complicada	1		
	Fácil	2	X	
Presencia de metáforas	Nunca	0		1
	Siempre	1	X	
Calidad de la edición gráfica	Mala	0		2
	Buena	1		
	Excelente	2	X	
Grado de ajuste a las metodologías	Mala	0		1
	Buena	1	X	
	Excelente	2		
Grado de versatilidad en la navegación	Mala	0		1
	Buena	1	X	
	Excelente	2		
Grado de satisfacción en el tiempo de respuesta	Nada satisfactorio	0		2
	satisfactorio	1		
	Muy satisfactorio	2	X	
Calificación Ideal		15	Total	13

[LIB003] [www0033]

6.3.9 . FUNCIONALIDAD

Tabla 6.14 *Funcionalidad*

<i>FUNCIONALIDAD</i>				
Criterio	Subcriterio	Peso	Selec	Total
Cantidad de lenguajes en que genera el código	Bajo	1		2
	Considerable	2	X	
	Alto	3		
Cobertura del código generado	Muy poco	1		3
	Una o varias partes	2		
	Todo	3	X	
Grado de completitud del código generado.	Poca	1		3
	Considerable	2		
	Todo	3	X	
Calidad del código generado	Malo	1		3
	Bueno	2		
	Excelente	3	X	
Calidad de los reportes que genera	Mala	1		3
	Buena	2		
	Excelente	3	X	
Grado de completitud de los reportes	Poca	1		3
	Considerable	2		
	Todo	3	X	
Nivel de generación de prototipos	Bajo	1		3
	Medio	2		
	Alto	3	X	
Cantidad de lenguajes soportados para reingeniería	Pocos	1		2
	Varios	2	X	
	Muchos	3		
Calidad del producto de Reingeniería	Mala	0		1
	Buena	1	X	
	Excelente	2		
Calificación Ideal		26	Total	23

6.3.10 CURVA DE APRENDIZAJE

Tabla 6.15 Curva de aprendizaje

<u>CURVA DE APRENDIZAJE</u>				
Criterio	Subcriterio	Peso	Selec	Total
Es la herramienta intuitiva	No	0		1
	Si	1	X	
Grado de facilidad de edición	Bajo	1		2
	Medio	2	X	
	Alto	3		
Permite variación de visión de objetos	No	0		1
	Si	1	X	
Grado de satisfacción de las ayudas y tutoriales	Bajo	1		1
	Medio	2	X	
	Alto	3		
Calificación Ideal		7	Total	5

6.3.11 AYUDA EN LÍNEA

Tabla 6.16 Ayuda en línea

<u>AYUDA EN LÍNEA</u>				
Criterio	Subcriterio	Peso	Selec	Total
Rapidez del sistema de ayuda	Lenta	1		3
	Baja	2		
	Alta	3	X	
Facilidad de uso	Difícil	1		3
	No tan fácil	2		
	Fácil	3	X	
Sensibilidad al contexto	Baja	1		3
	Media	2		
	Alta	3	X	
Calificación Ideal		9	Total	9

[LIB003] [www0033]

6.3.12 FLEXIBILIDAD

Tabla 6.17 Flexibilidad

<u>FLEXIBILIDAD</u>				
Criterio	Subcriterio	Peso	Selec	Total
Capacidad de parametrización	Baja	1		3
	Buena	2		
	Excelente	3	X	
Grado de integración con otros software	Bajo	1		3
	Medio	2		
	Alto	3	X	
Calidad de integración	Baja	1		3
	Buena	2		
	Excelente	3	X	
Facilidad de cambio del producto terminado	Difícil	1		3
	No tan fácil	2		
	Fácil	3	X	
Grado de portabilidad	Bajo	1		3
	Medio	2		
	Alto	3	X	
Facilidad para detectar inconsistencia	Difícil	1		3
	No tan fácil	2		
	Fácil	3	X	
Grado de generación multiplataforma	Bajo	1		3
	Medio	2		
	Alto	3	X	
Calificación Ideal		21	Total	21

[LIB003] [www0033]

6.3.13 TIPO DE INTEGRACIÓN

Tabla 6.18 *Tipo de integración*

<i>TIPO DE INTEGRACIÓN</i>				
Criterio	Subcriterio	Peso	Selec	Total
Integración por datos	No	0		1
	Si	1	X	
Integración por interfaz	No	0		1
	Si	1	X	
Integración por actividades	No	0		1
	Si	1	X	
Calificación Ideal		3	Total	3

6.3.14 CIERRE DE REQUERIMIENTOS INTERNOS

Tabla 6.19 Cierre de requerimientos internos

<i>Aspectos Internos</i>					
Criterio	Resultado Ideal	Resultado Obtenido	Peso Ideal	% Total	Calif
Fases del ciclo de vida que cubre	20	17	10	10%	8.5
Metodologías soportadas	15	10	4	4%	2.7
Usuarios concurrentes	8	7	5	5%	4.4
Componentes	23	20	10	10%	8.8
Control de proyectos	3	3	4	4%	4
Plataformas de software	7	6	5	5%	4.3
Manejadores de bases de datos	24	19	20	20%	16
Facilidades de uso	15	13	5	5%	4.3
Funcionalidad	26	23	20	20%	18
Curva de aprendizaje	7	5	5	5%	5
Ayuda en línea	9	9	4	4%	4
Flexibilidad	21	21	4	4%	4
Tipo de integración	3	3	4	4%	4
Total			100	100%	88

La calificación total de los aspectos internos es: **88/100**

A continuación se va a evaluar los aspectos externos, aquí constan criterios que están fuera del alcance de lo que es propiamente la herramienta, se evaluarán los conceptos que tienen que ver con los proveedores conjuntamente con las garantías que brindan a los usuarios de dicha herramienta.

6.4. ASPECTOS EXTERNOS

6.4.1 SOPORTE TÉCNICO

Tabla 6.20 Flexibilidad

<i>SOPORTE TÉCNICO</i>				
Criterio	Subcriterio	Peso	Selec	Total
Soporte técnico permanente	Malo	1		2
	Bueno	2	X	
	Excelente	3		
Horas mensuales de soporte técnico personalizado	0	0		4
	1 – 4	4	X	
	5 – 10	5		
Soporte técnico vía web	Nada	0		1
	Poco	1	X	
	Mucho	2		
Soporte adicional	Nada	1		2
	Poco	2	X	
	Mucho	3		
Experiencia positiva en otras empresas	Mala	0		2
	Buena	1		
	Excelente	2	X	
Cobertura y flexibilidad de horarios de soporte técnico	Mala	0		2
	Buena	1		
	Excelente	2	X	
Grado de satisfacción en llamadas a soporte técnico	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Préstamo de equipos por fallas en la herramienta	No	0		1
	Si	1	X	
Calificación Ideal		20	Total	15

[LIB003] [www0033]

6.4.2 ENTRENAMIENTO

Tabla 6.21 Entrenamiento

<u>ENTRENAMIENTO</u>				
Criterio	Subcriterio	Peso	Selec	Total
Variación de los tipos de cursos	Mala	0		2
	Bueno	1		
	Excelente	2	X	
Cursos básicos anuales	Nunca	0		2
	Rara vez	1		
	Siempre	2	X	
Cursos avanzados anuales	Nunca	0		2
	Rara vez	1		
	Siempre	2	X	
Cobertura de los cursos básicos	Usuario con privilegios	1		2
	Todo usuario	2	X	
Cobertura de los cursos avanzados	Usuario con privilegios	1	X	1
	Todo usuario	2		
Horas requeridas para lograr el dominio del case	Muchas	0		2
	Pocas	1		
	Muy pocas	2	X	
Calificación Ideal		12	Total	11

[LIB003] [www0033]

6.4.3 MATERIAL DE APOYO

Tabla 6.22 *Material de apoyo*

<i>MATERIAL DE APOYO</i>				
Criterio	Subcriterio	Peso	Selec	Total
Grado de completitud del material de apoyo por funcionalidad	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Grado de variedad del material de apoyo de acuerdo al usuario que está dirigido.	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Cobertura del material de apoyo de acuerdo a los usuarios que está dirigido	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Variedad del material de apoyo de acuerdo a sus presentaciones.	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Cobertura del material de apoyo, en cuanto a sus presentaciones	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Calidad en la clasificación de los temas	Mala	0		2
	Buena	1		
	Excelente	2	X	
Profundidad en el tratamiento de los temas	No profundiza	1		3
	Mediano	2		
	Profundo	3	X	
Consistencia del material	Mala	0		2
	Buena	1		
	Excelente	2	X	
Calificación Ideal		17	Total	14

6.4.4. PRESTIGIO DEL DESARROLLADOR DE LA HERRAMIENTA

Tabla 6.23 *Prestigio del desarrollador*

<i>PRESTIGIO DEL DESARROLLADOR</i>				
Criterio	Subcriterio	Peso	Selec	Total
Publicaciones expertas positivas	Corrientes	1		2
	Buenas	2	X	
	Excelentes	3		
Desempeño en otras empresas	Malo	0		1
	Bueno	1	X	
	Excelente	2		
Posee certificaciones	No	0		1
	Si	1	X	
Grado de bienestar financiero de la empresa desarrolladora	Malo	0		2
	Corriente	1		
	Excelente	2	X	
Tiempo en el mercado de la empresa con esa herramienta	Corto	1		3
	Medio	2		
	Largo	3	X	
Nivel de mejoramiento organizacional de la empresa	Malo	0		2
	Bueno	1		
	Excelente	2	X	
Calificación Ideal		16	Total	11

[LIB003] [www0033]

6.4.5 COSTOS

Tabla 6.24 Costos

<i>COSTOS</i>				
Criterio	Subcriterio	Peso	Selec	Total
Costos de adquisición	Elevados	1		3
	Medios	2		
	Bajos	3	X	
Cobertura de los costos	Una parte	1		3
	Varias partes	2		
	Todo	3	X	
Costos de entrenamiento	Adicionales	1		3
	Primeras etapas	2		
	Incluido	3	X	
Costos de hardware	Elevados	1		2
	Bajos	2	X	
Cobertura del costo de hardware	Una parte	1		2
	Varias partes	2	X	
	Todo	3		
Costos de software	Elevados	1		2
	Bajos	2	X	
	Gratis	3		
Cobertura de los costos de software	Una parte	1		3
	Varias partes	2		
	Todo	3	X	
Calificación Ideal		20	Total	18

6.4.6. CIERRE DE LOS REQUERIMIENTOS EXTERNOS

Tabla 6.25 Cierre de requerimientos externos

<i>Cierre de Requerimientos Externos</i>					
Criterio	Resultado Ideal	Resultado Obtenido	Peso Ideal	% Total	Calif
Soporte técnico	20	15	20	20%	15
Entrenamiento	12	11	10	10%	9
Material de apoyo	17	14	20	20%	16.4
Prestigio del desarrollador	16	11	10	10%	6.9
Costos	20	18	40	40%	36
Total			100	100%	83.3

La calificación de los requerimientos externos es: **83.3 / 100**

Ahora se va a proceder a cerrar y tabular todos los indicadores evaluados anteriormente, dicho en otras palabras se va a cerrar todo el proceso de evaluación.

Los aspectos preliminares tienen una importancia del veinte por ciento, los aspectos internos tienen el cincuenta por ciento y los aspectos externos el treinta por ciento.

6.5. CIERRE GENERAL DE LA EVALUACIÓN

Tabla 6.26 Evaluación general de la Herramienta CASE “GeneXus 7.5”

<i>EVALUACIÓN GENERAL DE GENEXUS 7.5</i>				
Criterio	Calificación Obtenida	Calificación Ideal	Porcentaje General	Calificación Total
Aspectos Preliminares	66	100	20%	13.2
Aspectos Internos	88	100	50%	44
Aspectos Externos	83.3	100	30%	26.5
Total			100%	83.7

Según la evaluación realizada, Genexus 7.5 tiene una calificación de **83.7/100**

Tabla 6.27 Calificación Descriptiva

Calificación General Descriptiva		
Descripción	Rango	Herramienta
Excelente	81 – 100	X
Muy Buena	61 – 80	
Buena	41 – 60	
Regular	0 – 40	

Por el resultado obtenido de 83.7 / 100, le herramienta Genexus 7.5 se la considera: **Muy Buena.**

6.6. INTERROGANTES QUE SE DEBEN HACER UNA VEZ IMPLANTADA LA HERRAMIENTA CASE

Se deben plantar estas interrogantes, para poder apreciar si valió la pena cambiar la metodología tradicional por Herramientas CASE.

- Maximizó la productividad del software?
- Aumentó la calidad del software?
- Disminuyó el número de errores?
- Simplificó el proceso de desarrollo de software?
- Posibilitó un empleo más eficiente de los recursos del ordenador?
- Se redujeron los costos del software?
- Aumentó la fiabilidad de los procesos de desarrollo?
- Proporcionó un mejor ambiente hombre máquina?
- Revolucionó los procesos de desarrollo de software?
- Automatizó la generación de código ejecutable?
- Enfatizó la generación de prototipos?
- Automatizó la comprobación de errores en cada etapa de desarrollo?
- Automatizó la gestión del proyecto?
- Formalizó y estandarizó el proceso de desarrollo?
- Integró las herramientas de desarrollo?
- Normalizó la documentación del software?
- Generó ayudas automáticamente?
- Promovió la reutilización del software.
- Promovió un control general de mantenimiento?
- Mejoró la portabilidad del software?
- Está seguro de haber implantado una cultura de desarrollo?
- Prefiere la metodología CASE a la convencional ?
- Es mejor no separar los datos del comportamientos de los objetos?
- Aumentó la productividad de los desarrolladores?
- Aumentó el grado de satisfacción del cliente final?
- Cree que las herramientas CASE solucionan gran parte sus problemas de desarrollo?

Si ha contestado positivamente a la mayoría de preguntas desplegadas; la utilización de la tecnología CASE, ha sido un éxito en su organización, caso contrario falló en alguno de los pasos de implantación o utilización.

6.7 HERRAMIENTAS CASE MÁS CONOCIDAS EN EL MERCADO.

Tabla 6.28 Herramientas CASE más conocidas en el mercado

<i>Herramienta</i>	<i>Fase del ciclo que cubre</i>
Genexus (Artech)	Todo
Erwin	Análisis / Diseño
Acción Request System	Todo
Bpwin	Generador de código
Meta Edit	Diseño
Designer (Oracle)	Análisis / Diseño
Power Designer (Sybase)	Análisis / Diseño
Power Builder (Sybase)	Generador de código
Rational Rouse	Análisis /Diseño / UML
Rational Unified Process	Gestión de procesos
Visio (Microsoft)	Varias Partes
Visual Age For Java	Diseño Web
Developer	Análisis / Diseño
Foundation (Arthur Andersen)	Todo el ciclo
Desing Machine (Ken Orr & Associates)	Análisis y diseño
Auto Mate Plus	Diseño matemático
TAGS (Teledyne Brown)	Todo
Analyst Designer	Análisis y diseño
Inspector / Recoder (Language Technology)	Mantenimiento
Verify (Online Software Internetal)	Mantenimiento
Cobol / SF (IBM)	Mantenimiento
PM/SS (Adpac)	Reingeniería
Maestro (SoftLab)	Reingeniería
Teamwork (Cadre Technologies)	Diagramación

CAPÍTULO VII

CONCLUSIONES Y RECOMENDACIONES



- ❖ Verificación de la Hipótesis
- ❖ Conclusiones
- ❖ Recomendaciones

7.1. VERIFICACIÓN DE LA HIPÓTESIS.

Todo el estudio realizado en esta investigación, tanto en la parte teórica como en la práctica, a servido para comprobar positivamente lo que se mencionó en la hipótesis inicial: ***“Las Herramientas CASE, sí son capaces de generar soluciones multicapa y multiplataforma de gran complejidad, en tiempos mínimos y a costos bajos”***

Para poder comprobar el contenido de la hipótesis de una manera real, se realizó un sistema de control de inventarios y bodegas para la empresa petrolera: DYGOIL. Cia. Ltda. utilizando la Herramienta CASE: GENEXUS.7.0.

La técnica de comprobación es la siguiente: Se diseñó una sola base de conocimiento del sistema, esta base de conocimiento se la generó en diferentes capas y plataformas, y se evaluó: costos, tiempos, facilidades de generación de una plataforma a otra y calidad del software generado.

La base de conocimiento modelo fue diseñada para que funcione bajo una arquitectura cliente / servidor de dos capas, utilizando Novell Netware 5.1 como sistema operativo de red, con un sistema gestor de base de datos Oracle 8i y en la parte del cliente Windows 98 con Visual Foxpro como lenguaje de programación.

Una vez terminada esta aplicación, se tomó la misma base de conocimiento y con un simple cambio en algunas de las propiedades de la herramienta, se generó aplicaciones para: Windows 2000 Server, con SQL Server 7.0 en el servidor y Windows 98 con Visual Foxpro en el cliente, generando de igual manera un sistema cliente servidor de dos capas.

Luego de esto, utilizando la misma base de conocimiento se generó aplicaciones centralizadas o nativas conmutando: Acces Visual Basic y Visual Foxpro. A manera de demostración se realizó también un pequeño aplicativo totalmente para el web, utilizando Genexus 7.5 Trial Versión.

Con esto se demuestra que las herramientas CASE sí son capaces de generar sistemas multicapa y multiplataforma, ya que se generó soluciones: centralizadas, dos capas y web, tanto para plataformas Windows y plataformas Novell y la calidad del software fue excelente en cada arquitectura generada.

Los costos y tiempos de generación de una misma aplicación, en distintas capas y plataformas, utilizando Herramientas CASE y programación común, son extremadamente impresionantes, ya que si no se utilizaría Herramientas CASE se necesitaría por lo menos un experto para cada arquitectura generada, esto implica tiempos más largos y precios más altos por cada solución a generarse.

El tiempo que se tardó en el diseño de la base de conocimiento con Genexus 7.0 fue de cuatro meses y el tiempo de generación en cada arquitectura mencionada varió de dieciséis a veinte segundos y el recurso humano fue de una sola persona experta en el manejo de la herramienta.

Se estima que con programación común, para generar la misma aplicación en una arquitectura diferente se necesita un tiempo igual al de generación de la primera aplicación, en este caso de tres a cuatro meses, comparado con veinte segundos que tarda generar la aplicación con Genexus la relación programación común versus Herramientas CASE es sumamente grande.

Con esta explicación queda demostrada la segunda parte de la hipótesis, que es: Las Herramientas CASE sí son capaces de generar soluciones a tiempos mínimos y con costos bajos. La tabla 7.1 indica los tiempos y costos de generación de una manera más específica.

TABLA 7.1 Tiempos y costos de generación de una aplicación ya diseñada, en diferentes arquitecturas.

Metodología Utilizada	Arquitecturas a Generarse	Tiempo de Generación	Recursos Humanos	Costos de Generación
Programación Común	Novell - Oracle Windows 98 Visual Foxpro	8 Meses 2 Meses por cada arquitectura generada	4 Expertos Un experto para cada arquitectura	8000 USD Mil dólares mensuales por cada experto
	Windows 2000 Serv SQL Server Windows 98 Visual Foxpro			
	Windows Acces Visual Basic			
	Windows Visual Foxpro			
HERRAMIENTAS CASE (GENEXUS)	Novell - Oracle Windows 98 Visual Foxpro	80 Segundos 20 Segundos por cada arquitectura generada.	1 Experto Un experto en la herramienta	15 USD Una hora veinte minutos del experto
	Windows 2000 Serv SQL Server Windows 98 Visual Foxpro			
	Windows Acces Visual Basic			
	Windows Visual Foxpro			

7.2. CONCLUSIONES

Las conclusiones a las que se llegó en esta tesis fueron obtenidas a base del estudio de la parte teórica y plasmadas en la realización del aplicativo.

CON RESPECTO A LAS HERRAMIENTAS CASE

- La utilización de Herramientas CASE en todo el ciclo de vida de un sistema informático, le ayudan al desarrollador de aplicaciones a automatizar las partes automatizables, como: creación, normalización y generación de la base de datos, generación de documentación, generación de diagramas, generación de código, generación de pantallas, generación de paneles de trabajo, entre otras. Dándole la oportunidad al programador de que su trabajo se centre en las partes medulares de una aplicación y que solo el hombre las puede realizar como son: el análisis y el diseño.
- Las Herramientas CASE maximizan la calidad y la productividad del software generado y de todo el equipo de desarrollo. Ya que no se está frente a una herramienta común de programación sino frente a una metodología de diseño de aplicaciones; que está enfocada a crear una cultura de programación ordenada y sistemática tanto en los sistemas a desarrollar como en el personal desarrollador.
- Las Herramientas CASE reducen costos y tiempos de generación; porque para cada aplicación generada no se vuelve programar nuevamente todo el sistema; solamente se debe realizar pequeñas modificaciones en la base de conocimiento que tiene el sistema de referencia o prototipo, permitiendo de esta manera la reutilización total del código ya generado, propiedad que lleva a reducir: tiempos, personal y requerimientos de generación.
- Las mayoría de Herramientas CASE facilitan la generación de soluciones informáticas en diferentes capas y plataformas, para distintos lenguajes de programación y bases de datos existentes en el mercado, en algunos casos sin importar el fabricante. Con Herramientas CASE a la misma aplicación se la puede observar en modos: Windows, Texto (AS/400) y Web; sin que el cambio de una interfase a otra afecte en lo mínimo al desempeño de la aplicación.
- Las Herramientas CASE facilitan un entorno netamente gráfico e interactivo de desarrollo, con un tiempo de respuesta suficientemente rápido, encaminado a la comprobación de errores desde el principio de las etapas de desarrollo. Cuando el desarrollador puede corregir los errores al principio de cada etapa, le es más fácil obtener un software de calidad, ya que va depurando poco a poco a toda la aplicación; con programación común se acentúa más la posibilidad de encontrar errores al final de las etapas de desarrollo y su corrección puede generar problemas sumamente complejos, en muchas de las veces toca volver nuevamente a

programar toda la aplicación; en cambio las Herramientas CASE brindan la oportunidad de corregir todo error en cada etapa de la aplicación.

- Las Herramientas CASE aportan a la generación de prototipos, los cuales son copias exactas de la aplicación real, donde se puede realizar: pruebas, cambios, adaptaciones, etc. Sin que esto afecte a la aplicación en producción; afectará a esta aplicación siempre y cuando los cambios realizados en prototipo hayan sido completamente probados y aprobados por el grupo desarrollador. Los prototipos sirven también para demostrar al cliente un demo del software que va a adquirir.

CON RESPECTO AL APLICATIVO Y A GENEXUS

- Con la utilización de la Herramienta CASE “Genexus 7.0” en el aplicativo de esta tesis; el proceso de desarrollo tuvo las siguientes conclusiones: Se maximizó la productividad y la calidad del software, disminuyó el número de errores, simplificó el proceso de desarrollo de software, permitió un empleo más eficiente de los recursos del ordenador, se redujeron los costos del software, aumentó la fiabilidad de los procesos de desarrollo, proporcionó un mejor ambiente hombre máquina, revolucionó los procesos de desarrollo de software, automatizó la generación de código ejecutable, enfatizó la generación de prototipos, automatizó la comprobación de errores en cada etapa de desarrollo, formalizó y estandarizó todos los procesos, integró todas las herramientas de desarrollo, normalizó la documentación, generó ayudas automáticamente, se pudo reutilizar el software, se facilitó el mantenimiento, se facilitó la portabilidad y aumentó el grado de satisfacción del cliente final y del desarrollador.
- Genexus 7.5 Trial Versión, tal como está almacenada y publicada en Internet no genera soluciones cliente servidor para bases de datos que necesitan conexiones ODBC o JDBC solo se puede realizar soluciones para acces, se comprobó en diferentes máquinas y con diferentes expertos de Genexus. Es por eso que el aplicativo se desarrollo con Genexus 7.0.

7.3. RECOMENDACIONES.

CON RESPECTO A LAS HERRAMIENTAS CASE

- Conciba a las Herramientas CASE como metodologías de desarrollo de software y no como herramientas de generación de aplicaciones.
- Es mejor utilizar herramientas tipo Workbench ya que utilizaría una sola metodología y cultura de desarrollo en todo el ciclo de vida de la aplicación.
- Se recomienda utilizar herramientas que soporten metodología incremental para que el desarrollador pueda elaborar poco a poco la aplicación.
- Cuando ya se decidió por utilizar la tecnología CASE , se recomienda empezar con un plan piloto bien diseñado.
- No piense que las Herramientas CASE son la solución total a sus problemas de software; estos problemas tienen diferentes causas.
- No ponga sus sistemas en producción sin antes haber probado y aprobado en prototipo.
- Tenga mucho cuidado con la seriedad del distribuidor de la herramienta, ya que lo pueden sorprender con cosas que usted no sabía el momento de adquirir la herramienta; esto sucede incluso en las versiones educativas y de libre acceso.
- Si se va a especializar en algún lenguaje de cuarta generación o en una determinada base de datos se le recomienda especializarse mejor en una Herramienta CASE ya que con esta puede generar las mismas cosas pero para múltiples lenguajes y bases de datos.
- Otra de las recomendaciones de la utilización de herramientas workbench es que no separan los datos de los procesos y su comportamiento. El resultado de la aplicación con estas herramientas es un software compacto y muy bien relacionado con todos los objetos, más no es un modelo de datos independiente.
- En definitiva con todas las ventajas de las Herramientas CASE presentadas en esta investigación, se recomienda cambiar la programación convencional por metodologías CASE, ya que su objetivo principal es “mejorar y automatizar el proceso de desarrollo de software.”

CON RESPECTO A GENEXUS

- Todos los atributos que desea utilizar deben estar en la estructura de la transacción, por ejemplo quiere modificar el stock de productos desde compras, en la estructura de la transacción compras debe estar el stock del producto, caso contrario no se puede realizar las reglas.
- Es preferible poner claves primarias independientes en cada nivel de la transacción, no se recomienda poner como clave primaria a la clave primaria de otra transacción, ya que al hacer consultas, el resultado es general y no local como se debería esperar.
- No se recomienda utilizar objetos como: combo box, check box, radio button, porque estos no los puede leer el promp generado por defecto, se recomienda realizar transacciones pequeñas, que le permitan la selección de información.
- Se recomienda nombrar los atributos con un número máximo de seis caracteres, ya que algunas bases de datos cumplen este estándar de generación y al pasar de una arquitectura a otra va a tener problemas.
- Para quitar el menú por defecto que genera Genexus cuando se corre la aplicación, se puede crear un panel de trabajo y determinarlo como objeto main.
- Muchas veces Genexus genera sin problemas a toda la aplicación, pero cuando se desea ver la base de datos las tablas no constan en ella; para solucionar esto debe activar la preferencia: all tables in the server, que quiere decir que todas las tablas estén en el servidor.
- Para generar una aplicación ya realizada en una arquitectura diferente, es mejor crear un nuevo modelo y generar la aplicación, caso contrario si genera en el mismo modelo anterior al momento de ejecutarla la aparecerá algunos errores.

Todas estas recomendaciones no constan en los manuales de Genexus disponibles en nuestro medio.

El Autor:

ÍNDICE

CAPÍTULO I

INTRODUCCIÓN A LAS HERRAMIENTAS CASE

1.1. GENERALIDADES.....	2
1.2. DEFINICIONES DE HERRAMIENTAS CASE.....	4
1.2.1. EL PROCESO DE DESARROLLO DE SOFTWARE	5
1.3. CLASIFICACIÓN DE LAS HERRAMIENTAS CASE.....	6
1.3.1. CLASIFICACIÓN GENERAL DE LAS HERRAMIENTAS CASE.....	6
1.3.2. OTROS TIPOS DE CASE.....	7
1.3.3. CASE QUE GENERAN UNA PARTE DEL CICLO DE VIDA.....	8
1.3.4. TIPOS DE CASE POR SU FUNCIONALIDAD.....	8
1.3.5. OTRA CLASIFICACIÓN DE LAS HERRAMIENTAS CASE	10
1.3.6. VENTAJAS Y DESVENTAJAS DEPENDIENDO DEL TIPO DE CASE UTILIZADA	12
1.3.7. CASE EN EL CICLO DE VIDA DE UN SISTEMA	13
1.4. OBJETIVOS DE LAS HERRAMIENTA CASE.....	14
1.5. ESTRUCTURA DE LAS HERRAMIENTAS CASE.....	15
1.5.1. MÓDULO DE REPOSITORIO O ENCICLOPEDIA.....	15
1.5.2. MÓDULO DE DIAGRAMACIÓN Y MODELAMIENTO	17
1.5.3. MÓDULO DE PROTOTIPADO.....	18
1.5.4. MÓDULO DE GENERADORES DE CÓDIGO.....	19
1.5.5. MÓDULO GENERADOR DE DOCUMENTACIÓN.....	20
1.6. PROCESO DE DESARROLLO DE SOFTWARE CON HERRAMIENTAS CASE.....	20
1.6.1. ARQUITECTURA JERÁRQUICA DE LAS HERRAMIENTAS CASE.....	20
1.6.2. OPCIONES DE INTEGRACIÓN.....	21
1.6.2.1. INTERCAMBIO DE DATOS PUNTO A PUNTO	22
1.6.2.2. ACCESO COMÚN A HERRAMIENTAS.....	22
1.6.2.3. INTEGRACIÓN DE DATOS.....	23
1.6.2.4. INTEGRACIÓN TOTAL.....	24
1.7. PONDERACIONES PARA LA UTILIZACIÓN DE HERRAMIENTAS CASE	26
1.8. REQUISITOS PARA ADQUIRIR HERRAMIENTAS CASE	26
1.8.1. REQUISITOS FUNCIONALES.....	27
1.8.2. FACILIDADES DE USO QUE DEBE PRESTAR UNA HERRAMIENTA CASE.....	27
1.8.3. PROCESO DE ADQUISICIÓN DE HERRAMIENTAS CASE	28
1.9. ESTRATEGIAS PARA LA CORRECTA IMPLANTACIÓN HERRAMIENTAS CASE	29
1.10. RECUPERACIÓN DE COSTOS DE COSTOS DE LAS HERRAMIENTAS CASE	31
1.10.1. OBJETIVOS DE LA RECUPERACIÓN DE COSTOS.....	32
1.10.1.1. OBJETIVOS CUANTITATIVOS	32
1.10.1.2. OBJETIVOS CUALITATIVOS.....	33

CAPÍTULO II

SOLUCIONES CLIENTE / SERVIDOR GENERADAS POR HERRAMIENTAS CASE

2.1. GENERALIDADES DE CLIENTE / SERVIDOR.....	35
2.2. ARQUITECTURA CENTRALIZADA.....	35
2.3. DEFINICIÓN DE CLIENTE / SERVIDOR.....	36
2.3.1. SERVIDOR.....	38
2.3.2. CLIENTE.....	38
2.4. CONCEPTOS FUNDAMENTALES UTILIZADOS EN C/S CON HERRAMIENTAS CASE.....	40
2.4.1. MIDDLEWARE.....	40
2.4.2. SISTEMA ABIERTO.....	40
2.4.3. DOWNSIZING.....	40
2.4.4. ROCESO DISTRIBUIDO.....	40
2.4.5. SMARTSIZING.....	40
2.4.6. OUTSOURZING.....	42
2.4.7. UPSIZING.....	42
2.4.8. RIGHTSIZING.....	42
2.5. ESTRUCTURA DE CLIENTE / SERVIDOR.....	43
2.6. CLASIFICACIÓN DE CLIENTE / SERVIDOR.....	45
2.6.1. POR EL TAMAÑO DE SUS COMPONENTES.....	45
2.6.2. POR PLANOS O CAPAS (Tier).....	48
2.6.2.1 PLANOS A NIVEL DE SOFTWARE.....	48
2.6.2.2 PLANOS A NIVEL DE HARDWARE.....	52
2.6.3. POR LA NATURALEZA DEL SERVICIO.....	52
2.7. HERRAMIENTAS CASE EN CLIENTE SERVIDOR.....	53
2.7.1. REQUERIMIENTOS DEL CLIENTE.....	54
2.7.2. REQUERIMIENTOS DEL SERVIDOR.....	55
2.8. FORMAS DE ENLACE.....	56
2.9. REINGENIERÍA Y ANÁLISIS DE IMPACTOS CASE.....	59
2.10. TÁCTICAS PARA SOLUCIONES CLIENTE / SERVIDOR GENERADAS CON CASE.....	64
2.11. CARACTERÍSTICAS DE UNA HERRAMIENTA CASE CLIENTE SERVIDOR.....	65
2.12. ASPECTOS CONSIDERABLES DE CLIENTE / SERVIDOR.....	66
 <i>CAPÍTULO III</i>	
<i>SOLUCIONES INTERNET GENERADAS POR HERRAMIENTAS CASE</i>	
3.1. GENERALIDADES DE INTERNET.....	69
3.1.1. HISTORIA DE INTERNET.....	71
3.1.2. INTERNET EN CIFRAS.....	73

3.2. TRES CAPAS Y APLICACIONES WEB	75
3.2.1. GENERALIDADES.....	75
3.2.2. COMPONENTES DE UNA APLICACIÓN WEB TRES CAPAS.....	77
3.2.3. PROGRAMACIÓN DE APLICACIONES WEB.....	77
3.3. HERRAMIENTAS CASE EN SOLUCIONES WEB	79
3.4. ESTRUCTURA DE LAS HERRAMIENTAS CASE PARA EL WEB.....	87
3.4.1. GENERALIDADES DE JAVA.....	87
3.4.2. EL GENERADOR JAVA CON RESPECTO A GENEXUS	88
3.4.3. ARQUITECTURA DE LAS HERRAMIENTAS CASE EN APLICACIONES DISTRIBUIDAS.....	88
3.5. UTILIZACIÓN DE COMPONENTES	90
3.5.1. UTILIZACIÓN DE CORBA	90
3.5.2. UTILIZACIÓN DE RMI.....	91
3.5.3. UTILIZACIÓN DE DCOM.....	92
3.6. OBJETOS GENERADOS POR HERRAMIENTAS CASE EN APLICACIONES WEB.....	92
3.5.1. TRANSACCIONES WEB.....	92
3.5.2. WEB PANELS	93
3.5.3. SMART STATIC PANELS.....	94
3.5.4. COMPRESIÓN DE PÁGINAS WEB.....	94

CAPÍTULO IV

GENEXUS DESARROLLO DE APLICACIONES

4.1. INTRODUCCIÓN A GENEXUS.....	96
4.1.1. ¿CÓMO CONSEGUIR GENEXUS?.....	98
4.1.2. GENEXUS 7.5 TRIAL VERSIÓN.....	98
4.1.2.1. LIMITACIONES DE GENEXUS 7.5 TRIAL VERSION	99
4.1.2.2. REQUERIMIENTOS DE HARDWARE Y SOFTWARE	99
4.1.2.3. INSTALACIÓN	100
4.1.3. OBJETOS BÁSICOS GENERADOS POR GENEXUS.....	102
4.2. DEFINICIÓN DE REQUERIMIENTOS PARA SOLUCIONES GENEXUS.....	102
4.3. DISEÑO DE TRANSACCIONES	103
4.3.1. OBJETOS DE LAS TRANSACCIONES.....	103
4.3.2. ESTRUCTURA DE UNA TRANSACCIÓN.....	104
4.3.3. RELACIÓN ENTRE LA ESTRUCTURA Y EL MODELO DE DATOS.....	109
4.3.4. NOMENCLATURA GENEXUS (GIK).....	110
4.3.5. TIPOS DE CONTROLES GENEXUS.....	111
4.3.6. ATRIBUTOS VINCULADOS E INTEGRIDAD REFERENCIAL	114
4.3.7. NIVELES DE UNA TRANSACCIÓN.....	115
4.3.8. TIPOS DE RELACIONES ENTRE OBJETOS.....	118
4.3.9. FORM DE UNA TRANSACCIÓN (FORMULARIOS O PANTALLAS).....	120
4.3.10. ATRIBUTOS Y FÓRMULAS	121
4.3.11. REGLAS	123
4.3.12. CALL, AFTER Y ORDEN DE DISPARO DE LAS REGLAS.....	125

4.4. DISEÑO DE REPORTES	126
4.4.1. DEFINICIÓN DE OBJETOS DE UN REPORTE	126
4.4.2. LAYOUT	126
4.4.3. FOR EACH - ENDFOR.....	129
4.4.4. ORDENAMIENTO DE DATOS ORDER (ÍNDICES).....	131
4.4.5. FOR EACH CON CONDICIONES WHERE.....	135
4.4.6. FOR EACH ANIDADOS.....	138
4.4.7. DEFINICIÓN DE VARIABLES	141
4.4.8. CONDICIONES EN UN REPORTE (CONDITIONS).....	144
4.4.9. REPORTES GENERADOS CON WIZARD	144
4.5. DISEÑO DE PROCEDIMIENTOS	146
4.5.1. MODIFICACIÓN DE DATOS.....	146
4.6. DISEÑO DE PANELES DE TRABAJO (WORK PANELS).....	149
4.6.1. DEFINICIÓN DE OBJETOS DE UN WORK PANEL.....	149
4.6.2. PASOS PARA REALIZAR UN PANEL DE TRABAJO CON WIZARD	150
4.6.3. EVENTOS EN LOS PANELES DE TRABAJO.	156
4.6.4. CONDICIONES Y REGLAS DE LOS PANELES DE TRABAJO.....	158
4.7. GRAFICACIÓN DE DATOS	160
4.8. DISEÑO DE MENÚS	163
4.9. DISEÑO Y PUBLICACIÓN DE OBJETOS WEB.....	164
 <i>CAPÍTULO V</i> <i>APLICATIVO</i>	
5.1. INTRODUCCIÓN.....	171
5.2. SOLUCIONES A GENERARSE EN LA APLICACIÓN.....	172
PROPUESTA 1: NOVELL – ORACLE, WINDOWS 98 - VISUAL FOX PRO.....	173
PROPUESTA 2: WINDOWS 2000 SERVER - SQL SERVER, WINDOWS 98 - VISUAL FOXPRO.....	173
PROPUESTA 3: WINDOWS 98 – VISUAL FOXPRO.....	174
PROPUESTA 4: WINDOWS 2000 SERVER O WINDOWS 98, VISUAL BASIC – ACEES, INTERNET INFORMATION SERVER O PERSONAL WEB.....	174
5.3. ANÁLISIS DE NECESIDADES	175
5.3.1. FUNCIONAMIENTO DEL SISTEMA.....	175
5.4. DISEÑO DE LA APLICACIÓN	178
5.4.1. TRANSACCIÓN DE PRODUCTOS	178
5.4.2. TRANSACCIÓN ADQUISICIONES.....	186
5.4.3. TRANSACCIONES CON TRES NIVELES.	192
5.4.4. DISEÑO DE BODEGAS	196
5.4.5. ASIGNACIONES A BODEGAS.....	197
5.4.6. DEVOLUCIONES POR BODEGAS.....	200
5.4.7. ÓRDENES DE EGRESO E INGRESO	202
5.4.8. BAJAS DE PRODUCTOS: GENERALES Y POR BODEGAS.....	203
5.4.9. DISEÑO DE LA TRANSACCIÓN VENTAS.....	204

5.5. IMPLEMENTACIÓN DE LA APLICACIÓN	206
5.5.1. NOVELL - ORACLE CON WINDOWS 98 Y VISUAL FOXPRO	206
5.5.2. DBMS OPTIONS PARA ORACLE	208
5.5.3. PREFERENCIAS PARA ORACLE	209
5.5.3. OBJETOS GENERADOS EN LA APLICACIÓN	211
5.5.4. DIAGRAMAS RELACIONALES GENERADOS POR GENEXUS	213
5.5.4.1. DIAGRAMA ENTIDAD RELACIÓN POR TRANSACCIONES	214
5.5.4.2. DIAGRAMA ENTIDAD RELACIÓN POR TABLAS	215
5.5.5. BASE DE DATOS VISTA DESDE GENEXUS	216
5.6. PROTOTIPO / PRODUCCIÓN	219
5.7. APLICACIÓN PARA EL WEB	220
5.8. ESTRUCTURA GENERAL DEL APLICATIVO	225

***CAPÍTULO VI
EVALUACIÓN DE UNA HERRAMIENTA CASE***

6.1. GENERALIDADES PARA EVALUAR HERRAMIENTAS CASE	229
6.1.1. CAUSAS DEL FRACASO DE LAS HERRAMIENTAS CASE	230
6.1.2. TÉCNICA DE EVALUACIÓN	231
6.1.3. PESOS ASIGNADOS A CRITERIOS Y SUBCRITERIOS	231
6.2. ASPECTOS PRELIMINARES	232
6.2.1. CRITERIOS DE SELECCIÓN Y ADQUISICIÓN	232
6.2.2. TABLA PARA SELECCIÓN Y ADQUISICIÓN	236
6.2.3. CRITERIOS DE HARDWARE Y SOFTWARE DISPONIBLE	238
6.2.4. TABLA PARA EVALUAR EL HARDWARE DISPONIBLE	238
6.2.5. TABLA PARA EVALUAR EL SOFTWARE DISPONIBLE	239
6.2.6. CIERRE DE LA ETAPA DE ASPECTOS PRELIMINARES	239
6.3. ASPECTOS INTERNOS	240
6.3.1. FASES DEL CICLO DE VIDA QUE CUBRE	240
6.3.2. METODOLOGÍAS SOPORTADAS	241
6.3.3. USUARIOS CONCURRENTES	241
6.3.4. COMPONENTES	242
6.3.5. CONTROL DE PROYECTOS	243
6.3.6. PLATAFORMA DE SOFTWARE REQUERIDA	243
6.3.7. MANEJADOR DE BASES DE DATOS	244
6.3.8. FACILIDADES DE USO	245
6.3.9. FUNCIONALIDAD	246
6.3.10. CURVA DE APRENDIZAJE	247
6.3.11. AYUDA EN LÍNEA	247
6.3.12. FLEXIBILIDAD	248
6.3.13. TIPO DE INTEGRACIÓN	249
6.3.14. CIERRE DE REQUERIMIENTOS INTERNOS	250

6.4. ASPECTOS EXTERNOS	251
6.4.1. SOPORTE TÉCNICO	251
6.4.2. ENTRENAMIENTO	252
6.4.3. MATERIAL DE APOYO	253
6.4.4. PRESTIGIO DEL DESARROLLADOR DE LA HERRAMIENTA	254
6.4.5. COSTOS	255
6.4.6. CIERRE DE LOS REQUERIMIENTOS EXTERNOS	256
6.5. CIERRE GENERAL DE LA EVALUACIÓN	257
6.6. INTERROGANTES UNA VEZ IMPLANTADA LA HERRAMIENTA CASE	258
6.7. HERRAMIENTAS CASE MÁS CONOCIDAS EN EL MERCADO	259

CAPÍTULO VII

CONCLUSIONES Y RECOMENDACIONES

7.1. VERIFICACIÓN DE LA HIPÓTESIS	261
7.2. CONCLUSIONES	263
7.3. RECOMENDACIONES	265

BIBLIOGRAFÍA

ANEXOS