

**UNIVERSIDAD DON BOSCO  
FACULTAD DE INGENIERIA**



**INVESTIGACION SOBRE DIFERENTES HERRAMIENTAS CASE Y SU  
APLICACIÓN EN LAS DIVERSAS FASES DE LA INGENIERIA DEL  
SOFTWARE. DESARROLLO DE UN CASO PRACTICO.**

**TRABAJO DE GRADUACIÓN PREPARADO PARA LA  
FACULTAD DE INGENIERIA**

**PARA OPTAR AL GRADO DE  
INGENIERO EN CIENCIAS DE LA COMPUTACION**



**PRESENTADO POR:  
SANDRA IVETT ARENIVAR ALVARADO  
XIOMARA ARELY CHÁVEZ REYES**

**ASESOR  
LIC. REINA ELIZABETH DURAN DE ALVARADO**

**SEPTIEMBRE DE 1999  
SAN SALVADOR, EL SALVADOR, CENTROAMERICA**

**UNIVERSIDAD DON BOSCO**

**RECTOR  
ING. FEDERICO MIGUEL HUGUET RIVERA**

**SECRETARIO GENERAL  
PBRO. PEDRO JOSE GARCIA CASTRO**

**DECANO DE LA FACULTAD DE INGENIERIA  
ING. CARLOS BRAN**

**ASESOR  
DEL TRABAJO DE GRADUACION  
LIC. REINA ELIZABETH DURAN DE ALVARADO**

**JURADO EVALUADOR  
LIC. JULIO CESAR MONTES  
ING. JAIME ANTONIO ANAYA**

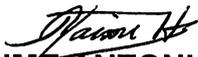
**UNIVERSIDAD DON BOSCO**

FACULTAD DE INGENIERIA

ESCUELA DE COMPUTACION

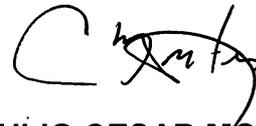
JURADO EVALUADOR DEL TRABAJO DE GRADUACION

**“INVESTIGACION SOBRE DIFERENTES HERRAMIENTAS CASE  
APLICADAS A LAS DIVERSAS FASES DE LA INGENIERIA DEL  
SOFTWARE. DESARROLLO DE UN CASO PRACTICO”**



**ING. JAIME ANTONIO ANAYA**

**JURADO**



**LIC. JULIO CESAR MONTES**

**JURADO**



**LIC. REINA ELIZABETH DURAN DE ALVARADO**

**ASESOR**

## DEDICATORIA

Gracias **Dios Nuestro Señor**, por darme sabiduría, luz y fuerza para poder finalizar ésta importante fase en mi vida.

A mis padres, **Fidencio y Elida**, ya que éste triunfo es tan suyo como mío, por todo el apoyo que me han brindado, por la confianza que han depositado en mi y principalmente por el amor que me dan ya que sin ustedes nunca lo hubiera logrado, los amo.

A mis hermanos **Mario, Lorena, Yanira, Michael y Reina** por que siempre estuvieron a mi lado brindadome consejos, amor y apoyo, los quiero mucho.

A mis abuelos, **Andrea de Chávez y Francisco Reyes** a los que quiero y dedico este triunfo con mucho cariño.

A toda mi demás familia, pero en especial a mis tíos **Edmundo Reyes y Adonai Chávez**, por el apoyo y tiempo que brindaron para la realización de ésta investigación y sobre todo por la confianza que pusieron en mi, los quiero mucho.

A **Sandra** por convertirse en una gran amiga y por compartir todos los momentos de angustia y las duras pruebas que pasamos, espero que siempre sigamos así.

A **David**, ya que sin tu apoyo, confianza y comprensión jamás hubiera llegado a culminar ésta etapa.

A mi asesora, **Lic. Reina de Alvarado** ya que fue nuestra guía y este trabajo se lo debo a usted. Gracias de corazón.

A mis amigos *Blanca, Monica, Brenda y Juan* por su ayuda y cariño.

**Xiomara Arely Chávez Reyes.**

## DEDICATORIA

Dedico esta tesis a Dios todopoderoso y a la Virgen María por darme la vida y este triunfo que llena de dicha a toda mi familia y sobre todo a mí.

A mi Madre: **Ana Silvia** porque es ella quien con su dedicación, sacrificio, apoyo y cariño y sobre todo por todos aquellos consejos que me ha brindado a lo largo de mi vida de estudiante. Gracias Mamí te quiero mucho este triunfo es mas tuyo que mío.

A mis Abuelitas: **Lucita y Mamá Vita** por todo el cariño, oraciones y además porque siempre estuvieron a mi lado en todo momento.

A mis Hermanos: **Luis Roberto y Fátima**, por sus consejos y alientos para seguir siempre adelante y sobre todo por la paciencia que me han tenido por mi mal carácter.

A mi Tía: **Romy**, porque a pesar que hace muy poco tiempo que la conozco me demostrado un gran cariño y por todo el apoyo incondicional que me han brindado.

A mis Tíos: **Beto, Mauricio y Hugo**, por estar siempre pendientes de mí y ayudándome en todo momento.

A mis primos: Por todo el apoyo incondicional que me han dado.

A mis amigos: Por toda la ayuda que me han dado y sobre por estar conmigo en los momentos más difíciles. Gracias los quiero mucho.

A mis compañeros de Trabajo: Especialmente a Sarita, Toto, Miguel Angel, José Luis, Francisco y todos los demás por todo el apoyo que me han dado para culminar esta etapa de mi vida.

A la familia Chavéz Reyes: Por todas las atenciones que me brindaron y sobre todo por todo el cariño y por considerarme y hacerme sentir parte de su familia. Gracias los quiero mucho.

A mi compañera de tesis: **Xiomara** porque no solo supo ser una excelente compañera de tesis sino además una gran amiga, por toda la paciencia y el cariño que demostró siempre. Gracias te quiero y admiro mucho.

A mi asosora: **Lic. Reyna de Alvarado**, porque todo el tiempo que nos dedico para hacer realidad uno de mis sueños y además de ser excelente guía.

**SANDRA IVETT ARENIVAR ALVARADO**

## AGRADECIMIENTOS ESPECIALES

De una manera especial agradecemos a todas aquellas personas que contribuyeron incondicionalmente al desarrollo de este trabajo de graduación:

- Ing. Ruben Mejía
- David Alfonso Rivas Cartagena
- William Ernesto Martínez
- Ing. Ricardo Enrique Herrera López
- José Edmundo Reyes Contreras
- Adonai Chávez Martínez
- Ing. Raúl Henríquez
- Ing. Ricardo Aguilar Abarca
- Lic. Miguel Angel López
- José Luis Rodríguez Flores
- Yanira del Carmen Reyes
- Omar Henríquez Calero

Pero sobre todo a la Lic. Reina Elizabeth de Alvarado pues es la persona que compartió su sabiduría, tiempo y dedicación, ya que sin su ayuda no hubiéramos logrado este triunfo, pues es tan suyo como nuestro.

Así mismo a todas aquellas personas que nos brindaron su tiempo, las diversas opiniones y recomendaciones para el desarrollo de la investigación.

## INDICE

	PAGINA
<b>INTRODUCCION.....</b>	<b>i</b>
<b>PROLOGO.....</b>	<b>ii</b>
<b>CAPITULO I. DESCRIPCION DEL PROBLEMA.....</b>	<b>1-10</b>
1.1.1 Objetivo General .....	1
1.1.2 Objetivos Específicos .....	1-2
1.2 Antecedentes.....	2
1.2.1 Reseña Histórica.....	2-5
1.2.2 Situación Actual .....	5-6
1.2.3 Tendencias Futuras.....	7
1.3 Planteamiento del Problema .....	7-8
1.4 Justificación e Importancia.....	8
1.5 Proyección Social.....	8-9
1.6 Alcances y Limitaciones .....	9-10
<b>CAPITULO II. METODOLOGIA DE INVESTIGACION</b>	
<b>    Y TECNICAS A UTILIZAR .....</b>	<b>11-12</b>
2.1.1 Estudio Bibliográfico .....	11

2.1.2 Estudio de Campo .....	11
2.1.2.1 Encuestas .....	12
2.1.2.2 Talleres .....	12
2.1.2.3 Entrevistas .....	12
2.1.3 Metodología de Prototipo .....	12

### **CAPITULO III. ANALISIS DEL MERCADO LOCAL DE LAS**

#### **HERRAMIENTAS CASE.....13-29**

3.1 Introducción.....	13
3.2 Definición del universo .....	14
3.3 Análisis de Pensum Curriculares.....	15
3.4 Análisis de Encuestas Dirigidas .....	15-22
3.5 Análisis de la Formación de Talleres .....	23-25
3.6 Análisis de Entrevistas Dirigidas.....	25-28

### **CAPITULO IV PROPUESTA DE HERRAMIENTAS CASE.....29-37**

4.1 Introducción.....	29-30
4.2 Propuesta A .....	30-34
4.3 Propuesta B.....	34-36
4.4 Propuesta para profesionales que deseen adquirir conocimientos sobre herramientas CASE .....	36-37

## **CAPITULO V CONCEPTOS BASICOS DE HERRAMIENTAS**

### **CASE EN EL DESARROLLO .....38-55**

5.1 Introducción.....	38
5.2 Herramienta .....	39
5.2.1 Beneficio del empleo de Herramientas.....	39-40
5.3 Herramienta CASE.....	40-43
5.3.1 Beneficio de CASE.....	43-45
5.4 Clarificación de CASE .....	45-49
5.5 Descripción general de distintos tipos de herramientas .....	49
5.6 Aplicaciones y usos de Herramientas CASE.....	49-55
5.7 Evaluación de una herramienta CASE .....	55
5.8 Herramienta CASE a utilizar .....	55

## **CAPITULO VI TECNICAS APLICADAS A LAS**

### **HERRAMIENTAS CASE EN EL DESARROLLO**

### **DE SOFTWARE .....56-68**

6.1 Introducción.....	56
6.2 Principales Técnicas .....	56-50
6.2.1 DFD .....	56-60
6.2.2 E/R.....	60-64
6.2.3 Diagramas de Estructuras .....	64-65
6.2.4 Matrices .....	65-66
6.2.5 Diagrama de Transición de Estados .....	66-67
6.2.6 Teoría de la Normalización .....	67-68

## **CAPITULO VII METODOLOGIAS APLICADAS A LAS HERRAMIENTAS**

### **CASE EN EL**

### **DESARROLLO DE SOFTWARE .....69-92**

7.1 Introducción.....	69-73
7.1.1 Reseña Histórica.....	69-70

7.1.2 Conceptos y Elementos del Método .....	70-72
7.1.3 Encapsulación de los métodos.....	72-73
7.1.4 Uso de método y herramientas .....	73
7.2 Principales Metodologías .....	74-92
7.2.1 Modelo NIAM .....	74-77
7.2.2 Diagrama Jackson.....	77-79
7.2.3 Diagrama de Warnier Orr .....	79-80
7.2.4 Prototipado y RAD .....	80-81
7.2.5 Metodología MERISE .....	82-86
7.2.6 Metodología SSAD .....	86-88
7.2.7 Metodología de Ingeniería de la Información .....	88-91
7.5.8 Metodologías Orientadas a Objeto.....	91-92

**CAPITULO VIII DIVERSAS HERRAMIENTAS CASE ..... 93-139**

8.1 Introducción.....	93
8.2 ER-Win.....	93-95
8.3 System Architect .....	95-98
8.4 Snap .....	98-103
8.5 Visible Analyst Workbech .....	103-105
8.6 Excelerator.....	105-107
8.7 Designer/2000.....	107-124
8.8 Easy-CASE.....	124-126
8.9 Power Designer .....	126-132
Cuadro Comparativo de Herramientas .....	133-139

**CAPITULO IX DEFINICION Y ANALISIS**

**DE CASO PRACTICO ..... 140-150**

9.1 Introducción.....	140
9.2 Definición del sistema de inventario .....	140
9.3 Metodología aplicada .....	141-142

9.4 Descripción de entidades involucradas .....	142-143
9.5 Diseño del sistema de inventario .....	143-150

**CONCLUSIONES .....151-153**

**RECOMENDACIONES .....154**

**GLOSARIO.....155-157**

**BIBLIOGRAFIA .....158-161**

**ANEXOS**

Anexo A. Formato De Encuestas .....	A-1 – A-4
Anexo B. Formulacion De Hipotesis .....	B-1
Anexo C. Propuestas de Contenidos .....	C-1 – C-7
Anexo D. Formato de Entrevistas .....	D-1
Anexo E. Simbología CASE .....	E-1 – E-9
Anexo F. Reporte del Modelo Conceptual .....	F-1 – F-21

## INTRODUCCION

Las herramientas CASE ( Ingeniería del Software Asistida por Computadora), surge como una necesidad a respuestas de problemas complejos en tiempos cortos. Los avances tecnológicos demandan de los desarrolladores de software que los sistemas estén funcionando de forma oportuna. Con el desarrollo de las herramientas CASE se busca mecanizar la labor de análisis, diseño, programación de las aplicaciones, exigiendo a los profesionales en el área de informática una buena elaboración conceptual de sus sistemas. Además en El Salvador estas herramientas se han venido popularizando a nivel de las instituciones financieras, gobierno y otras empresas privadas fuertes, lo que genera mayor demanda en el conocimiento de los mismos por parte de los profesionales y a la vez ofrece más oportunidades laborales.

Ha sido propósito de este documento desarrollar un estudio sobre las diferentes herramientas CASE sus características, aplicaciones y usos. Además de proponer su incorporación en el pensum curricular de los alumnos de ingeniería en computación de la Universidad Don Bosco, lo que además se completó con un temario que es desarrollado para proporcionar a los maestros y alumnos un documento base para el estudio de los mismos.

## PROLOGO

Dentro de los capítulos contemplados en esta investigación se encuentran los siguientes:

CAPITULO I: Consiste en la descripción del problema el cual contiene los siguientes apartados : objetivos que se alcanzaron durante el transcurso del proyecto, alcances y limitaciones planteadas, los antecedentes de dicho tema dividiéndolo en una breve reseña histórica, la situación actual y las tendencias futuras para luego explicar el planteamiento del problema, la relevancia y utilidad del desarrollo del mismo.

CAPITULO II: Comprende la metodología que se utilizó al desarrollar dicha investigación y las diversas técnicas empleadas para la recopilación de la información.

CAPITULO III: En el se plasma el análisis del mercado local de las herramientas CASE, tratando de abarcar los tres sectores importantes entre los que se encuentran los estudiantes, catedráticos y empresarios en informática.

CAPITULO IV: En el se elaboran tres propuestas, las cuales fueron desarrolladas en base a los resultados obtenidos en el análisis de mercado, las cuales consisten en : las dos primeras incorporan el conocimiento de las herramientas CASE dentro de la curricula de la carrera de ingeniería en computación; la última pretende dar a conocer una solución a todos aquellos profesionales que en el transcurso de su educación universitaria no adquirieron dicho conocimiento.

CAPITULO V: Es el complemento del capítulo anterior, ya que en el se desarrolla el contenido propuesto que servirá de guía a los catedráticos y alumnos en el conocimiento de las herramientas CASE.

CAPITULO VI: Aquí se muestran las diversas técnicas aplicadas en el desarrollo de aplicaciones.

CAPITULO VII: Contiene un resumen de diversas metodologías aplicadas a las herramientas CASE en el desarrollo de software.

CAPITULO VIII: Se dan a conocer una serie de herramientas CASE, que según el estudio realizado son las que se encuentran utilizando actualmente en el mercado, finalizando con un cuadro comparativo sobre las mismas, donde se resumen característica, beneficios, requerimiento y clasificación de diversas herramientas.

CAPITULO IX: Se da a conocer el funcionamiento y los objetivos del caso práctico desarrollado, el cual consiste en la realización de un prototipo (sistema de inventario de equipo informático), auxiliado por la herramienta Power Designer de Power Soft.

## **CAPITULO I**

### **DESCRIPCION DEL PROBLEMA**

#### **1.1. OBJETIVOS**

##### **1.1.1 OBJETIVO GENERAL**

Desarrollar una investigación sobre diferentes herramientas CASE (Ingeniería del Software Asistido por Computadoras) explorando sus características, aplicaciones y usos, además de analizar la pertinencia de incorporar su estudio en la curricula de la carrera de Ingeniería en Ciencias de la Computación.

##### **1.1.2 OBJETIVOS ESPECIFICOS**

- Analizar diversas herramientas CASE, características, beneficios, metodologías y técnicas.
- Investigar el conocimiento que los alumnos, maestros y profesionales de informática poseen sobre las herramientas CASE.
- Determinar si en el mercado laboral de profesionales en informática es importante el conocimiento sobre las herramientas CASE.
- Desarrollar propuestas que permita a la Universidad, específicamente a la escuela de computación incorporar las bases sobre herramientas CASE en el pensum curricular.

- Analizar cual es la forma adecuada de capacitación para quienes ya terminaron su formación académica en lo que a la carrera de ingeniería en Ciencias de la Computación respecta.
- Utilizar una herramienta CASE que sirva de apoyo al desarrollo de aplicaciones mediante un caso práctico desde su fase inicial<sup>1</sup> hasta el mantenimiento del mismo

## **1.2 ANTECEDENTES**

Antes de iniciar con la reseña histórica es importante definir que herramienta es cualquier dispositivo que cuando se emplea en forma apropiada, mejora la realización de una tarea.

Para el análisis de sistemas es importante el uso de algunas herramientas. Ellas mejoran la forma en que ocurre el desarrollo y tienen influencia sobre la calidad del resultado final.

CASE es Ingeniería del Software Asistido por Computadoras (COMPUTER Aided/Assisted Software/System Engineering), cuya idea es proporcionar un conjunto integrado de herramientas que enlazan y automatizan todas las fases del ciclo de vida del software

### **1.2.1 Reseña Histórica**

CASE surge a raíz de una crisis en el proceso del desarrollo de sistemas en la cual florece la necesidad de crear herramientas automatizadas para ayudar a los desarrolladores de "software". En los años 60 se comienzan a buscar soluciones a esta crisis por lo que surgen técnicas para el desarrollo y la documentación de diseños como lo fue los flujogramas estructurados.

Podríamos afirmar que, los entornos Unix e InterLisp constituyen un primer conjunto de herramientas que, junto a los lenguajes de cuarta generación (4GL), surgidos a finales de los 70, han influido en la concepción de las herramientas CASE.

---

<sup>1</sup> Análisis, Diseño, Codificación, Prueba

Fue hasta los años 80 que comienzan a introducirse nuevas técnicas enfocadas a las fases del desarrollo del ciclo de vida de un sistema. Inicialmente el enfoque de CASE fue crear herramientas que ayudaran en la realización de programas como los traductores, compiladores, ensambladores, macro procesadores y cargadores (loaders). A medida que las computadoras comenzaron a crecer en capacidad y complejidad, comenzaron a surgir nuevas herramientas; aparecen editores de programas, depurador, analizadores de código y otros.

Entonces surge una nueva visión de cómo desarrollar sistemas que cumplieran con la necesidad de establecer requerimientos adecuados, diseñar soluciones apropiadas, implementar esas soluciones, probarlas y documentarlas. Es un proceso largo en el que se produce "software" que requiere cambios según pasa el tiempo. La interacción con usuarios en cada una de las etapas del ciclo de vida del sistema es clave. Esta visión de programación en grande resulta de la utilización de todas las herramientas CASE.

A mediados de los 80, las CASE se popularizan y surgen las primeras herramientas de documentación y diagramación automática. Es una época en la que explotan el número de seminarios, cursos, revistas, libros y congresos dedicados al tema. También tienen un papel importante en este periodo la aparición de las estaciones de trabajo, que aportan una buena interfaz gráfica, asociada a una gran capacidad de proceso y que constituyen la plataforma original de muchos CASE.

Además surge el concepto de repositorio/diccionario como núcleo de un entorno CASE, así como generadores de programas y aplicaciones que automatizan gran parte de las últimas fases del ciclo de vida. En paralelo, también aparecen los gestores de proyectos, algunos de los cuales se integran con herramientas CASE. A finales de los años 80 se produce un considerable aumento en la difusión

de estos productos y empieza la etapa de asimilación de la tecnología, que fracasa debido, fundamentalmente a tres factores<sup>2</sup>: limitaciones de la primera generación de productos, Implantación incorrecta y falsas expectativas sobre sus posibilidades.

En definitiva la tecnología CASE ha experimentado la clásica evolución que sufren aquellos paradigmas que aparecen como panacea universal capaz de resolver todos los problemas de desarrollo de sistemas de información: técnicas estructuradas, inteligencia artificial, lenguajes de cuarta generación y en estos momentos la orientación a objetos.

Afortunadamente a mediados de los años 90 empieza a surgir una segunda generación de herramientas que superan gran parte de las limitaciones existentes. Además, los propios usuarios de las herramientas conocen mejor sus posibilidades y han aprendido a poner una expectativas mas justas sobre éstas, mejorando también los procesos de adopción de metodologías y herramientas.

La historia de herramientas de software se puede resumir en la figura 1.

Periodo	Actividad
Principio de los 80's:	Documentación Automatizada, Diagramas Computarizados, Herramientas de Análisis y Diseño.
Mediados de los 80's:	Análisis automático de Diseño y Verificación, Banco Automático de Información del sistema.
Finales de los 80's	Generación Automática de Código, Automatización de Diseño de Enlace.
Principios de los 90's	Dispositivo de metodología inteligente, interfaces reusables como metodología de desarrollo.

<sup>2</sup> Tomado del libro "Elementos y herramientas en el desarrollo de sistemas de información una visión actual de la Tecnología CASE" de Mario Piattini

Con los avances tecnológicos presentes el costo del hardware ha disminuido sustancialmente, pero por otro lado el costo del software va en ascenso. La razón principal del alto costo de software es que la tecnología envuelta en el desarrollo es mucho más intensa y demanda más trabajo. Los proyectos de software demandan al factor humano, también trae consigo el alza en la probabilidad de error. Las herramientas CASE buscan facilitar las tareas de desarrollo y mantenimiento de aplicaciones contribuyendo a la reducción de costos de los mismos.

### **1.2.2 Situación Actual**

Debido a su potencial de uso para facilitar el trabajo del desarrollador de software las herramientas CASE son cada vez más conocidas y aplicadas en el mundo entero.

Por su parte en El Salvador ya existen muchas empresas de sectores importantes que las están introduciendo como una herramienta de trabajo entre las que podemos mencionar al sector gubernamental, sector financiero, y la empresa privada.

Sin embargo al entrevistar a los usuarios de estas herramientas se perfila por un lado la dificultad de aprendizaje por no haber contado con conocimientos previos de las mismas y por otro la subutilización de dichas herramientas, ya que muchos los conocen solo como diagramadores.

Al realizar la investigación preliminar sobre el conocimiento en general de estas herramientas se detectó de que muchos tienen conceptos erróneos.

La mayoría de los profesionales que las conocen o que dicen conocerlas es porque en el desempeño laboral ha existido la necesidad de aprenderlas y ha sido sometido a capacitaciones especiales sobre alguna herramienta en particular.

El surgimiento y auge de estas herramientas ha motivado a ciertas instituciones educativas a emprender estudios más específicos sobre este tema, tal es el caso de la Universidad DON BOSCO que posee un trabajo de graduación sobre el tema "Desarrollo de Aplicaciones Utilizando una metodología auxiliada por herramientas C.A.S.E.", la cual presenta lineamientos y procedimiento que faciliten el desarrollo de aplicaciones de manera estructurada dentro de una metodología de desarrollo formal.

Por su parte, en otra universidad<sup>3</sup> se han llevado a cabo los temas: "Introducción al desarrollo de sistema de información utilizando herramientas CASE un caso de aplicación", en el que se presenta la evaluación de una herramienta CASE y analiza como ésta mejora la confiabilidad y el mantenimiento del sistema desarrollado y, el "Desarrollo de herramientas de análisis y diseño aplicadas a El Salvador", que resume el uso de técnicas de diseño estructurado, y una metodología que permite al analista realizar un diseño coherente y eficiente.

Aún cuando estos trabajos inician un estudio sobre estas herramientas, ninguno hace un análisis comparativo de las mismas, explorando sus características, aplicaciones y usos.

Los trabajos de graduación antes mencionados y difusión de estas herramientas esta propiciando actualmente en El Salvador el estudio sistemático de los mismos pero que a la fecha solo en la Universidad Don Bosco se ha recién incorporado en la curricula de Ingeniería en Computación.

Los trabajos de graduación antes mencionados y la difusión sobre estas herramientas esta propiciando actualmente en El Salvador el estudio sistemático del mismo pero que a la fecha solo en la Universidad Don Bosco se ha recién incorporado en la curricula de Ingeniería en Computación.

---

<sup>3</sup> Universidad José Simeón Cañas, ver bibliografía.

### **1.2.3 Tendencias Futuras**

Técnicamente el futuro de las herramientas CASE vislumbra la continuación de soporte a múltiples metodologías al mismo tiempo. Además pretenden mejorar su soporte al desarrollo de lenguajes mixtos entre los cuales envuelve Java, C++, Visual Basic y otros; busca lograr la unión e interconectividad de componentes de diferentes lenguajes.

Las herramientas CASE, que actualmente almacenan su información de modelos en archivos pretenden utilizar "repositorios" que permitan hacer accesibles estos modelos a múltiples herramientas. Esto facilitará el intercambio de información entre desarrolladores de sistemas que les permitan crear esos modelos. Otra tendencia de CASE es lograr reducir la cantidad de herramientas necesarias para modelar una aplicación así como el esquema de base de datos.

Lo anterior incrementa la demanda de profesionales que posean los conocimientos adecuados sobre este tipo de herramientas ya que con esto se estará ahorrando tiempo y costos adicionales a las organizaciones.

## **1.3 PLANTEAMIENTO DEL PROBLEMA**

Existe por un lado la necesidad de preparar al futuro profesional en informática con conocimientos básicos sobre las herramientas CASE, que le permitan mejores oportunidades en el mercado laboral; para completar la solución a este problema se desarrolla un análisis de éstas herramientas explorando sus características, aplicaciones y usos para proveer a los usuarios de las mismas, criterios de análisis y bases para facilitar su aprendizaje ya que se determinó por medio del estudio

preliminar<sup>4</sup> un desconocimiento por parte de los Ingenieros en Sistemas a cerca de las herramientas CASE.

## **1.4 JUSTIFICACIÓN E IMPORTANCIA**

Desde el punto de vista que el objetivo principal de las herramientas CASE es facilitar el desarrollo y mantenimiento de aplicaciones mejorando la calidad del producto en menor tiempo y por lo tanto reduciendo los costos de desarrollo es importante el conocimiento fundamental de las mismas por parte de los profesionales en el área de informática ya que de ésta forma se agiliza su gestión dentro de la empresa. Favoreciendo a los empresarios al momento de invertir en capacitaciones para dichos empleados.

Además en El Salvador existe un mercado importante que demanda de profesionales con conocimientos de ésta tecnología como lo es el sector gubernamental, financiero, entre otros y es necesario conocer cuales son sus demandas específicas sobre el tema.

## **1.5 PROYECCIÓN SOCIAL**

A fin de facilitar la labor para la escuela de computación se ha llevado a cabo un estudio sobre la pertinencia de incorporar las herramientas CASE en el plan de estudio de la carrera de Ingeniería en Computación desde el punto de vista del empleador, estudiante y catedrático, permitiendo tomar la decisión más acertada que contribuya al logro de la misión que la escuela orienta a proveer los conocimientos científicos y tecnológicos para proponer soluciones informáticas de alto nivel en el área de: investigación, análisis, diseño e implementación de sistemas.

---

<sup>4</sup>Ver resultados de dicho estudio en el capítulo III

Además se proporciona un documento que beneficie a todos aquellos estudiantes universitarios y catedráticos de la carrera en ingeniería en ciencias de la computación, y de esta forma tendrán una guía en el cual se den a conocer los beneficios que se obtienen cuando se utilizan herramientas CASE junto con una metodología y técnicas adecuadas.

Así mismo se beneficia a todos aquellos analistas y programadores que desean utilizar herramientas CASE, y de esta forma se presenta un documento que permita verificar las características y beneficios que le proporciona una herramienta específica junto con su metodología y técnicas, con esto podrán observar lo útil que son muchas de estas herramientas en todas las fases del desarrollo de sistemas.

## **1.6 ALCANCES Y LIMITACIONES**

### **1.6.1 ALCANCES**

- Se investigó con los gerentes informáticos la pertinencia de estudio de los futuros profesionales en el ámbito informático a cerca de las herramientas CASE.
- Se investigó con los gerentes informáticos los tópicos que consideran fundamentales en el aprendizaje de las herramientas CASE.
- Se proporciona un documento que presenta tópicos que sirven de apoyo a los catedráticos y alumnos en el estudio de las herramientas CASE.
- Se desarrollaron encuestas y entrevistas con profesionales, maestros y alumnos sobre el conocimiento y uso de éstas herramientas y sus expectativas de aprendizaje.

- Se analizaron los pensum de diferentes universidades que ya incorporan el estudio de las herramientas CASE.
- Se desarrolló una investigación por medio de talleres en el que se incluirán alumnos, maestros y profesionales del ámbito informático para analizar causas del desconocimiento y/o subutilización de herramientas CASE.

### **1.6.2 LIMITACIONES**

- La recopilación de las herramientas CASE solo incluye aquellas herramientas mas utilizadas dentro del país.
- La aplicación desarrollada se realizó a nivel de prototipo, ya que con esto solo se busca demostrar el potencial de uso de las herramientas CASE la cual es efectiva en todo el ciclo de vida del desarrollo de aplicaciones.

## **CAPITULO II**

### **METODOLOGIA DE INVESTIGACION Y TECNICAS A UTILIZAR**

#### **2.1 METODOLOGIA DE LA INVESTIGACION**

##### **2.1.1. Estudio Bibliográfico**

Para el desarrollo de este proyecto de investigación se realizó un estudio exhaustivo de información bibliográfica sobre las herramientas CASE, a fin de conocer ampliamente sobre el tema, además se han consultado estudios publicados en Internet.

##### **2.1.2. Estudio de Campo**

El estudio de campo se realizó a través de las visitas a empresas del sector gubernamental, financieras y privadas con el fin de determinar el mercado que tienen hasta el momento las herramientas CASE, y ver la forma en que se están utilizando en el desarrollo de aplicaciones.

Con esta investigación se llegó a conocer las diversas opiniones de los empleadores, catedráticos y alumnos sobre la pertinencia de incorporar estas herramientas dentro del pensum curricular de la carrera en Ingeniería en Ciencias de la Computación cuyo propósito fue tener las bases fundamentales para desarrollar propuestas que sirvan al beneficio de los futuros profesionales en el área de Informática.

Para llevar a cabo el estudio de campo se utilizaron las técnicas siguientes:

### **2.1.2.1 Encuestas**

El objetivo de estas encuestas fue determinar el conocimiento que tienen los estudiantes universitarios, catedráticos y empleadores del área de informática a cerca de las herramientas CASE, y determinar la utilización de dichas herramientas.

### **2.1.2.2 Talleres**

Los cuales fueron dirigidos a estudiantes universitarios en la carrera de Ingeniería en computación, catedráticos y profesionales de esa misma área con la finalidad de confrontar opiniones y llegar a resultados concretos.

### **2.1.2.3 Entrevistas**

Las entrevistas han permitido identificar el conocimiento que tienen los diferentes Administradores de Centros de Cómputo sobre este tema. Además de conocer las expectativas que tienen los futuros profesionales de Informática sobre lo que son herramientas CASE. Así mismo han servido de base para fundamentar las propuestas de incorporación de dichas herramientas dentro de la curricula para determinar el contenido apropiado, y además para conocer de que forma se están empleando estas herramientas.

## **3.1.3 Metodología de Prototipo**

Se utilizará para el desarrollo del sistema, el cual nos sirve como un mecanismo de definición de requisitos para que se realice la ingeniería del software como una visión hacia la calidad y facilidad de mantenimiento.

## **CAPITULO III**

### **ANALISIS DEL MERCADO LOCAL DE LAS HERRAMIENTAS CASE**

#### **3.1 INTRODUCCION.**

El presente capítulo ha sido elaborado como parte del estudio de campo realizado en varias empresas del país. Con el fin de determinar la utilización de las herramientas CASE además de conocer el mercado laboral y la pertinencia de incorporar dichas herramientas dentro de la formación profesional de los ingenieros en computación. Este análisis comprendió las etapas siguientes:

- 1) Análisis de Pensum Curriculares de varias universidades con el fin de determinar si incorporan el estudio de dichas herramientas y de que forma están incluidas dentro de la carrera en Ciencias de la Computación.
- 2) Encuestas dirigidas las cuales sirvieron para determinar el conocimiento y utilización de dichas herramientas.
- 3) Formación de Talleres para confrontar opiniones de empleados, alumnos y catedráticos.
- 4) Análisis de Entrevistas dirigidas a empleadores desempeñándose en el área de informática del sector gubernamental, financiero y empresa privada.

### **3.2 DEFINICION DEL UNIVERSO**

Para realizar el estudio de campo se ha especificado el universo a partir del cual se establece la muestra para llevar a cabo la investigación. .

El Universo en este caso esta compuesto por empresarios, docentes y alumnos.

A Continuación se define cada uno de los sectores comprendidos en el universo:

#### **EMPRESARIOS:**

Los empresarios fueron de gran de importancia para el estudio de campo ya que son ellos los que demandan las necesidades tecnológicas y son los encargados en reclutar a los ingenieros al momento de hacer una contratación.

#### **ALUMNOS:**

Este sector nos ayudó a conocer las expectativas que tienen los alumnos de la importancia que existe en incorporar dentro su formación profesional este tipo de herramientas, además de la ubicación dentro del Pensum curricular.

#### **DOCENTES:**

Los docentes son los facilitadores de transmitir las bases del conocimiento a los alumnos, por lo que fue necesario conocer las diversas opiniones que tienen sobre estas herramientas y si es necesario la incorporación dentro de la carrera de ingeniería en ciencias de la computación.

### **3.3 ANALISIS DE PENSUM CURRICULARES**

Este estudio consistió en comparar diversos pensum de 6 universidades Salvadoreñas que imparten la carrera de Ingeniería o Licenciatura en Ciencias de la Computación con el fin de determinar si incluyen alguna materia con el nombre de herramientas CASE ó tópicos dentro del contenido de otra asignatura, pudiendo observar que en estos momentos cinco de éstas no la incorporan dentro de una materia obligatoria o electiva técnica. Pero sí dos de éstas universidades han hecho convenios con distribuidores de herramientas CASE, los cuales les han proveído de dicha herramienta y les han donado capacitaciones gratis para su personal académico, por lo que una institución universitaria ya incluyó dentro de su pensum una electiva técnica con el nombre Herramientas CASE; se cree que en un futuro cercano la otra universidad incorporará dicha materia a su curricula.

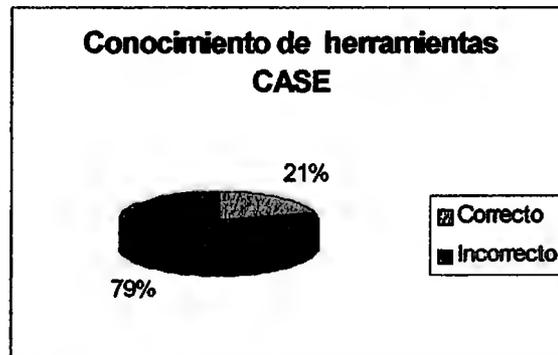
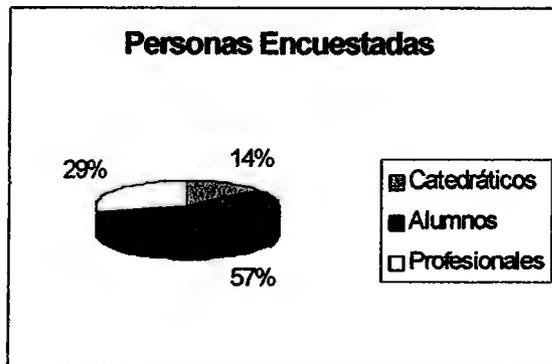
Así mismo sé tuvo la oportunidad de revisar pensum curriculares extranjeros en los cuales se pudo dar cuenta que en dichas universidades si imparten contenidos de éstas herramientas, pero dentro de materias obligatorias de la carrera en Ingeniería en Ciencias de la Computación.

### **3.4 ANALISIS DE ENCUESTAS DIRIGIDAS**

Estas encuestas<sup>1</sup> se realizaron con la finalidad de analizar el conocimiento que los estudiantes, empresarios y catedráticos tienen a cerca de las herramientas CASE, así mismo el uso que los desarrolladores de aplicaciones le están dando a las mismas, cuyas gráficas se presentan en las figuras a continuación.

---

<sup>1</sup> Ver formato en el Anexo A



**Figura A. Gráfica que muestra la clasificación y porcentaje de Personas que participaron en la encuesta.**

**Figura B. Gráfica que presenta el porcentaje de las Personas que poseen un concepto correcto sobre las herramientas CASE**



**Figura C. Conocimiento correcto de los sectores encuestados sobre lo que son las herramientas CASE.**

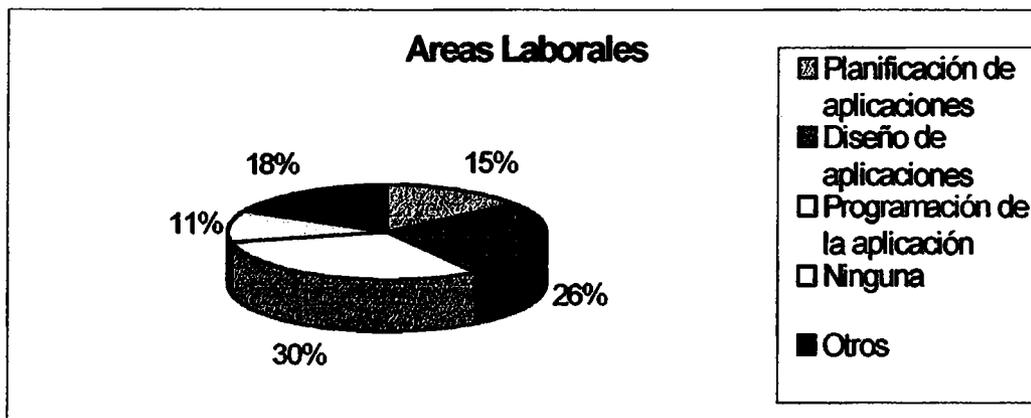
En las gráficas anteriores se presentan los datos obtenidos de las encuestas realizadas; la gráfica A muestra la clasificación de los diversos sectores que participaron en dicha encuesta, entre los cuales el 57% fueron estudiantes universitarios, el 14% fueron catedráticos y el resto profesionales en el ámbito informático. La gráfica B muestra el porcentaje total del conocimiento sobre el concepto correcto de lo que son las herramientas CASE, como se puede observar existe un desconocimiento por parte de las personas involucradas en el estudio del 79%. La gráfica C muestra cuantos de los encuestados poseen un conocimiento acertado sobre dichas herramientas pero detallando alumnos,

catedráticos y profesionales, dentro de lo que se puede recalcar que un 76% de los alumnos no conocen lo que son éste tipo de herramientas y solo un 23% poseen un conocimiento correcto de CASE, así mismo un 66% de los profesionales no conocen los beneficios que este tipo de herramientas les pueden proporcionar y un dato muy importante es con respecto a los catedráticos ya que solo un 33% de éstos saben el concepto de lo que son y para que sirven las herramientas CASE.



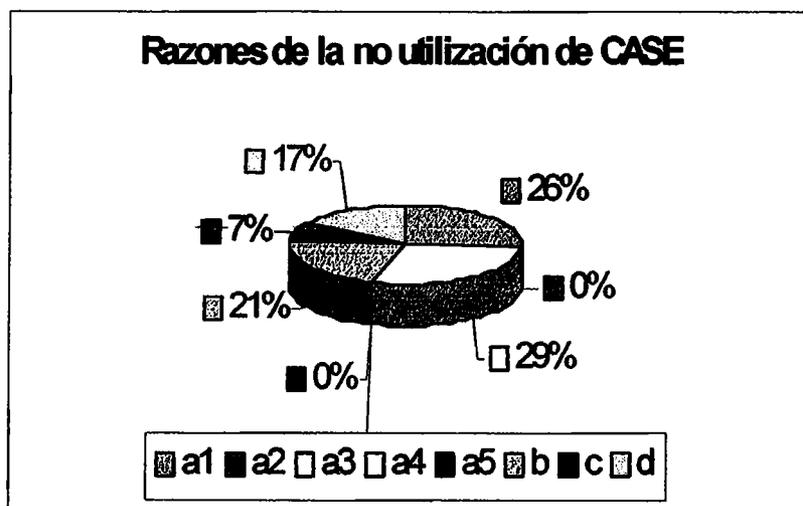
**Figura D. Gráfica que presenta quienes utilizan las herramientas CASE**

La gráfica anterior muestra que solo un 46% de los alumnos, 20% de los catedráticos y un 50% de los profesionales están utilizando las herramientas CASE y el resto no las utiliza, más adelante se explican los diversos motivos por lo cuales no se auxilian de dichas herramientas.



**Fig. D. Gráfico que muestra las áreas de desempeño laboral.**

En la gráfica anterior se establecen las diversas áreas en la que laboran las personas encuestadas, pudiéndose observar que un 30% se encuentran desempeñándose en la programación de aplicación, seguidamente un 26% labora en el diseño de aplicaciones, las cuales son áreas que pueden ser auxiliadas por las herramientas CASE.



**Figura F. Gráfica que muestra el porque no se utilizan las herramientas case**

- a1: Nunca le han mencionado este concepto.
- a2: Las ha escuchado pero no le interesa conocerlas.
- a3: Falta de tiempo para estudiarlas
- a4: Lleva alguna materia en la cual la incluyen pero todavía no la ha cursado.
- a5: Otros motivos
- b: Falta del recurso económico
- c: Las conoce pero no las puede utilizarlas
- d: Otras Razones:
  - No les dan herramienta sin capacitación
  - No cuentan con el equipo
  - No conozco ninguna aplicación

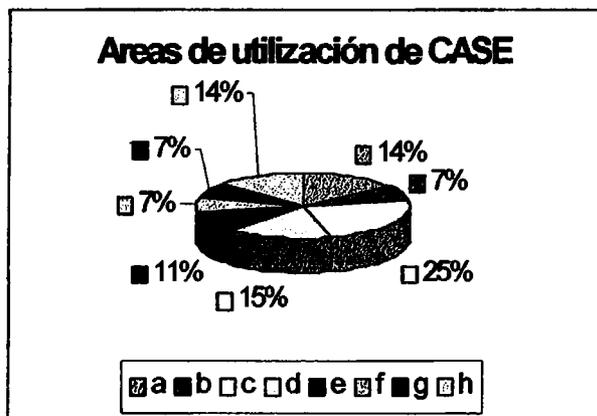
Como podemos observar la figura F, el 29% de la población incluida en el estudio corresponde a la falta de recurso económico, además con muy poca diferencia se encuentra un 26% que corresponde a las personas que dicen que nunca les han mencionado este concepto.



- a) ER-Win
- b) System Architect
- c) Case Designer
- d) Case Project
- e) Designer 2000
- i) VAW
- j) Excelerator
- k) Open Snap

Figura G Gráfica que indica las herramientas más conocidas en el país

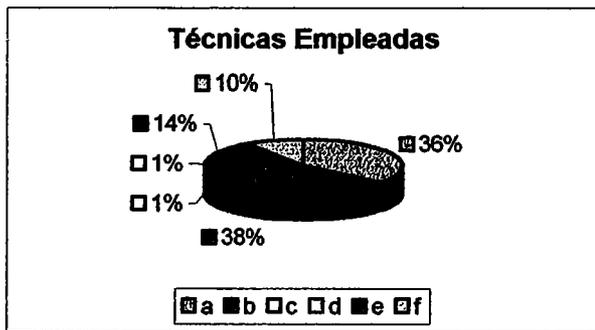
Como podemos observar según la gráfica anterior las herramienta más utilizada por parte de las personas encuestadas son con un 29% la VAW, seguidamente se encuentra la herramienta E-R Win con un 19%.



- a) Diagramación
- b) Planificación de sistemas y admón. De proyectos
- c) Análisis y diseño
- d) Generadores de código
- e) Programación
- f) Generadores de Interfaces
- g) Simulación y creación de prototipos
- h) Documentación.

Figura H. Gráfica que muestra las áreas en las que utilizan las diversas herramientas CASE

La figura H presenta las áreas en las que se están auxiliando por las herramientas CASE, observando que el 25% se encuentra en el análisis y diseño de sistemas, seguidamente un 15% se auxilia con los generadores de código y con un 14% se encuentra la documentación y diagramación. Observando que existe una subutilización de dichas herramientas ya que todos las utilizan de manera aislada y no las integran, pudiendo así obtener mejores resultados.



- a) DFD
- b) E/R
- c) NIAM
- d) Jackson
- e) Diagramas de estructuras
- f) Matrices

Figura I. Gráfica que presenta las diversas técnicas empleadas en el desarrollo de aplicaciones

En la gráfica anterior se observa que un 38% de los desarrolladores utilizan el Modelo Entidad Relación al momento de realizar sus aplicaciones y un 36% los DFD, siendo las técnicas más empleadas por los analistas.

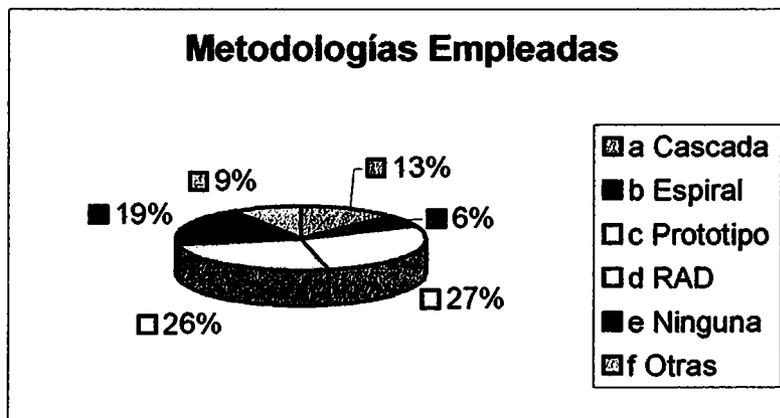
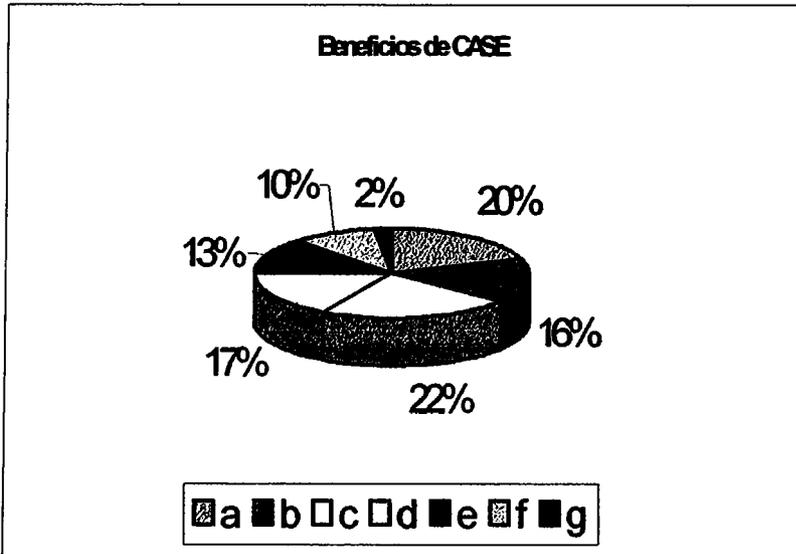


Figura J. Gráfica que presenta las metodología que utilizan los desarrolladores de aplicaciones

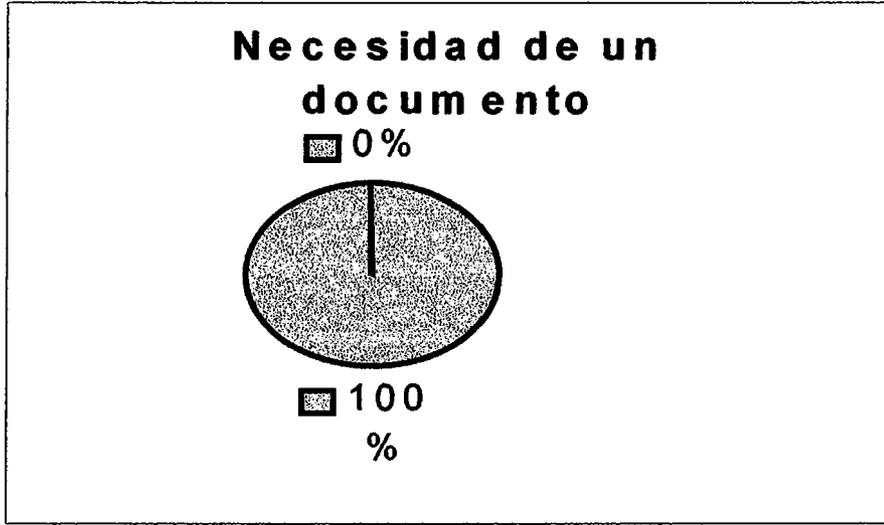
Con la gráfica anterior podemos observar que la metodología más utilizada es la de Prototipo (con el 27%) y seguidamente con un 26% se encuentra la Metodología RAD(Desarrollo Rápido de Aplicaciones).



- a) Mejora productividad del análisis y usuario final
- b) Mejora la eficiencia
- c) Mejora la calidad
- d) Aumenta la velocidad y disminuye el tiempo para contemplar tareas para completar tareas de desarrollo.
- e) Consistencia de procedimiento
- f) Captura almacenamiento, procesamiento y recuperación de detalles de sistema

**Figura K. Beneficios más notables al momento de utilizar herramientas CASE**

De acuerdo a las diversas opiniones de los encuestados se llegó a determinar que uno de los beneficios más notables al momento de utilizar este tipo de herramientas es mejorar la calidad y productividad del Sistema de Información, como también aumentar la velocidad y disminuir el tiempo necesario para completar las tareas de desarrollo, éstos puntos son muy importantes ya que en las gráficas anteriores se pudo observar que los desarrolladores de aplicaciones tienen un concepto errado de lo que son las herramientas CASE, debido a esto las utilizan de una mala manera pero aun así si logran percibir beneficios de dichas herramientas.



**Figura L. Gráfica que presenta la necesidad de un documento en donde se den a conocer diversas herramientas CASE.**

En base a la muestra se comprobó que el 100% de los encuestados creen que es necesario un documento en el que se recopilen diferentes herramientas CASE, técnicas y metodologías utilizadas en el desarrollo de aplicaciones.

Por medio de los resultados obtenidos podemos concluir lo siguiente:

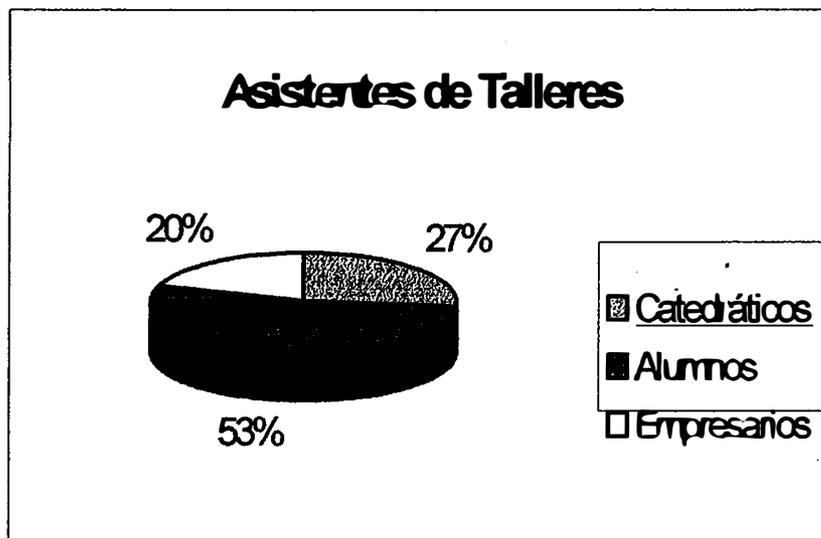
- De acuerdo a los resultados en las encuestas se concluye que existe desconocimiento o un conocimiento a medias sobre el uso de las herramientas CASE.
- La mayoría de desarrolladores de sistemas que utilizan herramientas CASE se quedan hasta la fase de documentación o diagramación, esto se debe a que no tienen el concepto correcto de lo que son dichas herramientas y así mismo se desconocen los beneficios que éstas proporcionan, en el incremento de productividad, calidad de desarrollo y ahorro en sus costos.

### 3.5 ANALISIS DE LA FORMACION DE TALLERES

Para éste análisis se incluye el universo antes mencionados excluyendo a los proveedores de herramientas CASE.

El taller se desarrolló en dos fases. La primera consistió en presentar una hipótesis<sup>1</sup> con el fin de escuchar las opiniones de los participantes y llegar a una conclusión en la cual se dé a conocer si es necesario la incorporación de dichas herramientas en el pensum curricular de la carrera de ingeniería en computación y además cual sería la mejor manera de impartir este conocimiento, en la segunda fase se presentaron tres propuestas de contenidos<sup>2</sup> dentro de los cuales las personas involucradas en el taller debían analizar y seleccionar según su criterio que tópicos tienen más énfasis.

En la siguiente gráfica se presenta la definición de la muestra.



Gráfica M. Muestra la clasificación de los diferentes sectores que participaron en los talleres realizados.

<sup>1</sup> El planteamiento de la hipótesis se encuentra en el Anexo B

<sup>2</sup> Las propuestas de contenidos se encuentran en el Anexo C

Con lo que obtuvimos los siguientes resultados:

### **Análisis de Empresarios**

Se concluye que la mayoría de los empresarios creen en la necesidad de incorporar este tipo de Herramienta dentro de los conocimientos de los futuros profesionales y dando una solución para aquellos que ya terminaron sus estudios universitarios sugirieron que si las empresas necesitan invertir sobre estas herramientas deberán incluir las capacitaciones necesarias a todos aquellos que se encuentren desempeñando laboralmente en el área de desarrollo de aplicaciones.

El 100% de los empresarios asistentes al taller determinó que la mejor forma de incluir estas herramientas dentro de la curricula universitaria debe de ser dentro de contenidos de materias ya existentes entre las cuales se puede mencionar análisis y diseño de sistemas, ya que en esta materia se dan a conocer las bases del ciclo de vida de desarrollo de sistemas, en las cuales las herramientas CASE vendrían a aplicar dichas fases.

### **Análisis de Alumnos**

Dentro de los resultados obtenidos por parte de los alumnos podemos concluir:

Actualmente el conocimiento acerca de las herramientas CASE, expresaron que es mínima, ya que en esos momentos solo un 5% de la muestra tenía un concepto básico de este tipo de herramientas.

El 100% de los alumnos creen que la mejor forma de incorporar este tipo de herramientas es dentro de materias ya existente, como pueden ser análisis y diseño de sistemas e ingeniería del software en un nivel introductorio, ya que en estas materias se dan a conocer las etapas del ciclo de vida de desarrollo de aplicaciones y de que forma se analiza y diseña un sistema.

Además la incorporación de una electiva técnica dedicada a este tipo de herramientas, sería conveniente para aquellos alumnos que deseen especializarse en el área de análisis y diseño de aplicaciones.

### **Análisis de Catedráticos**

Dentro de las opiniones por parte de catedráticos expusieron lo siguiente:

Para ellos este tipo de herramienta se debe incluir como una electiva obligatoria a un nivel de 8° ciclo cuando ya se tengan los conocimientos a cerca de análisis y diseño de sistema, pero la incorporación de las herramientas CASE va ha depender de la demanda que exista por parte de los empresarios ya que son ellos los que se beneficiaran en un futuro, ya que algunos de los docentes se preocupan mas por dar a conocer los beneficios de una paquetería y pierde la visión de lo que realmente están formando.

Dentro de las propuestas que seleccionaron más apropiadas para el conocimiento de los alumnos y que posteriormente les será de ayuda en el desempeño profesional a los ingenieros en computación se encuentran la A y B

### **3.6 ANALISIS DE ENTREVISTAS DIRIGIDAS**

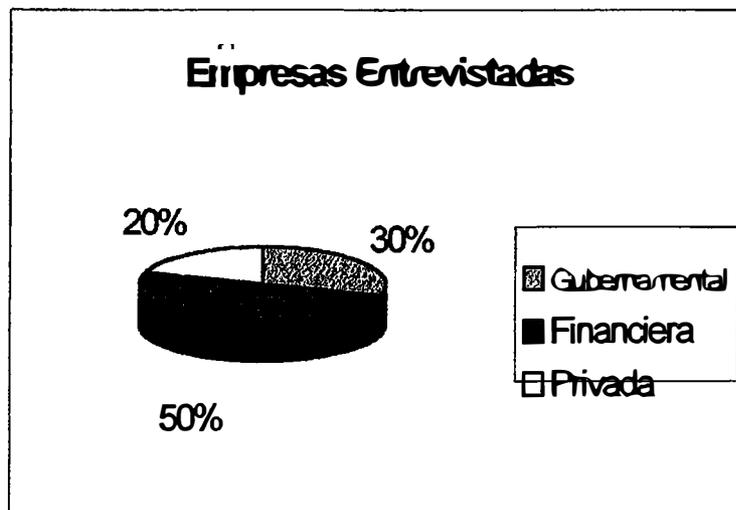
En estas entrevistas<sup>3</sup> estuvieron involucrados los distribuidores de herramientas CASE con el propósito de determinar el mercado local de dichas herramientas, así como también diversos empresarios que se encuentran utilizando estas herramientas a fin de conocer si es conveniente para

<sup>3</sup> Ver formato de entrevista en Anexo D

los futuros profesionales en el ámbito informático incluir este tipo de tecnología dentro de su formación universitaria, pero una de las principales razones de estas entrevistas fue fundamentar las conclusiones presentadas en los talleres.

### **Análisis de las entrevistas realizadas a los empresarios**

Como primer punto se muestra el gráfico de las empresas entrevistadas:



**Figura N. Estas son los diversos sectores que se entrevistaron.**

Dentro de los resultados obtenidos de dichas entrevistas se encuentran:

#### **Pregunta 1:**

### ***¿ Utiliza alguna Herramienta CASE?***



Basándose en los resultados anteriores nos podemos dar cuenta que muchas empresas se encuentran utilizando este tipo de herramientas, dentro de las cuales el 50% pertenecen al sector Gubernamental, un 80% al financiero y un 75% al Privado pero no todas las utilizan adecuadamente<sup>5</sup>.

### ***Pregunta 2***

***¿ Es necesario que a las Universidades impartan conocimientos de lo que son las herramientas CASE?***

Se concluye que la mayoría de los empresarios creen en la necesidad que los profesionales que se contraten tengan ya las bases sobre estas herramientas de manera tal que en su desarrollo laboral se facilite (como se puede observar en la gráfica el 95% manifiesta la necesidad de que los futuros profesionales en el ámbito de informático conozcan sobre lo que son las herramientas CASE).

***Pregunta 3***

***¿De qué forma se podría incorporar a las herramientas dentro del Pensum Curricular?***

Con lo anterior nos damos cuenta que según las expectativas se debe de incluir como una materia Electiva Técnica Obligatoria o también como una materia dentro de contenidos de materia, ya que los empresarios requieren que los futuros profesionales posean conocimiento de las necesidades tecnológicas entre las cuales se encuentran las Herramientas CASE.

## CAPITULO IV

### PROPUESTA DE ESTUDIO DE HERRAMIENTAS CASE

#### 4.1 INTRODUCCION

Las propuestas<sup>1</sup> resultantes del estudio mostrado en el capítulo anterior para cubrir las expectativas de los empleadores, alumnos y finalmente de los catedráticos que poseen la experiencia de enseñanza se presentan en este capítulo. Las cuales beneficiaran a la escuela de Ingeniería en Ciencias de la Computación con un estudio exhaustivo para sustentar la incorporación de las herramientas CASE dentro del Pensum curricular de la carrera.

Para este estudio se analizaron tres propuestas: Las cuales consistían en la incorporación de dichas herramientas como una materia obligatoria, electiva técnica o como dentro de contenidos de materias ya existentes, de las cuales se debía seleccionar una y de esta forma tener fundamentos para dicha incorporación.

La primera propuesta fue descartada por parte de las personas involucradas en el estudio, porque la tecnología esta cambiando constantemente y van surgiendo nuevas formas de hacer las cosas, siempre buscando métodos más efectivos, confiables con mayor calidad y menos riesgos por lo que creen que no es conveniente incorporarla como materia obligatoria ya que es una herramienta que proyectándose dentro de 10 años puede no ser indispensable para la formación del profesional en informática y esto llevaría al constante cambio del pensum curriculares de la carrera.

<sup>1</sup> Ver planteamiento de hipótesis en Anexo B

La segunda y tercera propuesta se diferencian en que una servirá de especialización para todos aquellos estudiantes que deseen desempeñarse en el análisis y desarrollo de aplicaciones, siendo ésta de gran ayuda para todos aquellos profesionales que tienen una visión al desarrollo de aplicaciones. Y la segunda pretende dar a conocer los conceptos básicos de las herramientas CASE, para evitar el desconocimiento de dichas herramientas.

Finalmente se presenta una última propuesta para todos aquellos profesionales de Informática que no poseen conocimientos sobre dichas herramientas y desean actualizar sus estudios con las tecnologías de punta.

#### **4.2 PROPUESTA A: TECNICAS Y METODOLOGIAS APLICADAS A LAS HERRAMIENTA CASE. UNA ELECTIVA TECNICA**

Partiendo del criterio que una materia electiva técnica debe profundizarse una área específica del conocimiento de la carrera, ayudando a precisar mejor una especialización. Se recomienda que las herramientas CASE podrían ser incorporadas dentro del pensum curricular de la carrera de Ingeniería en Ciencias de la Computación como una electiva técnica a nivel de 8° ciclo, pero previamente es necesario tener un conocimiento de programación orientada a objetos y análisis y diseño de sistemas las cuales deben ser requisitos de dicha materia, ya que con estas materias los estudiantes tendrán un conocimiento apropiado sobre las fases del ciclo de vida de desarrollo de sistemas, además estas herramientas hacen uso una programación orientada a objetos; inicialmente al estudiante se le podría dar a conocer el concepto básico sobre estas herramientas y así éste pueda percibir qué son y como le beneficiarán en el futuro, con el propósito de despertar el interés en el momento de cursar dicha electiva.

Analizando los contenidos<sup>2</sup> sugeridos y con las opiniones de los empresarios, alumnos y docentes que participaron en entrevistas dirigidas y talleres se hizo un consenso para la temática que se detalla a continuación, se cree que es el más conveniente, el cual muestra el objetivo siguiente:

### **Objetivo General:**

Comprender el estudio de la Ingeniería del Software asistida por computadoras, analizando sus técnicas, metodologías, componentes, aplicaciones y uso, con el propósito de conocer las características y beneficios que las herramientas CASE proporcionan.

### **Objetivos Específicos:**

- Conocer el concepto, objetivos y beneficios de las Herramientas CASE.
- Conocer las diversas técnicas y metodologías aplicadas a las herramientas CASE
- Distinguir la clasificación de herramientas CASE, como también las características y requerimientos.
- Desarrollar aplicaciones utilizando una herramienta CASE.

## **CONTENIDO**

### ***1. CONCEPTOS GENERALES***

#### *1.1 Conceptos Básicos de Herramienta y Herramienta CASE*

##### *1.1.1 Beneficios del empleo de herramientas*

##### *1.1.2 Objetivos de las herramientas CASE.*

##### *1.1.3 Beneficios del las Herramientas CASE*

<sup>2</sup>Ver propuestas de contenidos en anexo C

## *1.2 Clasificación de las herramientas CASE*

*1.2.1 Front End. Se utiliza Planificación y Definición de Requerimientos, Etapa de Análisis y Diseño, Modelo y generación de Prototipos.*

*1.2.2 Back End. En compiladores y Generadores de Código.*

*1.2.3 Integrales. Conjunto de herramientas integradas que interactúan unas con otras en forma consistente.*

## *1.3 Descripción General de los distintos tipos de herramientas*

*1.3.1 Nombre de la herramienta*

*1.3.2 Tipo de Clasificación*

*1.3.3 Breve Descripción*

*1.3.4 Características*

*1.3.5 Beneficios*

*1.3.6 Requerimientos*

## **2. APLICACIONES Y USOS DE LAS HERRAMIENTAS CASE**

*2.1 Herramientas de Análisis y Diseño*

*2.2 Herramientas de Programación*

*2.3 Herramientas de para Diagramación*

*2.4 Herramientas de Creación de Prototipo*

*2.5 Herramientas de Soporte*

*2.6 Herramientas de Administración*

*2.7 Herramientas de planificación de sistemas de administración de proyecto*

## **3. TECNICAS Y METODOLOGIAS APLICADAS A LAS HERRAMIENTA CASE**

#### **4. EVALUACION DE UNA HERRAMIENTA CASE**

*4.1 Selección de herramientas CASE adecuadas*

*4.2 Implementación de una herramienta CASE*

*4.3 Problemas y Soluciones del Software*

*4.4 Establecimiento de Necesidades*

*4.5 Adaptación de la tecnología de CASE*

*4.6 Auditoria del entorno CASE*

#### **5. HERRAMIENTAS C.A.S.E A UTILIZAR**

*5.1 Introducción*

*5.2 Componentes*

*5.3 Herramientas y Métodos*

*5.4 Generadores, Administración de Repositorios*

#### **6. PRACTICA DE ESTUDIO**

*Se sugiere que la práctica de esta asignatura sea con la utilización de una herramienta CASE*

#### **BIBLIOGRAFIA**

1. Alfaro Burgos, Guadalupe, "*Desarrollo de aplicaciones utilizando metodología auxiliada por herramientas case*", Universidad Don Bosco, mayo de 1996.
2. Cuevas Gonzalo "*Ingeniería del Software*" Iberoamericana 1995
3. Cruz, Dueñas, Duran, Hernández; "*Desarrollo de herramientas de análisis y diseño aplicadas a El Salvador*", UCA; 1992.
4. Molina, Navas, Zepeda; "*Introducción al desarrollo de sistemas de información utilizando herramientas C.A.S.E., un caso de aplicación*"., UCA, Septiembre 1993.

5. Piattini Mario, *"Elementos y Herramientas en el desarrollo de sistemas de información, una visión actual de la tecnología CASE"* Wesley Iberoamericana 1997
6. Senn James A., *"Análisis y Diseño de Sistemas de Información"* México 1992.

#### **4.3. PROPUESTA B: COMO CONTENIDOS DE MATERIAS YA EXISTENTES**

Las herramientas CASE deberían ser incorporadas dentro del pensum curricular de la carrera de ingeniería en ciencias de la computación como contenidos de materias obligatorias, ya existentes dentro de las cuales se puede incluir en ingeniería del software o análisis y diseño de sistemas, ya que en estas materias se dan a conocer las fases del ciclo de vida de un sistema en las cuales se podría aplicar dichas herramientas, proporcionándole al estudiante una visión amplia sobre la lógica que utilizan estas herramientas y metodologías que se aplican debido ya que le facilitan al analista el desarrollo de sistemas.

Con estos tópicos se proporcionarán los conceptos fundamentales de este tipo de tecnología evitándole el desconocimiento de dichas herramientas a los futuros profesionales, dándole a conocer que una herramienta CASE consiste en un conjunto de herramientas operando en una máquina con una plataforma común donde el ingeniero de sistemas puede integrar toda la información y los procesos de una sistema con el fin de lograr la creación del mismo, como también conocer las técnicas y metodológicas apropiadas para el análisis y diseño de sistemas.

Por lo que analizando los contenido sugerido para dicha propuesta fue seleccionado por los empresarios, alumnos y docentes que participaron en entrevistas dirigidas y talleres el cual se detalla a continuación:

## **Objetivos Específicos:**

- Conocer el concepto, objetivos y beneficios de las Herramientas CASE.
- Conocer las diversas técnicas y metodologías aplicadas a las herramientas CASE
- Distinguir la clasificación de herramientas CASE.

## **CONTENIDO**

### **1. CONCEPTOS GENERALES**

#### *1.1. Conceptos Básicos de Herramienta y Herramienta CASE*

##### *1.1.1. Beneficios del empleo de herramientas*

##### *1.1.2. Objetivos de las herramientas CASE*

##### *1.1.3 Beneficios del las Herramientas CASE*

#### *1.2. Clasificación de las herramientas CASE*

*1.2.1. Front End. Se utiliza Planificación y Definición de Requerimientos, Etapa de Análisis y Diseño, Modelo y generación de Prototipos.*

*1.2.2. Back End. En compiladores y Generadores de Código.*

*1.2.3. Integrales. Conjunto de herramientas integradas que interactúan unas con otras en forma consistente*

#### **1.3. TECNICAS Y METODOLOGIAS APLICADAS A LAS HERRAMIENTA CASE**

## **BIBLIOGRAFIA**

1. Alfaro Burgos, Guadalupe, "*Desarrollo de aplicaciones utilizando metodología auxiliada por herramientas case*", Universidad Don Bosco, mayo de 1996.

2. Cruz, Dueñas, Duran, Hernández; *"Desarrollo de herramientas de análisis y diseño aplicadas a El Salvador"*, UCA; 1992.
3. Molina, Navas, Zepeda; *"Introducción al desarrollo de sistemas de información utilizando herramientas C.A.S.E., un caso de aplicación"*., UCA, Septiembre 1993.
4. Piattini Mario, *"Elementos y Herramientas en el desarrollo de sistemas de información, una visión actual de la tecnología CASE"* Wesley Iberoamericana 1997
5. Senn James A., *"Análisis y Diseño de Sistemas de Información"* México 1992.

#### **4.4. PROPUESTA PARA PROFESIONALES QUE DESEEN ADQUIRIR CONOCIMIENTOS SOBRE HERRAMIENTAS CASE.**

Como solución al problema sobre el desconocimiento de herramientas CASE se propone que aquellos profesionales que no tienen conocimiento sobre esa tecnología estudien un diplomado sobre dichas herramientas, el cual puede ser impartido en diversas universidades nacionales.

Para el desarrollo de esta propuesta se analizó la mejor forma de impartir las bases sobre estas herramientas, entre las alternativas evaluados están: un diplomado o una maestría.

La segunda alternativa fue descartada debido a que las herramientas CASE no requieren de mucho tiempo para conocer sobre sus conocimientos fundamentales y el estudio de una maestría se basa en la especialización de la carrera por lo que no amerita que el profesional invierta en una maestría a cerca de herramientas CASE.

Siendo la primera alternativa la más indicada ya que es de gran ayuda para todos aquellos profesionales que tienen una visión al desarrollo de aplicaciones estudiar un diplomado sobre

herramientas CASE dando a conocer sus fundamentos y beneficios en el desarrollo de aplicaciones evitando así el desconocimiento de dichas herramientas y ayudándoles a facilitarles las tareas de análisis, diseño y codificación.

## CAPITULO V

### CONCEPTOS BASICOS DE HERRAMIENTAS CASE

#### 5.1 INTRODUCCION

La Ingeniería del Software comprende muchas actividades cuyo objetivo es proporcionar sistemas más confiables y menos costosos. El término Ingeniería de software fue utilizado por primera vez por Fritz Bauer en la conferencia de la NATO de 1968, siendo posteriormente impulsada por el D.O.D (Departamento de Defensa) de los E.E.U.U., es interdisciplina: usa matemáticas, para analizar y certificar algoritmos, ingeniería para estimar costes y definir lo más conveniente de varios factores, ciencia de la dirección para definir requerimientos, fijar riesgos, supervisar personal y controlar el progreso.

Es una disciplina para el desarrollo del software que combina métodos y procedimientos para realizar cada una de las etapas en las que se divide este desarrollo, establece las técnicas mas adecuadas para realizar la construcción y determinar las herramientas a utilizar para llevar a cabo la realización de las actividades.

Además la Ingeniería del Software es el tratamiento y uso de determinadas técnicas modelos, productos o herramientas destinados a facilitar el desarrollo de sistemas, bajo el nombre de herramientas CASE (Computer-Aided Software Engineering) y cuyo ciclo de vida es la elaboración de sistemas atacando en primer lugar la generación de programas.

## **5.2 HERRAMIENTAS**

Cualquier dispositivo que cuando se emplea en forma apropiada, mejora la realización de una tarea. Las herramientas son esenciales para el análisis de sistemas. Ellas mejoran la forma en que ocurre el desarrollo y tienen influencia sobre la calidad del resultado final.

### **5.2.1 Beneficios del empleo de herramientas.**

Las herramientas extienden en tres formas la capacidad del analista de sistemas: proporcionan el potencial para mejorar la productividad del analista, facilitan el desarrollo de procesos más eficaces y mejoran la calidad del sistema.

En otras palabras, tanto el proceso de desarrollo de sistemas como el producto que se obtienen con él, pueden mejorarse con el uso de herramientas apropiadas. Las cuales se detallan a continuación.

#### ***5.2.1.1 Mejora en la productividad.***

En algunos casos, las herramientas correctas contribuyen a alcanzar un nivel de productividad que hace factible una tarea que de otro modo no sería posible realizar.

Las herramientas aumentan la productividad del analista al disminuir la cantidad de tiempo necesaria para documentar, analizar y desarrollar sistemas de información. Cuando se utilizan adecuadamente, aumentan la eficiencia del analista.

#### ***5.2.1.2 Mejora en la eficacia.***

Las herramientas sugieren procedimientos que conducen al empleo de procesos más eficientes. Si la productividad significa realizar la tarea correcta, la eficiencia significa hacer esta tarea en forma correcta. Las herramientas pueden sugerir la mejor forma para abordar una tarea.

En el campo del analista de sistemas, tener las herramientas correctas significa sugerir formas más eficientes para realizar tareas. La disponibilidad de estas herramientas para el flujo de datos, estimula al analista a poner mayor hincapié, antes de iniciar el desarrollo del sistema, sobre la determinación de los requerimientos de sistemas. Las decisiones eficientes con respecto a la herramienta ahorran recursos: personal, tiempo y dinero.

### ***5.2.1.3 Mejoran en la calidad del sistema de información.***

Cuando las herramientas mejoran los procesos, por lo general también ocurre lo mismo con los resultados. Los usuarios de los sistemas de información desean calidad en el sistema con un tiempo razonable. Hace algún tiempo no existían muchas herramientas, debido a esto no era posible el desarrollo de prototipos de aplicación ni tampoco el análisis estructurado.

La invención de los lenguajes de cuarta generación y de diagramas de flujo de datos, dos herramientas esenciales para realizar respectivamente estas tareas, cambiaron en las organizaciones los procedimientos para analizar sistemas.

## **5.3 HERRAMIENTAS CASE**

Desde que, a finales de los años sesenta, se acuñó el término crisis del software, numerosos expertos han venido ocupándose del tema, describiendo sus síntomas y causas (como por ejemplo, BROOKS (1975)) y proponiendo distintas técnicas, metodologías y herramientas para paliar esta situación.

Entre todas ellas, destacan las conocidas bajo el nombre de CASE (COMPUTER Aided/Assisted Software/System Engineering) que, en su acepción más amplia, se puede definir como las herramientas y metodologías que soportan un enfoque de ingeniería en el desarrollo de software para todas las fases de este proceso.

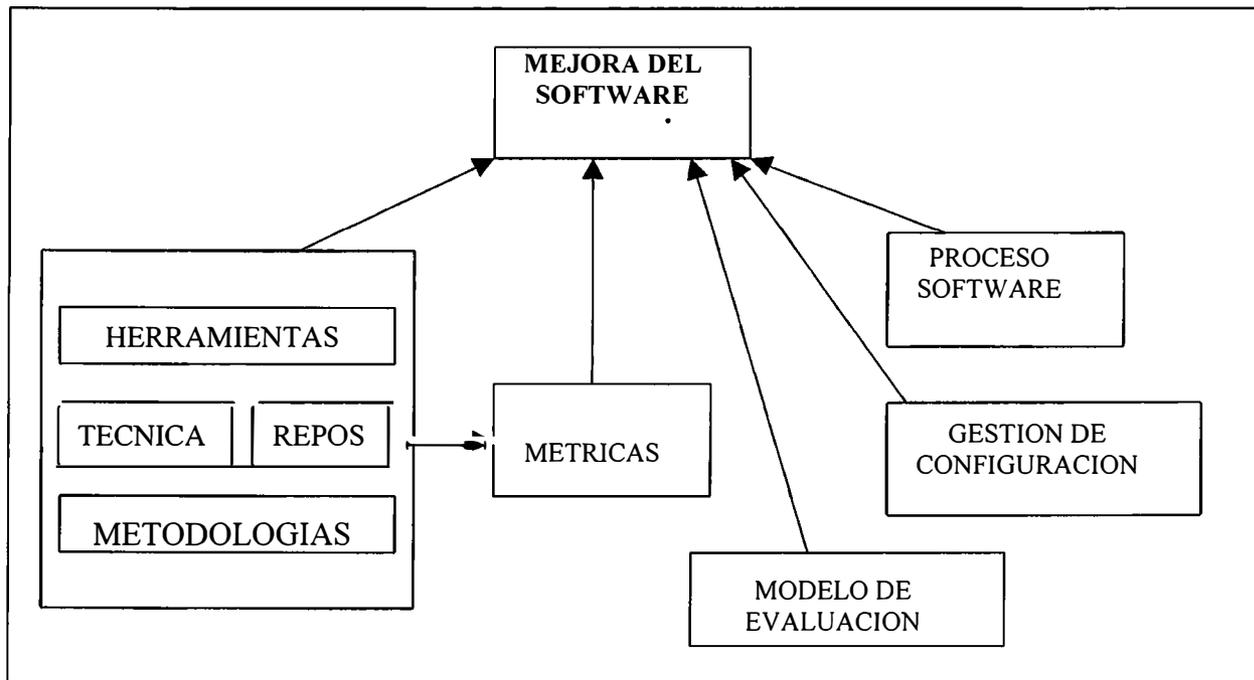
La automatización aplicada a diversas áreas, como, por ejemplo, CAD/CAM, ha producido a un aumento de la productividad; de forma análoga, las herramientas CASE contribuyen a elevar la productividad y la calidad en el desarrollo de sistemas de información.

Las herramientas CASE son un elemento muy importante en la mejor del software, pero no son suficientes si no van acompañadas de las técnicas y metodologías adecuadas, no se implantan de la forma correcta, o si no se completan por otros aspectos relativos a la calidad.

En la actualidad, el entorno CASE se ha integrado con otros aspectos que también persiguen la mejora del software, como son las métricas, el modelo de proceso software, modelos para evaluar la capacidad de desarrollo y calidad (CMM o ISO 9000) o la gestión de configuración, tal y como se muestra en esta figura 1.

La tecnología CASE corresponde a la Ingeniería del Software asistido por ordenador. Es un conjunto de herramientas software para automatizar las tareas del desarrollo de software. La idea es proporcionar un conjunto integrado de herramientas que enlazan y automatizan todas las fases del ciclo de vida del software y su administración.

La tecnología CASE es una combinación de herramientas software y metodología. Enfoca sobre el problema de producción de software.



**Figura 1. Interrelaciones entre diversos aspectos en el desarrollo de SI inspirada en TATE et al. (1992).**

Entre los objetivos de la tecnología CASE tenemos:

- Aumentar la productividad en el desarrollo
- Dar calidad a los productos desarrollados
- Reducir el coste del software
- Automatizar los chequeos de errores.
- Acelerar el desarrollo de las aplicaciones
- Automatizar tareas de desarrollo
- Automatizar la generación de documentación y código
- Dar portabilidad al software
- Implantar metodologías de desarrollo
- Datos reutilizables y compartidos
- Administrar el proyecto

- Ingeniería hacia atrás (reverse).

CASE, "Computer-Aided Software Engineering", tal como lo indica sus siglas, son herramientas en "software" que se han creado para ayudar y asistir al personal de computadoras al momento de desarrollar sistemas de información. Permite el uso de asistencia computadorizada en todo el proceso de definición y desarrollo del ciclo de vida de una aplicación.

Consiste en la combinación de gráficas o diagramas, diccionarios o repositorios, generadores y compiladores, manejadores de proyectos y otras herramientas que le dan asistencia a la mayoría de las actividades que lleva a cabo un ingeniero de "software", como lo son: la planificación, el manejo de proyectos, la definición y especificación de requerimientos a tomar en consideración, el análisis y diseño de un sistema, la documentación, la creación de prototipos de un sistema, la codificación y la corrección de errores en programación ó depuradores.

Un ambiente CASE consiste en un conjunto de herramientas CASE operando en una máquina con una plataforma común donde el ingeniero de sistemas puede integrar toda la información y los procesos de un sistema con el fin de lograr la creación del mismo.

Los objetivos principales esta metodología son: crear sistemas que cumplan con las necesidades y los requisitos de los usuarios, evitar errores en los sistemas, obtener un sistema de calidad y precisión, completar la creación del sistema en un tiempo razonable, lograr que las técnicas estructuradas sean prácticas al desarrollador y aumentar la productividad del ingeniero de información.

### **5.3.1 Beneficios de las herramientas asistidas por computadora.**

La automatización mejora los beneficios que se pueden obtener con el empleo de herramientas. Con ella disminuye el tiempo necesario para llevar a cabo las tareas, se reduce la intensidad del trabajo, y

el seguimiento de todos los procedimientos se lleva a cabo de manera consistente; también se capturan los datos que describen el sistema para tenerlos almacenados en un formato que pueda leer una computadora. Estas se describen a continuación:

#### ***5.3.1.1 Disminución de tiempo***

La introducción de herramientas asistidas por computadora en los esfuerzos de análisis y desarrollo aumenta los beneficios que se derivan del uso de herramientas. Las herramientas de análisis asistido por computadora mejoran la velocidad y disminuyen el tiempo necesario para completar la tarea de desarrollo. Tanto el análisis como las actividades de desarrollo se llevan a cabo en un tiempo menor. Por tanto, resulta claro que para obtener resultados aceptables es esencial que el analista esté entrenado en el uso de las herramientas.

#### ***5.3.1.2 Automatización de tareas tediosas***

La automatización también se hace cargo de algunas tareas que son pesadas. El desarrollo de diagramas de flujo de datos, parte esencial del método de análisis estructurado, es una tarea que puede consumir mucho tiempo. Las herramientas automatizadas para el flujo de datos, hacen posible dejar al software de la computadora el proceso de dibujo.

#### ***5.3.1.3 Garantizar la consistencia de los procedimientos***

Cuando los procedimientos forman parte del software, éstos se realizan en forma más consistente. Se convierten en rutinas. La consistencia que pueden ofrecer los procedimientos es una excelente razón para ampliar el conjunto de herramientas asistidas por computadora para el desarrollo de sistemas. Por ejemplo la tarea de examinar diagramas de flujo de datos para determinar si son o no consistentes y completos. La automatización de este proceso garantiza que, cada vez que sea necesario, las evaluaciones se efectúen en forma consistente.

#### 5.3.1.4 *Captura de los datos del sistema*

Una ventaja que distingue a muchos sistemas automatizados es la captura, almacenamiento, procesamiento y recuperación de los detalles de un sistema. Una vez en forma procesable por la computadora, los detalles del sistema pueden utilizarse para muchas finalidades.

### 5.4 CLASIFICACIÓN DE HERRAMIENTAS AUTOMATIZADAS

Como sucede en otras áreas de la informática, la tecnología CASE emplea una terminología que puede resultar a veces confusa. Existen numerosas clasificaciones de las herramientas CASE; citaremos a nuestro juicio, las más importantes.

En primer lugar, existe una división clásica atendiendo a la fase del ciclo de vida que soportan:

- **CASE Frontales** (front-end) o **Superiores** (Upper CASE), abarcan las primeras fases de análisis y diseño.
  - Herramientas de planificación y definición de requisitos.
  - Herramientas de análisis y diseño
  - Herramientas modelización y generación de prototipos
- **CASE Dorsales** (back-end) o **Inferiores** (Lower CASE), cuyo objetivo suele ser el diseño detallado y la generación de código.
  - Compiladores
  - Generadores de estructura y código
- **Gestión**
  - Gestión de proyectos (dirección proyecto)
  - Gestión de proyectos o ciclo de vida
  - Gestión requisitos
  - Gestión cambios y configuraciones

Se denomina **ICASE** (Integrated CASE) a las herramientas que engloban ambos aspectos, e **IPSE** (Integrated Programming Support Environment) a aquellas que, además, incluyen componentes para la gestión de proyectos y la gestión de configuración.

Otra clasificación bastante conocida es la que se establece en McCLURE (1992), que distingue tres categorías de herramientas CASE.

- **Juegos de herramientas** (toolkit), son el tipo más simple, y están formados por un conjunto de herramientas que automatizan un tipo de tarea del ciclo de vida, como el análisis. Dentro de éstas, se incluyen los **armazones** (frameworks), que sirven para proporcionar una infraestructura en la cual acoplar y adaptar herramientas individuales.
- **Bancos de trabajo** (workbenchs), conjuntos de herramientas integradas que abarcan las fases básicas del ciclo de vida: análisis, diseño e implementación.
  - Conforme a un conjunto de estándares
  - Ante el usuario conforman un bloque único sin funciones o mensajes duplicados.

En lo que respecta a las capacidades funcionales básicas el workbench proporciona asistencia mediante ordenador para el desarrollo, mantenimiento y administración de sistemas software de forma integrada.

Debe tener las siguientes características:

- Interfaz gráfico para dibujar diagramas estructurado.
- Un repositorio (enciclopedia) para almacenar y administrar toda la información del sistema software.
- Conjunto integrado de herramientas que comparten una interfaz de usuario común.
- Herramientas para asistir en cada fase del ciclo de vida.
- Herramientas para prototipo.

- Herramientas para administración del proyecto.
- Generación automática de código desde las especificaciones de diseño.
- Soporte metodología ciclo vida del software con verificación incorporada en la herramienta.
- **Compañeros de metodologías** (methodology companions), que pueden ser cualquiera de los anteriores, siempre que soporten todas las fases y reglas de una metodología en particular.

Un conocido experto en el área de la ingeniería del software, el profesor Sommerville, distingue, por su parte en SOMMERVILLE(1992), entre:

- **Entornos de programación**, pensados para soportar la programación, prueba y depuración de sistemas.
- **Bancos de herramientas CASE** orientados principalmente al análisis y diseño.
- **Entornos de ingeniería de software**, cuyo objetivo principal es la producción de grandes sistemas software, y que soportan todas las actividades de desarrollo y mantenimiento.

Por último, cabe destacar una clasificación bastante reciente, que se puede encontrar en FUGGETTA ( 1993), donde se distinguen entre:

- **Herramientas**, que soportan una tarea específica del proceso de producción de software, como la edición, programación, verificación y validación, gestión de configuración, métricas o gestión de proyectos.
- **Bancos de herramientas**, que integran varias herramientas y que soportan: planificación y modelados estratégico, desarrollo de la interfaz de usuario, programación, verificación y validación, mantenimiento e ingeniería inversa, gestión de configuración o gestión de proyectos.

- **Entornos**, que se caracterizan por soportar los procesos software, y que este autor subdivide en:
  - **Juegos de herramientas**, conjunto de productos débilmente integrados.
  - **Entornos integrados**, que utilizan mecanismos estándar para integrar sus componentes.
  - **Entornos de cuarta generación**, que soportan una clase específica de programas, como el procesamiento de datos
  - **Entornos centrados en el proceso**, basados en la definición formal del proceso software y que se componen de dos partes: la destinada a la ejecución del modelo de proceso y la utilizada para producir el modelo de proceso. Dentro de esta categoría se incluirían los **generadores de entorno** y las **metaherramientas CASE**, capaces de generar entornos de desarrollo siguiendo los procedimientos y políticas descritos por el usuario en el modelo de proceso.

Existen otras muchas clasificaciones, como las de PRESSMAN (1992) y SHARON (1993), donde existe un apartado especial dedicado a los repositorios y diccionarios, o PERRY y KAISER (1991), en la que las distintas categorías de herramientas CASE se comparan con los cuadros de diferentes entornos, compuestos por el individuo, la familia, la ciudad y el estado, analizando así los problemas que se pueden dar a esas cuatro escalas.

Además de la disparidad de criterios a la hora de clasificar las herramientas, hay que destacar que las fronteras entre las distintas categorías no son tan nítidas, pudiendo un mismo producto catalogarse en distintos apartados.

Algunas herramientas están vinculadas con metodologías específicas de desarrollo como por ejemplo, análisis estructurado. Otras soportan solo lenguajes específicos como COBOL, o a determinado fabricante de hardware como IBM o Digital. De acuerdo con las necesidades de la

organización estas características tal vez limitan la utilidad de ciertas herramientas. A menudo la clave se encuentra en la base de datos central.

## **5.5 DESCRIPCION GENERAL DE DISTINTOS TIPOS DE HERRAMIENTAS**

En este apartado se muestran algunas de las herramientas CASE que se están utilizando actualmente como también un cuadro comparativo de las mismas, todo esto aparece en el capítulo 6 con un breve resumen y de manera extensa en el capítulo VIII.

## **5.6 APLICACIONES Y USOS DE LAS HERRAMIENTAS CASE**

En general, las herramientas CASE incluyen los siguientes componentes: herramienta de análisis y diseño, herramientas de programación, herramientas de creación de prototipos, herramientas de soporte, herramienta de planificación y herramientas de gestión, herramientas para diagramación, Deposito centralizado de información, Generador de interfaces, Generadores de código y Herramientas de administración, los cuales se describen a continuación:

### **5.6.1 Herramientas de Análisis y diseño**

Estas herramientas permiten al ingeniero del software crear un modulo del sistema a construir. Ayuda en la creación del modelo y también en la evaluación de la calidad de este, las herramientas de análisis y diseño además estas proporcionan al el conocimiento y soporte necesario para eliminar los errores antes de que se vuelvan complejos.

Dentro de las herramientas de análisis y diseño podemos mencionar:

- Herramientas de AE/DE(Análisis / Diseño estructurado) El AE/DE combina una estación de trabajo específica, heurística de análisis y diseño de transformaciones para producir representaciones realizadas del software.

- Herramientas PRO/SIM(Creación de prototipo y de simulación). Estas herramientas proporcionan la capacidad de predecir el comportamiento de una sistema de tiempo real antes que sea construido. Además permiten desarrollar prototipos de sistemas de tiempo real proporcionando al cliente una visión general de la función.
- Herramientas para el diseño y desarrollo de interfaces. Son un conjunto de componentes de software tales como menús, botones , estructuras de ventanas, iconos, mecanismos de visualización controladores de dispositivos y otros elementos de este tipo.

### **5.6.2 Herramientas de programación**

En esta categoría de programación se encuentran los lenguajes de programación orientados a objetos 4GL, sistemas avanzados de consultas a bases de datos y un amplio espectro de herramientas para PC.

### **5.6.3 Herramientas para diagramación**

Dan soporte al análisis y documentación de los requerimientos de una aplicación. Por lo general, incluyen facilidades para producir diagramas de flujos de datos. Estas herramientas de alto nivel son esenciales para brindar apoyo a la metodología de análisis estructurado. Las herramientas CASE, incorporan de manera extensa, métodos propios del análisis estructurado.

Estas herramientas ofrecen la capacidad de dibujar diagramas y cartas, además de guardar los detalles en forma interna. Cuando es necesario realizar cambios, la naturaleza de éstos se describen en el sistema, el cual puede entonces volver a dibujar todo el diagrama de manera automática. La capacidad para cambiar y volver a dibujar elimina una actividad que los analistas encuentran tediosa y poco deseable.

#### **5.6.4 Depósito centralizado de información**

Lo que es la captura, análisis, procesamiento y distribución de todos los sistemas de información es asistida por el depósito de información centralizado o diccionario de datos, el cual contiene detalles sobre los componentes del sistema, tales como datos, flujo de datos y procesos; asimismo, también incluye información que describe el volumen y frecuencia de cada una de las actividades. Aunque los diccionarios son diseñados para que el acceso a la información sea sencillo, también incluyen controles y medidas de protección que preservan la exactitud y consistencia de los detalles del sistema. El uso de 1. Niveles de Autorización, 2. Validación de procesos y 3. Procedimientos para verificar la consistencia de las descripciones, asegura que el acceso a las definiciones y a las revisiones hechas a ellos en el depósito de información, ocurran en forma apropiada y acorde con los procedimientos ya establecidos.

#### **5.6.5 Generador de interfaces**

Las interfaces con el sistema son los medio que permiten a los usuarios interactuar con una aplicación, ya sea para dar entrada a información y datos o para recibir información. Los generadores de interfaces ofrecen la capacidad para preparar imitaciones y prototipos para las interfaces con los usuarios. Por lo general, soportan la rápida creación de menús de demostración para el sistema, de pantallas de presentación y del formato de los informes. Los generadores de interfaces son importantes para el desarrollo de prototipo de aplicación, aunque también son de utilidad para los demás métodos de desarrollo.

### **5.6.6 Generadores de código**

Automatizan la preparación del software. Estos incorporan métodos que permiten convertir las especificaciones del sistema en código ejecutable. La generación de códigos aun no ha sido perfeccionada, los mejores generadores de código producen aproximadamente el 75% del código fuente de una aplicación. El resto debe ser escrito por los programadores. La codificación manual, que es el nombre que recibe este proceso, sigue siendo necesaria.

Dado que las herramientas CASE son de propósito general, es decir no están limitadas a ciertas áreas específicas de aplicación, como el control de procesos de manufactura, análisis de portafolio o administración de cuentas, resulta que el desafío de automatizar el proceso de generación de software es sustancial. Los mayores beneficios se obtienen cuando los generadores de código se encuentran integrados con un depósito central de información. Esta combinación alcanza el objetivo de crear un código que pueda volverse a emplear. Cuando las especificaciones cambian. Se puede volver a generar el código al alimentar los detalles del diccionario de datos, a través de generador de código. El contenido del diccionario puede emplearse de nuevo para preparar el código ejecutable.

### **5.6.7 Herramienta de Creación de Prototipo**

Estas herramientas abarcan un amplio conjunto de herramientas de muy variada simulación para el análisis de la temporización y dimensionamiento de sistemas de tiempo real. A nivel básico, estas herramientas se centran en la creación de pantallas e informes que permitan al usuario entender el ámbito de entrada y de salida de un sistema de información o de una aplicación de ingeniería. En un nivel sofisticado estas herramientas permiten crear un modulo del sistema para aplicaciones de tiempo real. Después, se puede analizar y, en algunos casos, ejecutar los módulos creados mediante las herramientas, para así evaluar el rendimiento de la ejecución del sistema propuesto antes de construirlo.

### 5.6.8 Herramientas de soporte

Aquí se encuentran las herramientas de producción de documentos, el software de sistemas de red, las bases de datos, el correo electrónico, todas ellas para administrar y controlar la información que se genera a medida que se desarrolla el software.

A continuación se describen cada una de las herramientas de soporte mencionadas.

- **Herramientas de Documentación:** Estas herramientas se utilizan en casi todos los aspectos de la ingeniería del software, representando así una buena oportunidad para los desarrolladores de aplicaciones, puesto que en algunas de las organizaciones se emplea mucho tiempo en el desarrollo de documentos y a veces resulta muy ineficiente.
- **Herramientas para soporte de comunicación:** las herramientas CASE conforman una tecnología de estaciones de trabajo, que deben estar soportadas por sistemas de redes de comunicación de alta calidad, correo electrónico, boletines electrónicos, etc., para aumentar los beneficios de desarrollo en grupo.
- **Herramientas de Prueba:** las herramientas de éste tipo proporcionan muchos niveles de soporte diferentes para los pasos que con lleva la prueba del software. Algunas herramientas proporcionan un soporte directo para el diseño de la prueba y se utilizan durante las primera etapas de la misma. Otras herramientas, como las pruebas de regresión automática y las herramientas de generación de datos de prueba, se utilizan durante la prueba de validación y durante la integración ayudando a reducir la cantidad de esfuerzo aplicado al proceso de realización de pruebas.
- **Herramientas de BD:** éstas sirven como fundamento sobre el cual establecer el repositorio de información de la aplicación la que también se denomina BD del proyecto. Poniendo énfasis en

los objetos de la configuración, las herramientas de gestión de BD pueden evolucionar de los sistemas relacionales a los sistemas basados en objetos.

#### **5.6.9 Herramientas de administración.**

Este componente ayuda a los gerentes de desarrollo a calendarizar las actividades de análisis y diseño así como la asignación de recursos a las diferentes actividades del proyecto. Algunas de estas herramientas permiten que los gerentes de proyecto especifiquen elementos de su propia elección. Por ejemplo pueden seleccionar los símbolos gráficos que deseen definir metodologías de desarrollo propias, incluyendo las reglas de validación y los estándares para datos y nombres de procedimientos. Sin embargo la mayor parte de los sistemas CASE pretenden en gran medida que la notación, principios y prácticas del análisis estructurado.

#### **5.6.10 Herramientas de planificación de sistemas de administración de proyectos**

Estas herramientas proporcionan un metamodelo del cual se puede obtener sistemas de información específicos. La información de gestión se modela según va pasando a través de las distintas entidades organizativas de una empresa. El objetivo principal de las herramientas de ésta categoría es ayudar a comprender mejor como fluye la información entre las distintas unidades organizativas. Debe tomarse en cuenta que las herramientas de este tipo no son adecuadas para todas las organizaciones, ya que se requiere compromisos por parte de la dirección de la empresa en cuanto a los recursos que son necesarios y por ende el entendimiento de la misma con respecto al tiempo que debe emplearse al hacer uso de éstas herramientas, así como también el costo respectivo, para producir un modelo y actuar según la información obtenida de éste. Sin embargo éstas 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.

Las herramientas de administración de proyecto se dividen en:

- Herramientas de Planificación de proyectos.

Este tipo de herramienta se centra en dos áreas principales: el esfuerzo y goce de un proyecto y la planificación del mismo. La estimación del coste permite al administrador calcular el tamaño del proyecto, utilizando una medida indirecta (por ejemplo líneas de código) y describir las características globales del proyecto (complejidad, experiencia del personal, madurez del proceso). Estas herramientas calculan el esfuerzo estimado, la duración del proyecto y el número de personal recomendado.

## **5.7 EVALUACION DE UNA HERRAMIENTA CASE**

En este apartado se sugiere que el catedrático evalúe una herramienta que esté de acuerdo a las necesidades del mercado local.

## **5.8 HERRAMIENTA CASE A UTILIZAR**

En este apartado se sugiere que el catedrático utilice la herramienta analizada en el capítulo anterior y de a conocer las bases fundamentales para la utilización de la misma.

# **CAPITULO VI**

## **TECNICAS APLICADAS A LAS HERRAMIENTAS CASE EN EL DESARROLLO DE SOFTWARE.**

### **6.1 INTRODUCCION**

En este capítulo se dan a conocer algunas de las técnicas más utilizadas en el desarrollo de aplicaciones dentro de las que podemos mencionar diagramas de flujo de datos cuyo objetivo principal es la descomposición de un problema complejo en otros más sencillos y manejables, otra de las técnicas utilizadas en el desarrollo de aplicaciones es Modelos Entidad/Relación es una de las técnicas más populares y utilizadas dentro del mundo del diseño de aplicaciones entre otras.

### **6.2 PRINCIPALES TÉCNICAS APLICADAS EN EL DESARROLLO DE APLICACIONES**

#### **6.2.1 Diagramas de Flujos de Datos**

Los diagramas de Flujos de Datos se encuentran entre las técnicas estructuradas de análisis y es el método más extendido en modelado conceptual de procesos. Esta técnica que también es conocida por las siglas DFD, o por diagramas de burbujas fue propuesta por Yourdon a finales de los años 70.

El objetivo principal de ésta técnica es la descomposición de un problema complejo en otros más sencillos y manejables, facilitando la modularidad del sistema así como aprovechar la comunicabilidad a través de modelos gráficos. Pretende, además separar la estructura física del sistema de la lógica, aumentando la facilidad de mantenimiento de los análisis. Los DFD constan de muy pocos elementos y además son fáciles de aprender.

Estos elementos son:

- **Procesos:** son los lugares donde se transforma o descompone la información de un flujo de datos. Se representan con un círculo indicando su nombre dentro.
- **Entidades externas:** son aquellas organizaciones, departamentos, personas o entidades en general que no pertenecen al sistema de información, pero que tienen relación con él. Es de donde proceden los datos que necesitan el sistema o hacia donde van los generados por él. Solo aparecen en el diagrama de contexto inicial. Se representa mediante un rectángulo con su nombre en el interior.
- **Almacenes de Datos:** Son aquellos lugares donde se almacena de forma estática la información dentro del sistema. Su nombre no indica como está organizado se representa gráficamente con dos líneas paralelas.
- **Flujos de Datos:** son las tuberías o caminos por donde fluye la información entre los diferentes elementos de sistemas. Se representa a través de un arco orientado.

### **Proceso de Análisis**

El proceso de análisis consiste en un refinamiento progresivo a través de la siguientes fases:

- Construcción del diagrama de contexto. Es aquel DFD compuesto por un solo proceso que representa todo el sistema, las entidades externas que participan en él y los flujos de entrada y de salida.
- Descomposición de cada proceso en los subsiguientes niveles de DFD hasta conseguir todos los procesos primitivos (aquellos que no se pueden seguir descomponiendo).
- Descripción de todos los elementos obtenidos en el diccionario de datos

### **Propiedades de los DFD**

Existe una gran serie de características generales que hay que tomar en cuenta a la hora de enseñar un diagrama de flujo de datos óptimos , dentro de las características tenemos:

- Evitar la utilización de estructuras ilegales. Estas son:
  - Flujo de datos que se subdivide en diagramas. Es decir, para que un flujo de datos se pueda desdoblar es necesario un proceso que realice esta operación. No podemos olvidar que una de las funciones de los procesos es descomponer flujos de datos.
  - Señales de control, y en concreto señales de activación. No pueden aparecer flujos de datos cuyo único contenido sean flags (datos booleanos, señales de final de ficheros, pasos de control por llamadas a procedimientos, etc.)
  - Bucles. Si un flujo entra y sale en el mismo proceso significa que es algo que solo le interesa a dicho proceso, por lo tanto, debe quedar escondido.
  - Flujo de datos entre dos almacenes de datos. Para poder hacer una lectura en un almacén y una escritura en otro, es necesario la existencia de un proceso que realice dicha función.
  - Procesos aislados. No tiene sentido la existencia de un proceso que no emita o reciba flujos de datos sin conexión alguna con el resto del sistema.

- Flujo entre entidades externas. El paso de información que puede existir entre dos entidades externas es algo que no interesa al sistema que se está diseñando. Si nos damos cuenta que si interesa es porque ésta información debe pasar a través del sistema.
- Procesos sumidero y proceso fuente. Son aquellos que sólo remiten o reciben información. No son válidos en un DFD.
- Procurar mantener la propiedad de conservación de los datos.
  - Los datos no se crean solo se transforman. No pueden existir datos en el sistema que surjan de la nada . deben aparecer en el sistema viniendo del exterior o como resultado de la transformación de otros flujos.
  - Balanceo entre niveles. Según ésta propiedad los flujos de entrada y de salida de un proceso de nivel  $n$  tienen que ser los mismos que en el diagrama  $n+1$ . Esta propiedad se rige por los siguientes puntos:
    - Todas las entradas de un diagrama  $n+1$  deben estar en el diagrama  $n$
    - Todas las salidas de el diagrama  $n+1$  deben ser las mismas del diagrama  $n$ , excepto los rechazos triviales, entendiendo por estos últimos aquellos flujos de salida que representan un mensaje de error o de cualquier otro tipo que no sean muy interesante hacer notar desde el primer nivel.
- Propiedades Prácticas. Existe una serie de normas que parten de la experiencia:
  - Dividir el sistema de manera natural no por empeñarnos en descomponer el sistema va a estar mejor analizado. Solo debemos descomponer si realmente es necesario y cuando la propia naturaleza del sistema lo determine.

- Establecer conexiones simples. Cuantos menos flujos de datos hayan entre dos procesos mejor. Si se ve que son necesarios, agruparemos primeros los flujos y lo descompondremos en los niveles sucesivos.
- Limitar el número de procesos dentro de un nivel de descomposición determinado.  
El número ideal es  $7 + 2$ .
- Explosionar cuanto sea necesario. El sistema no va a ser más complicado por que tenga más niveles todo lo contrario.
- Descomponer procesos si tienen varios flujos de entrada y/o salida. Si existen varios flujos entrando o saliendo de un proceso, eso nos indica que seguramente existen partes de la lógica de dicho proceso que en niveles inferiores serán procesos individuales.
- Nominar correctamente todos los objetivos que surjan en los diagramas. Los elementos lingüísticos que se recomiendan para cada tipo de objeto son los siguientes:
  1. Flujo de datos: un sustantivo más un adjetivo
  2. Procesos: un verbo más un sustantivo
  3. Almacenes de datos: un sustantivo

### **6.2.2 Modelos Entidad/Relación**

El modelo Entidad / Relación (ME/R) es una de las técnicas más populares y utilizadas dentro del mundo del diseño de aplicaciones, concretamente, en las bases de datos. Puede incluso ser una técnica utilizada por personas poco involucradas en el mundo de la informática y según algunos críticos, hasta bastante exitosa, debido a que es una técnica gráfica.

El ME/R tiene uno de sus componentes en el diagrama de Entidad/Relación (DE/R) , que se atribuye básicamente a Peter Chen, en los comienzos de los años 70, Chen (1976). Pero desde entonces han

aparecido muchas variantes y extensiones, DE MIGUEL Y PIATTINI (1993). Hoy día y en las metodologías más modernas como MARTIN (1987), los diagramas entidad/relación se utilizan como una base para el desarrollo de modelos datos en el diseño de aplicativos; éstas variantes (como MER extendido) destacan en muchas metodologías. No por ello, el MER deja de ser una de las técnicas más distinguidas y populares entre los analistas e ingenieros del software.

En un MER se puede identificar los siguientes conceptos:

Entidad, Relación (Asociación), Cardinalidad, Atributos, Ocurrencia, Identificador, Tipos de Datos y dependiendo del nivel de desarrollo de los modelos, se utilizarán otros conceptos que alcanzarán directamente con la técnica de diseño de bases de datos (Con elementos como subtipos, supertipos, opcionalidad, dominio, etc.).

Entidad significa todo objeto del que se desea guardar información; es decir, se pretende almacenar una serie de datos para su posterior recuperación y tratamiento.

Relación significa básicamente cualquier asociación o interdependencia entre varias entidades.

Las Cardinalidades significan las reglas del negocio por las cuales las entidades existen en el modelo; es decir, definen la manera de asociar entidades.

Los atributos son características o propiedades que describen a las entidades. Los atributos toman distintos valores. Estos atributos se definen durante el desarrollo de los diagramas y siempre se realiza de forma iterativa, es decir, es un proceso cíclico donde todos los atributos se definen después de varios estudios. A veces algunos atributos se construyen por propia necesidad en el modelo para poder identificar valores de formas ordenadas. Una ocurrencia de una entidad es un dato definido siguiendo el modelo diseñado.

Un identificador (clave) es un atributo que reúne una serie de condiciones y que tiene la utilidad de poder definir una ocurrencia de manera única. Un identificador debe tener un solo valor que no se puede repetir y no puede ser nulo.

Los tipos de datos son la manera en que los atributos se representarán de forma física en el diseño de las Bases de Datos. Estas formas físicas son propias del entorno sobre el cual se están desarrollando y por lo tanto son características físicas.

La normalización es frecuentemente aplicada sobre los métodos MER para poder tener los modelos diseñados de una forma correcta, eliminando toda redundancia que pudiese existir en el diseño de las bases de datos. Normalmente el proceso comienza con el desarrollo de los MER y termina aplicando la teoría de la normalización.

Existe otras muchas formas de modelar los datos, pero ciertamente el MER es el más implementado en las herramientas CASE hoy en día existen muchas herramientas CASE que soportan las distintas notaciones de los MER con los correspondientes diccionarios.

## **Técnicas**

### **Diagramas de clases:**

Las metodologías representan los conceptos de clase y de objeto. Un aspecto muy importante a destacar es la existencia de servicios y atributos que pertenecen a las clases.

En los diagramas de clases se suelen representar las Interrelaciones como en los diagramas E/R, se representa la generalización(herencia) formando retículos o jerarquías de clases, bien mediante flechas que suelen ir de la subclase a la super clase o mediante triángulos o semicírculos.

### **Tarjetas de Clase**

Consiste en elaborar por cada clase una tarjeta en la que se apuntan las principales características de la clase, y que permite validar un sistema orientado al objeto jugando con las clases, de una manera antropomórfica, examinando como una clase envía mensajes a otra y cuales son sus responsabilidades, para ver si posee las características necesarias y atender la las peticiones que le lleguen.

### **Diagramas de Comunicación/Colaboración de objetos**

Se utiliza para mostrar la iteración de los objetos a través del paso de mensajes. También pueden identificarse los mensajes que se intercambian los objetos.

### **Diagramas de Transición de Estados**

Permiten la agregación y generalización de estados. Para evitar la exposición de estados que podrían darse en un sistema orientado al objeto, se suele definir un diagrama de estado por cada clase.

### **Escenarios(Casos de uso)**

Encargado de describir el dialogo entre los usuarios. A la representación gráfica le acompaña una secuencia de transacciones que representa el dialogo del usuario con el sistema y que permite validar su implementación.

Entre las técnicas mas utilizadas en la metodología orientada a objeto podemos mencionar: El desarrollo incrementa, las responsabilidades y los casos de uso.

### **Limitaciones de la metodología actual y tendencias futuras**

Las carencias mas importantes en la actualidad son las siguientes:

- No poseer un modelo formal que sirva de base para la metodología.

- No abarcar todas las fases del desarrollo; algunas se limitan simplemente a técnicas específicas de una fase, otras se centran en el diseño o en el análisis pero muy pocas cubren ese tipo de desarrollo.
- No soportar todos los aspectos importantes para el modelado de datos.
- No soportar la trazabilidad, que debe ser bidireccional, de los requisitos al código y viceversa.
- No incluir los aspectos de prueba que prácticamente ignoran o descuidan la mayor parte de la metodologías.

Podemos suponer que en un futuro cercano los propios diseñadores irán adoptando las metodologías y técnicas que sean mas fáciles de utilizar, sólidas y rigurosas y que se encuentran ampliamente soportadas; mientras que el resto de las metodologías irán desapareciendo, aunque algunas de sus ideas se incorporen en las metodologías que sobrevivan, al igual que sucedió con las técnicas estructuradas.

### **6.2.3 Diagramas de Estructura (Structure Charts)**

La programación estructurada fue la primera técnica que contempló la complejidad de los sistemas modernos. La técnica sugerida por la programación estructurada para hacer una progresión desde la visión general hasta el detalle, se conoce como diseño top-down.

Todos los conceptos del diseño top-down pertenecen en el diseño estructurado. La principal herramienta usada en diseño estructurado para representar la estructura de un sistema de estructura.

#### **Elementos Principales del Diagrama de Estructura.**

- Módulo es un número de sentencias que hacen una actividad, se representa por una caja rectangular con su nombre dentro.

- Conexión Intermodular: un sistema está compuesto por módulos organizados jerárquicamente, cooperando y comunicándose entre sí para realizar una tarea.
- Comunicación Intermodular: los signos de la comunicación son los datos y los flags.
- Símbolo de centro de transacción.
- Conectores y continuidad de página para evitar el cruce de líneas en el diagrama de estructuras, se utilizan los conectores.

#### **6.2.4 Matrices**

Ayudan principalmente a realizar la verificación y validación entre el mundo de los datos y el de los procesos (matriz evento/entidad), éstas técnicas sirven para que se realicen con éxito otras (matriz entidad/entidad en la realización del modelo de datos entidad/relación).

##### **Matriz entidad/entidad**

Cuando una relación entre dos entidades se duplica por lazos que van vía una tercera entidad, entonces no se debe incluir esa relación. A veces se puede ser más riguroso indicando la Cardinalidad en la intersección.

##### **Matriz evento/entidad**

Una vez identificadas las entidades y los eventos se deben relacionar mediante la matriz: se ha de reconocer que efecto tienen cada evento sobre las entidades y se ha de marcar en la casilla correspondiente de la matriz.

### **Matriz papel del usuario/función**

La matriz muestra que papel de usuario puede iniciar que conjunto de funciones. Cada intersección en la matriz representa un dialogo y por lo tanto cada fila identifica todos los diálogos que necesita un usuario dado.

### **6.2.5 Diagrama de Transición de Estados**

Es una técnica de modelado que se basa en el comportamiento dependiente del tiempo de un sistema. Normalmente se utiliza para representar el comportamiento de sistemas de tiempo real donde el SW debe responder a sucesos del mundo real en un tiempo muy limitado lo define como un diagrama que representa los estados que puede tomar un componente o un sistema y muestra los eventos o circunstancias que implican en el cambio de un estado a otro.

#### **Componentes de un Diagrama de Transición de Estados**

Los componentes son : El estado y la transición.

**El estado** representa un modo externo de comportamiento. El nombre del estado es el nombre del comportamiento exhibido por el sistema. Uno de los estados de un DTE será el inicial, y se representa con una flecha de transición apuntando hacia el y sin ningún estado origen.

**Las transiciones** representan el paso de un estado a otro.

También puede ocurrir que una transición termine en el mismo estado del que parte o que hay múltiples transiciones desde o hacia otro estado.

Las condiciones provocan una transición en el sistema una vez que cumple la condición, se toman las acciones indicadas y se cambia de estado.

## **Aplicaciones del Diagrama de Transición de Estado**

Una aplicación muy interesante de los diagramas de transición de estados es su utilización para especificar transformaciones de datos y de control.

Las transformaciones son elementos básicos utilizados en metodologías de modelado de sistemas en tiempo real. Su trabajo es describir de una forma gráfica y sencilla el comportamiento del sistema.

Una transformación de datos simboliza, mediante un círculo la función que se realiza para generar unos flujos de datos de salida a partir de unos flujos de datos de entrada.

Una transformación de control es aquella que solo admite como entrada y salida flujos de control sin datos.

### **6.2.6 Teoría de la Normalización**

Es una técnica bastante difundida en las metodologías de desarrollo de sistemas de información que suelen emplearse tanto en el aspecto conceptual como lógico.

Esta técnica ayuda a los diseñadores a prevenir problemas de redundancia y anomalías de modificación, inserción o borrado en los esquemas de datos, aunque a costa de penalizar su recuperación.

Consiste en ir descomponiendo los registros (relación, entidad) en otros de menor tamaño, de forma que satisfaga una serie de restricciones específicas que definen lo que se reconoce de forma normal.

#### **Formas Normales más Usuales**

La primera forma normal definida por Cood junto con el modelo relacional prohíbe que *en un registro hayan grupos repetitivos*, es decir, todos los campos deben de ser atómicos.

La segunda y tercera formas normales, que también fueron definidas por Cood se formulan tomando en cuenta *la relación entre los campos claves y los que no forman parte de ninguna clave*.

Así decimos que un registro está en segunda forma normal sí, además de estar en primera forma normal todos los campos que no forman parte de ninguna clave candidata suministra información acerca de la clave completa.

La tercera forma normal además de englobar a las dos anteriores añade la siguiente restricción: *los campos que no forman parte de la clave candidata deben facilitar información solo acerca de las claves candidatas y no acerca de otros campos*. Sus campos deben ser mutuamente independientes y completamente dependientes de las claves candidatas.

La cuarta y la quinta forma normales fueron propuestas por Fagin en 1977 y en 1979, respectivamente, demostrándose que la quinta forma normal es la última posible utilizando esta forma de descomposición. Estas formas normales se encuentran relacionadas con las dependencias multivaluadas y las de proyección/combinación, que, debido a su complejidad y falta de espacio nos vemos obligados a omitir.

Concluyendo podemos decir que la teoría de normalización es, como señala DATE (1990), ni más ni menos que la formalización del sentido común, por lo que resulta muy práctica en el desarrollo de sistemas de información, incorporándose como en mayor o menor medida, en las actuales herramientas CASE.

# **CAPITULO VII**

## **METODOLOGIAS APLICADAS A LAS HERRAMIENTAS CASE EN EL DESARROLLO DE SOFTWARE.**

### **7.1 INTRODUCCION**

Este capítulo desarrolla un resumen de diversas metodologías aplicadas a las herramientas CASE en el desarrollo del software.

Además muestra una breve reseña histórica en donde se da a conocer que un método es la secuencia de modelados que ayuda a construir, a partir de la original realidad de una o varias cadenas de modelos, derivados unos de otros, con el objetivo de lograr un modelo material final o sistema que concrete la nueva parcela de realidad deseada en relación con la original. Como también se menciona que la metodología es el discurso, ciencia o estudio de los métodos.

#### **7.1.1 Reseña Histórica**

Todo el esfuerzo carente de ordenadores, modeladores e incluso herramientas, han quedado como en la prehistoria de los métodos y la posterior historia de los métodos tienen en común cierto enfoque pragmático que empieza por reestudiar la mejora de la programación en sí (en los años 70 más preocupada por el simbólico sintáctico, pasando por la estructuración, la orientación al objeto y otras prometedoras modalidades) en cada etapa sucesiva aparece, pese al cambio de enfoque, el mismo problema: el código, incluso impecable técnicamente no cubre las expectativas.

Esta constante tiene como explicación la insuficiente comunicación de ideas y necesidades de los clientes con los proveedores, y entre estos, de los diseñadores por los ejecutores y mantenedores. Así, esa mejora de comunicación se ha ido abordando tras cada mejora de la programación por la extensión de ésta a otras fases preparatorias, sean previas (diseño de especificaciones, análisis de requerimientos, estudio y oportunidad, planificación) y/o cíclicas (mantenimiento, prototipo). La programación se ha ido así clarificando no solo internamente sino externamente por adjunción de fases preparatorias.

### **7.1.2 Conceptos y Elementos De Métodos**

Un método es la secuencia de modelados que ayuda a construir, a partir de la original de la realidad una o varias cadenas de modelos, derivados unos de otros, con el objetivo de lograr un modelo material final o sistema que concrete la nueva parcela de realidad deseada en relación con la original. En sentido estricto, metodología es el discurso, ciencia o estudio de los métodos.

Lo anterior cubre toda acción de la parcela de realidad llamada información desde sus concepciones más clásicas, como las de SHANNON hasta sus enfoques recientes, como el de BOOCH en la orientación a objetos.

Un método es también un sistema de información: al aplicarle la visión metódica, o sea, siguiendo un proceso metametódicos, el método puede analizarse como un conjunto de entidades relacionadas entre sí y como una estructura de procedimientos que sintetizan un flujo de procesos generador de resultados. Las entidades primarias objeto de modelado detallado por los enfoques metametódicos como el modelo V describe las actividades, tareas, estados, funciones y productos de todo método en general. Sus relaciones articulan la estructura, los procedimientos(técnicas) y los resultados de

cada método particular. La **estructura** del método se acopla al ciclo de desarrollo del sistemas de información deseado, y para alcanzar este, sigue un proceso organizado y por pasos. La organización del proceso se basa en descripciones de los modelos sucesivos realizadas en los lenguajes abstractos apropiados a cada paso y entendibles por el receptor adecuado(que en el ultimo paso será el hardware y el utilizados).

El **ciclo de desarrollo** (amplia, pero impropriamente llamado ciclo de vida) tiene tres grandes grupos de fases: preejecutor(planificación, especificación, análisis), ejecutor( diseño, codificación pruebas) y postejecutor ( documentación, formación de usuarios, mantenimiento, rectificaciones); unos métodos no incluyen todas las fases en su estructura y otros incluyen fases paralelas al desarrollo como gestión del proyecto, de su calidad y configuración.

Las fases se subdividen en subfases (como módulos, etapas, etc.) hasta que la habitual estructura alcanza las **tareas** actividades elementales realizadas en condiciones atómicas, (un objetivo, un tipo de actor, un puesto). A ciertas fases o tareas que suponen ensanchamiento de opciones(para buscar los requerimientos) le siguen otras de restricción (con valoraciones y decisiones que buscan soluciones específicas).

Cada tarea se apoya en un **procedimiento** que prescribe su forma de ejecución y es vehículo de comunicación entre los conocedores de la realidad original, sus modeladores y los usuarios del sistema final. El procedimiento obtiene un **resultado** a partir de los resultados de otros procedimiento previamente complementados. El conjunto de resultados finales o productos (código ejecutable, descripciones, informes) constituye el sistema deseado, si se prueba su consistencia interna y externa.

Las reglas de un procedimiento, al menos textuales y a menudo mixtas se han estabilizados en un número de técnicas, que son comunes a muchos métodos y a menudo se profundizan al margen de estos. Las **herramientas** CASE y cada vez más entornos, informatizan las técnicas diagramáticas, algorítmicas y textuales: no solo formalizan documentaciones, sino que deben contener los mecanismos rutinarios formalizados para verificar las consistencias de las técnicas soportadas. Según el alcance de las técnicas que cubra la herramienta será común a varios o específicas a cada uno de ellos.

### **7.1.3 Encapsulación de los métodos**

Los métodos se empiezan a encapsular como una panoplia de otros dispositivos abstractos, por emplear el paradigma de los que rodean el microprocesador central de un computador personal: la placa-rack de soporte sería un metamétodo mientras que la carcasa envolvente comprendería un frontal de perimétodos y los laterales, cubiertos con mecanismos paralelos al desarrollo, como los mecanismos para gestionar calidad, configuración, seguridad o reparto o bien extensores a enfoque y/o sectores complementarios o suplementarios al clásico núcleo de gestión de organizaciones.

El soporte metametodico a porta la estructuración decisiva de los procesos y relaciones básicas subyacentes a una gran mayoría de métodos sean tradicionales o nuevos. Por cierto la articulación de esas actividades, tareas, estados, roles, resultados y productos tienen una infraestructura conceptual similar a la que subyacen al modelo de gestión de flujos de trabajo, ligado por otra parte a la reingeniería de procesos de organizaciones.

Las interfases perimetódicas como eurométodo se centra en la construcción de una estrategia, ósea una forma de articular los modelos intermedios que permitan acercarse óptimamente al objetivo de conseguir el sistema deseado. La estrategia parte de la situación del problema, un primer modelo de

la parcela de realidad original formado por un conjunto de factores que caracterizan su complejidad y su incertidumbre.

La articulación de los dispositivos encapsuladores con los proyectos de desarrollo concreto, realizables en diversos grados y formas, promete ser uno de los aspectos mas importantes de investigación metodológicas de problemas inmaduros que requieren nuevas vías de avance y solución.

#### **7.1.4 Uso de métodos y Herramientas**

Según los estudios realizados se puede concluir que los métodos se emplean relativamente pocos se eligen mas por limitaciones fácticas que por criterios técnicos; no se suelen aprender a fondo ni aplicar rigurosamente. Reducen la satisfacción esperable de sus resultados poco objetivables y su uso complicado para los analistas, e impracticable para los usuarios.

Este panorama practico de los métodos contrasta con un potencial, mas que optimista imprescindible de ventajas inmediatas y mediatas. Las inmediatas mejora la identificación de problemas y errores al producir los sistemas, sobre todos los mas complejos e inciertos. Las ventajas mediatas permiten entender rápida y establemente el problema, controlar con un plan riguroso y no redundante y el desarrollo de proyecto.

En cuanto a las herramientas informatizadas de técnicas y métodos su difusión relativamente reciente dificulta consideraciones ampliamente aceptadas sobre su clasificación y experiencia de uso. Es innegable que la aplicabilidad de un método en problemas de cierta complejidad dependerá cada ves más de su soporte herramental.

## 7.2 PRINCIPALES METODOLOGÍAS

### 7.2.1 Modelo NIAM

El modelo de análisis de la información NIAM (Nijssen Information Analysis, Methodology) fue desarrollado por el doctor G. M. Nijssen y reconocido por ISO (International Standard Organization) en 1982.

NIAM permite obtener, a partir de un conjunto simple, un diagrama conceptual basado en relaciones binarias y en un conjunto de restricciones definidas sobre estas. En dicho diagrama se representan todas las fases elementales que permiten la descripción del universo del discurso dado. Dentro de los conceptos que permiten llevar a cabo el modelado de las fases elementales, así como, el formalismo gráfico que corresponde a dicho modelo están:

#### **Tipos de Objeto**

- El tipo de objeto no léxico (NOLOT) se utiliza para representar las entidades que forman el universo del discurso. No deben existir 2 NOLOT distintos que representen la misma entidad. Se representa mediante un círculo con trazos continuos.
- El tipo de objeto léxico (LOT) representa las características de las entidades a las cuales representan los NOLOT . Su representación es un círculo de trazo discontinuo.
- El tipo idea representa una relación binaria entre dos NOLOT, que pueden ser el mismo, en donde cada uno de ellos tiene un papel definido dentro de la relación.
- El tipo puente representa una relación binaria entre un NOLOT y un LOT. En este tipo de objeto de la misma forma que ocurría con el tipo IDEA, el LOT tienen un papel asociado y el NOLOT otro.

## **Tipos de Restricciones**

Existen varios tipos de restricciones con las que se puede llevar a cabo la descripción del sistema de información con más exactitud y rigor:

### *Restricciones de Integridad sobre un papel de una relación binaria:*

- Restricciones de unicidad: a toda ocurrencia de un objeto le corresponde una única ocurrencia del otro objeto involucrado en la idea o puente se representa mediante una doble flecha. Se trata de una relación de tipo 1: N.
- Restricción de Obligatoriedad: a toda ocurrencia de un objeto NOLOT le corresponde al menos una ocurrencia de otro objeto (NOLOT o LOT). Se representa por el símbolo “v”

### *Restricciones de Integridad entre Papeles, Ideas o Puentes:*

- Restricción de Unicidad entre varios papeles de ideas o puentes; un objeto queda determinado de manera única por otro de tipo LOT o NOLOT. Se representa por el símbolo “U”.
- Restricción de Obligatoriedad entre varios papeles ó ideas: todas las ocurrencias de un objeto intervienen al menos una vez en uno de los papeles de una de las ideas. Se representa mediante el símbolo “T” uniendo los papeles implicados.

### *Restricciones de Igualdad entre Papeles e Ideas*

- Restricciones de igualdad entre dos papeles de dos ideas: toda ocurrencia de un objeto que participa en uno de los papeles implicados en la restricción de igualdad también participa en otro papel de la otra idea implicada. Se representa mediante el símbolo “=” uniendo los papeles implicados.

- Restricción de igualdad entre dos ideas: el objeto de ocurrencias de dos ideas son iguales. Se representa gráficamente mediante un símbolo “=” que une las dos ideas implicadas.

#### *Restricciones de Exclusión entre Papeles e Ideas*

- Restricciones de exclusión entre dos papeles y dos ideas: una ocurrencia de un objeto solo puede participa en uno de los papeles implicados. Se representa mediante el símbolo “X”
- Restricción de exclusión entre dos ideas: el conjunto de ocurrencias perteneciente a una ideas es disjunto con respecto al de la otra idea implicada en la restricción. Se representa gráficamente mediante un símbolo “X” uniendo las ideas implicadas.

#### *Restricciones de Inclusión entre Papeles o entre Ideas*

- Restricciones de inclusión entre dos papeles de dos ideas: dado un objeto que participa en dos ideas distintas las ocurrencias de dicho objeto que participan en el papel de una de las ideas participan también en el de la otra idea implicada en la relación ya que constituye un subconjunto de ellas. Se representa mediante una flecha que va del papel incluido hacia el papel que incluye.
- Restricción de inclusión entre dos ideas: el conjunto de ocurrencias que intervienen en una relación es un subconjunto del conjunto de ocurrencias de la otra relación implicada. Se representa con una flecha que va desde la idea incluida a la idea que la incluye.

#### **Subtipos**

Este tipo de restricción lleva asociada la idea de herencia de propiedades entre NOLOT. Se representa mediante una flecha que va desde el subtipo al supertipo y que significa es un (a).

- Restricciones de totalidad entre varios subtipos: el conjunto de ocurrencias que el objeto supertipo se obtiene como unión del conjunto de ocurrencias de sus objetos subtipos se representa mediante un símbolo “T”
- Restricciones de exclusión entre subtipos: donde el conjunto de las ocurrencias de cada uno de los subtipos implicados en la relación son disjuntos. Se representa con una “X”.

### **7.2.2 Diagrama Jackson**

La metodología Jackson desciende directamente de la programación estructurada que surgió a finales de la década de 1960 y principio de los años 70. En 1975 fue publicado en el libro Principles of Program Design, y su utilización se difundió ampliamente. Favorecida principalmente por la elegancia, eficiencia y corrección de la técnica.

El método de Jackson se basa en el principio de que el punto inicial de diseño de programa son los datos del problema y no los requisitos funcionales exigidos. Esta forma de enfocar el desarrollo de software está basada en los datos, ya que los datos del problema son los únicos elementos de las especificaciones que tienen una estructura objetiva y que pueden ser usados como una base lógica y racional para la estructura del programa, evitando utilizar una descomposición fundamental que tiene un alto grado de juicio subjetivos o bien un desarrollo mediante organigramas que aportan una visión poco clara de la estructura completa .

Las tareas a realizar son:

- Formar las estructuras de datos de salida y entradas a partir de los datos del problema. Mediante un proceso de determinación de correspondencia entre estas estructuras de datos se obtienen la estructura única del programa.

- A esta estructura del programa se le asignan las operaciones ejecutables de programa derivadas de las especificaciones funcionales.
- El resultado se traduce en un formato de pseudo código , conocido como lógica esquemática cuya traducción a código es muy simple.

Las ventajas inherentes de éste proceso son:

- La estructura del programa tienen una base consistente
- Se puede asignar las operaciones en el lugar correcto del programa con un mínimo de intuición.
- La estructura del programa refleja la estructura del mundo real del problema a resolver, siendo fácil de entender y mantener.

### **Conceptos básicos**

La estructura puede ser muy variada para unos mismos datos, pero la jerarquía debe ser la que mejor se ajuste al objetivo impuesto por el programa y la estructura debe ser estrictamente jerárquica; es decir, un solo camino a cada entidad desde la entidad raíz.

El formato exige que cada nivel esté compuesto exclusivamente por:

- Todos los elementos de una secuencia
- Todos los elementos de una selección
- Un único elemento repetitivo

Por lo tanto no admite niveles mixtos. Los diagramas utilizan tres estructuras básicas, que son:

**Secuencia:** Se tienen una estructura de secuencia cuando dos o mas componentes son colocados juntos en estricto orden secuencial para formar un componente mayor.

**Repetición:** Se usa una construcción de repetición cuando un elemento de datos se repite varias veces, la iteración, a diferencia de las otras estructuras está formado por un único subcomponente.

**Selección:** La selección se muestra cuando se debe escoger entre dos o más componentes. Pueden ser una o más alternativas.

### 7.2.3 Diagrama de Warnier Orr.

Se desarrollo en Francia. Plantea la programación lógica más que como un método, como una forma de razonar; tratando de evitar los problemas que surgieron durante la fase de introducción de la programación estructurada.

La metodología de Warnier es que la forma o estructura de los datos a procesar determinará la forma o estructura del programa. Uno de los principios en el que se basa es que todo conjunto de información debe subdividirse en subconjunto.

El diagrama de Warnier Orr es una representación gráfica que permite mostrar la estructura lógica de los ficheros de entrada y salida, así como la del programa.

### Conceptos Básicos

La metodología de Warnier se basa en la aplicación de dos principios fundamentales:

- Principio de Ordenación Jerárquica de los Conjuntos de Información (salida, entrada y programa).
- Principio de Correspondencia en la Organización de los Conjuntos de Información.

Para ordenar jerárquicamente los conjuntos, se utilizan las estructuras básicas:

- Repetición de ocurrencias dentro de un mismo conjunto: que se representan en los diagramas indicando el número mínimo y máximo de las mismas.

- Selección entre ocurrencias de un conjunto: Se efectúa la división en subconjunto cuya presencia es aleatoria y son excluyentes entre sí, y se representa por medio del símbolo  $\dot{A}$ .

El principio de correspondencia formula que:

- La organización jerárquica de los datos de entrada estará determinada por los datos de salida.
- La organización del programa tienen determinada por los datos de entrada.
- El control del programa se realiza a partir de los datos de salida.

#### **7.2.4 Prototipado y Desarrollo Rápido de Aplicaciones (RAD)**

Es una de las técnicas más antiguas y prácticas de la humanidad. De hecho, la mayoría de los grandes proyectos se desarrolla con técnicas de prototipado y de maquetas. En el caso de la Ingeniería del SW tiene infinitas ventajas.

Se utiliza y se recomienda para definir sistemas y tener un conjunto de requerimientos diseñados, compuesto, visto o incluso probado por el usuario antes de ser construido. Ofrece al usuario un proyecto evolutivo que puede ir viendo durante la fase de diseño y construcción y finalmente obtener la aplicación en la cual a participado durante su construcción.

Existe una gran diferencia entre el prototipo evolutivo y el prototipo de maquetas, éste es aquel en el que después de realizar el trabajo las maquetas no sirven de nada y hay que volver a construir de forma real y desde cero.

Según MARTIN (1991) son cuatro los ingredientes que forman los elementos para el uso de prototipado: Personas, herramientas, metodología y gestión.

Las personas son la clave del éxito del prototipado o proyecto RAD y éstas deben ser formadas y bien entrenadas las personas deben disponer de herramientas potentes como L4G, incluso son

muchas veces necesarias herramientas C.A.S.E. también haría falta una metodología formal de trabajo y un proceso libre de burocracia para el desarrollo de prototipos.

El RAD es una metodología muy disciplinada que se compone de cuatro pasos para su realización. Se rodea de varios elementos como son un diagrama PERT, un ICASE, que permite el análisis gráfico, diseño, construcción de la aplicación, un diccionario potente, herramientas de prueba y depuración en la creación de prototipo.

El RAD se compone de las siguientes fases:

- **Planificación de requerimientos:** que describe a partir de los usuarios los objetivos y el como de la aplicación .
- **Diseño con el usuario:** que consiste en la participación de los usuarios, de forma no técnica.
- **Construcción:** Con la ayuda de un ICASE se logra tener la aplicación generada con aveces muy pocas modificaciones por parte de los programadores.
- **Implantación:** una vez terminada la aplicación ésta debe pasar las pruebas, ser aprobada y realizarse los cursos de formación.

Las desventajas más grandes del prototipado es que no apto para proyectos muy grandes a largo plazo ni incluso para aplicaciones pequeñas de menos de un mes, el tiempo medio estimado para el éxito del prototipado es en aplicaciones medias y su duración puede ser fijada entre tres y cinco meses.

Sin duda alguna el prototipado reduce drásticamente el ciclo de vida y los tiempos de desarrollo.

Esta es la ventaja principal a tener en cuenta en su utilización.

### **7.2.5 Metodología MERISE**

A medida de los años 70 el ministro de industria francés dándose cuenta de las debilidades que presentaban estos métodos en relación con las nuevas necesidades de enfoque global, se propuso confrontar los diversos puntos de vista de una selección de profesionales de la ingeniería informática para normalizar los elementos de un método informático nacional que fuera la síntesis enriquecida de los métodos existentes.

De esta manera surgió en 1977 la metodología RACINES, que consideraba a la empresa como un todo a ser estudiado, tratando su informatización como un acto estratégico que manejaba un recurso estratégico como es la información y que por lo tanto debe acometerse de manera reflexiva y ordenada mejorando la calidad de los procesos estudiados a fin de proporcionar un mejor nivel de servicios a sus usuarios.

Esta orientación aporta:

- Un marco de análisis global que tienen en cuenta las perspectivas de evolución.
- Una herramienta de planificación del desarrollo
- Una progresión por etapas, implicando controle, validaciones, elaboración de documento y obtención de resultados.
- Una estructura de dialogo entre los participantes en el sistema de información.

En 1979 y también a partir de una agrupación de expertos en ingeniería del software pertenecientes a diversas empresas de servicios coordinadas por el Ministerio de Industria Francés nació MERISE como metodología de análisis y diseño de sistemas de información, aportando un plan de trabajo y técnicas de modelado para la concepción de aplicaciones coherentes para el área de gestión de las empresas, pasando de lo que hasta entonces se había considerado como análisis, a lo que apartir de

entonces se denominada concepción de sistemas de información. Suponiendo dicha concepción una verdadera intersección entre el informático y organización e integrando los sistemas así diseñados en el marco común diseñados por RACINES.

### **Características de la metodología MERISE**

- Aproximación racional para la resolución de problemas mediante la combinación de los diferentes niveles de abstracción considerados y las etapas de concepción y desarrollo definidas, lo que permite jerarquizar los problemas y aportar soluciones construidas apoyándose en técnicas de modelado.
- El modelo de tratamientos basado en las redes de Petri, que proporciona una visión cronológica y completa de las relaciones entre procesos.
- La coherencia de los sistemas resultantes, obtenida gracias a la construcción de modelos de acuerdo con la empresa y el dominio.
- El amplio grado de difusión alcanzado por el método, lo que garantiza las inversiones realizadas y facilita el hallazgo de expertos en el mismo.

### **Etapas**

- Estudio previo. Define de manera global las soluciones conceptuales, organizativas y técnicas del futuro SI.
- Estudio detallado. Define explícitamente las especificaciones funcionales de la aplicación objeto del estudio.
- Estudio Técnico. Realiza la distribución de datos en ficheros físicos y el tratamiento en módulos de programas en función del sistema informático a usar.

- Realización y puesta en marcha. Producción de los módulos de programación e implantación de los medios técnicos y organizativos necesarios.
- Mantenimiento. Etapa que abarca el resto de la vida del sistema hasta que sea sustituido por otro nuevo.

Los modelados utilizados en el método MERISE se apoyan en las siguientes bases metodológicas.

### **Separar los datos de los tratamientos**

Desde el punto de vista del análisis, lo estable reside en la información o memoria colectiva del sistema, mientras que lo inestable reside en las manipulaciones que se aplican a dicha memoria colectiva las cuales serán cambiantes según la organización y necesidades de la empresa.

### **Fragmentar la dificultad**

En los modelos se sigue un orden de graduación de dificultades:

- Primero se trata de comprender la esencia del problema.
- Posteriormente una vez fragmentado el problema se estudiará quien va a ser que, donde lo va hacer y cuando lo va hacer.
- Finalmente se estudiará el como: es decir, se considerará los medios técnicos para implementar la solución diseñada.

### **Comunicar**

Debe basarse en los siguientes principios :

- Máxima formalización del lenguaje para evitar mal entendidos.
- La comunicación se hará con la ayuda de modelos formalizados de carácter eminentemente gráficos.

- Los informes elaborados deberán ser simples, claros, atractivos y manejables.
- La comunicación deberá implicar al máximo a los usuarios haciéndolos reflexionar sobre el nuevo sistema, facilitándole la posibilidad de ejecutar cambios en cada etapa antes de que sea demasiado tarde.

La metodología MERISE considera tres ciclos:

- Ciclo de Vida.
- Ciclo de Abstracción. Subdividido en tres niveles
  - Conceptual. Trata la información y los procesos independientemente de las particularidades organizativas o de los medios utilizados
  - Organizativo/Lógico. Tienen en cuenta la organización establecida, pero no considera los medios técnicos que entraran en juego.
  - Físico. Selecciona los medios tanto materiales como inmateriales
- Puntos de Validación establecidos

Los diferentes niveles del ciclo de abstracción dan lugar a una serie de modelos.

- Modelo Conceptual de Datos (MCR). Representa la información manejada por la empresa y sus Interrelaciones independientemente de su utilización
- Modelo Conceptual de Tratamientos (MCT). Representa gráficamente los procesos y sus relaciones en forma de intercambios de información.
- Modelo Lógico de Datos (MLD). Muestra los caminos de acceso a los diferentes datos y supone el primer nexo de unión entre datos y tratamientos.
- Modelo Organizativo de Tratamiento (MOT). Representa la información expresada en el MCT, teniendo en cuenta la organización existente.

- Modelo Físico de Datos (MFD). Es el reflejo de la memoria colectiva del sistema implantada en un soporte físico dado.
- Modelo Operativo de Tratamientos (MopT). Tienen por misión servir de base para la posterior programación.

Los conceptos de etapa, fase y tarea responden a las definiciones siguientes:

- **Etapa:** Distancia que separa dos pausas consecutivas en la realización del estudio que se caracteriza por una progresión armoniosa de los tres ciclos (Vida, Abstracción y Decisión).
- **Fase:** Subdivisiones de las etapas que persiguen objetivos concretos.
- **Tarea:** Subdivisiones de las fases que corresponden con la menor unidad de trabajo controlable

#### **Puntos más sobresalientes de ésta metodología:**

- Aproximación racional, que abarca el ciclo de vida completo de un sistema de información, asegurando su coherencia.
- Definición estricta del método de trabajo
- Utilización de los tres niveles del ciclo de abstracción
- Amplia utilización de diversas técnicas de modelado
- Visión cronológica de los procesos.
- Amplio grado de difusión.
- Método en constante movimiento, lo que le permite un alto grado de adaptación a las nuevas tecnologías informáticas.

### **7.2.6 Metodología SSAD**

Cuando nació esta metodología, lo hizo solo con las bases de análisis y diseño.

En 1990 apareció la versión 4 que puso un remodelado importante de la estructura del método haciéndola más modular menos interactiva e incorporando nuevas técnicas fundamentales en el diseño de procesos.

SSAD proporciona un conjunto de especificaciones y procedimientos para llevar a cabo las tareas de análisis y diseño de sistemas. No cubre la planificación estratégica ni el control de proyectos y el diseño físico llega hasta el máximo grado de detalle en especificaciones pero sin entrar en la construcción del código.

### **Ventajas e Inconvenientes**

Dentro de las ventajas del SSAD tenemos:

- La alta participación del usuario en la definición de los modelos asegura que el sistema diseñado coincida con el sistema requerido.
- Nos proporciona tres vistas independientes del sistema:
  - La funcionalidad, a través de los diagramas de flujo de datos (DFD) donde vemos los procesos que se desarrollan en nuestro sistema.
  - Los datos con un estudio exhaustivo del modelo (técnica top-down y bottom-up).
  - La lógica de nuestro sistema con una técnica que prácticamente es utilizada solo por esta metodología: la historia de vida de la entidad.
- La separación de conceptos lógicos o físicos facilita en cierto modo la implementación del sistema analizado en distintos entornos.
- Produce gran cantidad de documentación a lo largo de todas sus etapas lo que supone una importante ayuda para el mantenimiento y producción del sistema diseñado

Dentro de los Inconvenientes se tienen:

- El carácter rígido y exigente de su origen británico.
- Cada paso de cada fase está rigurosamente definido, así como los subproductos que se vana ir obteniendo en el camino
- Las técnicas de estudios de datos y procesos son similares a las de otras metodologías pero aporta las suyas propias lo que requiere una formación y entrenamiento adicionales si se pretende seguir el método.

### **7.2.7 Metodología de Ingeniería de la Información**

Es ya una de las metodología mas populares. Desarrollada inicialmente por James Martín (1991) el cual la definió como: La aplicación de un conjunto interrelacionado de técnicas formales para la planificación, análisis, diseño y construcción de sistemas de información en toda la organización o en un área de la misma. La aplicación de la ingeniería de información se basa hoy día en un repositorio de Herramientas CASE. De lo contrario, el proceso puede llegar a ser muy complejo y pesado. La metodología pretende establecer desde la fase inicial de planificación un modelo de sistema de información, basado en la estrategia de la organización, independientemente de la plataforma física.

#### **Características de la Metodología**

- Proporcionar un entorno metodológico para cada fase e integrar éstos con técnicas y herramientas dentro de una sola planificación del proyecto y una estructura de descomposición del trabajo de gestión.
- Uso de técnicas que están integradas en cada fase de la metodología.

Por todo esto, los objetivos generales son:

- Ayudar a la organización a conseguir y defender ventajas competitivas en su mercado mediante la identificación del uso estratégico de la tecnología de la información.
- Proporcionar una arquitectura para el soporte integrado de la información ejecutiva, toma de decisiones, y sistema de información a nivel operativo o intermedio.
- Soportar las necesidades de información de la gestión ejecutiva de una manera eficiente.
- Enfocar los esfuerzos de grupo de SI de la empresa, relacionando el desarrollo del SI con los objetivos de negocio de la empresa y los factores críticos de éxito.
- Mejorar la calidad de los SI mediante el empleo de técnicas rigurosas a través de todo el ciclo de vida de desarrollo.
- Disminuir el tiempo requerido para desarrollar nuevas aplicaciones, así como reducir los costes de mantenimiento de las ya existentes.

### **Las fases de las metodologías**

Las fases de son: Planificación, análisis, diseño y construcción que forman el ciclo de vida del desarrollo de los sistemas de información.

En la fase de planificación se elabora un modelo que soporte los objetivos del negocio y el desarrollo de planes de sistemas de información estratégico y táctico.

La fase de análisis consiste en formalizar los requisitos de información y realizar un modelo conceptual que lo soporte.

La fase de diseño, transforma el modelo conceptual de la aplicación en una especificación detallada de datos y procesos.

La fase de construcción e implantación consiste en construir la nueva aplicación.

Se puede describir la metodología como una manera de trabajar que reúne el conjunto de información, datos o elementos en un repositorio. Las ventajas de trabajar con este método son bien claras, debido a que el SI estará orientado a soportar la estrategia global de la empresa. Establece un enfoque de arquitectura basados en los datos de la empresa, implica al usuario final en las decisiones de desarrollo al nivel apropiado durante la creación de las distintas aplicaciones.

### **Las técnicas de la ingeniería de información**

La ingeniería de información utiliza una serie de técnicas la mayor parte ya sea descrito anteriormente salvo el diagrama de acción, que es propia de metodología de la ingeniería de información.

La fase de planificación utiliza las técnicas de diagrama de descomposición, entidad/relación y matriz de aplicación.

La fase de análisis utiliza E/R, DFD, Diagrama de descomposición y la matriz de asociación, especialmente sobre las entidades y procesos.

La fase de diseño utiliza el diagrama de estructura de datos, el diagrama de descomposición y como técnica mas importante, que es muy propia de la metodología, el diagrama de acción que consiste en dar el máximo de información posible en la descripción de la lógica o incluso en la combinación de una misma técnica para dos usos compatibles en distintas fases.

### **Beneficios a corto y largo plazo de las metodologías**

Los beneficios a corto plazo son:

- Las necesidades de una información son normalmente muy rápidas de identificar.

- La atención primaria se concentra de forma ordenada sobre los objetivos, problemas y factores críticos de éxito.

Los beneficios a largo plazo son:

- Se utiliza la información y la tecnología última para una mayor y mejor construcción de los sistemas.
- La dirección consigue una mejor comprensión de negocios mediante las técnicas utilizadas.
- La misma información se utiliza en distintos lugares de manera integrada, y todo el mundo utiliza un mismo idioma para el trabajo.

#### **6.2.8 Metodologías Orientadas a objeto.**

En el paradigma de la orientación al objeto, un sistema se concibe como un conjunto de objetos que se comunican entre sí mediante mensajes en el aspecto conceptual, un objeto es una entidad percibida en el sistema que se está desarrollando, mientras que al nivel de implementación, un objeto corresponde con un encapsulamiento de un conjunto de operaciones que pueden ser invocadas externamente y de un estado que recuerda el efecto de los servicios. El encapsulamiento es un principio de abstracción que agrupa datos y procesos, permitiendo ocultar a los usuarios de un objeto.

Las metodologías de desarrollo orientadas al objeto han sufrido grandes influencias a lo largo de este paradigma.

Podemos observar un cambio filosófico entre las metodologías clásicas de análisis y diseño estructurado y las de orientación a objeto. En las primeras, se examinan los sistemas desde el punto de vista de las funciones o tareas que deben realizar, tareas que se van descomponiendo en mas

pequeñas las cuales forman los bloques o módulos de las aplicaciones. En la orientación al objeto, por su parte cobra, mucho más importancia el aspecto de modela de sistemas, examinando el dominio del problema como un conjunto de objetos que interactúan.

## **CAPITULO VIII**

### **DIVERSAS HERRAMIENTAS CASE**

#### **8.1 INTRODUCCION**

En resumen se da a conocer algunas de las herramientas CASE existentes, como también mostrar una breve descripción, los beneficios, ventajas, características de cada una de estas herramientas. Además se da a conocer el tipo de requerimientos que se necesitan al momento de implementar dichas herramientas.

#### **8.2 ERWIN (Modelamiento de Data Warehouse y de B.D.)**

Es una nueva versión de la herramienta de diseño de la B.D. reforzando la posición como la herramienta de modelaje de datos líder en el mercado, las capacidades de la versión incluye:

- Nuevo soporte para el establecimiento de estándares de diseño y rehusos
- Propiedades metas expandibles
- Modelaje Dimensional Integrado para el Diseño de Data Warehouse

Estas herramientas requieren diseño interactivo entre el lógico y físico y frecuentemente interacción de diseño durante el despliegue y su evolución; proporciona una ventaja para el ambiente Data Warehouse así como para otro tipo de aplicaciones, aumentando la inversión de la organización en la herramienta Erwin trae consigo nuevas capacidades de diseño incluyendo:

Soporte optimizado para el Modelamiento dimensional con notación tipo estrella "star schema" y reglas de validación para tablas tipo fact, dimensión y Outtrigger.

Las nuevas fuentes de datos identifican el origen de las columnas y las nuevas reglas de utilización de datos capturan la forma en que los datos son mantenidos en el Warehouse. Erwin conjuntamente con ModelMart y mantenimiento del metal Data completamente integrado para Data Warehouse.

Promueve el diseño bajo estándares de la organización, además del nuevo soporte del diseño de Data Warehouse Erwin 3.5 es una herramienta de diseño que ofrece propiedades integradas por el usuario (UDPs) esta característica permite al usuario expandir Erwin y capturar directamente en el modelo, cualquier información adicional que sea importante para su negocio proporcionando así un nivel sin precedentes de personalización, por ejemplo La documentación externa tal como página Web u hojas de cálculo pueden ser asociadas a cualquier objeto en el modelo de manera de obtener fácil referencia en cualquier momento.

Erwin incluye soporte para atributos independientes con el fin de aumentar el rehuso de diseño y los estándares de datos. Los nuevos nombres consistentes de atributos y definiciones a través de todas la B.D. Erwin también incluye capacidades volumétricas para asistir a los usuarios en la planificación de capacidad permitiéndoles estimar el tamaño físico y el crecimiento de una B.D. basándose en la información almacenada en éste modelo incluso antes de generar la B.D.

Con esta herramienta se aumentará la productividad en el diseño y mantenimiento de B.D. Tiene un nuevo soporte para el establecimiento en estándares de diseño y reutilización. Propiedades meta expandibles y modelado dimensional para el diseño y modelado de Data Warehouse. Además de

otras aplicaciones basadas en bases de datos relacionales en la cual se requiere un diseño interactivo e inteligente entre sus niveles físicos y lógicos.

### **8.3 SYSTEM ARCHITECT (Herramienta adquirida por el BBA)**

Posee un repositorio único que integra todas las herramientas y metodologías usadas. En la elaboración de los diagramas el System Architect conecta directamente al diccionario de datos, los elementos asociados, comentarios, reglas de validación, normalización, etc.

En el modelo DER/OOA se provee la creación automática de Foreign Keys, además de la reutilización de los datos contenidos en la biblioteca. A partir de los atributos, el System Architect también crea, importa y exporta esquemas de bases de datos, tablas, campos, reglas y triggers.

Proporciona una arquitectura abierta para diccionario de datos y enciclopedias, dando la posibilidad de interface con otros diccionarios a través de ficheros en formato Dbase 111 Plus y ASCII.

El System Architect posee control automático de diagramas y datos, normalizaciones y balanceamiento entre diagramas "Padre e Hijo", además de balanceamiento horizontal que trabaja integrado con el diccionario de datos asegurando la compatibilidad entre el modelo de datos y el modelo funcional.

Esta herramienta es considerada un UPPER CASE, que puede ser integrada a la mayoría de los generadores de código. Traduce módulos de entidad - relación en esquema para:

- Sybase
- DB2
- Oracle ó Oracle 7

- Ingress
- SQL Server
- RDB
- XDB
- Progress
- Paradox
- SQL Base
- AS400
- Otros

Posee esquemas de seguridad e integridad a través de contraseñas que posibilitan el acceso al sistema de diversos niveles, pudiéndose integrar a la seguridad de la red Novell o Windows NT de ser necesario. Posee también un completo HELP sensible al entorno.

Posee un modelo específico para ingeniería reversa desde la B.D. SQL más populares, la ingeniería reversa posibilita la creación actualización y manutención, tanto el modelo lógico como en su documentación. A través de ODBC el Systems Architect logra leer bases de datos y construye el modelo lógico y físico, alimentando su diccionario de datos con las especificaciones de las tablas y de sus elementos de datos, incluyendo las relaciones entre tablas y su cardinalidad.

Posee customización desarrollada por Choose Technologies para empresas que deseen emplear la técnica de análisis esencial como base de la metodología de desarrollo de sistemas MDS.

Este modelo posibilitara a la documentación del System Architect de una manera rápida e interactiva. La estructura de documentación fue modificada basándose en las técnicas de análisis

esencial utilizando todos los recursos y gráficos para tomar más fácil la diagramación y documentación de sistemas.

System Architect posee múltiples metodológicas para diseño y análisis, incluyendo:

- Análisis Estructurado(DFD): en los modelos De Marco/ Yourdon y Gane/Sarson.
- Análisis de tiempo real en el modelo Ward & Mellor
- Análisis esencial de sistemas
- Análisis orientado a objetos(OOA) en los modelos UML, Booch ('91 y '94), Coad/Yourdon, Rumbaugh, Shaaler/Mellor.

Diagrama de entidad relación(DER) en los modelos Peter Chen, James Martín, Bachnan o Booch, Gráfico de estructuras, diagramas de descomposición. Planeamiento estratégico de informaciones, entre otras.

Estas metodológicas para desarrollo de sistemas, proporcionan amplio soporte, para la construcción de modelo conceptual, funcional y operacional.

El System Architect, es una herramienta case de última generación creada específicamente para la arquitectura cliente servidor, por esto posee control total de versiones y de Access, así como la administración completa de múltiples equipos de desarrollo. Independientemente de la topología de la red de comunicación, System Architect es operable sobre Novell, Windows NT, OS2 y posee interfaces específicas con diversos utilitarios " Front End" como power builder, Visual Basic, Sql Windows, etc.

El System Architect posee una arquitectura abierta que posibilita la customización, la catalogación y la manipulación de pantallas, informes y datos.

Así mismo posee un modulo para prototipación automática de sistemas que posibilita la elaboración de GUI( Graphic User Interface) o pantalla de caracteres las cuales pueden ser reaprovechadas de los sistemas.

Esta herramienta posibilita:

- El aumento de la productividad en el desarrollo de sistemas
- Una mejora de la calidad del software producido
- La aplicación de nuevas técnicas de ingeniería de información y reusabilidad.
- Automatización y estandarización de la documentación del sistema.
- Aumento de la interacción con el usuario en la definición del sistema.
- Liberación del analista, para concentrarse en la parte creativa del desarrollo de sistemas.

#### **8.4 SNAP**

Es una CASE para el desarrollo de aplicaciones en el sistema AS400 de IBM. Proporciona el ambiente integral de trabajo brindando la posibilidad de construir sistemas de inmejorable calidad a los estándares S.A.A. de IBM, totalmente documentados y ajustados a los requerimientos específicos de la organización, en una fracción de tiempo y coste del que se invertiría si se utilizaran herramientas tradicionales.

Snap se ha considerado como el CASE más poderoso y con mejor historial de resultados, genera los programas nativos de mejor rendimiento, proporciona dos ambientes de trabajo y genera aplicaciones nativas y/o Clientes/Servidor con el mismo esfuerzo de desarrollo.

## *Características*

### **Genera aplicaciones de alta calidad**

Las aplicaciones que genera el Snap son de alta calidad, 100% nativas de AS400, totalmente documentadas y libres de errores, los cuales se ajustan a los estándares S.A.A. (Arquitectura Abierta del Software) de IBM.

### **Produce resultados a muy corto plazo**

La rápida curva de aprendizaje, la infraestructura de soporte y las asesorías profesional disponibles en 14 países, hacen que las organizaciones obtengan el máximo provecho de la inversión con Snap a corto plazo.

### **Apoyo de etapas en Análisis y Diseño**

Snap se integra directa y naturalmente, con herramientas de análisis y diseño tipo Upper Case, tales como V.A.W. (Visible Analyst Workbech) permitiéndole a la organización ampliar la cobertura de desarrollo y mantenimiento de aplicaciones bajo ambiente CASE.

### **Arquitectura**

En una arquitectura Snap implementa de manera adecuada el esquema metodológico de entidad - relación, facilitando las herramientas y guías necesarias para construir aplicaciones que exploten al máximo las virtudes y potencial del AS400 en su modalidad nativa, siguiendo los alineamiento técnicos y presentación que propone la filosofía S.A.A de IBM.

Snap se compone de 4 grandes áreas:

- Modelo de datos
- Método de desarrollo Acelerado MDA

- Utilitarios
- Seguridad

### **Modelo de Datos**

En el modelo de datos el analista introduce el diseño conceptual o representación de la estructura de la información de la aplicación, siguiendo, paso a paso la metodología entidad - relación. Este diseño puede ser transferido directamente desde otras herramientas o bien tecleado directamente en Snap, además, prevee la facilidad de incorporar o trasladar DDS's existentes al modelo de datos.

Los pasos del Modelamiento son:

- 1° Identificar las entidades que participan en el sistema
- 2° Definir interrelaciones existentes entre ellas
- 3° Describir los atributos de cada entidad

Luego de haber introducido el modelo de datos Snap genera automáticamente la B.D. al mismo tiempo el analista sin necesidad de introducir ni una línea de código queda habilitado para especificar y generar la base de programas que dan mantenimiento integral al modelo, incluyendo aquellos de soporte de consulta, integridad referencial, navegación por lista e informes necesarios, para conformar un sistema

Todos los programas generados en el modelo de datos quedan totalmente funcionales, sin embargo el analista tiene la posibilidad de ajuste por medio del modelo de desarrollo acelerado (MDA), segunda gran componente del Snap

### **Método de Desarrollo Acelerado (MDA)**

Es una plataforma de trabajo para ajustar, en forma individual, los programas generados automáticamente en el modelo de datos, permite crear y generar programas nuevos.

Al igual que con el modelo de datos, al trabajar con el MDA el analista se sirve estratégicamente de los elementos del repositorio central y lo actualiza con el resultado de su gestión.

Además proporciona la herramienta necesaria para ajustar programas individuales, con un altísimo grado de productividad y rendimiento, sin necesidad de recurrir a lenguajes tradicionales. Esta herramienta incluye entre otras un modelo de especificación de procesos y lógica, y un formateador de informes

### **Utilitarios**

Estos utilitarios ayudan al analistas a administrar y controlar el proceso de desarrollo de aplicaciones.

Entre las funciones que se ofrecen están: herramientas para definir los estándares de la organización en Snap, utilitario de regeneración automática de sistemas, utilitarios de administración y control de repositorio, comandos para salvar, restaurar y recrear sistemas o elementos de repositorio central, documentación integral, y una gama de misceláneos de apoyo a la gestión de los analistas

### **Seguridad**

Snap incorpora un ambiente muy sofisticado para controlar y ayudar a la administración de procesos de desarrollo de sistemas. Se soporta los elementos necesarios para proteger hasta en 5 niveles las distintas definiciones y recursos del repositorio central.

### ***Snap Cliente/Servidor.***

Es un modulo adicional que permite al desarrollador construir aplicaciones (nativas o cliente/servidor) trabajando en un ambiente gráfico (Windows).

## ***Aplicaciones Generadas***

### **Generación de la aplicación en dos versiones**

Uno de los grandes beneficios de Snap C/S, es habilitar a los desarrolladores la posibilidad de generar una misma aplicación en dos entornos partiendo del mismo modelo de datos y especificación de programas automáticos. Ahora es posible generar una versión basada enteramente en AS400 con el Snap tradicional y otra bajo ambientes cooperativos C/S

### **Entorno de Trabajo de Alto Nivel**

Snap C/S selecciono a MS Windows como ambiente clave y estratégico para la implantación del entorno de trabajo del cliente. MS Windows es hoy en día el estándar de la industria y plataforma de trabajo en ordenadores personales de mas instalaciones de productos de software disponible en el mundo entero. Próximamente Snap generará también Java que permitirá la ejecución de aplicaciones de AS/400 desde navegadores.

### **Reducción de Carga de Trabajo**

Al liberar el sistema AS/400 del manejo de pantallas y control de ciertas validaciones "locales" y asignarla a ordenadores personales, se reduce sustancialmente la carga de trabajo sobre el procesador central así como proveer de mayor funcionalidad y amigabilidad a las aplicaciones en el cliente.

### **Consulta Gráfica**

El programa de consulta gráfica (CONGRA) permite generar en forma automática interfaces gráficas utilizando los datos almacenados en el servidor AS/400. Se cuenta con la disponibilidad de desplegar los datos en múltiples formatos, tales como barras, líneas o tartas en dos o tres

dimensiones y representando valores porcentuales o absolutos. Este programa es sumamente útil en todos los niveles empresariales, y es clave para la toma de decisiones y evaluación de rendimiento

### **8.5 VISIBLE ANALYST WORKBECH (V.A.W.)**

Es una herramienta CASE para análisis y diseño de sistemas, es decir es de tipo UPPER CASE. Se presenta como una serie de herramientas que se pueden operar con bases de menús y corren bajo un ambiente de microcomputadoras IBM y compatibles. El V.A.W. y sus herramientas poderosas de programación, análisis, Modelamiento de datos y prototipos, aumentan no solo la productividad en el desarrollo de sistemas sino que también la eficiencia global en el análisis y el diseño. Adicionalmente la facilidad de uso permite que los usuarios se involucren en el proceso de desarrollo de sistemas.

Por medio del V.A.W. los usuarios finales pueden comunicar eficientemente su requerimientos objetivo y modificaciones definidas.

Los principales componentes para el ambiente de trabajo se describen a continuación:

#### **Herramienta de Diagramación:**

Esta herramienta es utilizada pro medio del Mouse, la cual le permite dibujar virtualmente cualquier tipo de diagrama. Los símbolos estándar, líneas de texto, combinaciones con las características de edición manejadas por medio de menús permiten crear diagramas relacionados con cualquier metodología o regla

#### **Las Reglas:**

Estas dependen de la metodología y se encuentran incorporadas dentro de las herramientas de Modelamiento de procesos y datos para brindar una guía y proveer consistencia a sus actividades.

Las metodologías soportadas están codificadas en reglas las cuales se aplican mediante se dibujan los diferentes ítems en los diagramas o cuando el usuario la solicita aplicando la función ANALIZE.

### **El repositorio:**

Se incluye una base de datos que actúa como el repositorio central para toda la información perteneciente de cada proyecto este repositorio esta relacionado con los modelos de procesos y herramientas de Modelamiento de datos, para proveer soporte de los diagramas que son creados bajo las guías de las metodologías soportadas, todas las adiciones y modificaciones a los diagramas se encuentran reflejadas en el repositorio.

### **Modo de operación:**

Soporta las siguientes etapas del desarrollo de sistemas:

Modelaje de datos:

- Generación automática de diagramas entidad - relación
- Notación bachman o equivalente
- Normalización de datos automática y en línea
- Balanceo de elementos de datos y Entidad vrs. Procesos

Análisis Estructurado:

- Balanceo automático y en línea de flujo de datos
- Nivelación automática
- Revisión de completitud
- Descomposición jerárquica sin límites de niveles
- Numeración de procesos automática

- Soporte de metodología top down
- Diccionario de datos integrados y probado automáticamente
- Generación automática de diagramas de descomposición
- Metodología de Tom de Marco

#### Diseño Estructurado

- Metodología de Yourdon
- Revisión sintáctica y de completitud
- Integración con la etapa de análisis estructurado
- Diccionario integrado y probado automáticamente
- Capacidad de medición de calidad de diseño

## 8.6 EXCELERATOR

Es un instrumento de ingeniería del software desarrollado por INDEX TECHNOLOGIES, apoya a los analistas de sistemas para mejorar su productividad y para comunicarse mejor con otros analistas y usuarios finales. Ha logrado su aceptación ya que:

- Es un instrumento integrador que se apega a todos los métodos de diseño y análisis en un solo paquete
- Esta disponible para la IBM PC y ciertas compatibles
- Su filosofía permite que los analistas representen cualquier cosa en la pantalla, pudiendo distanciarse de ciertas reglas.

#### **Funciones:**

**Gráficas:** Permite a los analistas elaborar diagramas que apoyan a la productividad de los analistas en el desarrollo de sistemas. Entre estos tenemos: Diagramas de flujos de datos, diagramas

estructurados, modelos de datos lógicos, diagramas de entidad - relación, diagramas de estructuras, gráficas de presentación.

**Diccionario XL:** Se encuentran obtenidos los elementos de datos, tales como el registro de los datos, los flujos de los datos, las tablas de los códigos y las relaciones de los datos. Los procesos funcionales en los diagramas de las estructuras y entidades externas, los módulos y las gráficas de presentación también pueden almacenarse en el diccionario.

**Pantallas y Reportes:** El analista es capaz de preparar una muestra de salida para pantallas o reportes impresos.

**Análisis:** Exceletor permite que el analista dibuje casi todo. Esto es una ventaja ya que permite que el analista siga cualquier patrón de pensamiento, mas que forzarlo a conceptualizar en una manera predeterminada.

La utileria análisis contiene 4 reportes para verificar el trabajo gráfico:

- Reporte de verificación, el cual lista cada conexión ilegal y cada objeto no conectado del diagrama de flujo de datos.
- Reporte de análisis, el cual lista el flujo de los elementos datos para cada uno de los 4 procesos y cada almacén de datos y lista los datos capturados, los de salida o derivados de ellos.
- Reporte explosión, el cual crea una tabla jerárquica del contenido, que se utiliza para detectar explosiones faltantes o inconsistencias en las definiciones.
- Reporte de balance entre niveles, el cual compara los niveles en el diagrama de flujo de datos para verificar que el flujo de datos de un nivel tenga una contraparte en el siguiente nivel superior.

**Interface XLD:** Este modulo permite que el analista importe y exporte datos de otros diccionarios de datos. Estos se extienden a la transferencia de datos entre otras implementaciones del Excelerator o de computadoras centrales.

**Documentación:** la utileria de documentación prepara la salida impresa de todas las gráficas y diccionarios en las que halla trabajado el analista.

**Mantenimiento:** La utileria de mantenimiento permite establecer la configuración de sistemas mantener proyectos y asignar privilegios de acceso.

## 8.7 DESIGNER/2000

### Versión 2.1

El designer/2000 marca un significativo salto desde la mayoría de modelamiento de aplicaciones hasta un punto en que se vuelve una herramienta invaluable que los departamentos de informática podrán utilizar diariamente para diseñar y utilizar sus tareas. Es una herramienta que incluye soporte para las Bases de Datos de Oracle y las de cualquier otra compañía que se les parezca. También soporta la generación de opciones para las aplicaciones, así como para integrar componentes de terceros como: Visual Basic.

Además Soporta la generación de aplicaciones Web, en la forma de html dinámica, ésta solo puede ser soportada vía un servidor de aplicaciones. Ya que no se encuentran opciones para generar aplicaciones en Java.

Aunque las capacidades de ésta versión no eliminaran el código de desarrollo, las herramientas, el diseño expandido y el soporte de desarrollo deberían reducir en forma

significativa la cantidad de tiempo y esfuerzo que ocupan los grupos de creación de proyectos.

#### Características del Designer/2000 Vr. 2.1

- Modelamiento de aplicaciones basada en el repositorio
- Análisis y Diseño orientado a objetos
- Proceso reversibles de reingeniería
- Soporte expandido nativo para bases de datos

#### Bases de Datos que soporta Designer/2000 Vr. 2.1

- Oracle 7, Oracle 8, Oracle Lite, Oracle Rdb,
- DB2 de IBM
- Microsoft SQLServer
- Soporte ODBC para otras Bases de Datos

#### Generación Automática de Aplicaciones entre las que tenemos:

- HTML Dinámico
- Developer/2000
- Visual Basic
- C++

#### Beneficios y Limitantes del Designer/2000 Vr. 2.1

##### ***Beneficios:***

- Generación Automática de Código
- Soporte para Data Store de otras compañías

- Excelente capacidad para revertir procedimientos ingenieriles
- Interfaz simplificada

***Limitantes:***

- Aplicaciones de HTML Dinámico Limitado por el momento
- No posee soporte para Java

**Versiones Anteriores**

Designer/2000 soporta el diseño de complejos de sistemas con reingeniería de procesos de negocios (BPR), análisis y diagramadores de diseño. Con un repositorio común soporte de metodologías y diseño flexible, un entorno de desarrollo de servidor y cliente unificado, y una arquitectura portable abierta se constituye como la única herramienta de desarrollo y diseño de cliente/servidor de segunda generación en la industria.

Permite a los usuarios desarrollar sistemas con amplio rango de complejidad . Las herramientas C.A.S.E. de Oracle evolucionaron hacia Designer/2000 la cual posee los componentes siguientes:

- ***Desarrollo de escala empresarial***

Disegner/2000 capacita a la organización para brindar soluciones a las prioridades corporativas de negocios. Todo el personal experto en reingeniería de procesos de negocios (BPR), analistas de negocios, desarrolladores de aplicación y diseño de sistemas, obtienen la sinergia del proyecto en un repositorio compartido complementando por un juego integrado de herramientas de diseño y generadores de sistemas. Diseña y proporciona sistemas precisos y oportunos de cliente/servidor empresariales adaptables a las cambiantes

necesidades de depósitos de datos y aplicaciones de negocios.

- ***Reingeniería de procesos y negocios***

Las herramientas de Designer/2000 BPR se utilizan para analizar procesos fundamentales de negocios mediante técnicas centralizadas en la administración, construye modelos de flujos de procesos de multimedia que incluye iconos, imágenes, sonidos y videos para captar definiciones de proceso de negocios y su alineación de unidad de negocios. La animación de procesos da vida a los modelos de procesos con escala en el tiempo y ilustrando problemas y oportunidades que desafían a su organización. Con las técnicas BPR, las organizaciones obtienen una profunda comprensión de las oportunidades y métodos de los negocios para seguirlos.

- ***Sistemas Visuales y Modelaje de Diseño***

Se pueden crear con los modelos basados en BPR ó nuevos directamente con los sistemas Designer/2000 para diseñar componentes de modelaje. Los diagramas de modelaje de función e información para construir un modelo que capte las necesidades de los usuarios y los negocios mediante el registro de estructura lógica y física requeridas en los sistemas de depósito de datos y transaccionales.

- ***Desarrollo de sistemas dirigidos por modelos***

Utiliza los modelos de requerimientos dirigidos por repositorio para acelerar la construcción y mantenimientos en proceso de los sistemas de producción. Para las aplicaciones Developer/200 y Visual Basic la configuración GUI, el acceso a la BD, la lógica de servidor y la lógica de aplicación, se crean automáticamente incorporando particionamiento de

aplicación flexible para procesamiento óptimo de C/S. Emplea planillas bien definidas para asegurar interfaces de usuario de aplicación consistente a nivel empresarial que adopten los estándares de configuración, de aplicación y presentación nativa de GUI.

- ***Repositorio abierto***

Integra los repositorios y herramientas que no son de Oracle en el entorno Designer/2000. Utiliza la interfaz de programador de aplicaciones (API) abierta de Designer/2000 para integrar herramientas y aplicaciones de terceros. Emplea Oracle Case Exchange para incorporar o exportar información de modelo de proceso, función y entidad a otros marcos y herramientas de repositorio conocidos.

### **Generalidades de Designer/2000**

- Modelaje de procesos: Visualiza y mejora radicalmente los procesos fundamentales de negocios con las capacidades de modelaje de proceso del designer/2000. Obtiene una considerable ventaja competitiva, reduce costos y mejora la calidad mediante la comprensión de dependencia entre organizaciones y duraciones del ciclo del proceso. Identifica unidades propuestas y existentes de organizaciones de múltiples niveles y visualiza luego en (SWIM LANES) Intuitivos. Aplica Iconos para identificar pasos de procesos y las iteraciones e ingresa parámetros de costo y tiempo para cada paso de proceso.

Utiliza animación de procesos y multimedios para revelar problemas y oportunidades a la administración y los usuarios a medida que observa toda la ejecución de pasos de procesos de principio a fin a un entorno de escala en el tiempo.

- Modelaje de sistemas: El Designer/2000 simplifica el rápido desarrollo dirigido por modelo de poderosos e intuitivos métodos de modelaje de estándar de industria. Utiliza técnicas de revelación de entidades, jerarquía de funciones, flujo de datos, y modelaje de matriz para capturar la estructura y las interrelaciones de todos los objetos del sistema. Mediante el repositorio del Designer/2000, controla el compartir y reutilizar los objetos de aplicación a través de múltiples proyectos. Utiliza el entorno intuitivo de ventanas para integrar conocidas aplicaciones de desktop con los diagramas y publica los diagramas a través del object linking and Embedding (OLE2).

Además emplea un navegador de objetos de repositorio y despliega avanzados diagramas de toda la estructura del repositorio. Mejora la productividad y equipo a través de la reutilización de objetos de drag and drop, la fácil modificación de relaciones y propiedad e informes totalmente documentados en la estructura y contenido de repositorio.

La claridad y la facilidad de uso de las herramientas de modelaje de sistemas combinadas con las sofisticadas instalaciones de navegación, aseguran un sólido cimiento para el diseño e implementación de soluciones de depósitos de datos y aplicaciones.

- Diseño de sistemas: El Designer/2000 se utiliza para implementar los sistemas distribuidos de C/S y para aplicaciones de depósito. Toma INPUT directamente de los desarrolladores y utiliza Wizards de diseño para derivar rápidamente diseños de sistemas de la definición de requerimiento. La estructura de datos, y los diseños de programas de Developer/2000 y Visual Basic se materializan rápidamente y con

precisión.

Utiliza una combinación de diagramadores y navegadores para crear y manejar detalles de tablas, claves foráneas y reglas de validación de datos; y define triggers basados en servidores junto con las condiciones, funciones y procedimientos de ejecución (triggering). El Designer/2000 facilita el particionamiento de código de C/S. Se pueden registrar las reglas declarativas como implementaciones de cliente, C/S ó Servidor en preparación para la generación de aplicaciones equilibradas de C/S. Controla al diagramador de estructura de modulo para definir la estructura de aplicaciones, sus módulos de programas de componentes y las interacciones entre los mismos. Aplica el navegador de lógica de módulos para la gráfica e inteligente construcción y modificación de definiciones de procedimientos dirigida por sintaxis. El diagramador de datos de módulo define fácilmente las interacciones entre programas individuales y los datos a los que acceden con el uso de ventanas y páginas. Modifica las características de acceso de datos de módulos mediante la definición de operaciones autorizables de selección, inserción, actualización y supresión y el agregado de especializaciones específicas de modulo de diseño generales.

Utiliza navegador de referencia y plantillas para definir las características generales y visuales y semánticas de los sistemas generados que elimina tiempo dedicado a pintar pantallas y define una presentación estándar de todas las aplicaciones.

Designer/2000 provee las herramientas que necesita para diseñar e implementar rápidamente sistemas de C/S y aplicaciones de depósito y para responder las

cambiantes necesidades de negocios.

- Generación de aplicación de cliente: Utiliza Designer/2000 para construir automáticamente aplicaciones Developer/2000. Las aplicaciones generadas incluyen configuraciones de pantallas múltiples, informes sofisticados, navegación a través de menús, botones y listas desplegables, lógica de aplicación del lado del cliente y funcionalidad total de acceso a la Base de Datos; todo con origen en los diseños de tablas, modulo de programas y preferencia almacenados en el repositorio. Las plantillas dirigen la apariencia de una gran variedad de controles de GUI en pantallas generadas, tales como imágenes, diálogos, listas desplegables, grupos de radio, casilleros de control, botones de iconos y otros. Además genera aplicaciones que no solo corren en múltiples plataformas con presentación nativa completa, sino que también explotan las funciones nativas de cada plataforma tales como controles OLE2 y VBX. Aprovecha su inversión generando configuraciones de GUI, lógica de acceso a Bases de Datos y código de administración de transacción utilizando el mismo modelo de diseño mantenido por repositorio. Provee productivamente interfaces comunes de usuario utilizando este enfoque de generación abierta desde un repositorio de diseño a nivel empresarial.
- Generación de servidor: Genera no sólo aplicaciones del lado del cliente sino que también completa la entrega de aplicaciones empresariales y soluciones de deposito con la creación de todos los componentes del lado del servidor. ANSI, SQL, DDL, pueden crearse desde el diseño de servidor mantenido por repositorio, incluyendo las estructuras de tablas y columnas, la validación del lado del servidor de restricciones de claves foráneas, condiciones de control y claves primarias.

Además para Oracle 7 Designer/200 crea definiciones que dirigen SNAPSHOTS, el modelo de seguridad basado en rol, lógica de aplicación del lado del servidor en triggers, procedimientos y funciones almacenadas y parámetros de almacenamiento de datos. Soporta el desarrollo de modelos de implementación distribuida a través del particionamiento de componentes de aplicaciones en servidor y nodos específicos. Estas definiciones se crean luego en un Form conveniente para la distribución de esas ubicaciones junto con todas las definiciones de red requeridas para acceder a servidores distribuidos. Se dispone de ingeniería reversa para otorgar a los desarrolladores y administradores acceso a las sofisticadas herramientas de diagramación para la administración y mantenimiento de servidores distribuidos. Designer/2000 se utiliza para convertir rápidamente diseño de sistemas distribuidos a BD en funcionamiento así como también manejar efectivamente los servidores existentes.

- Administración de Repositorio: Designer/2000 brinda un abarcativo juego de instalaciones para administrar los contenidos y el acceso al repositorio de Designer/2000. Divide y subdivide definiciones a través de múltiples proyectos y otorga a los usuarios privilegios que perfilen exactamente sus necesidades del acceso. Promueve la reutilización de objetos a través del entorno de desarrollo proporcionando definiciones compartidas a través de múltiples proyectos.

Rastrear y administrar precisamente múltiples sistemas de desarrollo y producción a través de poderosas instalaciones de control de versión. En entornos que requieren múltiples implementaciones de repositorio de Designer/2000, explota las funciones de extracto, carga, y fusión para administrar los desarrollos distribuidos.

Designer/2000 provee las sofisticadas y aun así fáciles de usar instalaciones de administración de repositorios que necesita para asegurar la explotación de bajo costo y alto valor del entorno de repositorio integrado.

- Administración de la Complejidad del Diseño de Cliente/Servidor: Designer/2000 se pone a la altura de los desafíos de las exigencias de la industria mediante la rápida creación de sistemas de cliente/servidor escalables. Además, a medida que su portafolio de aplicaciones basadas de aplicaciones Designer/2000 crece, valor del el del portafolio de aplicaciones basadas en Designer/2000 crece, el valor de los objetos reutilizables en el repositorio compartido se sumará al éxito de los esfuerzos de depósito y desarrollo de sistemas.

### **Características claves :**

#### **Diseño de cliente Servidor**

- La integración directa entre diagramadores y repositorios asegura la integridad de los modelos y la disponibilidad inmediata a otros miembros del equipo de desarrollo.
- El navegador de objetos de repositorio provee una manera altamente intuitiva de visualizar y manipular objetos de repositorio.
- Los diálogos claros y de fácil uso y las hojas de propiedad captan información no gráfica y semántica para completar definiciones de modelos de negocios.
- Las múltiples interfaces de documentos permiten el trabajo simultáneo en más de un modelo.

- La integración de OLE 2.0 permite a los usuarios combinar conocidas herramientas de desktop con técnicas de modelaje.
- Se provee soporte de idioma nacional completo, único y de múltiples bytes.
- Documentación y ayuda total sensible al contexto disponible.
- Rango abarcativo de impresión y otras opciones de output disponibles.

### **Modelaje de Procesos**

- Soporte flexible para técnicas BPR
- Integración con herramientas de modelaje de sistemas
- Representación gráfica de alto impacto de organizaciones y sus procesos
- Captura de métricas de procesos de negocios tales como tiempo, costo y producción para autorizar modelaje preciso de procesos de negocios actuales y futuros.
- Ejecución de procesos de negocios incluyendo los componentes de multimedios mediante animación.
- Soporte de multimedios
- Integración de desktop incluyendo interface de plantilla de cálculo (Lotus 1-2-3)

### **Diagramador de Relaciones de Entidades**

- Soporta la construcción de modelos de relaciones de entidad utilizando entidades y atributos, relacionales, UIDs.

- Subtipos y supertipos de entidades
- Atributos, identificadores y sinónimos visibles en diagramas
- Instalación de configuración automática sofisticada
- Utilitarios e informes de control de calidad extensivos

### **Modelador de Jerarquía de Funciones**

- Soporta la construcción de jerarquía de funciones.
- Drag and Drop para resecuenciación y re-parenting de funciones.
- Reducción, expansión de ramas de jerarquía para mayor claridad.
- Opciones de configuración híbridas, verticales y horizontales.
- Módulos de información abarcativo incluyendo análisis de punto de función.
- Controles de calidad.

### **Modelador de Flujo de Datos**

- Soporta la construcción y control de los modelos de flujo de datos incluyendo funciones, almacenes de datos, flujos de datos y entidades externas.
- Selección y navegación por árbol, rápida a través de jerarquía de funciones
- Equilibrio y nivelación automática de flujo a través de modelos
- Flujos de datos divididos y resueltos

- Despliegue de propiedades de funciones, flujos de datos y almacenes de datos en modelos.

### **Diagramador de Matriz**

- Soporte de referencia cruzada abarcativa de objetos mantenidos por repositorios
- Filtra la selección y refinamiento de objetos de repositorio
- Preferencia de matriz personalizable
- Presentación de planilla de cálculos
- Ordenamiento automático de datos desplegados
- Soporte para definiciones de repositorio extendidas

### **Wizards de Diseño**

- El Wizards de diseño de BD automático crea un esquema de BD de primer corte que incluye tablas, columnas, índices y restricciones referenciales de integridad.
- El Wizards de diseño de modulo de aplicación automático toma la información de modelos funcionales y de flujo de datos, y crea completa definiciones de módulos para pantallas, informes y menús para su revisión y trabajo adicional de diseño para generación de código.

### **Diagramador de Datos**

- Poderosas técnicas de modelaje de base de datos.
- Selección de apuntar y hacer click de reglas de particionamiento de C/S para generación

de código.

- Instalación sofisticada de configuración automática
- Representación de iconos de funciones de diseño de bases de datos.

### **Diagramadores de Datos de Módulo**

- Definición de alta productividad de acceso a datos y requerimientos de uso de módulos de programa.
- Configuración de diagrama productivo e intuitivo.
- Instalación sofisticada de configuración automática.
- Integración directa con otras herramientas de generación y modelaje de Designer/2000.

### **Diagramador de Estructuras de Módulo**

- Definición rápida y flexible de interdependencia entre módulos de programas.
- Integración directa con otras herramientas de generación y modelaje de Designer/2000.

### **Navegador de Lógica de Módulos**

- Guía completa de estilo GUI en línea.
- Poderosa representación jerárquica de preferencias de GUI y de codificación.
- Rango de tipos de objeto de diseño para el que pueden fijarse preferencias.
- Juegos de preferencias nombrados para reutilización de standards.

- El congelamiento de nivel de módulos evita que el módulo generado sea reconstruido con diversas preferencias.
- Completos valores por defecto de preferencias se envían para su uso inmediato.

### **Funciones de Generación de aplicación Developer/2000**

- Completa instalación de generación de informes y forms de Developer/2000 para inteligentes generación de código GUI dirigida por modelo.
- Genera aplicaciones de alta funcionalidad, portables, escalables.
- Las preferencias automatizan la guía de estilo de GUI para obtener consistencia y productividad.
- Integración estrecha con herramientas de diseño de sistemas para programación gráfica.
- Ingeniería reversa de definiciones existentes de forms e informes del repositorio.
- Las sofisticadas operaciones de agrupación y resumen pueden definirse en el repositorio para la incorporación de informes.
- Los controles VBX y OLE 2.0 pueden incluirse en el código generado.

### **Generación de Visual Basic**

- Modelo de diseño común tanto para generación Visual Basic como para Developer/2000.
- Creación completa de configuración GUI y SQL DML.
- Crea todo código de administración de transacciones, navegación y validación de datos.

- Acceso optimizado para servidores Oracle 7
- Integrado con el código servidor de Oracle 7

### **Funciones de Generación de Servidor**

- Finaliza la entrega de aplicaciones empresariales creando todos los componentes del lado de servidor automáticamente de las definiciones mantenidas por repositorio.
- Información completa de tablas y columnas justo con validación de lado de servidor de restricciones de claves foráneas, condiciones de control, claves primarias, etc. Se crea DDL, SQL standar de ANSI.
- Se generan definiciones de Oracle 7 de lógica de aplicación de lado del servidor(PL/SQL), incluyendo triggers, procedimientos y funciones almacenadas, y los parámetros de almacenamiento.
- Soporte completo para modelo de seguridad basado en rol de Oracle 7.
- También se crean definiciones de replicación de la información que se guarda dentro del repositorio de Designer/2000.
- Se registran las definiciones de bases de datos para la implementación en servidores y nodos específicos
- Provee definiciones de una manera conveniente para lograr distribución a las ubicaciones distribuidas y crea todas las definiciones de red requeridas.
- Ingeniería reversa de definiciones existentes de bases de datos.

## **Soporte de Depósito de Datos**

- Modela los requerimientos de negocios para el depósito de datos a nivel de negocios, sistemas o diseño.
- Genera DDL SQL para la implantación de depósito.
- Genera aplicaciones estructuradas para acceso de depósito utilizando Developer/2000 o Visual Basic.
- Completa Discoverer/2000 End User Layer desde el repositorio Designer/2000.

## **Funciones de Administración de Repositorio**

- Instalaciones abarcativas de administración de repositorio para DBAs de desarrollo.
- Reutilización y administración de múltiples datos de proyecto.
- El navegador de repositorio brinda una forma altamente intuitiva de administración objetos de repositorio.
- Control de versión de objetos de sistemas de aplicaciones.
- Control de acceso.
- Poderosos utilitarios soportan sofisticados entornos de repositorios distribuidos con Extract, Load y Merge.
- Funcionalidad check-in, check-out.
- API completo de repositorio que utiliza procedimientos almacenados en el servidor de

Oracle7.

- CASE Exchange soporta el movimiento bidireccional de las definiciones entre el repositorio Designer/2000 y muchos otros productos de CASE.
- La extensibilidad de usuario permite que los contenidos del repositorio sean personalizados de modo que cumplan con las necesidades específicas de desarrollo.

## **8.8 EASY-CASE**

Es una herramienta CASE que cubre las fases de análisis y diseño. Su arquitectura está montada sobre dos productos: EasyCASE Professional y EasyCASE DBE. El primero de ellos automatiza las técnicas soportadas en las fases de análisis y diseño (diagrama de flujo de datos, modelo entidad - relación, diagrama de estructuras, historia de la vida de la entidad, etc.), mientras que el segundo permite la generación de esquemas de bases de datos e ingeniería inversa de datos mediante conexión vía ODBC.

Las características principales que hacen de EasyCASE una herramienta diferente son:

- Sencillez de manejo (menos de cinco horas de aprendizaje)
- Contenido público del repositorio (fácil integración con otros productos)
- Soporte de múltiples métodos y técnicas (flexibilidad de implantación).
- Bajo coste.
- Posibilidad de correr bajo DOS o Windows.

La integración de EasyCASE - MultiBase nace de la búsqueda de una solución sencilla, con

costes razonables y de fácil implantación a una serie de herramientas que cubran el ciclo de vida completo.

### **Apertura del Diccionario**

Una de las características fundamentales de EasyCASE es la apertura del diccionario a la integración con otros productos, gracias a la cual ha sido posible su integración con MultiBase.

Esta integración se realiza no sólo desde el aspecto físico, sino también desde el lógico, ya que con ello el desarrollo va ligado totalmente al uso de una metodología y unas técnicas estructuradas. La integración EasyCASE - MultiBase permite, además, que esta metodología sea Métrica v2, ya que es una de las soportadas por EasyCASE, posibilitando de esta forma su utilización.

El ámbito que se pretende alcanzar con esta integración es bidireccional, de tal manera que los cambios que se realicen a lo largo del desarrollo, independientemente de donde se produzcan, se actualicen en el repositorio común.

Esta integración abarcará distintas etapas, como son la creación de un prototipo, una interfaz de acoplamiento débil, una siguiente conexión de los repositorios comunes y, por último, la incorporación del camino de retorno entre los dos productos.

EasyCASE DBE, junto con EasyCASE Professional, constituye una herramienta de ayuda de gran valor en instalaciones que dispongan de MultiBase como gestor de BD.

EasyCASE dispone de capacidad para definir las estructuras de las tablas y de los campos de las BD, definir índices primarios y secundarios, incluyendo índices únicos, y establecer su orden, así como el de las opciones para la generación de esquemas MultiBase, y para añadir restricciones a

nivel de registro.

EasyCASE permite elegir las entidades a procesar de un diagrama entidad - relación en el proceso de la generación de esquemas. Con ello se pueden crear vistas del modelo.

## **8.9 POWER DESIGNER**

### **8.9.1 Alcance de la herramienta**

PowerDesigner es una herramienta CASE, que abarca las siguientes fases del ciclo de vida de una aplicación:

- Análisis
  - De Datos
    - Diagramas de Entidad - Relación ( Modelo Conceptual ).
  - De Procesos.
    - Diagramas de Flujos de Datos.
- Diseño
  - Diseño de bases de datos.
- Implementación
  - Creación de bases de datos.
- Mantenimiento
  - Mantenimiento de bases de datos
  - Ingeniería Inversa
  - Reingeniería.

### 8.9.2 Características Generales

PowerDesigner está formado por tres módulos independientes entre sí. Dos de estos módulos están dedicados al análisis DataArchitect y ProcessAnalyst. El tercer módulo, MetaWorks, está dedicado a la gestión de los proyectos analizados con los dos módulos anteriores.

A continuación se detallan las características de los distintos módulos :

#### **METAWORKS**

MetaWorks permite la gestión de distintos proyectos concurrentemente.

MetaWorks puede generar diversas Base de Datos como: Informix, SQL Server, Sybase SQL Anywhere, ..., mediante una aplicación anexa llamada SYBASE CENTRAL.

Se generan dos ficheros con extensión .DB y .LOG.

Una vez creada la base de datos, desde el MetaWorks vía ODBC, se realiza la conexión con la misma.

Dentro de una mismo tipo de base de datos, podemos crear distintos diccionarios de datos atendiendo a los diversos proyectos que tengamos que gestionar.

MetaWorks que permite a varios usuarios trabajar sobre el mismo proyecto de PowerDesigner.

Para ello, trabaja sobre un diccionario de datos, donde guarda toda la información de un proyecto.

Un diccionario de datos permite:

- La existencia de **grupos de trabajo** ya que permite que varios diseñadores compartan una aplicación.
- Asegura la **integridad de los datos**.
- Asegura la **consistencia y compatibilidad**.
- Permite **compartir datos** entre diferentes proyectos.

La posibilidad de crear un proyecto facilita la tarea de los grupos de trabajo. Un proyecto está formado por un conjunto de modelos físicos, lógicos y de procesos.

La integridad del diccionario de datos se controla de dos formas:

- Directamente: no permite algunas acciones incorrectas.
- A través del corrector: avisa de los errores semánticos, un objeto sin relaciones, balanceado

## **POWERDESIGNER DATA ARCHITECT**

### ***Características:***

DataArchitect es una herramienta muy poderosa para el diseño de bases de datos. Con esta herramienta CASE, podemos realizar las siguientes operaciones:

- Modelar un sistema de información utilizando un diagrama Entidad-Relación, llamado Modelo de Datos Conceptual
- Generar el correspondiente Modelo Físico de Datos , para un determinado gestor de base de datos (DBMS) , teniendo en cuenta las especificaciones del DBMS .
- Personalizar el Modelo físico de Datos (PDM) para satisfacer consideraciones físicas y de rendimiento.
- Generar una escritura de creación de base de datos para el DBMS designado.
- Personalizar e imprimir Reportes
- Generar Ingeniería Inversa de base de datos y aplicaciones .
- Definir los atributos extendidos para los objetos del Modelo Físico de Datos.

Una vez especificado el tipo de base de datos, con el que vamos a trabajar, podemos pasar del modelo físico al lógico, y a la inversa, directamente.

### **Modelo Lógico:**

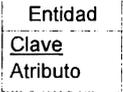
El modelo lógico representa los datos de una empresa, independientemente de su implementación.

En el caso de PowerDesigner, se realiza el diseño lógico según la notación de James Martin.

El modelo lógico se utiliza para:

- Representar los datos de forma gráfica.
- Verifica y valida el diseño de los datos
- Genera el modelo físico automáticamente según el sistema gestor de bases de datos definido.

### Elementos de un Modelo Lógico

OBJETO	DESCRIPCIÓN
Entidad 	Representa un elemento ( cosa, persona, concepto..), que interesa al sistema que estamos modelando.
Relación 	Relación entre entidades del sistema.
Atributos 	Parte elemental en un sistema de información que forman las entidades. Toda entidad debe de tener un atributo que la identifica univocamente y que denominaremos clave.

### Cardinalidad de las Relaciones

La cardinalidad, indica el tipo de relación existente entre dos entidades. Así como las restricciones de las mismas. PowerDesigner controla la cardinalidad, de manera que no aparezcan relaciones innecesarias. Las cardinalidades posibles son:

SIMBOLO	CARD MINIMA	CARD MAXIMA	DESCRIPCIÓN
	1	1	Debe de haber una y sólo una relación con la tabla.
	0	1	Como máximo sólo se puede relacionar con una tabla, pero la relación es opcional.
	1	M	Al menos debe de haber relación con una tabla, puede haber con muchas.
	0	M	Puede existir relación con ninguna o con muchas tablas.

La cardinalidad de una relación, indica el número de instancias de una tabla que se relacionan con otra. Por lo tanto, todas las tablas de una relación poseen su cardinalidad. Sin embargo, no todas las combinaciones de las cardinalidades tienen sentido y PowerDesigner lo controla. No se permite la relación R:(1,1)-(1,1) ya que no es una relación entre dos entidades diferentes, sino una entidad dividida en dos.

### Relaciones de Dependencia

Tendremos una relación de dependencia entre dos tablas, cuando una tabla no puede identificarse por sí misma y necesita algún atributo de otra tabla con la que está relacionada para obtener su clave.

Las relaciones de dependencia se representan:



Y deben de cumplir estas normas:

- No se permite dependencia en los dos sentidos
- No puede haber dependencia en una relación reflexiva (una entidad consigo misma )

### Reglas en un Modelo Lógico

Todo modelo lógico correcto debe de cumplir estas reglas:

- Toda entidad debe de tener algún atributo.
- Toda entidad debe de tener clave principal.
- No pueden existir dependencias circulares.
- Una entidad no puede tener restricciones de dependencia con ella misma.

### **Modelo Físico:**

Un modelo físico, representa la implementación física de una base de datos. El modelo físico se utiliza para:

- Modelar mediante un gráfico una base de datos.
- Generar una base de datos.
- Generar un modelo lógico.

Existen tres formas de generar un modelo físico:

- A partir de un modelo lógico.
- Directamente, de forma manual.
- A través de ingeniería inversa.

Objetos de un Modelo Físico

OBJETO	DESCRIPCIÓN
<p>Tabla</p> <div data-bbox="178 443 422 564" style="border: 1px solid black; padding: 5px; margin: 10px 0;"><p style="text-align: center;">Tabla</p><p>Clave Primaria &lt;pk&gt;</p><p>Clave Ajena &lt;fk&gt;</p><p>Columna</p></div>	<p>Representa a una tabla de la base de datos.</p>
<p>Referencia</p> <div data-bbox="155 659 447 764" style="margin: 10px 0;"><p>Referencia</p><p>Clave_Ajena = Clave_Ajena</p></div>	<p>Enlaces entre tablas, a través de las claves ajenas. Las referencias pueden tener un nombre, o pueden identificarse a través de los campos con los que se establece la relación.</p>

### CUADRO COMPARATIVO DE DIVERSAS HERRAMIENTAS CASE

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
System Architect	Upper Case	Una herramienta que automatiza los procesos de generación, manejo y organización de diversos modelos de diagramas	<ul style="list-style-type: none"> <li>* Diccionario de Datos</li> <li>* Generación de esquemas y creación de BD</li> <li>* Ingeniería Reversa</li> <li>* Análisis esencial</li> <li>* Soporte a metodologías</li> <li>* Arquitectura C/S</li> <li>* Productos Específicos</li> <li>* SA-BPR</li> <li>* Implementación de metodolo de desarrollo de sistemas</li> </ul>	<ul style="list-style-type: none"> <li>* Aumento de la productividad en el desarrollo de sistemas.</li> <li>* Mejora la calidad del software producido.</li> <li>* Aplica nuevas técnicas de Ingeniería de información y reusabilidad</li> <li>* automatiza y estandariza la documentación del sistema</li> <li>* Aumenta la interacción con el usuario en la definición del sistema</li> <li>* Liberación del analista para concentrarse en la parte creativa del desarrollo de sistema.</li> </ul>	<ul style="list-style-type: none"> <li>Cualquier PC</li> <li>* PC/AT IBM o computador compatible</li> <li>* Driver de alta densidad 3.5</li> <li>* Disco duro con al menos 16 MB en RAM</li> <li>* Un CD-ROM</li> <li>En ambiente WIN</li> <li>* 1 MB en RAM</li> <li>* SO/MSDOS 3.3. o ver. posterior.</li> <li>* Una pantalla graficadora para Microsoft WIN</li> <li>* Un mouse.</li> <li>En ambiente OS/2</li> <li>* 3 MB en RAM</li> <li>* Versión 1.2 o 1.3 de OS/2</li> <li>* Una pantalla graficadora que soporte el OS/2</li> <li>* Mouse que soporte OS/2</li> </ul>
Easy CASE	Upper CASE	Su arquitectura esta montada sobre dos productos: EasyCASE Profesional y EasyCASE DBE.	<ul style="list-style-type: none"> <li>* Sencillez de manejo (menos de 5h de aprendizaje)</li> <li>* Contenido público del repositorio (fácil integración)</li> <li>* Soporte de multiples métodos y técnicas</li> </ul>	<ul style="list-style-type: none"> <li>Integración con Multibase</li> <li>* Integración bidireccional</li> <li>* Creación e vistas del modelo.</li> <li>* Explosionar objetos del diagrama mediante el editor de programa</li> <li>* Adicionamiento de res-</li> </ul>	

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
			(Flexibilidad de implantación). * Bajo coste * Posibilidad de correr bajo DOS o Windows	tricciones de integridad referencial a claves ajenas.	
ER-Win	Upper CASE	Es una poderosa herramienta fácil de usar, para el diseño de BD	* DataWareHouse * Diccionario e Interfaz Drag and Drop * Modelamiento dimensional tipo estrella "STARSCHEMA" * Ofrece propiedades integradas definidas por el usuario (UDPs)	* Aumenta la productividad en el diseño y mto de la BD * Facilita la obtención de nombres consistentes e atributos y definiciones * Permite al usuario expandir y capturar directamente en el modelo, cualquier información adicional que sea importante. * Proporciona un nivel sin precedentes de personalización.	Intel/P86 Pentium con Win 3.x, 95, NT con 16MB en RAM y 50 MB en disco
Designer 2000	Integral	Esta herramienta permite la construcción del modelado de sistemas complejos con reingeniería de procesos, de negocios (BPR) análisis y diagramadores de diseño	* Modelaje de procesos y sistemas. * Generación de aplicaciones de C/S * Administración de repositorio * Diagramador de Relación de Entidades	* Soporta una metodología y diseño flexible * Proporciona al usuario un entorno de desarrollo C/S unificado * Posee una arquitectura portable abierta	

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
				<ul style="list-style-type: none"> <li>* Asegura la integridad de los modelos y la disponibilidad inmediata a los miembros del equipo de desarrollo mediante la integración directa entre diagramadores y repositorios.</li> </ul>	
Excelerator	Integral	Es una herramienta de Ingeniería del software que apoya a los analistas de sistemas para mejorar su productividad y para comunicarse mejor con otros analistas y usuarios finales	<ul style="list-style-type: none"> <li>* Posee una opción de gráficas.</li> <li>* Diccionario de Datos</li> <li>* Interfaz XLD</li> <li>* Apoya la documentación y mtto.</li> </ul>	<ul style="list-style-type: none"> <li>* Permite a los analistas elaborar diagramas que apoyan la productividad en el desarrollo de sistemas.</li> <li>* Su filosofía permite que los analistas representen cualquier cosa en la pantalla, pudiendo distanciarse de ciertas reglas.</li> <li>* Es un instrumento integrador que se apega a todos los metodos de diseño y análisis en un solo paquete.</li> <li>* Permite que el analista importe y exporte datos de otro diccionarios de datos.</li> <li>* Permite establecer la configuración del sistema, mantener proyectos y asignar privilegios de</li> </ul>	<p>Funciona en diversas computadoras tales como:  IBM, XT, PC-AT, 3270 PC,  COMPAQ PLUS,  AT&amp;T6300, WANGPC.  Equipadas con:  512K en RAM  10 MB en Disco Duro  1 Mouse</p>

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
V.A.W.	UpperCASE	Es una herramienta para análisis y diseño de sistemas y puede operar con bases de menus	<ul style="list-style-type: none"> <li>* Herramienta de Diagramación.</li> <li>* Posee reglas para que se encuentran incorporadas dentro del modelamiento de procesos y datos</li> <li>* Incluye una BD que actúa como un repositorio central</li> <li>* Soporta las etapas de modelaje de datos, análisis y diseño estructurado.</li> </ul>	<p>acceso.</p> <ul style="list-style-type: none"> <li>* Aumento no solo la productividad en el desarrollo de sistemas, sino que la eficiencia global en el analisis y diseño.</li> <li>* Facilidad de uso del producto permitiendole al usuario que se involucre en el proceso de desarrollo de sistemas.</li> <li>* Cuenta con una poderosa herramienta de diagramación la cual le permite dibujar cualquier tipo de diagrama.</li> <li>* Posee control sobre todas la adiciones y modificaciones de los diagramas ya que estas se reflejan en el repositorio.</li> </ul>	
Snap	Integral	Herramienta para el desarrollo de aplicaciones	<ul style="list-style-type: none"> <li>* Modelo de datos</li> <li>* Método de desarrollo acelerado (MDA)</li> <li>* Posee una serie de utilitarios</li> <li>* Seguridad</li> <li>* Arquitectura C/S</li> </ul>	<ul style="list-style-type: none"> <li>* Las aplicaciones que genera son de alta calidad 100% nativas de AS400, totalmente documentadas y libres de errores, los cuales se ajustan a los standares S.A.A. (Arquitectura Abierta del Software) de IBM.</li> <li>* Produce resultados a muy corto plazo, hacen</li> </ul>	

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
				<p>que las organizaciones obtengan al máximo provecho de la inversión.</p> <ul style="list-style-type: none"> <li>* Apoya etapas de análisis y diseño, con herramientas de tipo Upper Case tales como VAW permitiendole a la organizacion ampliar la cobertura de desarrollo y Mfto. De aplicaciones bajo ambiente CASE.</li> <li>* Implementa de manera adecuada el esquema metodológico E-R, facilitando la sherramientas y guias necesarias para la construcción de aplicaciones que exploten al máximo las virtudes y potencialidad del AS400</li> <li>* Incorpora un ambiente sofisticado para controlar y ayudar a la administración de procesos y desarrollo de sistemas.</li> <li>* Provee una serie de utilitarios para ayudar al analista a administrar el proceso de desarrollo de aplicaciones.</li> </ul>	

NOMBRE	TIPO DE CLASIFICACION	BREVE DESCRIPCION	CARACTERISTICAS	BENEFICIOS	REQUERIMIENTOS
				<ul style="list-style-type: none"> <li>* Permite al desarrollador crear aplicaciones (Nativas o C/S), trabajando en ambiente gráfico.</li> <li>* La programación de C/S se genera de forma automática y simultanea partiendo del mismo repositorio central.</li> </ul>	

## **CAPITULO IX**

### **DEFINICION Y ANALISIS DE CASO PRACTICO**

#### **9.1 INTRODUCCION**

El presente capítulo se da a conocer el funcionamiento y los objetivos del caso práctico a desarrollar.

El cual consiste en la realización de un sistema de inventario, auxiliado por la herramienta CASE Power Designer de Power Soft, cuyo desarrollador es Power Builder.

Se muestra las definiciones y las relaciones que están establecidas en cada una de las tablas involucradas en la realización del sistema. Además se explica la metodología que ha sido empleada en el desarrollo de dicha aplicación.

#### **9.2 DEFINICION DEL SISTEMA DE INVENTARIO**

El objetivo de realizar esta aplicación es beneficiar a una organización<sup>1</sup> para llevar un control informático de sus recursos. Con el fin de determinar la ubicación exacta de un equipo de trabajo, como también el empleado que está a cargo de dicho equipo, y los diversos accesorios que lo componen. Además se puede conocer a los proveedores que le distribuyen a la empresa con el propósito de solventar cualquier problema de mantenimiento que se presente u otro inconveniente.

---

<sup>1</sup> El Análisis para el desarrollo de la aplicación fue elaborado en base a las necesidades de la empresa privada TransTOOLS.

### **9.3 METODOLOGIA Y TECNICA APLICADA PARA EL DESARROLLO DE LA APLICACIÓN**

Para la realización de este proyecto se utilizó el Modelo de Construcción de Prototipos. El cual comienza con la recolección de requisitos que posteriormente son utilizados para la construcción del modelo conceptual. Este prototipo puede servir como un primer sistema, al cual se le pueden ir haciendo mejoras según las necesidades de la organización y las validaciones necesarias para el buen funcionamiento del mismo.

Son cuatro los elementos para el uso del prototipado: personas, herramientas, metodología y gestión. A continuación se dan a conocer la aplicación de cada uno de estos:

Las personas son las claves del éxito del prototipado y estas deben ser formadas y bien entrenadas; las personas deben de imponer de herramientas potentes que este caso en particular fue la herramienta CASE Power Designer.

También es necesario una metodología formal de trabajo, auxiliándonos del modelo RAD(desarrollo rápido de aplicaciones). Además es un modelo de proceso de desarrollo del software lineal secuencial que enfatiza un ciclo extremadamente corto. El modelo RAD es una adaptación a alta velocidad del modelo lineal secuencial en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes. Esta metodología fue utilizada para el desarrollo de la aplicación debido a que se disponía de un tiempo limitado para el desarrollo de la misma ya que es apropiada para este tipo de proyectos y además esta ligado al modelo lineal secuencial.

Otra técnica utilizada fue el diagrama entidad relación ya que se utiliza para el desarrollo de modelos de datos en el diseño de aplicaciones, en el cual se pueden identificar entidad, relación, cardinalidad, atributos, ocurrencias, identificador, tipos de datos, normalización y se aplico para el diseño de este sistema ya que es el mas implantado en las herramientas CASE, hoy en día existen muchas herramientas CASE que soportan las distintas notaciones de los diagramas entidad relación.

#### **9.4 DESCRIPCION DE ENTIDADES INVOLUCRADAS EN EL DISEÑO**

Dentro de las entidades involucradas en el diseño de aplicaciones tenemos las siguientes:

- Equipos de Trabajo: los cuales están compuestos por Insumos Informáticos, Periféricos, CPU, Software Instalado y Empleados.
- Empleados: La que contendrá los diversos empleados que poseen cada equipos de trabajo.
- Proveedores: Esta entidad se registrarán todas las personas que le distribuyen a la organización: CPU, Periféricos, Software e Insumos Informáticos.
- Actas: Es un registro en donde se refleja si el equipo se encuentra dentro o fuera de la organización ya sea por reparación u otras causas.
- Periféricos: Se registrarán las diversas características que poseen los accesorios de cada equipo de trabajo.
- Tipos de Hardware: Conformará un registro de los diferentes periféricos que componen un equipo de trabajo.
- Accesorios: Son todos aquellos elementos que conforman un periférico por ejemplo: memoria de impresora, protector de pantalla de un monitor, etc.
- Organización: Se establecerá la ubicación del empleado dentro de la organización.

- Cargos Funcionales: Se establecerá el cargo del empleado dentro de la organización.
- CPU: Esta entidad representa componentes y características de cada unidad central de procesamiento.
- Software: Se incluirá con todo el software que cuenta la organización y sus respectivas características para la instalación de éste.
- Software Instalado: Los software que se encuentran instalados en cada equipo de trabajo.

## **9.5 DISEÑO DEL SISTEMA DE INVENTARIO**

### **9.5.1 MODELO LÓGICO**

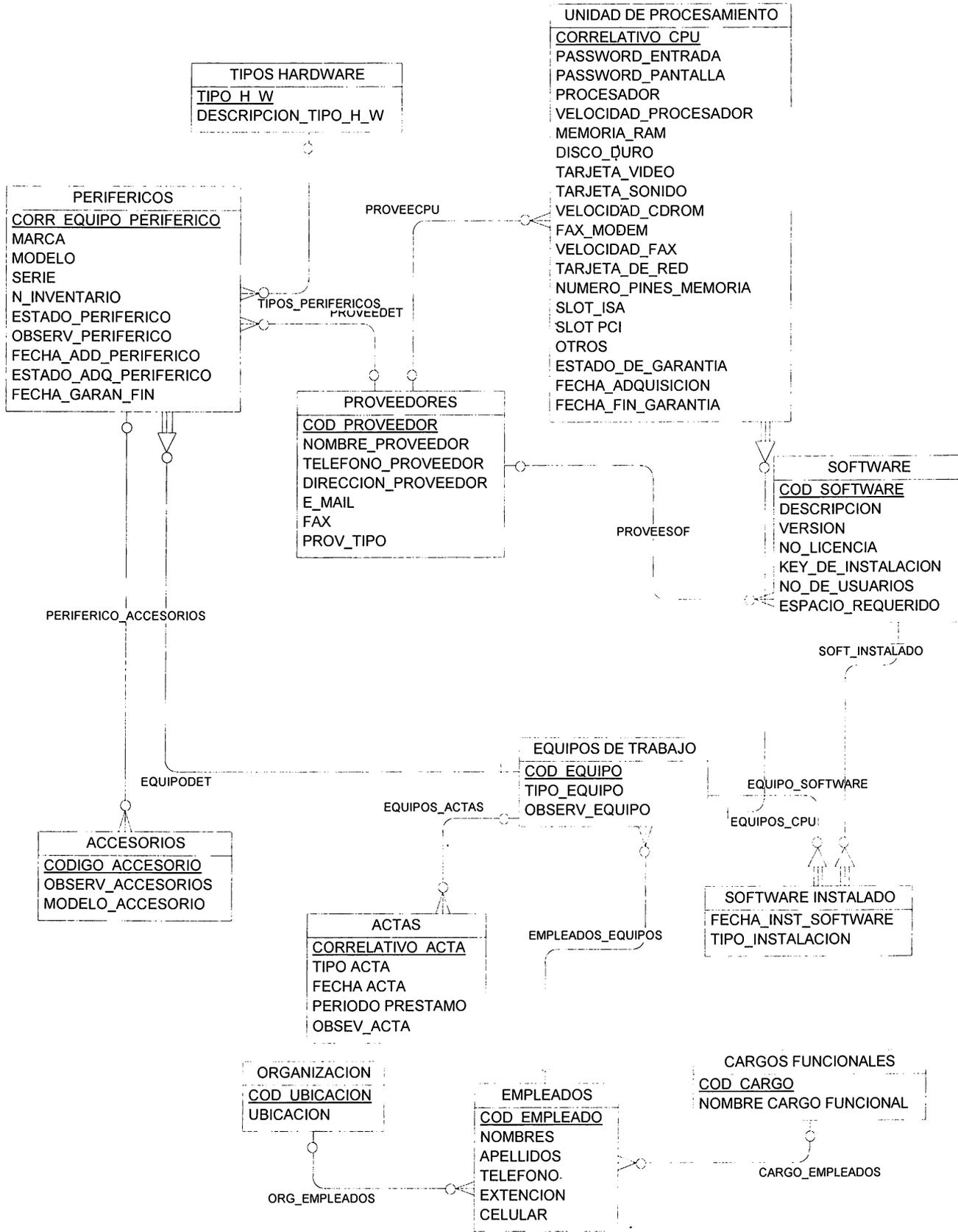
El modelo lógico representa los datos de la empresa, independientemente de su implementación.

En el caso de PowerDesigner, se realiza el diseño lógico según la notación de James Martín.

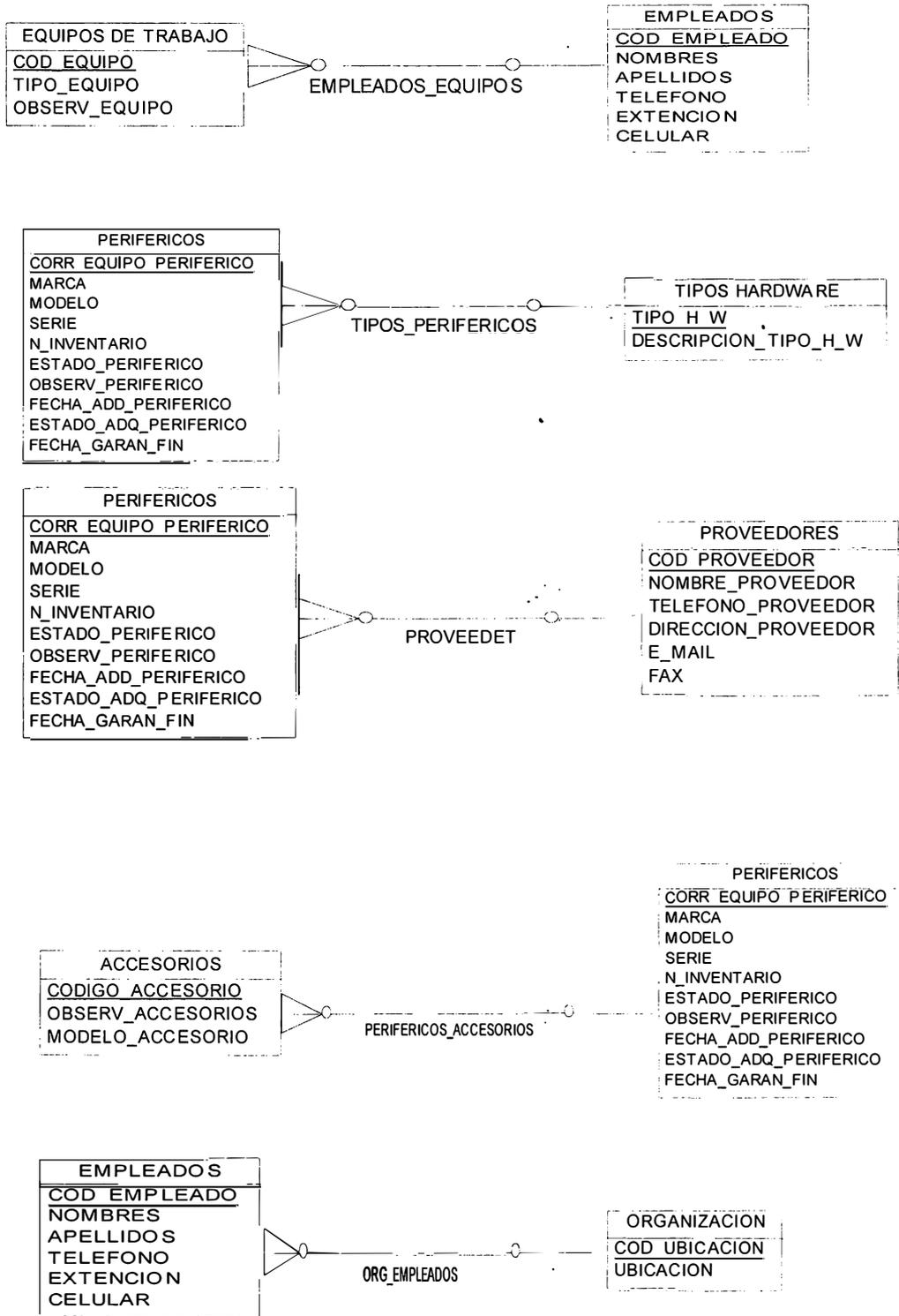
El modelo lógico se utiliza para:

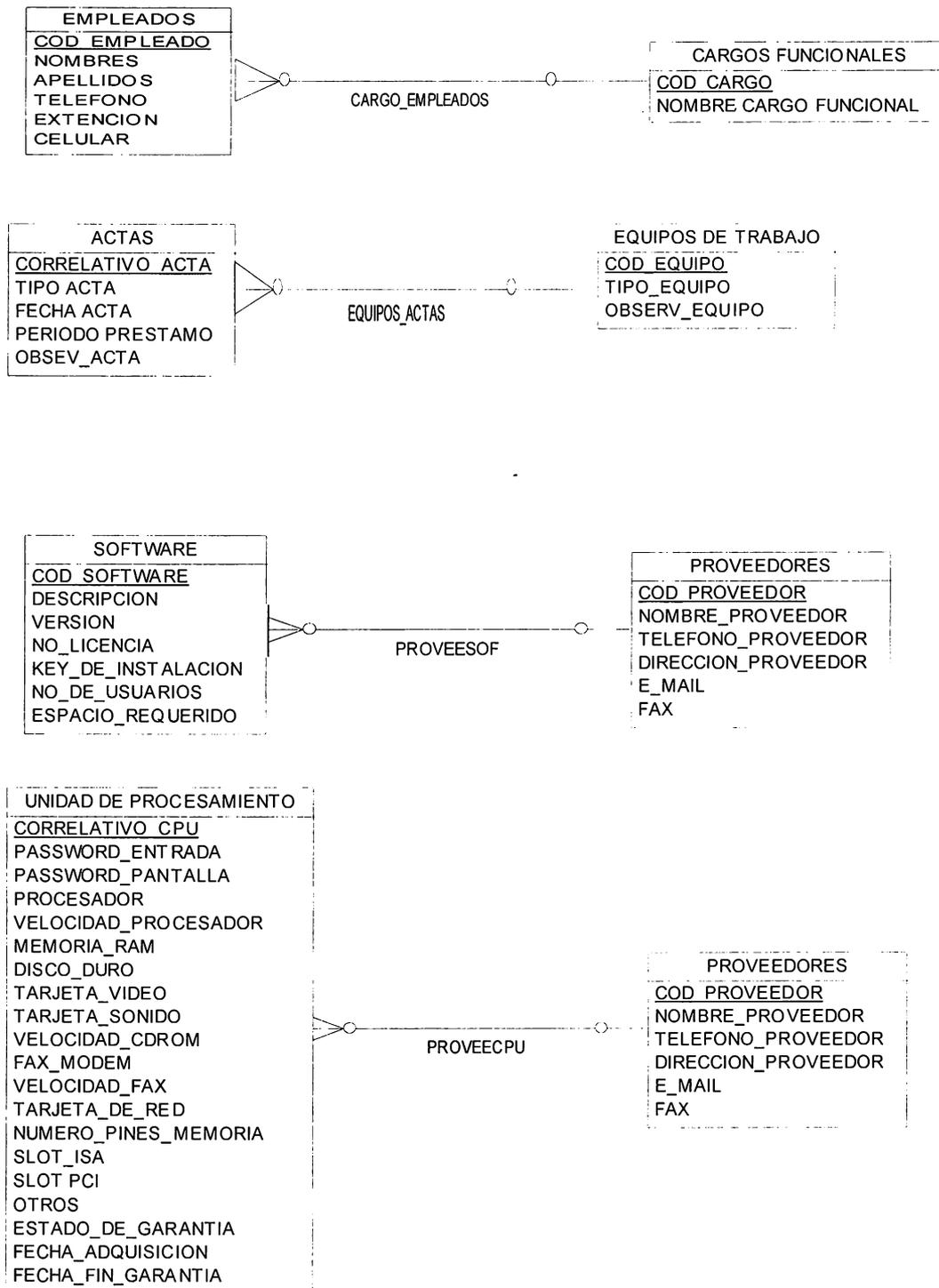
- Representar los datos de forma gráfica.
- Verifica y valida el diseño de los datos
- Genera el modelo físico automáticamente según el sistema gestor de bases de datos definido.

De acuerdo al sistema de inventario la representación del modelo lógico se presenta a continuación:

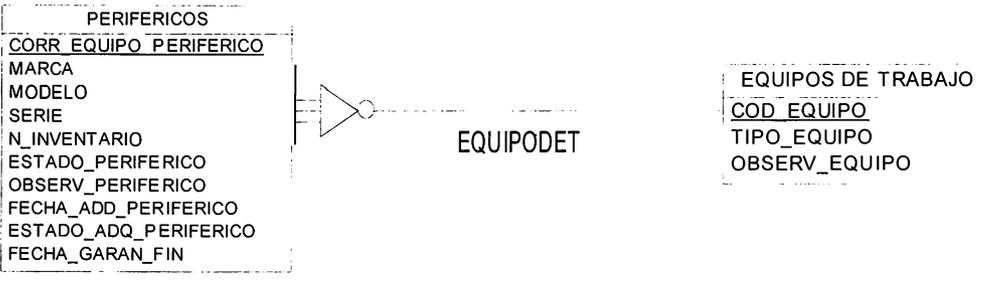
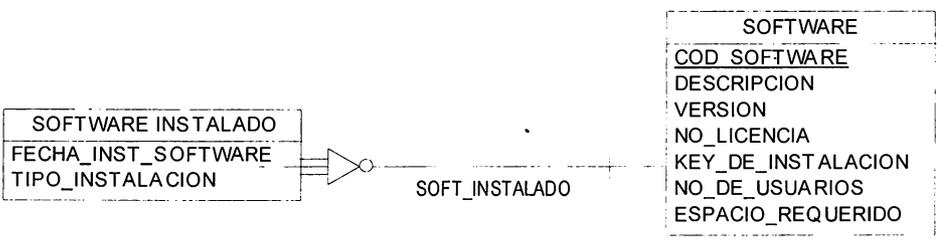
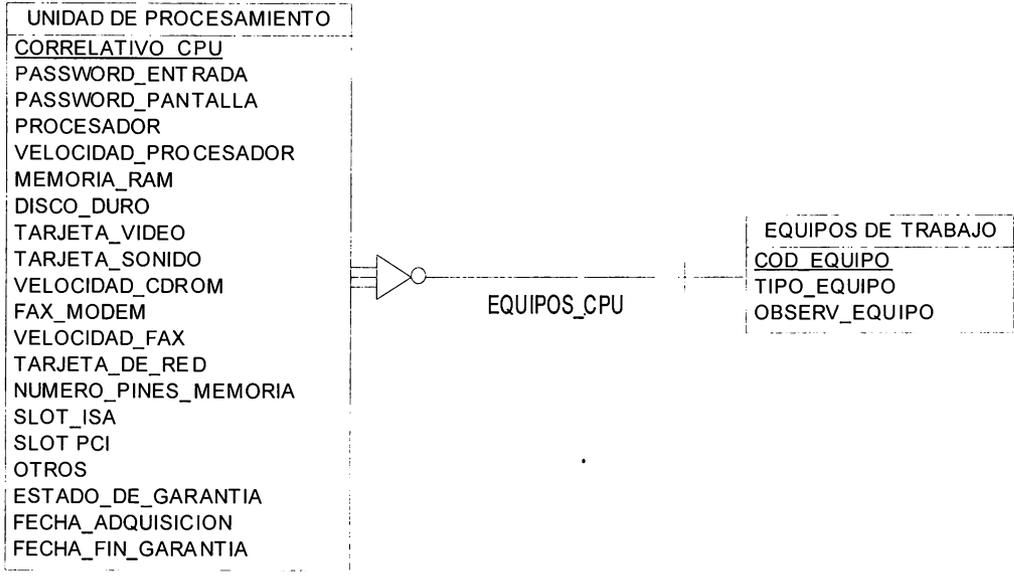


### 9.5.1.1 RELACIONES ENTRE ENTIDADES





Estas entidades poseen una cardinalidad de muchos a uno y además son entidades dependientes heredando la llave primaria de la otra entidad, pudiendo haber o no elementos entre ellos.



Las entidades anteriores poseen una cardinalidad de muchos a uno, y la primera entidad es mandatoria y dominante de la segunda, es decir hereda la llave primaria y se necesita un mínimo de uno para su existencia.

### **9.5.2 MODELO FÍSICO**

Un modelo físico, representa la implementación física de una base de datos. El modelo físico se utiliza para: Modelar mediante un gráfico una base de datos, Generar una base de datos, Generar un modelo lógico.

Existen tres formas de generar un modelo físico: A partir de un modelo lógico, directamente, de forma manual, A través de ingeniería inversa.

Para la generación del modelo físico del sistema de inventario se realizó a partir del modelo lógico el cual se presenta en la siguiente figura.



### **9.5.3 GENERACIÓN DE LA BASE DE DATOS**

En el proceso de desarrollo de la base de datos, una vez hecho el análisis del sistema construimos nuestra base de datos. Este proceso se realiza automáticamente con PowerDesigner o desde Sybase que es la base de datos utilizada para esta aplicación.

Esta herramienta, permite generar tanto una base de datos como el sql de creación de la misma. Sin embargo previamente deberemos de haber construido el modelo físico basándonos en el sistema gestor de bases de datos con el que vamos a trabajar.

Los pasos para generar una base de datos son:

- Primero se crean los database device
- Segundo se crea la BD con su data y log de transacciones.

### **9.5.4 GENERACIÓN DE LA APLICACIÓN**

En el desarrollo de la creación de aplicación se siguieron los siguientes procedimientos:

- A partir de los modelos antes mencionados y la BD en Sybase ya creada se selecciono la opción Generate Database del menú Database de Power Designer y ejecuto Generate Script.
- Para conectarse a la BD se realizar mediante la opción Execute SQL dentro de Database
- Finalizando se selecciona Generate Power Builder Application dentro del menú cliente de Power Designer.

## CONCLUSIONES

- Las herramientas CASE buscan asistir a cada fases del ciclo de vida de una aplicación de forma tal que permita al ingeniero de sistemas desarrollar un sistema completo, eficiente, de alta calidad y amoldable a cambios futuros o a diferentes plataformas.

Son fundamentales en las etapas de análisis y diseño pues toman en consideración todos los posibles aspectos del sistema. Las CASE traen como resultado una reducción significativa en el costo de mantenimiento al ser más cauteloso y específico al crear las aplicaciones.

Las metodologías CASE han sufrido grandes cambios y han evolucionado a base de las necesidades del usuario. Surgen como facilitadoras del proceso y desarrollo de un sistema. El futuro de CASE vislumbra muchos cambios dirigidos a el intercambio y múltiple acceso de información que promete mayor efectividad de procesos al hacerlo más accesible a una mayor variedad de plataformas, lenguajes y metodologías.

- Para la Macroempresa Salvadoreña el desarrollo, la actualización y mantenimiento de aplicaciones es uno de los problemas más costoso del área de la informática. Una manera de enfrentar este problema es utilizando herramientas CASE que incrementen la productividad y mejoran la calidad en el trabajo.

- Actualmente existe desconocimiento de dichas herramientas, y la mayoría de empresas no cuentan con personal capacitado en el empleo de este tipo de tecnología. Sin embargo es un obstáculo que se puede solucionar, incorporando dentro de la formación profesional de los Ingenieros en Ciencias de la Computación este tipo de conocimientos fundamentales sobre las CASE, y de esta forma agilizar su gestión dentro de las empresas salvadoreñas.
- De acuerdo a las necesidades que actualmente se están presentando en algunas empresas nacionales se ha podido determinar que es importante que los estudiantes de Ingeniería en Computación conozcan de los beneficios que este tipo de herramientas proporcionan, así mismo la lógica con que funcionan ya que esto les servirá como base en su desempeño laboral, generando más oportunidades de trabajo.
- El conocimiento sobre esta herramientas desde la educación superior viene a ahorrarle al empresario tiempo y dinero en el entrenamiento de dichas herramientas.
- Una herramienta CASE no es solo un desarrollador de sistemas como muchas personas creen entre los que podemos mencionar a Power Builder y Developer.
- Las propuestas aquí presentadas se basan en el estudio de la importancia que tiene que los estudiantes de Ingeniería en Computación conozcan de las herramientas de ingeniería del software asistida por computadora (C.A.S.E.), pero es responsabilidad de la escuela de computación asignar un profesional que posea un conocimiento amplio de este tipo de herramientas y así mismo utilizar un software que se esté utilizando en ese momento como

también aplicaciones reales que permitan que el estudiante visualice la forma en que se desempeñará posteriormente.

- Es necesario que exista más apoyo por parte de la Universidad en este tipo de investigación ya que para la realización de los talleres con maestros fue difícil la coordinación de los mismos.
- Es importante que cada vez que se desee incorporar materias dentro del pensum curricular de la carrera de Ingeniería en Ciencias de la Computación, se debería de realizar un estudio sistemático, en donde se den a conocer las necesidades del mercado laboral, así como también tomar en cuenta las diversas expectativas de catedráticos y alumnos.
- El aprendizaje de las herramientas CASE no es difícil, siempre y cuando se tengan los conocimientos de Análisis y Diseño de Sistemas, Bases de Datos y los conceptos básicos de las herramientas CASE.

## RECOMENDACIONES

- Se recomienda a la Escuela de Computación que siempre que visualice la necesidad de incorporar una materia al pensum curricular de la carrera, se realice un estudio exhaustivo, para fundamentar la necesidad de dicha incorporación, dicha incorporación y trasladar conocimientos más acertados a los estudiantes según las expectativas de todos los involucrados .
- Se sugiere a la escuela de computación asignar a la materia de herramientas CASE un catedrático que posea un conocimiento amplio de este tipo de herramientas y así mismo utilizar un software que se esté utilizando en el momento a impartirla, como también aplicaciones reales que permitan que el estudiante visualice la forma en que se desempeñará posteriormente.
- Se recomienda a la Escuela de Computación brindar más apoyo a los estudiantes al momento de realizar su trabajo de graduación, ya que en muchos casos existe falta de comunicación y coordinación.
- Dentro de las propuestas formuladas y en base a los resultados obtenidos, consideramos que la más acertada es la incorporación de herramientas CASE dentro de contenidos de materias obligatorias ya existentes, en las cuales se aplique el ciclo de vida de desarrollo de sistemas.
- Se sugiere que para incorporar el estudio de estas herramientas se establezca la ruta crítica de las materias a cursar previamente.

## **GLOSARIO TECNICO**

### **Encapsulamiento**

Agrupamiento de procedimientos y sus datos asociados.

### **Estructura**

El ciclo de desarrollo de un sistema de información deseado, y para alcanzar éste, sigue un proceso organizado y por pasos.

### **Frameworks**

Sirven para proporcionar una infraestructura en la cual acoplar y adaptar herramientas individuales.

### **Front-End ó Upper CASE**

Clasificación de herramientas que abarcan las primeras fases de análisis y diseño.

### **Herencia**

Clasificación de los objetos de acuerdo con las propiedades que comparten.

### **Herramientas**

Soportan una tarea específica del proceso de producción de software, como la edición, programación, verificación y validación, gestión de configuración, métrica o gestión de proyectos.

### **Metametódico**

El método puede analizarse como un conjunto de entidades relacionadas entre sí y como una estructura de procedimientos que sintetizan un flujo de procesos generador de resultados.

## **Método**

Secuencia de modelados que ayuda a construir a partir del original de la realidad, una o varias cadenas de modelos, derivados unos de otros, con el objetivo de lograr un modelo material final o sistema que concrete la nueva parcela de realidad deseada en relación con el original.

## **Metodología**

Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.

## **Objeto**

Son las unidades estructurales primarias del programa.

## **Polimorfismo**

Compartir ciertas características comunes como la capacidad para colocarse en una sola colección y la capacidad para responder a los mismos mensajes.

## **Programación Orientada a Objetos**

Aquella programación por medio del envío de mensajes a objetos de tipo desconocido.

## **Tareas**

Actividades elementales realizadas en condiciones atómicas.

## **Técnica**

Conjunto de procedimientos de que se sirve una ciencia o arte.

## **Back-End ó Lower CASE**

Clasificación de herramientas cuyo objetivo suele ser el diseño detallado y la generación de código.

## **Integrated CASE**

Herramientas que engloban Upper y Lower CASE, además incluyen componentes para la gestión de proyectos y la gestión de configuración.

## **Toolkit**

Conjunto de herramientas que automatizan un tipo de tarea del ciclo de vida, como el análisis.

## **Workbenchs**

Conjunto de herramientas integradas que abarcan las fases básicas del ciclo de vida: análisis, diseño e implementación.

## BIBLIOGRAFIA

- [1] Alfaro Burgos, Guadalupe, "*Desarrollo de aplicaciones utilizando metodología auxiliada por herramientas case*", Universidad Don Bosco, mayo de 1996.
  
- [2] Baca Urubina, "*Evaluación de Proyectos*", México 1995
  
- [3] Cuevas Gonzalo "*Ingeniería del Software*" Iberoamericana 1995
  
- [4] Cruz, Dueñas, Duran, Hernández; "*Desarrollo de herramientas de análisis y diseño aplicadas a El Salvador*", UCA; 1992.
  
- [5] Molina, Navas, Zepeda; "*Introducción al desarrollo de sistemas de información utilizando herramientas C.A.S.E., un caso de aplicación*"., UCA, Septiembre 1993.
  
- [6] Pressman Rogers, "*Ingeniería del Software un enfoque Práctico*". Madrid 1998.
  
- [7] Piattini Mario, "*Elementos y Herramientas en el desarrollo de sistemas de información, una visión actual de la tecnología CASE*" Wesley Iberoamericana 1997
  
- [8] Senn James A., "*Análisis y Diseño de Sistemas de Información*" México 1992.

## **ENTREVISTAS**

1. Almacenes SIMAN, Lic. Raaúl Antonio Henríquez, Gerente de Informática.
2. APEX ,Ing. René Alas, Administrador de centro de cómputo.
3. Avicola Salvadoreña S.A. de C.V., Ing. José Dolores Hernández, Administrador del centro de cómputo.
4. BCR, Lic. Dario Alvarez, Auditor de Sistemas.
5. BCR, Lic Manuel Velazco, Auditor de Sistemas.
6. BCR, Lic. José Hernández, jefe del depto. De análisis y sistemas.
7. C.E.L. Central , Ing. Juan Carlos Recinos, Administrador del centro de cómputo.
8. CEPA , Ing. René Ventura, Subgerente de Informática.
9. CNR, Ing. Ruben Mejía, Administrador de Redes
10. Consisa, Lic. Ring Salazar, Jefe de Proyectos
11. Corte de Cuentas de El Salvador, Lic. Nery Balcaceres, Jefe de Análisis de Sistemas.
12. DATUM , Lic. Morales, Jefe de Soporte Técnico.
13. DATUM , Lic. Jose Luis Catellanos, Gerente General.

14. Escuela Superior de Economía y Negocio. Willian E. Martínez Rivera, Administrador del centro de cómputo.
15. INPEP , Ing. Brenda Jeres, Jefe de Desarrollo de Sistemas.
16. ISSS Central, Lic. Walter Turcios, Administrador del centro de cómputo
17. Monelca S.A. de C.V., Lic. José Roberto Cea, Administrador del centro de cómputo.
18. Pollo Campero de El Salvador , Lic. Daniel López, Gerente de Informática.
19. Shell El Salvador, Ing. Dennie de Larín, Administrador de centro de cómputo.
20. TransTOOLS, Lic. Miguel Angel López, Administrador de Proyectos.
21. Superintendencia de Pensiones, Ing. José Eduardo Pleitez Tecun, Administrador de centro de cómputo.
22. Siemens El Salvador., Lic. Carlos Castillo, Administrador del centro de cómputo.
23. Solaire El Salvador ,Ing. Santos Rivera, Gerente de Informática.
24. UCA, Lic. Rafael Ibarra, Administrador de centro de cómputo.

## **DIRECCIONES DE INTERNET**

<http://agamenon.unidadess.edu.co/pfiguero/mgmo/herramientas.html>

<http://dc2.uni-bielefeld.de/stc/tt/producto/multiba1.html>

[http://www.oracle.cl/ora\\_3200.html](http://www.oracle.cl/ora_3200.html)

<http://www.oracle.cl/ora3221.html>

<http://www.oracle.cl/ora3231.html>

<http://www.sybase.com/productos/powerdesigner>

<http://www.alfamega.com.mx>

<http://www.upco.es/pag/euinf/ingsoft.html>

<http://dichato.dcc.uchile.el/DCC/docenc...>

<http://osiris.sunderland.ac.uk/sst/case2/welcome.html#I>

[http://web.cs.ualberta.ca/~softeng/se\\_defn.html#casetools](http://web.cs.ualberta.ca/~softeng/se_defn.html#casetools)

[http://www.sei.cmu.edu/technology/dynamic\\_systems/case/](http://www.sei.cmu.edu/technology/dynamic_systems/case/)

<http://www.db,smag.com/9707d16.html>

<http://www.db,smag.com/9601d16.html>

<http://www.oramag.com/archives/45APP.html>

<http://osiris.sunderland.ac.uk/sst/case2/welcome.html#I>

[http://web.cs.ualberta.ca/~softeng/se\\_defn.html#casetools](http://web.cs.ualberta.ca/~softeng/se_defn.html#casetools)

[http://www.sei.cmu.edu/technology/dynamic\\_systems/case/](http://www.sei.cmu.edu/technology/dynamic_systems/case/)

<http://www.db,smag.com/9707d16.html>

<http://www.db,smag.com/9601d16.html>

<http://www.oramag.com/archives/45APP.html>

[http://www.sei.cmu.edu/ata/ata\\_init.html](http://www.sei.cmu.edu/ata/ata_init.html)

<http://www.sei.cmu.edu/legacy/case/index.html>

<http://www.sei.cmu.edu/engineering/engineering.html>

***ANEXO A***  
***FORMATO DE ENCUESTA***



## ENCUESTA SOBRE EL CONOCIMIENTO Y UTILIZACIÓN DE HERRAMIENTAS CASE

La presente encuesta tiene como finalidad, analizar el conocimiento que los estudiantes tienen y el uso que los desarrolladores del software(aplicaciones) le están dando a las herramientas CASE.

De antemano gracias por su colaboración.

1. Seleccione el grado académico:

- Bachiller
- Técnico en Computación
- Estudiante Universitario
- Egresado
- Graduado Universitario

En que año \_\_\_\_\_

2. Seleccione en que área se está desempeñando actualmente(Puede seleccionar más de una):

- a) Planificación de aplicaciones
- b) Diseño de aplicaciones
- c) Programación de la aplicación
- d) Ninguna
- e) Otras(Mencionelas): \_\_\_\_\_

3. Señale cual cree usted que es concepto de herramientas CASE (Seleccione solo una):

- a) Desarrollador de flujogramas
- b) Software que automatiza parte del proceso de desarrollo de aplicaciones.
- c) Combinación de herramientas software y métodos que enfocan el problema de producción de software.

- d) Herramientas analíticas y de recolección de información sin la mínima intervención del elemento humano.
- e) Herramientas que dan soporte a todas las fases del ciclo de vida de desarrollo de un proyecto.
- f) Otro: \_\_\_\_\_

4. Esta utilizando o ha utilizado herramientas CASE: SI  NO

5. Si la respuesta anterior es NO, marque las razones(Puede seleccionar más de uno):

a) Porque No las conoce

- 1. Nunca le han mencionado este concepto
- 2. Las ha escuchado pero no le interesa conocer acerca de ella
- 3. Simplemente no ha tenido tiempo para estudiarlas
- 4. Lleva alguna materia en el cual la incluyen pero todavía no la ha cursado
- 5. Otros motivos: \_\_\_\_\_

b) No cuenta con el recurso económico para poderlos adquirir.

c) Si las conoce pero no las puede utilizar

d) Otras razones: \_\_\_\_\_

6. Si la respuesta del numeral 4 es SI, marque cuales(Puede seleccionar más de uno):

- a) E-R Win For Power Builder
- b) System architect
- c) CASE designer de Oracle
- d) CASE proyect
- e) Oracle Designer/2000
- f) Open Snap
- g) PCCS(DFD)
- h) Excelerator
- i) VAW
- j) Natural Construct
- k) SYNON
- l) Otras: \_\_\_\_\_

7. Señale para que las esta utilizando(Puede seleccionar más de uno):

- a) Para diagramación
- b) Para planificación de sistemas y administración de proyectos.
- c) Análisis y diseño
- d) Generadores de código.
- e) Programación
- f) Generadores de Interfaces
- g) Simulación y creación de prototipos
- h) Documentación
- i) Otros \_\_\_\_\_

8. Cual de estas técnicas emplea para el desarrollo de sus aplicaciones(Puede seleccionar más de una).

- a) Diagrama de flujo de datos(DFD)
- b) Modelo E/R
- c) Modelo NIAM
- d) Diagramas Jackson
- e) Diagramas de estructuras
- f) Matrices
- g) Diagramas de Warnier-Orr.
- h) Diagramas de transición de estados.
- i) Otras(especifique) \_\_\_\_\_

9. Cual de estas metodología emplea para el desarrollo de sus aplicaciones(Puede seleccionar más de una).

- a) Metodología de cascada
- b) Metodología de espiral
- c) Metodología de prototipo
- d) Metodología de Desarrollo rápido de aplicaciones(RAD).
- e) Ninguna
- f) Otras(especifique): \_\_\_\_\_

10. Si alguna vez ha utilizado herramientas CASE como cree que le beneficio(Puede seleccionar más de una):

- a) Mejora la productividad del analista y del usuario final
- b) Mejora la eficacia
- c) Mejora la calidad del Sistema Información
- d) Aumenta la velocidad y disminuye el tiempo necesario para completar las tareas de desarrollo
- e) Garantiza la consistencia de procedimientos
- f) Captura, almacenamiento procesamiento y recuperación de los detalles de un sistema.
- g) Otros: \_\_\_\_\_

11. Cree que le sea útil proporcionarle un documento en el cual se encuentre la recopilación de las diferentes técnicas y metodologías utilizadas en el desarrollo de aplicaciones utilizando estas herramientas.

SI  NO

\_\_\_\_\_  
Firma

***ANEXO B***  
***FORMULACION DE HIPOTESIS***

Las Herramientas CASE deben  
estar estudiadas como una  
asignatura estratégica en la  
carrera de Ingeniería Ciencias  
de la computación.

***ANEXO C***  
***PROPUESTA DE CONTENIDOS***

# **PROPUESTA A**

*A Continuación Se Presenta Una Serie De Temas Que Se Encuentran Relacionados Con Las Herramientas C.A.S.E. Según Su Criterio Seleccione Cual De Estos Temas Son Importante Para La Formación Profesional De Un Ingeniero En Sistemas.*

---

## **1. TÉCNICAS Y METODOLOGÍAS EN EL DESARROLLO DE SOFTWARE**

### **1.1 Conceptos Generales de las Técnicas más utilizadas**

- 1.1.1 DFD (Diagrama de Flujo de Datos).** *Se utiliza para representar el flujo de datos dentro de una organización.*
- 1.1.2 Diagrama Entidad-Relación.** *Refleja el modelo de datos en base a los requerimientos de la empresa.*
- 1.1.3 Diagrama de Warnier-Orr.** *Ayuda al diseño de Estructuras de Programas identificando las salidas y el resultado del procedimiento, es decir, trabaja hacia atrás para determinar los pasos y combinaciones de entradas necesarios para producirlos.*
- 1.1.4 Método Jackson.** *Es una técnica sistemática para resolver problemas por módulos es decir, es una manera estructurada.*

### **1.2 Clasificación de los métodos**

- 1.2.1 Metodología de Prototipo**
- 1.2.2 Metodología Secuencial o Lineal**
- 1.2.3 Metodología Incremental**
- 1.2.4 Metodología Espiral**
- 1.2.5 Metodología R.A.D.**

## **2. CONCEPTOS GENERALES**

- 2.1 Conceptos Básicos de la Tecnología C.A.S.E.**
- 2.2 Clasificación de las herramientas C.A.S.E.**

- 2.2.1 *Front End. Se utiliza Planificación y Definición de Requerimientos, Etapa de Análisis y Diseño, Modelo y generación de Prototipos.*
- 2.2.2 *Back End. En compiladores y Generadores de Código.*
- 2.2.3 *Integrales. Conjunto de herramientas integradas que interactúan unas con otras en forma consistente.*
- 2.3 *Descripción General de los distintos tipos de herramientas*
  - 2.3.1 *Nombre de la herramienta*
  - 2.3.2 *Tipo de Clasificación*
  - 2.3.3 *Breve Descripción*
  - 2.3.4 *Características*
  - 2.3.5 *Beneficios*
  - 2.3.6 *Requerimientos*
- 3. **3. APLICACIONES Y USOS DE LAS HERRAMIENTAS C.A.S.E.**
  - 3.1 *Herramientas de Análisis y Diseño*
  - 3.2 *Herramientas de Programación*
  - 3.3 *Herramientas de Integración y Prueba*
  - 3.4 *Herramientas de Creación de Prototipo*
  - 3.5 *Herramientas de Mantenimiento*
  - 3.6 *Herramientas de Soporte*
  - 3.7 *Herramientas de Planificación*
  - 3.8 *Herramientas de Gestión*
- 4. **4. IMPLEMENTACIÓN DE C.A.S.E.**
  - 4.1 *Problemas y Soluciones del Software*
  - 4.2 *Establecimiento de Necesidades*
  - 4.3 *Adaptación de la tecnología de C.A.S.E.*
  - 4.4 *Evaluación de C.A.S.E.*
  - 4.5 *Auditoría del entorno C.A.S.E.*
- 5. **5. REPOSITARIOS Y DICCIONARIOS DE DATOS**
  - 5.1 *Visión General de los Repositorios y Dicionarios*
  - 5.2 *Estándares ISO/IEC: Para sistemas de diccionario de recurso de información*
  - 5.3 *PCT Repositorio Estándar para Herramientas Portátiles*

#### *5.4 CDIF Estándar para el Intercambio de Datos entre Herramientas C.A.S.E*

### **6. HERRAMIENTAS C.A.S.E DE ORACLE**

#### *6.1 Introducción*

#### *6.2 Oracle Designer 2000*

#### *6.3 Componentes*

#### *6.4 Herramientas y Métodos*

#### *6.5 Generadores, Administración de Repositorios*

#### *6.6 Reingeniería*

### **7. PRACTICA DE ESTUDIO**

*La práctica de esta asignatura consistirá en la utilización de una herramienta C.A.S.E. Según su criterio cual cree que es la más conveniente a utilizar.*

*7.1 System Architect : Herramienta que automatiza los procesos de generación, manejo y organización de diversos modelos de diagramas.*

*7.2 Easy-CASE: Herramienta que cubre la fase de análisis y diseño*

*7.3 I-CASE: Se concibe como un conjunto de cuatro herramientas que toda las disciplinas que van desde la estrategia de la empresa y la concepción del sistema de información hasta el análisis, diseño y generación de los mismos programas.*

*7.4 Erwin: Herramienta fácil de usar para el diseño de Bases de Datos*

*7.5 Developer: Herramienta que permite la construcción de modelado de sistemas complejos con reingeniería de procesos, análisis y diseño.*

*7.6 Designer 2000: Herramienta que permite la construcción de modelado de sistemas complejos con reingeniería de procesos, de negocios (BPR), análisis y diagramadores de diseño.*

*7.7 Discovery 2000: Herramienta que soporta análisis multidimensional de rápida consulta e información de data WareHousing y poderosa búsqueda de datos.*

*7.8 Exceleator: Herramienta de Ingeniería del Software que apoya a los analistas de sistemas para mejorar su productividad para comunicarse de una mejor manera con otros analistas y usuarios finales.*

*7.9 V.A.W.: Herramienta para análisis y diseño de sistemas y puede operar con bases de menús.*

*7.10 Snap: Herramienta para el desarrollo de aplicaciones*

**7.11 Power Builder: Herramienta profesional para el desarrollo de aplicaciones, corre en ambiente Windows que permite a programadores profesionales construir sofisticadas aplicaciones gráficas.**

# **PROPUESTA B**

*A Continuación Se Presenta Una Serie De Temas Que Se Encuentran Relacionados Con Las Herramientas C.A.S.E. Según Su Criterio Seleccione Cual De Estos Temas Son Importante Para La Formación Profesional De Un Ingeniero En Sistemas.*

---

## ***Unidad 1. Introducción al Software Asistido por Computadoras***

### ***1.1 Conceptos Generales***

### ***1.2 Clasificación de C.A.S.E. (Upper CASE, Lower CASE)***

### ***1.3 Componentes y Clasificación Básica de una herramienta C.A.S.E.***

## ***Unidad 2. Aplicaciones y Uso en el ciclo de Vida del Desarrollo de un Sistema***

### ***2.1 CASE aplicados al análisis***

#### ***2.1.1 Procesos***

#### ***2.1.2 Modelo de Bases de Datos***

#### ***2.1.3 Objetos***

### ***2.2 C.A.S.E. Aplicaciones al diseño de Software***

#### ***2.2.1 Procesos (Flujo de Datos)***

#### ***2.2.2 Diagramas Entidad-Relación (Bases de Datos)***

### ***2.3 C.A.S.E. Aplicadas al desarrollo***

#### ***2.3.1 Generadores de código para***

- *Lenguaje de tercera generación***
- *Lenguaje de cuarta generación***
- *Herramientas de Desarrollo Orientada Objetos (Power Builder)***

#### ***2.3.2 Generadores de Bases de Datos***

- *Generadores de SQL***
- *Generadores para Bases de Datos Relacionales***

## ***Unidad 3. Plan de Estudio para cada una de las etapas de datos relacionales***

### ***3.1 Análisis (S.Designer Process Analyst)***

**3.2 Diseño (S.Designer Entidad-Relación)**

**3.3 Desarrollo (S.Designer App modelos para: Visual Basic, Power Builder,  
Optima++ (Java))**

# **PROPUESTA C**

*A Continuación Se Presenta Una Serie De Temas Que Se Encuentran Relacionados Con Las Herramientas C.A.S.E. Según Su Criterio Seleccione Cual De Estos Temas Son Importante Para La Formación Profesional De Un Ingeniero En Sistemas.*

---

- 1 DIAGRAMAS ENTIDAD-RELACION, DIAGRAMAS FLUJOS DE DATOS:** *Introducción al modelado de datos. Elementos. Tipos de Entidades. Tipos de Relaciones*
- 2 DICCIONARIO DE DATOS DE DIAGRAMAS ENTIDAD-RELACION Y DE DIAGRAMAS DE FLUJOS DE DATOS:** *Introducción al diccionario de datos. Elementos que lo componen.*
- 3 INTRODUCCION A LAS HERRAMIENTAS C.A.S.E.:** *Qué es una herramienta C.A.S.E., Ventajas de las herramientas C.A.S.E., Evolución de las herramientas C.A.S.E.*
- 4 INTRODUCCION A UNA HERRAMIENTA ESPECIFICA:** *Definición de proyectos. Inicio de un proyecto. Selección de metodología a utilizar en un proyecto. Selección de técnicas a utilizar. Descripción de las opciones del menú.*
- 5 DESARROLLO DE UNA ACTIVIDAD PRACTICA POR MEDIO DE LA HERRAMIENTA:** *El alumno desarrollara con ayudan del docente varios casos de practica del modelo de Entidad-Relación y el análisis de un sistema sobre la herramienta estudiada.*

***ANEXOD***  
***FORMATO DE ENTREVISTAS***



## **ENTREVISTA SOBRE LA PERTINENCIA DE INCORPORACION DE HERRAMIENTAS CASE DENTRO DEL PENSUM CURRICULAR DE LA CARRERA DE INGENIERIA EN CIENCIAS DE LA COMPUTACION**

La presente entrevista tiene como finalidad, conocer las diversas opiniones que los profesionales en el ámbito informático poseen sobre la pertinencia de incorporar dichas herramientas dentro del pensum curricular de la carrera de Ingeniería en Ciencias importante que son las Herramientas C.A.S.E. dentro de su formación profesional.

---

1. ¿Utiliza alguna herramienta CASE?
2. ¿ Es necesario que en las universidades les impartan conocimientos de lo que son herramientas CASE?
3. ¿ De que forma se podría incorporar ésta materia, como una obligatoria, como electiva ó como dentro de contenidos diversas materias?
4. ¿Cuál de las propuestas de los contenidos cree que es más relevante para ser impartida?
5. ¿ Qué esperan de los nuevos ingenieros en sistemas?

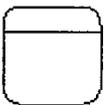
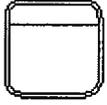
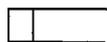
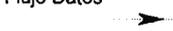
***ANEXO E***  
***SIMBOLOGIA CASE***

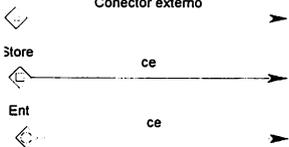
## SIMBOLOGIA DE HERRAMIENTAS CASE

### ELEMENTOS DE LOS DIAGRAMAS DE FLUJO DE Gane & Sarson.

Los elementos utilizados en un diagrama de flujo de datos Gane & Sarson son:

#### ELEMENTOS GRÁFICOS

ELEMENTO	DESCRIPCIÓN
Entidad Externa 	Representa a un objeto que se encuentra fuera del sistema definido La entidad externa obtiene o introduce información en el sistema.
Entidad Externa Duplicada 	Representa una entidad externa que se encuentra más de una vez en un diagrama.
Proceso de Datos 	Es un indicativo, para decir que los datos que entren (inputs), se utilizarán para producir datos de salidas (outputs).
Proceso de Datos Repetido 	Indica que el proceso indicado, aparece más de una vez en el diagrama.
Data Store 	Se utiliza para almacenar los datos que son proporcionados por unos procesos de datos, y utilizados posteriormente por otros. Deben aparecer en el nivel más alto del diagrama, cuando es utilizado por dos o más procesos, pudiendo aparecer en más de un lugar del diagrama y puede ser utilizado en niveles inferiores del mismo DFD.
Data Store Duplicado 	Almacenamiento que se encuentra más de una vez en un mismo diagrama.
Flujo de Datos Flujo Datos 	Muestra la transferencia de datos entre dos símbolos distintos, cumpliendo las reglas que posteriormente se definirán. Los Flujos de datos pueden tener una única dirección (La punta de flecha en el

	destino), o dos direcciones (La flecha tendrá dos puntas).
<p>Estímulo</p> 	Este objeto, se utiliza para indicar que un determinado proceso que tiene como entrada uno de estos flujos, se iniciará cuando se produzca un cierto estímulo.
<p>Conectores Externos</p> 	Flujos de datos que provienen de un diagrama situado en un nivel superior. En cada flujo se indica el símbolo con el que está conectado. Existen conectores externos de datos, control y creación de procesos.
<p>Separación / Unión (Split/Merge)</p> 	Descompone un flujo de datos compuesto, en sus componentes, o "funde" distintas componentes, en un solo flujo de datos. También permite a un flujo simple, ser dirigido a varios procesos.

*REGLAS A SEGUIR EN UN DFD DE GENE & SARSON.*

Cada uno de los elementos de un diagrama Gene & Sarson deben de seguir un conjunto de reglas para que sea correcto.

Algunas de las normas que se deben cumplir, a la hora de diseñar un modelo de flujo de datos se citan a continuación, a pesar de que muchas de ellas serán controladas directamente por el programa.

*Diagrama de Contexto.*

El diagrama de contexto, que es el nivel más alto de la jerarquía de niveles en un DFD, debe contener como mínimo lo siguiente:

- Uno y SÓLO un proceso de datos, que tendrá como identificador el "0".
- Una Entidad Externa que introduzca datos al sistema.
- Una Entidad Externa que reciba datos del sistema.
- Un Flujo de datos que represente la entrada al sistema.

- Un Flujo de datos que represente la salida del sistema.

### *Diagrama de Flujo de Datos.*

Cualquier DFD que no sea el de contexto, debe contener como mínimo lo siguiente:

- Un Proceso de datos.
- Un Flujo de datos que representa la entrada al proceso.
- Un Flujo de datos que representa la salida del proceso.
- Una Entidad Externa y un Archivo, que estén conectados al Proceso de Datos mediante un Flujo de datos.

### *Procesos de Datos.*

Cada Proceso de datos debe tener al menos un Flujo de datos de entrada y otro de salida.

### *Data Store (Archivos).*

Cada componente de un Flujo de Datos que sale o entra de un Data Store, debe ser un componente del Data Store, o un nombre que identifique el contenido del flujo.

### *Flujos de Datos.*

En general, cada Flujo de datos, debe ser nominado y tener una entrada que le corresponda en el Diccionario de Datos. Normalmente, un Flujos de datos, conectado a un Data Store, (simple o repetido), no necesita ser nombrado y por tanto no debe tener una entrada en el Diccionario de Datos. En estos casos, se asume que el Flujo de Datos, llevará toda la información contenida en el Data Store.

Un Flujo de datos, debe conectar uno de los siguientes pares de objetos:

- Proceso de Datos - Proceso de Datos (Pero no el mismo).
- Proceso de Datos - Entidad Externa (simple o repetida).
- Proceso de Datos - Data Store (simple o repetido).
- Proceso de Datos - Espacio vacío (Interface).
- Proceso de Datos - Split/merge.
- Data Store (simple o repetido) - Split/Merge.
- Split/Merge - Espacio vacío (Interface).

*Estímulo.*

Un Estímulo debe conectar uno de los siguientes pares de objetos:

- Entidad Externa - Proceso de datos.
- Entidad Externa - Data Store.
- Entidad Externa - Split/Merge.
- Proceso de Datos - Data Store.
- Proceso de Datos - Split/Merge.
- Proceso de Datos - Proceso de Datos.
- Proceso de Datos - Interface.
- Data Store - Split/Merge.
- Split/Mege - Interface.

### *Conectores Externos*

Los conectores externos deben de tener al menos una conexión dentro del diagrama, además de cumplir las mismas normas que el tipo de flujo de datos del que proviene.

### *Split/Merges ( Separación / Unión).*

Un Split/Merge, debe tener un Flujo de datos de entrada, y uno de salida como mínimo.

Normalmente, lo que debería pasar sería:

SPLIT: Un flujo de entrada y varios de salida.

MERGE: Varios flujos de entrada y uno de salida.

El Contenido de los flujos de datos de entrada a un Split/Merge, debe ser igual al contenido de los flujos de salida.

### *Balanceado.*

El contenido de un flujo de datos conectado a un Proceso de datos, debe coincidir en contenido y dirección con los Flujo de datos originados y terminados en Interfaces, Entidades Externas, o Data Stores, en el DFD explosionado. Esto es lo que se denomina balanceado.

### *Nombrar y Numerar Objetos.*

Cada Entidad Externa simple, debe tener un único nombre y un único número (en las Entidades Externas este número no es tal, sino que es una letra minúscula), en un DFD dado. Las Entidades Externas repetidas, deben tener el mismo nombre y número.

Cada Data Store simple, debe tener un único nombre y número en un diagrama dado. Los Data Store repetidos, deben tener el mismo nombre y número.

Cada Proceso de datos, Proceso de datos repetido, Data Store, Data Store repetido, Entidad Externa, y Entidad Externa repetida, deben tener nombre y una entrada que le corresponda en el Diccionario de Datos.

El nombre de cada Proceso de datos, al igual que su número, deben ser únicos

El número de un Proceso de datos se construye automáticamente como sigue:

1. El proceso de datos simple del Diagrama de contexto, se numera con el 0.
2. Los Procesos de datos del siguiente nivel se numeran con el 1, 2, 3, etc.
3. Los números de los Procesos de datos en los niveles menores del diagrama, están compuesto del número del Proceso de datos del que el diagrama ha explotado, seguido de un ".", y seguido de un entero positivo 1, 2, 3, etc. Por Ejemplo, podríamos tener: 2.1.1, 2.1.2, 2.1.2.1, etc.

#### *Generalidades.*

Cada Entidad Externa (simple o repetida) y cada Data Store (simple o repetido), deben tener al menos un Flujo de Datos conectado a ellos.

Para cada Proceso de datos, Entidad Externa, Flujo de Datos, o Data Store, que exploten en algo, ese algo debe existir.

Se ha adoptado como regla general, que en los casos en que un proceso realiza las operaciones de *Añadir, Modificar, Consultar y Borrar*, sobre un archivo, el nombre del flujo de datos será "Mantenimiento".

A la hora de realizar el análisis o ingeniería inversa o reingeniería de una aplicación, es importante tener en cuenta las personas a las que va dirigida. Normalmente, el analista no es el único que va a

utilizar la documentación, y por lo tanto es de gran utilidad definir un conjunto de normas que hagan nuestros modelos más claros y fáciles de interpretar.

En el servicio de informática se utilizan normas que, para cada herramienta CASE, dan unas nociones básicas de su funcionamiento, a la vez que dictan las pautas a seguir durante el análisis o modelado de una aplicación.

Durante el proceso de reingeniería se ha seguido la norma de uso de PowerDesigner, creada por un becario de la Consellería de Sanidad y toda la documentación de este proyecto, se ha generado siguiendo la misma .

A continuación, se incluye un pequeño resumen de las convenciones utilizadas en la norma, que servirán de ayuda para interpretar la documentación.

### ***ENTIDAD - RELACIÓN***

Todos los elementos del entidad relación, se caracterizarán porque tendrán dos nombres:

- Nombre Lógico: (campo name) Nombre significativo de la entidad, independientemente de su longitud .
- Nombre Físico: ( campo code) Nombre con el que se denominará a la tabla de la base de datos que representa dicha entidad.

Además del nombre disponemos de tres campos que completan la definición de un elemento, que son:

- Etiqueta : Breve descripción.
- Descripción : Descripción más completa.
- Anotaciones : Registro de anomalías, excepciones...

Todo diagrama dispondrá de una etiqueta con una breve descripción del mismo; proyecto al que pertenece, aplicación, fecha...

El entidad relación dispone de dos vistas:

- Modelo Lógico: En el que lo fundamentan las entidades existentes y las relaciones entre las mismas. Por lo tanto, sólo se mostrarán las entidades, con la clave y las relaciones; ambas con el nombre representativo.
- Modelo Físico: Da una visión global de la estructura de la base de datos. Por lo tanto visualizaremos los modelos utilizando el nombre físico de sus elementos y presentaremos todos los atributos de las entidades.

#### *Entidades:*

Las entidades se modelan con un rectángulo; el cual sombrearemos en el caso en que la entidad represente a una relación con atributos.

#### *Relaciones:*

El nombre (en minúsculas) de las relaciones tendrán el siguiente formato:

inicial entidad origen + nombre representativo de la relación + inicial entidad destino

Sólo pondremos una etiqueta por cada relación, salvo en el caso en que tenga cardinalidad muchos a muchos .

Las ausencias de claves ajenas en la base de datos, se representarán a través del atributo “**nfk**”, se trata de un texto que nos avisa que aunque en el modelo, documentación CASE, si que aparece la relación esta no se ha actualizado todavía en la base de datos.

## ***DIAGRAMA DE FLUJO DE DATOS***

Todo modelo dispondrá de una etiqueta que describa sus características: aplicación, nombre, versión...

### *Proceso :*

Nombre representativo del proceso.

### *Flujos :*

Tendrán un nombre que representen los datos que pasan , el objeto que genera o el elemento de control .

### *Almacenamientos:*

En el caso en que un almacenamiento coincida con una entidad recibirá el mismo nombre que ésta.

Si no tendrá un nombre representativo.

### *Data Item:*

Coinciden con los atributos del modelo entidad relación, ya que se trasvasan de un diagrama a otro.

***ANEXO F***  
***REPORTE DEL MODELO CONCEPTUAL***

# Full CDM report

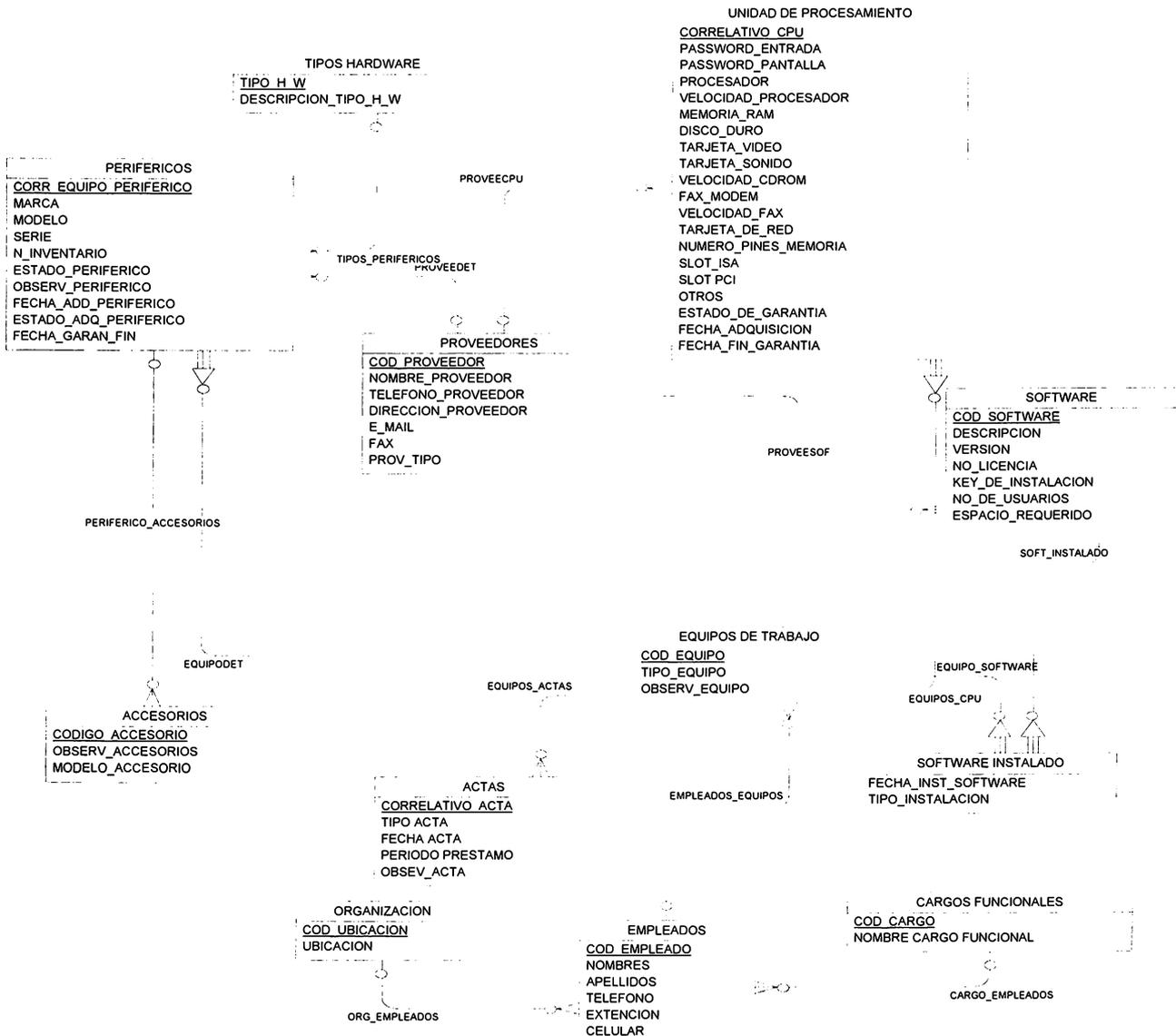
## Table of Contents

The 'Table of contents' field needs to be updated!

## Specifications Specifications

Include specifications here...

## Global model Graph Global model Graph



Model Information **Model Information**

<b>Project Name:</b>	SISTEMA DE INVENTARIO		
<b>Project Code:</b>	sistema_de_inventario		
<b>Name:</b>	INVENTARIO		
<b>Code:</b>	inventario		
<b>Label:</b>			
<b>Author:</b>			
<b>Version:</b>			
<b>Created On:</b>	9/14/99 2:05 PM	<b>Modified On:</b>	9/18/99 9:37 PM

Business Rules **Business Rules**Domains **Domains**Data Items **Data Items**Entities **Entities**Entity List **Entity List**

Name	Code
ACCESORIOS	accesorios
ACTAS	actas
CARGOS FUNCIONALES	cargos_funcionales
EMPLEADOS	empleados
EQUIPOS DE TRABAJO	equipos_de_trabajo
ORGANIZACION	organizacion
PERIFERICOS	perifericos
PROVEEDORES	proveedores
SOFTWARE	software
SOFTWARE INSTALADO	software_instalado
TIPOS HARDWARE	tipos_hardware
UNIDAD DE PROCESAMIENTO	unidad_de_procesamiento

## Entity ACCESORIO Entity ACCESORIOS

<b>Name:</b>	ACCESORIOS
<b>Code:</b>	accesorios
<b>Label:</b>	
<b>Number:</b>	<b>Generate Table:</b> Yes

## Attribute List

Name	Code	Type	I	M
CODIGO_ACCESORIO	codigo_accesorio	A2	Yes	Yes
OBSERV_ACCESORIOS	Observacion_accesorios	TXT	No	No
MODELO_ACCESORIO	modelo_accesorio	A15	No	No

## Data Item CODIGO\_ACCESORIO

## Data Item OBSERV\_ACCESORIOS

## Data Item MODELO\_ACCESORIO

## Reference List

Entity	Card	Dep.	Relationship
PERIFERICOS(perifericos)	0,1	No	PERIFERICO_ACCESORIOS(periferico_accesorios)

## Entity ACTA Entity ACTAS

<b>Name:</b>	ACTAS
<b>Code:</b>	actas
<b>Label:</b>	
<b>Number:</b>	<b>Generate Table:</b> Yes

**Attribute List**

Name	Code	Type	I	M
CORRELATIVO_ACTA	no_acta	I	Yes	Yes
TIPO ACTA	tipo_acta	A1	No	No
FECHA ACTA	fecha_acta	DT	No	No
PERIODO PRESTAMO	periodo_prestamo	A20	No	No
OBSEV_ACTA	obsev_acta	TXT	No	No

**Data Item CORRELATIVO\_ACTA**

**Data Item TIPO ACTA**

**Check**

<b>Low Value:</b>			
<b>High Value:</b>			
<b>Default Value:</b>			
<b>Unit:</b>			
<b>Format:</b>			
<b>Uppercase:</b>	No	<b>Lowercase:</b>	No
<b>List of Values:</b>	E	<b>Can't modify:</b>	No
	R		
			Entrega
			Recepción

**Data Item FECHA ACTA**

**Data Item PERIODO PRESTAMO**

**Data Item OBSEV\_ACTA**

**Reference List**

Entity	Card	Dep.	Relationship
EQUIPOS DE TRABAJO(equipos_de_trabajo)	0,1	No	EQUIPOS_ACTAS(equipos_actas)

**Entity CARGOS FUNCIONALES Entity CARGOS FUNCIONALES**

<b>Name:</b>	CARGOS FUNCIONALES
<b>Code:</b>	cargos_funcionales
<b>Label:</b>	
<b>Number:</b>	7
<b>Generate Table:</b>	Yes

**Attribute List**

Name	Code	Type	I	M
COD_CARGO	cod_cargo	I	Yes	Yes
NOMBRE CARGO FUNCIONAL	nombre_cargo_funcional	A35	No	No

**Data Item COD\_CARGO****Data Item NOMBRE CARGO FUNCIONAL****Reference List**

Entity	Card	Dep.	Relationship
EMPLEADOS(empleados)	0,n	No	CARGO_EMPLEADOS(cargo_empleados)

## Entity EMPLEADOSEntity EMPLEADOS

<b>Name:</b>	EMPLEADOS
<b>Code:</b>	empleados
<b>Label:</b>	
<b>Number:</b>	4
<b>Generate Table:</b>	Yes

**Attribute List**

Name	Code	Type	I	M
COD_EMPLEADO	cod_empleado	A8	Yes	Yes
NOMBRES	Nombres	A35	No	No
APELLIDOS	Apellidos	A35	No	No
TELEFONO	Telefono	A7	No	No
EXTENCION	Extencion	A15	No	No
CELULAR	Celular	A7	No	No

**Data Item COD\_EMPLEADO****Data Item NOMBRES****Data Item APELLIDOS****Data Item TELEFONO**

**Data Item EXTENCION**

**Data Item CELULAR**

**Reference List**

Entity	Card	Dep.	Relationship
CARGOS FUNCIONALES(cargos_funcionales)	0,1	No	CARGO_EMPLEADOS(cargo_empleados)
EQUIPOS DE TRABAJO(equipos_de_trabajo)	0,n	No	EMPLEADOS_EQUIPOS(empleados_equipos)
ORGANIZACION(organizacion)	0,1	No	ORG_EMPLEADOS(org_empleados)

Entity EQUIPOS DE TRABAJO Entity EQUIPOS DE TRABAJO

<b>Name:</b> EQUIPOS DE TRABAJO
<b>Code:</b> equipos_de_trabajo
<b>Label:</b>
<b>Number:</b> 3 <b>Generate Table:</b> Yes

**Attribute List**

Name	Code	Type	I	M
COD_EQUIPO	cod_equipo	I	Yes	Yes
TIPO_EQUIPO	tipo_equipo	I	No	No
OBSERV_EQUIPO	observ_equipo	A50	No	No

**Data Item COD\_EQUIPO**

**Data Item TIPO\_EQUIPO**

**Check**

<b>Low Value:</b>			
<b>High Value:</b>			
<b>Default Value:</b>			
<b>Unit:</b>			
<b>Format:</b>			
<b>Uppercase:</b>	No	<b>Lowercase:</b> No	<b>Can't modify:</b> No
<b>List of Values:</b>	1	PERSONAL	
	2	COMPONENTE EN RED	
	3	PORTATIL	

**Data Item OBSERV\_EQUIPO****Reference List**

Entity	Card	Dep.	Relationship
EMPLEADOS(empleados)	0,1	No	EMPLEADOS_EQUIPOS(empleados_equipos)
SOFTWARE INSTALADO(software_instalado)	0,n	Yes	EQUIPO_SOFTWARE(equipo_software)
PERIFERICOS(perifericos)	0,n	Yes	EQUIPODET(equipo_periferico)
ACTAS(actas)	0,n	No	EQUIPOS_ACTAS(equipos_actas)
UNIDAD DE PROCESAMIENTO(unidad_de_procesamiento)	0,n	Yes	EQUIPOS_CPU(equipos_cpu)

**Entity ORGANIZACION Entity Organización**

<b>Name:</b> ORGANIZACION
<b>Code:</b> organizacion
<b>Label:</b>
<b>Number:</b> 6 <b>Generate Table:</b> Yes

**Attribute List**

Name	Code	Type	I	M
COD_UBICACION	cod_ubicacion	I	Yes	Yes
UBICACION	ubicacion	A50	No	No

**Data Item COD\_UBICACION****Data Item UBICACION****Reference List**

Entity	Card	Dep.	Relationship
EMPLEADOS(empleados)	0,n	No	ORG_EMPLEADOS(org_empleados)

## Entity PERIFERICOS Entity PERIFERICOS

<b>Name:</b>	PERIFERICOS
<b>Code:</b>	perifericos
<b>Label:</b>	
<b>Number:</b>	5
<b>Generate Table:</b>	Yes

## Attribute List

Name	Code	Type	I	M
CORR_EQUIPO_PERIFERICO	corr_equipo_periferico	I	Yes	Yes
MARCA	marca	A15	No	No
MODELO	modelo	A20	No	No
SERIE	serie	A25	No	No
N_INVENTARIO	n_inventario	A15	No	No
ESTADO_PERIFERICO	estado_periferico	A15	No	No
OBSERV_PERIFERICO	observacion_periferico	A50	No	No
FECHA_ADD_PERIFERICO	fecha_add_periferico	DT	No	No
ESTADO_ADQ_PERIFERICO	estado_adq_periferico	A2	No	No
FECHA_GARAN_FIN	fecha_garan_fin	DT	No	No

## Data Item CORR\_EQUIPO\_PERIFERICO

## Data Item MARCA

## Data Item MODELO

## Data Item SERIE

## Data Item N\_INVENTARIO

## Data Item ESTADO\_PERIFERICO

## Data Item OBSERV\_PERIFERICO

## Data Item FECHA\_ADD\_PERIFERICO

## Data Item ESTADO\_ADQ\_PERIFERICO

**Check**

<b>Low Value:</b>			
<b>High Value:</b>			
<b>Default Value:</b>			
<b>Unit:</b>			
<b>Format:</b>			
<b>Uppercase:</b>	No	<b>Lowercase:</b>	No
<b>List of Values:</b>	CG	<b>Can't modify:</b>	No
	SG		
		Con Garantia	
		Sin Garantia	

**Data Item FECHA\_GARAN\_FIN**

**Reference List**

Entity	Card	Dep.	Relationship
EQUIPOS DE TRABAJO(equipos_de_trabajo)	1,1	Yes	EQUIPODET(equipo_periferico)
ACCESORIOS(accesorios)	0,n	No	PERIFERICO_ACCESORIOS(periferico_accesorios)
PROVEEDORES(proveedores)	0,1	No	PROVEEDET(proveedores_periferico)
TIPOS HARDWARE(tipos hardware)	0,1	No	TIPOS_PERIFERICOS(tipos_perifericos)

**Entity PROVEEDORESEntity PROVEEDORES**

<b>Name:</b>	PROVEEDORES
<b>Code:</b>	proveedores
<b>Label:</b>	
<b>Number:</b>	6
<b>Generate Table:</b>	Yes

**Attribute List**

Name	Code	Type	I	M
COD_PROVEEDOR	cod_proveedor	I	Yes	Yes
NOMBRE_PROVEEDOR	nombre_proveedor	A25	No	No
TELEFONO_PROVEEDOR	telefono_proveedor	A7	No	No
DIRECCION_PROVEEDOR	direccion_proveedor	A25	No	No
E_MAIL	e_mail	A25	No	No
FAX	fax	A7	No	No
PROV_TIPO	prov_tipo	I	No	No

**Data Item COD\_PROVEEDOR**

**Data Item NOMBRE\_PROVEEDOR**

**Data Item TELEFONO\_PROVEEDOR**

**Data Item DIRECCION\_PROVEEDOR**

**Data Item E\_MAIL**

**Data Item FAX**

**Data Item PROV\_TIPO**

**Check**

<b>Low Value:</b>			
<b>High Value:</b>			
<b>Default Value:</b>			
<b>Unit:</b>			
<b>Format:</b>			
<b>Uppercase:</b>	No	<b>Lowercase:</b>	No
<b>List of Values:</b>	1		<b>Can't modify:</b> No
	2	CPU	
	3	PERIFERICOS	
		SOFTWARE	

**Reference List**

Entity	Card	Dep.	Relationship
UNIDAD DE PROCESAMIENTO (unidad_de_procesamiento)	0,n	No	PROVEECPU(proveedor_cpu)
PERIFERICOS(perifericos)	0,n	No	PROVEEDET(proveedores_periferico)
SOFTWARE(software)	0,n	No	PROVEESOF(proveedor_sw)

Entity SOFTWARE Entity SOFTWARE

<b>Name:</b>	SOFTWARE
<b>Code:</b>	software
<b>Label:</b>	
<b>Number:</b>	<b>Generate Table:</b> Yes

**Attribute List**

Name	Code	Type	I	M
COD_SOFTWARE	cod_software	I	Yes	Yes
DESCRIPCION	descripcion	A25	No	No
VERSION	version	A15	No	No
NO_LICENCIA	no_licencia	A30	No	No
KEY_DE_INSTALACION	key_de_instalacion	A20	No	No
NO_DE_USUARIOS	no_de_usuarios	I	No	No
ESPACIO_REQUERIDO	espacio_requerido	A25	No	No

**Data Item COD\_SOFTWARE**

**Data Item DESCRIPCION**

**Data Item VERSION**

**Data Item NO\_LICENCIA**

**Data Item KEY\_DE\_INSTALACION**

**Data Item NO\_DE\_USUARIOS**

**Data Item ESPACIO\_REQUERIDO**

**Reference List**

Entity	Card	Dep.	Relationship
PROVEEDORES(proveedores)	0,1	No	PROVEESOF(proveedor_sw)
SOFTWARE INSTALADO (software_instalado)	0,n	Yes	SOFT_INSTALADO(soft_instalado)

Entity SOFTWARE INSTALADO Entity SOFTWARE INSTALADO

<b>Name:</b>	SOFTWARE INSTALADO
<b>Code:</b>	software_instalado
<b>Label:</b>	
<b>Number:</b>	<b>Generate Table:</b> Yes

**Attribute List**

Name	Code	Type	I	M
FECHA_INST_SOFTWARE	fecha_inst_software	DT	No	No
TIPO_INSTALACION	tipo_instalacion	A20	No	No

**Data Item FECHA\_INST\_SOFTWARE****Data Item TIPO\_INSTALACION****Reference List**

Entity	Card	Dep.	Relationship
EQUIPOS DE TRABAJO (equipos_de_trabajo)	1,1	Yes	EQUIPO_SOFTWARE(equipo_software)
SOFTWARE(software)	1,1	Yes	SOFT_INSTALADO(soft_instalado)

**Entity TIPOS HARDWARE Entity TIPOS HARDWARE**

<b>Name:</b>	TIPOS HARDWARE
<b>Code:</b>	tipos_hardware
<b>Label:</b>	
<b>Number:</b>	12
<b>Generate Table:</b>	Yes

**Attribute List**

Name	Code	Type	I	M
TIPO_H_W	tipo_h_w	I	Yes	Yes
DESCRIPCION_TIPO_H_W	descripcion_tipo_h_w	A25	No	No

**Data Item TIPO\_H\_W****Data Item DESCRIPCION\_TIPO\_H\_W****Reference List**

Entity	Card	Dep.	Relationship
PERIFERICOS(perifericos)	0,n	No	TIPOS_PERIFERICOS(tipos_perifericos)

## Entity UNIDAD DE PROCESAMIENTO Entity UNIDAD DE PROCESAMIENTO

<b>Name:</b>	UNIDAD DE PROCESAMIENTO
<b>Code:</b>	unidad_de_procesamiento
<b>Label:</b>	
<b>Number:</b>	<b>Generate Table:</b> Yes

## Attribute List

Name	Code	Type	I	M
CORRELATIVO_CPU	correlativo_cpu	I	Yes	Yes
PASSWORD_ENTRADA	password_entrada	A15	No	No
PASSWORD_PANTALLA	password_pantalla	A15	No	No
PROCESADOR	procesador	I	No	No
VELOCIDAD_PROCESADOR	velocidad_procesador	I	No	No
MEMORIA_RAM	memoria_ram	I	No	No
DISCO_DURO	disco_duro	A10	No	No
TARJETA_VIDEO	tarjeta_video	A1	No	No
TARJETA_SONIDO	multimedia	A1	No	No
VELOCIDAD_CDROM	cdrom	A10	No	No
FAX_MODEM	fax_modem	A1	No	No
VELOCIDAD_FAX	velocidad_fax	I	No	No
TARJETA_DE_RED	tarjeta_de_red	A1	No	No
NUMERO_PINES_MEMORIA	numero_pines_memoria	SI	No	No
SLOT_ISA	slot_isa	SI	No	No
SLOT_PCI	slot_pci	BT	No	No
OTROS	otros	TXT	No	No
ESTADO_DE_GARANTIA	estado_de_garantia	A2	No	No
FECHA_ADQUISICION	fecha_adquisicion	DT	No	No
FECHA_FIN_GARANTIA	fecha_fin_garantia	DT	No	No

**Data Item CORRELATIVO\_CPU**

**Data Item PASSWORD\_ENTRADA**

**Data Item PASSWORD\_PANTALLA**

**Data Item PROCESADOR**

**Data Item VELOCIDAD\_PROCESADOR**

**Data Item MEMORIA\_RAM**



## Relationships Relationships

## Relationship List Relationship List

Name	Code
CARGO_EMPLEADOS	cargo_empleados
EMPLEADOS_EQUIPOS	empleados_equipos
EQUIPO_SOFTWARE	equipo_software
EQUIPODET	equipo_periferico
EQUIPOS_ACTAS	equipos_actas
EQUIPOS_CPU	equipos_cpu
ORG_EMPLEADOS	org_empleados
PERIFERICO_ACCESORIOS	periferico_accesorios
PROVEECPU	proveedor_cpu
PROVEEDET	proveedores_periferico
PROVEESOF	proveedor_sw
SOFT_INSTALADO	soft_instalado
TIPOS_PERIFERICOS	tipos_perifericos

## Relationship CARGO\_EMPLEADOS Relationship CARGO\_EMPLEADOS

<b>Name:</b>	CARGO_EMPLEADOS
<b>Code:</b>	cargo_empleados
<b>Label:</b>	
<b>Entity 1:</b>	CARGOS FUNCIONALES
<b>Entity 2:</b>	EMPLEADOS
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	No
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	0, 1

## Relationship EMPLEADOS\_EQUIPOS Relationship EMPLEADOS\_EQUIPOS

<b>Name:</b>	EMPLEADOS_EQUIPOS
<b>Code:</b>	empleados_equipos
<b>Label:</b>	
<b>Entity 1:</b>	EMPLEADOS
<b>Entity 2:</b>	EQUIPOS DE TRABAJO
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	No
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	0, 1

## Relationship EQUIPO\_SOFTWARE Relationship EQUIPO\_SOFTWARE

<b>Name:</b>	EQUIPO_SOFTWARE
<b>Code:</b>	equipo_software
<b>Label:</b>	
<b>Entity 1:</b>	EQUIPOS DE TRABAJO
<b>Entity 2:</b>	SOFTWARE INSTALADO
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	Yes
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	Yes
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	1, 1

## Relationship EQUIPODETRelationship EQUIPODET

**Name:** EQUIPODET  
**Code:** equipo\_periferico  
**Label:**  
**Entity 1:** EQUIPOS DE TRABAJO  
**Entity 2:** PERIFERICOS  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** Yes

## Entity 1 --&gt; Entity 2:

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

## Entity 2 --&gt; Entity 1:

**Role:**  
**Mandatory:** Yes  
**Dominant:** Yes  
**Min, Max:** 1, 1

## Relationship EQUIPOS\_ACTASRelationship EQUIPOS\_ACTAS

**Name:** EQUIPOS\_ACTAS  
**Code:** equipos\_actas  
**Label:**  
**Entity 1:** EQUIPOS DE TRABAJO  
**Entity 2:** ACTAS  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** No

## Entity 1 --&gt; Entity 2:

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

## Entity 2 --&gt; Entity 1:

**Role:**  
**Mandatory:** No  
**Dominant:** Yes  
**Min, Max:** 0, 1

## Relationship EQUIPOS\_CPU Relationship EQUIPOS\_CPU

**Name:** EQUIPOS\_CPU  
**Code:** equipos\_cpu  
**Label:**  
**Entity 1:** EQUIPOS DE TRABAJO  
**Entity 2:** UNIDAD DE PROCESAMIENTO  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** Yes

**Entity 1 --> Entity 2:**

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

**Entity 2 --> Entity 1:**

**Role:**  
**Mandatory:** Yes  
**Dominant:** Yes  
**Min, Max:** 1, 1

## Relationship ORG\_EMPLEADOS Relationship ORG\_EMPLEADOS

**Name:** ORG\_EMPLEADOS  
**Code:** org\_empleados  
**Label:**  
**Entity 1:** ORGANIZACION  
**Entity 2:** EMPLEADOS  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** No

**Entity 1 --> Entity 2:**

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

**Entity 2 --> Entity 1:**

**Role:**  
**Mandatory:** No  
**Dominant:** Yes  
**Min, Max:** 0, 1

## Relationship PERIFERICO\_ACCESORIOS Relationship PERIFERICO\_ACCESORIOS

**Name:** PERIFERICO\_ACCESORIOS  
**Code:** periférico\_accesorios  
**Label:**  
**Entity 1:** PERIFERICOS  
**Entity 2:** ACCESORIOS  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** No

**Entity 1 --> Entity 2:**

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

**Entity 2 --> Entity 1:**

**Role:**  
**Mandatory:** No  
**Dominant:** Yes  
**Min, Max:** 0, 1

## Relationship PROVEECPU Relationship PROVEECPU

**Name:** PROVEECPU  
**Code:** proveedor\_cpu  
**Label:**  
**Entity 1:** PROVEEDORES  
**Entity 2:** UNIDAD DE PROCESAMIENTO  
**Cardinality:** One to Many  
**Entity 2 dependent of Entity 1:** No

**Entity 1 --> Entity 2:**

**Role:**  
**Mandatory:** No  
**Dominant:** No  
**Min, Max:** 0, n

**Entity 2 --> Entity 1:**

**Role:**  
**Mandatory:** No  
**Dominant:** Yes  
**Min, Max:** 0, 1

## Relationship PROVEEDETRelationship PROVEEDET

<b>Name:</b>	PROVEEDET
<b>Code:</b>	proveedores_periferico
<b>Label:</b>	
<b>Entity 1:</b>	PROVEEDORES
<b>Entity 2:</b>	PERIFERICOS
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	No
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	0, 1

## Relationship PROVEESOFRelationship PROVEESOF

<b>Name:</b>	PROVEESOF
<b>Code:</b>	proveedor_sw
<b>Label:</b>	
<b>Entity 1:</b>	PROVEEDORES
<b>Entity 2:</b>	SOFTWARE
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	No
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	0, 1

## Relationship SOFT\_INSTALADO Relationship SOFT\_INSTALADO

<b>Name:</b>	SOFT_INSTALADO
<b>Code:</b>	soft_instalado
<b>Label:</b>	
<b>Entity 1:</b>	SOFTWARE
<b>Entity 2:</b>	SOFTWARE INSTALADO
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	Yes
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	Yes
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	1, 1

## Relationship TIPOS\_PERIFERICOS Relationship TIPOS\_PERIFERICOS

<b>Name:</b>	TIPOS_PERIFERICOS
<b>Code:</b>	tipos_perifericos
<b>Label:</b>	
<b>Entity 1:</b>	TIPOS HARDWARE
<b>Entity 2:</b>	PERIFERICOS
<b>Cardinality:</b>	One to Many
<b>Entity 2 dependent of Entity 1:</b>	No
<b>Entity 1 --&gt; Entity 2:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	No
<b>Min, Max:</b>	0, n
<b>Entity 2 --&gt; Entity 1:</b>	
<b>Role:</b>	
<b>Mandatory:</b>	No
<b>Dominant:</b>	Yes
<b>Min, Max:</b>	0, 1

Inheritances Inheritances