Capítulo 2 Marco Teórico

Se pretende desarrollar un software que pueda ser aplicado como una herramienta útil para la administración de una empresa. Es necesario tener en cuenta que, en todo desarrollo de sistemas de software es de suma importancia definir una metodología. Esta permite a los desarrolladores seguir alguna especificación en cada una de las etapas del desarrollo del sistema, desde los requerimientos iniciales hasta las pruebas finales, que haga que el software sea coherente y además formal.

En éste capítulo abordaremos los conceptos computacionales tomados en cuenta durante todo el proceso de elaboración del software de este proyecto. Los conceptos que a continuación trataremos son la ingeniería de software y metodología orientada a objetos, las cuales darán la pauta sobre los estándares utilizados tanto para el análisis, diseño, implementación, pruebas y mantenimiento de la aplicación; la re-ingeniería examinará la aplicación existente para actualizarla y mejorarla; las bases de datos permitirán el manejo y manipulación de la gran cantidad de datos que existan; y los bussiness process o workflows ayudarán en la automatización de ciertas tareas.

2.1 Ingeniería de Software

El término 'Ingeniería de Software' fue introducido por primera vez a finales de 1960 en una conferencia destinada a su discusión, la cual fue posteriormente llamada 'crisis del software'. Esta crisis de software fue el resultado directo de la introducción del hardware de la tercera generación computacional [Sommerville, 1989].

Para tener una idea clara de lo que es la ingeniería de software vamos a definirlo según varios autores:

(1) *La aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software; es decir, la aplicación de ingeniería al software.*

(2) *Es una disciplina o área de la Informática o Ciencias de la Computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo* [Pressman, 1998].

El factor común en estas definiciones es que la ingeniería de software se enfoca a los sistemas computacionales, utilizando los principios de la ingeniería para el desarrollo de estos sistemas, y esta compuesta por aspectos técnicos y no técnicos.

La ingeniería de Software no es una disciplina que sólo deba aplicarse en proyectos de ciertas áreas, sino que también trata con áreas diversas dentro de las ciencias computacionales, tales como: construcción de compiladores, sistemas operativos, o desarrollos empresariales como es el caso de ésta aplicación de software. La Ingeniería de Software abarca todas las fases del ciclo de vida en el desarrollo de cualquier sistema de información aplicables a áreas tales como investigación científica, medicina, logística, y -para este caso particular- negocios.

En un nivel técnico la ingeniería de software empieza con una serie de tareas de modelado que llevan a una especificación completa de los requisitos y a una representación del diseño general del software a construir. Con los años se han propuesto muchos métodos para el modelado del análisis. Sin embargo, ahora dos tendencias dominan el modelado del análisis, el análisis estructurado y el análisis orientado a objetos.

2.2 Metodología orientada a objetos

Vivimos en un mundo de objetos. Estos objetos existen el a naturaleza, en entidades y en los productos que usamos. Los objetos pueden ser clasificados, descritos, organizados, combinados, manipulados y creados. Es por esto que se propuso un análisis y desarrollo orientado a objetos, que nos permita aprovechar las características, individualidad y facilidad de manipulación que nos ofrecen los objetos.

Es así que al estar hablando de objetos es importante describir las ideas fundamentales implícitas en la tecnología orientada a objetos incluyen [Martin, 1992]:

- **Objetos.** Un objeto es cualquier cosa, real o abstracta, acerca de la cual almacenamos datos y aquellos métodos que los manipulan.
- **Clases.** Una clase es la implementación de un tipo de objeto. Especifica la estructura de datos y los métodos operacionales permitidos que se aplican a cada uno de sus objetos.
- **Métodos.** Especifica la manera en la cual los datos de un objeto son manipulados. Los métodos en un tipo de objeto hacen solamente referencia a la estructura de datos de ese tipo de objeto. No deben de accesar directamente a la estructura de datos de otro objeto.
- **Peticiones.** Una petición solicita una operación específica debe ser invocada usando uno o varios objetos como parámetros.

Una vez que se han mencionado las ideas fundamentales del modelo orientado a objetos, es importante saber que existen tres conceptos importantes que diferencian el enfoque OO de la ingeniería del software convencional: (1) *encapsulamiento* empaqueta los datos y las operaciones que manejan estos datos en un objeto simple con denominación; (2) *herencia* permite que los atributos y operaciones de una clase sean heredados por todas las subclases y objetos que se instancian de ella; y (3) *polimorfismo* permite que una cantidad de operaciones diferentes posean el mismo nombre, reduciendo la cantidad de líneas de código necesarias para implementar un sistema y facilita los cambios en caso que se produzcan.

Como sabemos, los objetos están compuestos por *atributos* los cuales describen un objeto; que en esencia, son los que definen al objeto, a la vez que clarifican lo que se representa con el objeto en el contexto del espacio del problema.

Para poder manipular los atributos de los objetos existen los algoritmos que los procesan, los cuales son llamados *operaciones, métodos o servicios* y pueden ser vistos

como módulos en un sentido convencional. Cada una de las operaciones encapsuladas por un objeto proporciona una representación de uno de los comportamientos del objeto. Las operaciones definen el comportamiento de un objeto y cambian, de alguna manera, los atributos de dicho objeto.

No sólo se requiere conocer la forma en la que los objetos interactúan entre sí, sino también es necesario saber que el proceso se mueve a través de una espiral evolutiva, que comienza con la comunicación con el usuario. Es aquí donde se define el dominio del problema y se identifican las clases básicas del problema como se muestra en la figura 2.1 [Pressman, 1998]. Esta es la metodología que se empleará para el desarrollo de la aplicación.

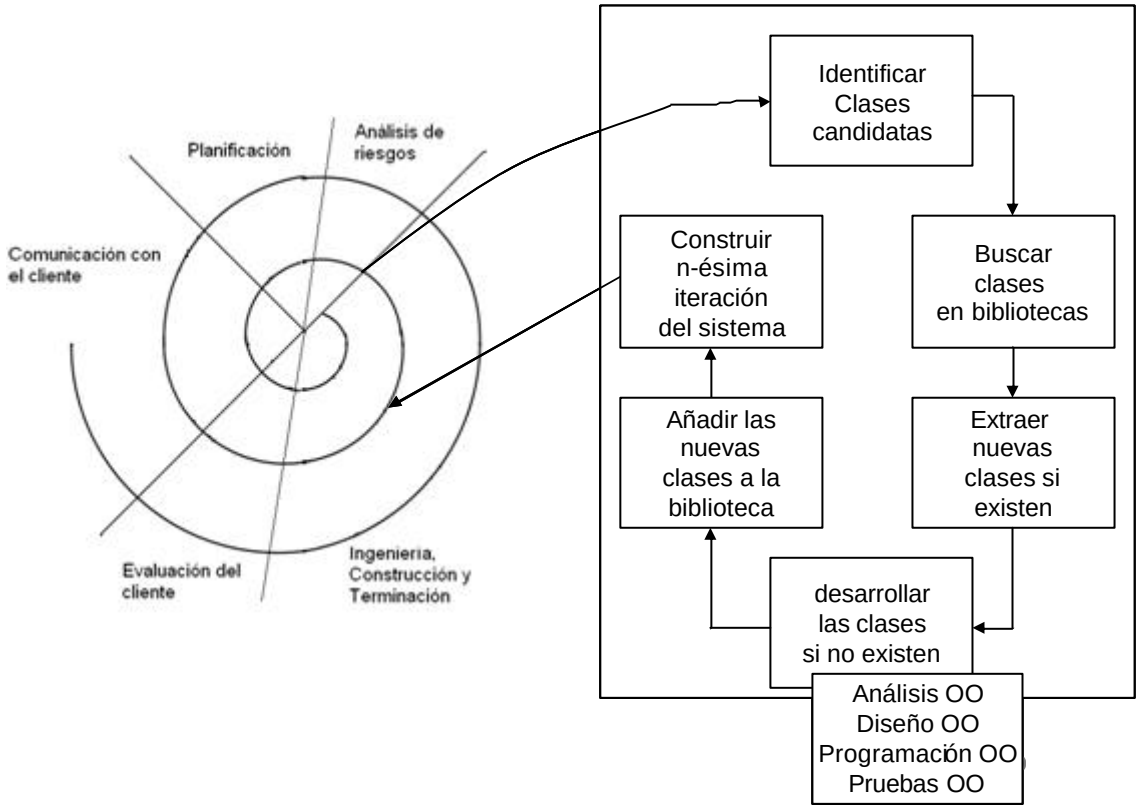

Figura 2.1 Modelo de procesos OO

El análisis y diseño orientado a objetos tiene dos aspectos. Al primer aspecto le conciernen los tipos de objeto, clases, relaciones entre los objetos y la herencia, y se conoce como el *Análisis de Estructura de Objetos* (AEO) y *Diseño de Estructura de Objetos* (DEO). Al otro aspecto le concierne el comportamiento de los objetos y que les pasa con el tiempo, y se conoce como el *Análisis del Comportamiento de Objetos* (ACO) y *Diseño del Comportamiento de Objetos* (DCO) [Martin, 1992].

2.3 Reingeniería

La reingeniería se produce en dos niveles distintos de abstracción. En el nivel de negocios, la reingeniería se concentra en el proceso de negocios con la intención de efectuar cambios que mejoren la competitividad en algún aspecto de los negocios. En el nivel del software la reingeniería examina los sistemas y aplicaciones de información con la intención de reestructurarlos o reconstruirlos de tal modo que muestren una mayor calidad.

La reingeniería de procesos de negocios (BPR) define los objetivos de negocios, identifica y evalúa los procesos de negocio ya existentes (en el contexto de los objetivos definidos), especifica y diseña los procesos revisados, y construye prototipos, refina e instancia esos procesos en el seno de un negocio. Al igual que la ingeniería de información, BPR suele ser la definición de formas en que las tecnologías de la información puedan prestar un mejor apoyo a los negocios [Pressman, 1998].

Es así que la re-ingeniería es el proceso de examinar un software, programa, existente y/o modificarlo con la ayuda de herramientas automatizadas para:

- Mejorar su futuro mantenimiento.
- Actualizar su tecnología.
- Extender su expectación de vida.
- Capturar sus componentes en un repositorio, donde las herramientas CASE (Computer-Aided Software Engineering) pueden ser utilizadas para mantenerlo.
- Incrementar su productividad de mantenimiento [McClure, 1992].

La reingeniería usualmente implica cambiar la forma, cambiar los nombres de los datos y sus definiciones, reestructurar los procesos lógicos, de un programa y mejorar su documentación. En este caso, la funcionalidad, comportamiento, del programa no cambia; sino, únicamente se modifica su forma. En otros casos, el proceso de reingeniería va más allá de la forma e incluye el rediseño cambiando la funcionalidad del programa para alcanzar los requerimientos del usuario. De los diferentes tipos de reingeniería existentes, la empleada en este proyecto fue la de análisis.

2.3.1 Tipos de reingeniería

Análisis

Es el proceso de examinar la cartera de sistemas existentes para entender mejor los componentes de los sistemas y como funciona el programa, para identificar los mejores candidatos para reingeniería, y para medir la calidad del sistema.

Reestructuración

Es el proceso de cambiar la forma del software, las definiciones y nombres de los datos y el código del programa, sin alterar su funcionalidad. El objetivo principal de la reestructuración es hacer el programa más fácil de entender.

Ingeniería inversa

Es el proceso de analizar un software, programa, para reconstruir la descripción de sus componentes y de la interrelación entre ellos. Una descripción de nivel superior del programa es recuperada de su nivel inferior, forma física. El objetivo de la ingeniería inversa es redocumentar el sistema y descubrir la información de diseño como una ayuda para incrementar el entendimiento del programa. Las herramientas de ingeniería inversa extraen información acerca de los datos, arquitectura y diseño de procedimientos de un programa ya existente.

Migración

Es el proceso de convertir un sistema computacional, programa, de un lenguaje a otro moviéndolo de un sistema operativo a otro, o actualizando su tecnología.

2.4 Arquitectura cliente-servidor

El término cliente-servidor se refiere a una arquitectura o división lógica de responsabilidades; donde el cliente (parte frontal o aplicaciones para el usuario o interfaces) es la aplicación que se ejecuta sobre el DBMS, aplicaciones escritas por el usuario y aplicaciones integradas; y el servidor (parte dorsal o servicios de fondo) es el DBMS y soporta la definición, manipulación, seguridad e integridad de los datos entre otros [Date, 2001].

El uso de la arquitectura cliente-servidor brinda ciertas ventajas como son:

- El servidor puede ser una máquina construida a la medida y por lo tanto proporcionar un mejor desempeño.
- Maneja el procesamiento paralelo normal, es decir el procesamiento del servidor y del cliente se están haciendo en paralelo, por lo que el tiempo de respuesta y velocidad real de trasporte mejoran.
- Varias máquinas cliente pueden acceder a la misma máquina servidor y por lo tanto una sola base de datos puede ser compartida entre varios sistemas clientes distintos.

La figura 2.2 muestra un ejemplo de la arquitectura cliente-servidor, donde existe un servidor y varios clientes.

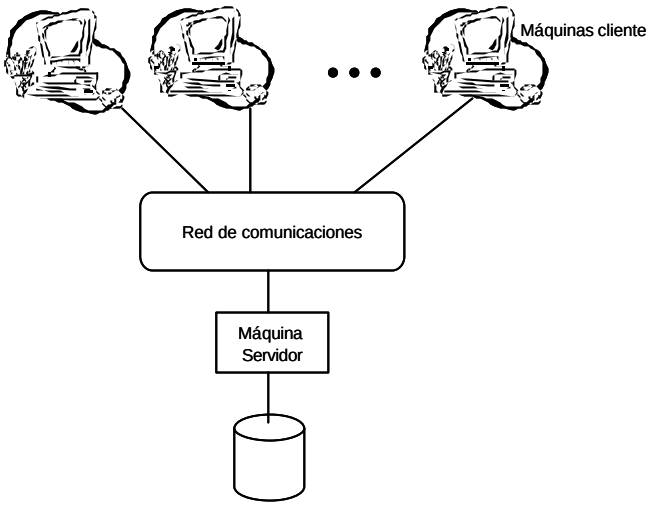

Figura 2.2 Arquitectura cliente-servidor

2.5 Modelo entidad-relación

Es un acercamiento descendente, cuya secuencia de operación es la siguiente:

a) selección de entidades, y de las relaciones entre ellas
b) asignación de atributos a esas entidades y relaciones de forma que se obtengan tablas completamente normalizadas.

Los conceptos básicos de este modelo son:

- **Entidad**: es una cosa, objeto, concepto, que la empresa reconoce que puede tener una existencia independiente, y puede ser identificado por si mismo. Por lo general se utilizan sustantivos para identificar las entidades. Por ejemplo maquinas, clientes, entre otros.
- **Atributo**: es una propiedad de la entidad. Por ejemplo los atributos de la entidad cliente pueden ser número de cliente, nombre.
- **Relación**: es una asociación entre dos o más entidades; por lo general se utilizan verbos para identificar las relaciones. Por ejemplo la entidad departamento puede estar asociada con la entidad empleado mediante la relación emplear.

Existen diferentes grados de relación (cardinalidad que es la especificación del número de ocurrencias de un objeto que se relaciona con ocurrencias de otro) dentro del diagrama entidad-relación como lo muestra la figura 2.3.

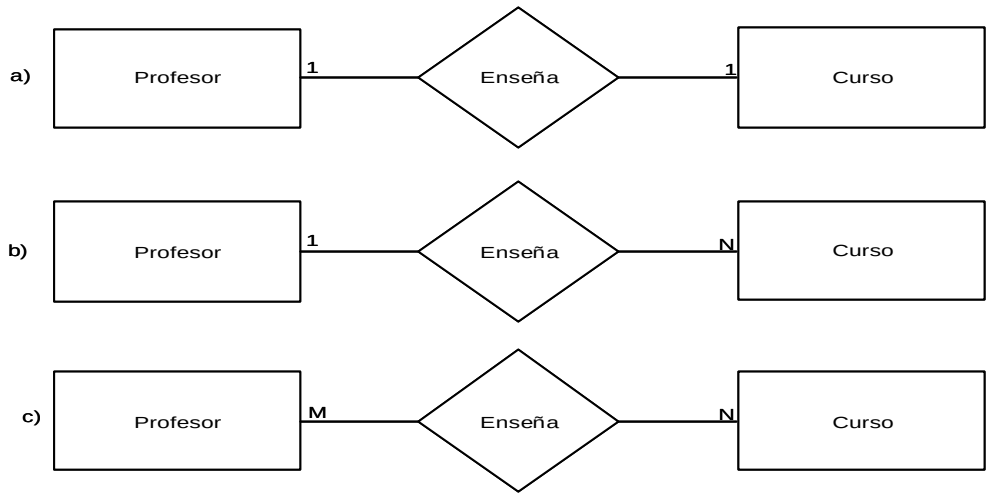

Figura 2.3 a) representa la relación 1 : 1, b) representa la relación 1 : muchos y c) representa la relación muchos : muchos.

2.5.1 Diagrama entidad-relación

Es un diagrama que muestra ocurrencias de entidades individuales y sus relaciones y proporciona un medio sencillo y de fácil comprensión para comunicar las características sobresalientes del diseño de cualquier base de datos.

La convención que se utilizará para dibujar el diagrama entidad-relación es la siguiente: las entidades serán representadas mediante rectángulos y las relaciones mediante rombos. Los conectores que mostrarán que entidades están asociadas a que relaciones son líneas. Tanto las entidades como las relaciones tendrán un nombre, como se muestra en la figura 2.4 [Howe, 1983].

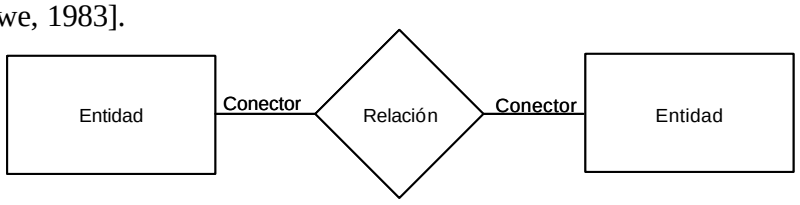

Figura 2.4 Diagrama ER

2.6 Workflows

Cada persona o grupo de personas están dedicados a la producción de un solo producto, pues sería ineficiente que cada uno creara todos los productos que necesita. Es por eso que la sociedad esta organizada en "unidades empresariales", en las cuales se satisfacen necesidades específicas de una manera eficiente y para esto es necesario un alto grado de especialización y control de los procesos que se llevan a cabo para obtener los productos.

Estas unidades empresariales tiene procesos empresariales (*Bussiness process*). Un proceso empresarial son descripciones centradas en el mercado, de las tareas de una organización, implementadas como procesos de información y/o procesos de materiales. Un proceso empresarial es creado para cumplir un contrato empresarial o satisfacer una necesidad específica de un consumidor. Por lo tanto, la noción de un proceso empresarial es conceptualmente un nivel más alto que la noción de procesos de información o de materiales.

Por otro lado, para comprender que es un workflow, primero debemos entender que es un trabajo. Existen muchos tipos de trabajos de los cuales obtenemos cosas. A estas "cosas" las llamaremos *casos*. Cada caso requiere de un *proceso*. Un proceso consiste de una serie de *tareas* que necesitan ser ejecutadas bajo un conjunto de *condiciones* que determina el orden de las tareas. Un proceso también es llamado *procedimiento*. Una tarea es una unidad lógica de trabajo que es realizada por un *recurso*, esto es, una persona o una máquina, es un proceso que no puede subdividirse más, un proceso atómico. Dos o más tareas que deben realizarse en un orden determinado que considera una *secuencia*. En ocasiones no es necesario realizar todas las tareas de un proceso y la decisión entre una u otra tarea es llamado *selección*. También hay tareas que pueden realizarse de manera *paralela* y necesita llevar una *sincronización*. Los procesos pueden tener *iteraciones* o repetición de algunas tareas.

A continuación presentamos un *diagrama de proceso* de reclamación de un seguro. La sintaxis del diagrama es:

1. Una flecha de la tarea A dirigida a la tarea B indica que la tarea A debe realizarse antes que la tarea B.
2. Cada tarea está representada por un rectángulo.
3. Si de una tarea surgen 2 o más flechas, debe seleccionarse una de las tareas siguientes.
4. Si una tarea tiene más de un predecesor, todos deben realizarse para poder continuar.
5. Los círculos blancos tienen varios precursores y sólo una tarea subsiguiente. Indican que solo una de las tareas siguientes necesita ser realizada para continuar.
6. Los círculos negros tienen un predecesor y varios sucesores. Indican que todas las tareas siguientes deben ser realizadas.

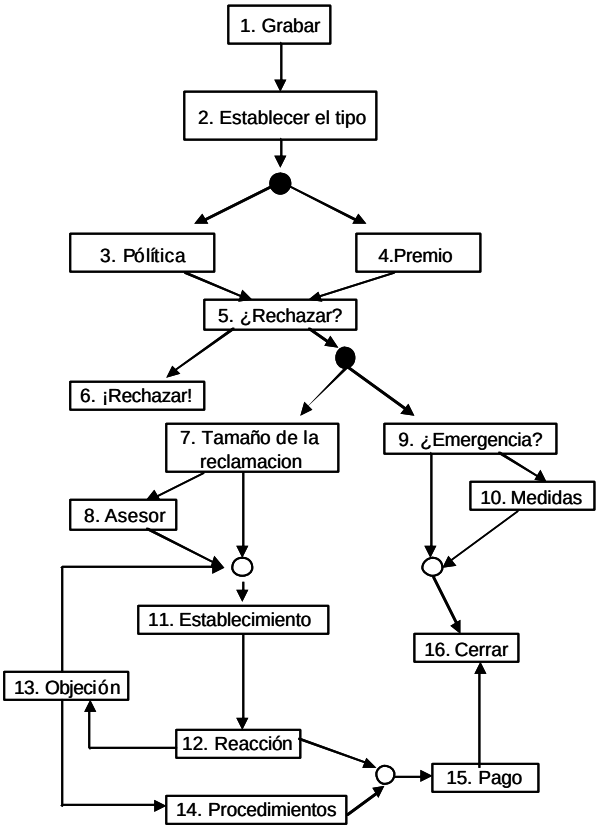

Figura 2.5 Diagrama de proceso

Para resumir, podemos identificar cuatro mecanismos básicos en las estructuras de los procesos: secuencia, selección, paralelización e iteración.

Los procesos pueden dividirse en:

Primarios: Son aquellos en los que se originan los productos de una compañía. También conocidos como *procesos de producción*. Son procesos que generan ingresos para la compañía y están claramente orientados a los consumidores. Este proyecto contiene un módulo para el control sobre las peticiones de producción.

Secundarios: Dan soporte a los primarios, es decir, son *procesos de soporte.* Por ejemplo: mantenimiento a la maquinaria o capacitación del personal.

Terciarios: Son los *procesos administrativos* que dirigen y coordinan los procesos primarios y secundarios. Es en estos en los que los administrativos de otros procesos se reúnen para establecer las condiciones y recursos a utilizarse en los procesos primarios y secundarios. En estos procesos también se incluyen la realización de presupuestos. Estos procesos conforman la mayor parte del sistema desarrollado.

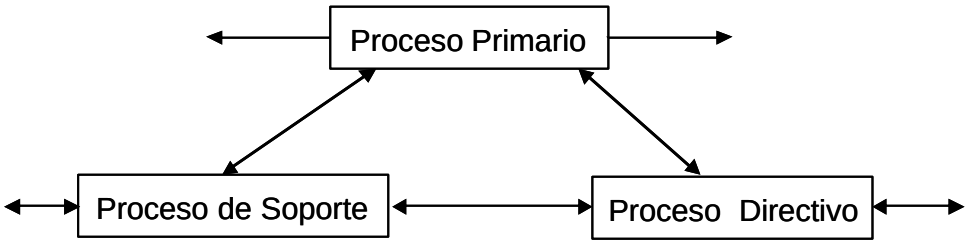

Figura 2.6 Relación de los procesos

Con los conceptos vistos anteriormente podemos dar una definición de workflow fácil de comprender. Un workflow es la representación computacional de una empresa. Especifica las diferentes tareas de un proceso empresarial que deben ser ejecutadas en un orden, el flujo de datos entre las mismas y los múltiples agentes que colaboran en la ejecución de éstas para cumplir un objetivo común [BeVaCo,2004]. Es una colección de tareas organizadas para ejecutar un proceso definiendo las condiciones bajo las cuales las tareas deben ser invocadas y su sincronización, en donde, eventualmente, un workflow impacta en otros.

Existen diferentes técnicas de modelado de procesos para definir la ruta detallada y los requerimientos para los procesos de un workflow típico, por ejemplo el modelo de proceso con cadenas de decisiones o el modelo de proceso con flujo de eventos.

El modelo de proceso con cadenas de decisiones. Utiliza milestones y puntos de decisión para seguir un proceso. El Modelo de proceso con flujo de eventos describe al proceso como una cadena de eventos manuales y automáticos y permite la inclusión de un buen nivel de detallado.

La integración de workflows en un sistema que los administre nos lleva a un *Sistema de administración de Workflows WFMS*, que es un sistema que define, administra, y ejecuta procesos en workflows durante la ejecución del software en el cual el orden de ejecución de estos es dirigida por una representación computacional de la lógica de los procesos en el workflow [Ellis, 1999].

Finalmente, mencionaremos que James G. Kobelius, en su libro Workflow Strategies propone cuatro categorías de workflows que se distinguen principalmente por el mecanismo de trasporte utilizado para dirigir los elementos del trabajo:

1. **Sistemas de workflows de producción.** Estos sistemas realizan la parte tradicional del mercado. En ocasiones son llamados sistemas basados en almacenamiento de archivos, sistemas de procesamiento de copias de documentos o sistemas de administración de formas. Estos sistemas envían los archivos a carpetas consistentes de una o más formas, o de diferentes tipos de documentos de la organización. Generalmente almacenan documentos en un repositorio central que provee check-in, check-out y control de versiones de esos documentos.
2. **Sistemas de workflows basados en mensajes**. También llamados Sistemas de Workflows Administrativos, que comprenden el segmento más bajo del mercado. Los productos contenidos son herramientas stand-alone que envían los documentos por medio de sistemas de correo electrónico ya existentes, ya sea como documentos o como archivos adjuntos.

3. **Sistemas de workflows basados en Web.** Estos sistemas son los más populares en el desarrollo de aplicaciones. Aprovechando la popularidad obtenida por el WWW, estos sistemas utilizan el mismo ambiente para implementar las capacidades de los workflows. Los sistemas en esta categoría utilizan clientes y servidores Web para liberar sus funcionalidades.
4. **Sistemas de workflows basados en conjuntos.** Los productos en esta categoría ofrecen conjuntos de aplicaciones integradas de oficina como procesador de texto, hojas de cálculo, presentaciones y correo electrónico. En un Sistema de Workflows basado en Conjuntos, todas las aplicaciones están integradas con el sistema de correo electrónico. Esta integración generalmente se logra a través de comandos para envío de correos en aplicaciones que no contienen correo electrónico.