

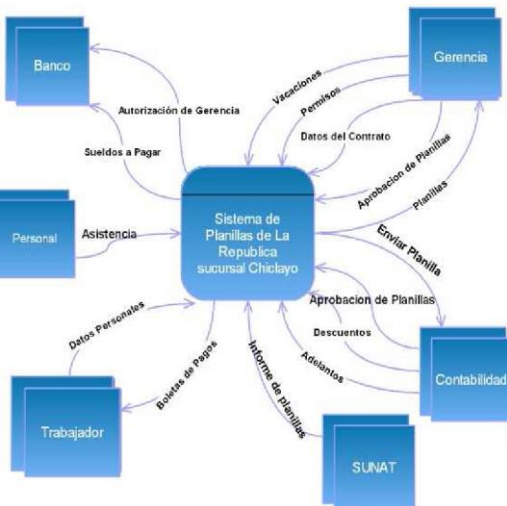


UNIVERSIDAD CATÓLICA
SANTO TORIBIO DE MOGROVEJO

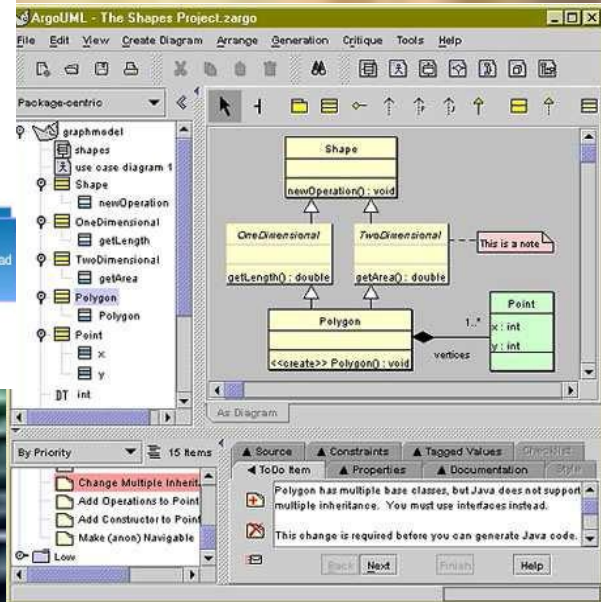
Análisis de Sistemas

MODELAMIENTO ESTRUCTURADO

Diagrama de Contexto:
Sistema de Planillas de la Empresa La República sucursal



MODELAMIENTO ORIENTADO A OBJETOS



El Autor:

Hugo José Luís Romero Ruiz, es profesor-asesor de la asignatura de Análisis de Sistemas de la Universidad Católica Santo Toribio de Mogrovejo, ciudad de Chiclayo, Región Lambayeque, PERÚ



Grado Académico

Bachiller en Ingeniería de Sistemas, Universidad Privada del Norte, Trujillo, Perú

Título Profesional

Ingeniero de Sistemas, Universidad Privada del Norte, Trujillo, Perú

Segunda Especialidad

Especialista en Tecnología Educativa: Mención en Administración y Gerencia Educativa, Facultad de Educación y Ciencias de la Comunicación, Universidad Nacional de Trujillo, Trujillo, Perú

Ha ejercido la docencia desde 1983, en diversas instituciones de nivel superior tecnológico, pedagógico y universitario, en la especialidad de informática y sistemas, conduciendo asignaturas relacionadas al análisis y desarrollo de sistemas, empleando herramientas CASE para el modelado de sistemas.

Asimismo, se ha desempeñado como ingeniero de software y liderado equipos de análisis, diseño, y construcción e implementación de sistemas, para empresas relacionadas con los sectores comercial e industrial del país.

Desde el 2006, es profesor del departamento de Ingeniería de Sistemas, Computación e Informática de la Universidad Católica Santo Toribio de Mogrovejo

INTRODUCCION

Desarrollar proyectos de software en el pasado causó siempre bastantes confusiones y malas interpretaciones entre clientes y usuarios, en parte debido a la abundancia de notaciones, metodologías y conceptos que dieron lugar a que los desarrolladores de sistemas no se pongan de acuerdo en que es lo que realmente estaban elaborando. En un esfuerzo para estandarizar las notaciones y procesos, se conformó un consorcio liderado por varias empresas del mundo de la industria, entre ellas, Microsoft, Oracle, Sun Microsystems, Intellicorp, IBM, AMD y otras, quienes desarrollaron una notación llamada UML siglas de Unified Modeling Language.

En la época previa existían una gama de técnicas y herramientas de desarrollo de software, donde todos los desarrolladores en la década de los 80's, realizaban el software de forma artística, es decir utilizando métodos y técnicas adhoc donde la experiencia (el ensayo-error). Este enfoque produjo grandes y exitosos productos de programación pero conforme los proyectos se volvieron más complejos debido al avance del hardware y software y la penetración cada vez mayor de la informática en todos los ámbitos de la sociedad, llevó a que se produjera software sin calidad, se incumplieran los presupuestos y se incrementaran dramáticamente los costos de mantenimiento. La solución propuesta fue aplicar métodos y principios que han sido utilizados y probados en la experiencia de desarrollo de software para producir de forma inequívoca productos que funcionen eficientemente y se ejecuten sobre máquinas reales. En la década de los 70 surgieron una gran variedad de metodologías y metodologías entre ellos se destacan Yourdon y Demarco cuyas investigaciones se basaban en los principios de la programación estructurada. En los 80's y 90's el paradigma estructurado evolucionó hacia el paradigma orientado a objetos, en el período de 1989 y 1994 se creó la llamada guerra de métodos dentro de la comunidad orientada a objetos existiendo un incremento de menos de diez a más de cincuenta metodologías, es así que los desarrolladores de software quedaron muy confundidos sin saber cual era la metodología más adecuada para elaborar sus proyectos.

El presente manual introducirá a los estudiantes de la asignatura de análisis de sistemas en el estudio básico de las dos estrategias más importantes para el modelamiento de proyectos de sistemas informáticos, estas son el modelamiento estructurado y el orientado a objetos.

INDICE TEMÁTICO

UNIDAD I CONCEPTOS DE ANÁLISIS DE SISTEMAS

- 1.1 Análisis y sistemas
- 1.2 Lo que no es el análisis de sistemas
- 1.3 Sistema, y términos asociados
- 1.4 Esquema de un sistema de información
- 1.5 Características del sistema de información
- 1.6 Tipos de sistemas de información
- 1.7 Sistemas de apoyo para la toma de decisiones (DSS: Decision Support Systems)
- 1.8 Sistemas de soporte para la toma de decisiones de grupo (GDSS - Group decision Support Systems).
- 1.9 Sistemas expertos de soporte para la toma de decisiones (DESS - Expert Decision Supprt Systems).
- 1.10 Sistemas de información para ejecutivos (EIS - Executive Information Systems).
- 1.11 Sistemas gerenciales
- 1.12 ¿Por qué es importante el análisis y diseño de sistemas?
- 1.13 Tipos de usuarios de sistemas
- 1.14 Usuarios operacionales
- 1.15 Usuarios supervisores
- 1.16 Usuarios ejecutivos o funcionarios
- 1.17 El analista de sistemas
- 1.18 Roles del analista de sistemas
- 1.19 El analista de sistemas como consultor
- 1.20 El analista de sistema como especialista de apoyo
- 1.21 El analista de sistema como agente de cambio
- 1.22 Cualidades del analista de sistemas

UNIDAD II CICLO DE VIDA DEL DESARROLLO DE SISTEMAS

- 2.1 Concepto
- 2.2 Objetivos del ciclo de vida del desarrollo de sistemas:
- 2.3 Etapas del ciclo de vida del desarrollo de sistemas:
- 2.4 Identificación de problemas, oportunidades y objetivos
- 2.5 Determinación de los requerimientos de información
- 2.6 Análisis de las necesidades del sistema
- 2.7 Diseño del Sistema Recomendado
- 2.8 Desarrollo y documentación del Software
- 2.9 Prueba y mantenimiento del sistema
- 2.10 Implantación y evaluación del sistema

UNIDAD III INVESTIGACIÓN PRELIMINAR

- 3.1. Fundamento de los Proyectos
- 3.2. Determinación de la Factibilidad
- 3.3. Factibilidad técnica
- 3.4. Factibilidad Económica
- 3.5. Factibilidad Operativa
- 3.6. Planeación y control de actividades
- 3.7. Administración de las actividades de Análisis y de Diseño
- 3.8. Productividad: El retraso en las aplicaciones.
- 3.9. Confiabilidad.
- 3.10. Mantenibilidad.

UNIDAD IV LOS REQUERIMIENTOS DEL SISTEMA

- 4.1 Acciones para determinar los requerimientos
- 4.2 Requerimientos básicos
- 4.3 Técnicas para relevar datos
- 4.4 La Entrevista
- 4.5 El Cuestionario
- 4.6 La Observación
- 4.7 El Muestreo

UNIDAD V ESTRATEGIAS PARA DETERMINAR LOS REQUERIMIENTOS DE SISTEMAS

- 5.1. Introducción
- 5.2. Tareas del análisis
- 5.3. Principios del análisis
- 5.4. El dominio de la Información
- 5.5. Partición
- 5.6. Visiones Lógicas y Físicas
- 5.7. Métodos de Análisis de Requerimientos
- 5.8. Metodologías de Análisis de Requerimientos
- 5.9. Métodos de Análisis Orientados al Flujo de Datos
- 5.10. Diagramas de Flujos de Datos
- 5.11. Diagramas físicos de flujo de datos
- 5.12. Dibujo del diagrama de contexto
- 5.13. Desarrollo de graficas de proceso
- 5.14. Desarrollo del primer nivel o diagrama cero de un diagrama físico de flujo de datos
- 5.15. Descripción del panorama lógico.
- 5.16. Diagrama lógico de flujo de datos – DFD
- 5.17. Deducción de panorama lógico:
- 5.18. Usos de diagramas físicos y lógicos de flujo de datos:
- 5.19. Reglas generales para el dibujo de diagramas lógicos de flujo de datos
- 5.20. Diagramas de Entidad-Relación

- 5.21. Características del diccionario de datos
- 5.22. Creación del diccionario de datos
- 5.23. Uso del diccionario de datos
- 5.24. Descripción de los elementos dato
- 5.25. Descripción de los datos
- 5.26. Descripciones funcionales
- 5.27. Métodos Orientados a la Estructura de Datos
- 5.28. Desarrollo de sistemas de Jackson
- 5.29. Requerimientos de las bases de datos
- 5.30. Características de las bases de dat

UNIDAD VI ESTRATEGIAS PARA EL ANÁLISIS DE DECISIONES

- 6.1. Tablas de decisión
- 6.2. Características de las tablas de decisión
- 6.3. Como construir tablas de decisión
- 6.4. Verificación de tablas
- 6.5. Tipos de entradas en tabla
- 6.6. Tablas múltiples
- 6.7. Procesadores de tabla de dedición
- 6.8. Español estructurado
- 6.9. Desarrollo de declaraciones estructuradas
- 6.10. *Estructuras de secuencia.*
- 6.11. *Estructuras de decisión.*
- 6.12. *Estructuras de iteración:*
- 6.13. Herramientas para documentar procedimientos de decisiones
- 6.14. Herramientas
- 6.15. Conceptos básicos sobre decisiones
- 6.16. Acciones
- 6.17. Árboles de decisión.
- 6.18. Características de los árboles de decisión
- 6.19. Uso de árboles de decisión
- 6.20. Identificación de los requerimientos de datos
- 6.21. Como evitar los problemas que se generan al utilizar árboles de decisión

UNIDAD VII DESARROLLO DE LA PROPUESTA DE SISTEMA

- 7.1. Introducción
- 7.2. Evaluación del equipo y del software
- 7.3. Identificación de costos y beneficios
- 7.4. Análisis de punto de equilibrio
- 7.5. Retorno de la inversión
- 7.6. Análisis del flujo efectivo
- 7.7. Valor presente

- 7.8. Lineamientos para el uso de los métodos
- 7.9. El valor presente
- 7.10. Redacción y presentación de la Propuesta
- 7.11. Recomendaciones del Analista de Sistema
- 7.12. Elección de un estilo de redacción
- 7.13. Uso eficaz de las tablas
- 7.14. Uso eficaz de las gráficas
- 7.15. Adopción de un estilo único para la propuesta
- 7.16. Presentación de la propuesta de sistema
- 7.17. Organización de la propuesta de sistema

ANEXO 1 CASO DE EJEMPLO SENCILLO DE ANÁLISIS DE SISTEMAS

Descripción de la organización

Empresa:

División Funcional:

Organigrama de la empresa

Funcionamiento de la empresa:

Selección del sub-sistema:

Sistema de ventas

Análisis de procesos

Diagrama de contexto (nivel 0)

Diagrama intermedio (nivel 1)

Diagrama detalle (nivel 2)

DFD de Detalle 1: Registrar Pedidos

DFD de Detalle 3: Almacenar:

DFD de Detalle 4: Asistir al Cliente

Análisis de datos

Entidades

Diagrama entidad-relación del sistema de ventas

Diccionario de datos

Análisis de procesos del Diagrama Nivel 1 (Intermedio)

Análisis de procesos del Diagrama Nivel 2 (Detalles)

Análisis de procesos: entidad-relación

ANEXO 2, CONSTRUYENDO DIAGRAMAS DE FLUJO DE DATOS.

1. DIAGRAMAS DE FLUJOS DE DATOS

Proceso.

Flujo.

Almacén.

Terminador.

Guía para la construcción de DFD.

- 1.1.- Escoger nombres con significado para los procesos, flujos, almacenes y terminadores.
 - 1.2.- Numerar los procesos.
 - 1.3.- Evitar los DFD excesivamente complejos.
 - 1.4.- Redibujar el DFD tantas veces como sea necesario estéticamente.
 - 1.5.- Asegúrese de que el DFD sea lógicamente consistente.
 - 1.6.- Extensiones del DFD para sistemas de tiempo real.
- 2 DIAGRAMAS DE ENTIDAD -RELACIÓN
- Tipos de objetos
 - Relaciones
 - Indicadores asociativos de tipo de objeto
 - Indicadores de subtipo/supertipo
 - Reglas para la construcción de diagramas de Entidad- Relación.

MODELAMIENTO ORIENTADO A OBJETOS

UNIDAD VIII INTRODUCCIÓN AL ANÁLISIS ORIENTADO A OBJETOS

- 8.1 Definiciones análisis orientado a objetos
- 8.2 ¿Qué es análisis orientado a objeto?
- 8.3 ¿Qué es desarrollo orientado objeto?
- 8.4 Técnicas para modelado
- 8.5 Modelando objetos
- 8.6 Modelo dinámico
- 8.7 Tópicos avanzados del modelado de objetos

UNIDAD IX PERSPECTIVA GENERAL DEL UML Y NATURALEZA Y PROPOSITO DE LOS MODELOS

- 9.1. Introducción al UML
- 9.2. Diagramas: Vistazo general
- 9.3. Diagrama de casos de uso (use case)
- 9.4. Modelado del contexto.
- 9.5. Modelado de requisitos
- 9.6. Diagrama de clases
- 9.7. La Clase
- 9.8. Relaciones entre clases
- 9.9. Ejemplo
- 9.10. Diagrama de objetos
- 9.11. Diagrama de componentes
- 9.12. Ejecutables
- 9.13. Código fuente
- 9.14. Diagramas de despliegue
- 9.15. Diagrama Secuencia

UNIDAD X LA VISTA ESTÁTICA

- 10.1 Descripción
- 10.2 Relaciones
- 10.3 Asociaciones
- 10.4 Generalización
- 10.5 Realización
- 10.6 Dependencias
- 10.7 Restricción
- 10.8 Instancias

UNIDAD XI LA VISTA DE CASOS DE USO

- 11.1 Diagrama de Casos de Uso
- 11.2 Actor
- 11.3 Casos de Uso
- 11.4 UML define cuatro tipos de relación en los Diagramas de Casos de Uso
- 11.5 La descripción del Caso de Uso comprende:

UNIDAD XII LA VISTA DE ACTIVIDADES

- 12.1 Descripción
- 12.2 Diagrama de actividades
 - 12.2.1 Describen cómo se coordinan las actividades, es útil en operaciones que alcanzan números de cosas distintas y Actividad

- 12.2.2 Transición
- 12.2.3 Barra de sincronización
- 12.2.4 Diamante de decisión
- 12.2.5 Calles

UNIDAD XIII

LA VISTA DE INTERACCION Y VISTAS FISICAS

- 13.1 Descripción de la vista de Interacción
- 13.2 Colaboración
- 13.3 Interacción
- 13.4 Diagrama de Secuencia
- 13.5 Activación
- 13.6 Diagrama de Colaboración
- 13.7 Patrones
- 13.8 Descripción de las vistas físicas
- 13.9 Componente
- 13.10 Nodo

UNIDAD XIV

LA VISTA DE GESTION DEL MODELO

- 14.1 Descripción
- 14.2 Paquete
- 14.3 Dependencias en los paquetes
- 14.4 Dependencia de acceso e importación
- 14.5 Modelo y subsistema

ANEXO 2.1

Etapas y actividades en el desarrollo Orientado a Objetos basado en UML

- Etapas
- Análisis de Requerimientos
- Diseño del Sistema
- Diseño detallado
- Implementación y pruebas
- Complemento
- Conceptos básicos en un Diagrama de Secuencia
- Mensaje
- Conceptos avanzados en un Diagrama de Secuencia
- Conceptos básicos en un Diagrama de Colaboración
- Objeto

Conceptos avanzados en un Diagrama de Colaboración
Elementos básicos en un Diagrama de Estructura Estática
Conceptos avanzados en un Diagrama de Estructura Estática
Conceptos básicos en un Diagrama de Estados
Conceptos avanzados en un Diagrama de Estados

ANEXO 2.2 .- Breve guía de ArgoUML

BIBLIOGRAFIA

CONCEPTOS DE ANÁLISIS DE SISTEMAS

CONTENIDOS

- | | |
|--|---|
| 1.23 Vista General del Análisis de Sistemas | 1.33 Sistemas de información para ejecutivos (EIS - Executive Information Systems). |
| 1.24 Análisis y sistemas | |
| 1.25 Lo que no es el análisis de sistemas | 1.34 Sistemas gerenciales |
| 1.26 Sistema, y términos asociados | 1.35 ¿Por qué es importante el análisis y diseño de sistemas? |
| 1.27 Esquema de un sistema de información | 1.36 Tipos de usuarios de sistemas |
| 1.28 Características del sistema de información | 1.37 Usuarios operacionales |
| 1.29 Tipos de sistemas de información | 1.38 Usuarios supervisores |
| 1.30 Sistemas de apoyo para la toma de decisiones (DSS: Decision Support Systems) | 1.39 Usuarios ejecutivos o funcionarios |
| 1.31 Sistemas de soporte para la toma de decisiones de grupo (GDSS - Group decision Support Systems). | 1.40 El analista de sistemas |
| 1.32 Sistemas expertos de soporte para la toma de decisiones (DESS - Expert Decision Support Systems). | 1.41 Roles del analista de sistemas |
| | 1.42 El analista de sistemas como consultor |
| | 1.43 El analista de sistema como especialista de apoyo |
| | 1.44 El analista de sistema como agente de cambio |
| | 1.45 Cualidades del analista de sistemas |

VISTA GENERAL DEL ANÁLISIS DE SISTEMAS

ANÁLISIS Y SISTEMAS

El análisis de sistemas se refiere al proceso de examinar la situación de una empresa con el propósito de mejorar con métodos y procedimientos más adecuados. El desarrollo de sistemas tiene dos componentes.

Análisis Es el proceso de clasificación e interpretación de hechos, diagnóstico de problemas y empleo de la información para recomendar mejoras al sistemas.

Diseño: Especifica las características del producto terminado.

Análisis: Especifica que es lo que el sistema debe hacer.

Diseño: Establece como alcanzar el objetivo.

LO QUE NO ES EL ANÁLISIS DE SISTEMAS

NO es:

El estudio de una empresa para buscar procesos ya existentes con el propósito de determinar cuáles deberían, ser llevados a cabo por una computadora y cuáles por métodos manuales. La finalidad del análisis está en comprender los detalles de una situación y decir si es deseable o factible una mejora. La selección del método, ya sea utilizando o no una computadora, es un aspecto secundario.

No es:

Determinar los cambios que deberían efectuarse.

No es:

Determinar la mejor forma de resolver un 'problema de sistemas de información. Sin importar cuál sea la organización, el analista trabaja en los problemas de ésta. Es un error hacer una distinción entre los problemas de la empresa y los de sistemas ya que estos últimos no existirían sin los primeros. Cualquier sugerencia debe primero considerarse a la luz de si beneficiará o perjudicará a la organización. No se debe ir tras ideas técnicamente atractivas a menos que estas mejoren el sistema de la organización.

Sistema, y términos asociados

Muchos autores han definido el concepto de Sistema, las definiciones se diferencian básicamente por el aspecto sobre el cual se hace énfasis:

- Conjunto de partes coordinadas y en interacción para alcanzar un objetivo

- Grupo de partes que interactúan bajo las influencias de fuerzas en alguna interacción definida
- Totalidad distinguible en un entorno o ambiente en el cual interactúa, compuesta a su vez de elementos que interactúan también
- Todo aquello que tiene un objetivo
- Grupo de unidades combinadas que forman un todo organizado
- Un todo integrado cuyas propiedades esenciales surgen de las relaciones entre sus partes
- Un grupo de componentes interrelacionados que trabajan en conjunto hacia una meta común mediante la aceptación de entradas y generando salidas en un proceso de transformación organizado

Una definición que abarque todas esas definiciones perfectamente validas, para nosotros será:

Sistema es el conjunto de elementos dinámicamente relacionados entre sí, que realizan una actividad para alcanzar un objetivo, operando sobre entradas y generando salidas o resultados. Todo dentro de un medio ambiente y constituyendo una totalidad diferente a otra.

Donde *elemento* se define como la parte integrante de una cosa o porción de un todo. También se puede hacer referencia al elemento utilizando los términos *parte* y *órgano*, eso depende del tipo de sistema que se esté evaluando, por ejemplo sistemas vivos o empresariales.

De los elementos de un sistema puede decirse que:

- Los elementos tienen características particulares que afectan o se ven expresadas en las características del sistema total. A su vez las características del sistema afectan o influyen en las características de los elementos. Esta particularidad se da en la medida en que el elemento está relacionado con otros
- Depende del analista del sistema determinar con qué detalle y qué elementos considerar en el momento en el cual evalúa un sistema
- Un elemento puede considerarse como un sistema, en este caso se le llama subsistema

Ejemplos:

- Partes de un computador.- Unidad Central de Proceso (CPU), teclado, monitor y ratón
- Partes de una planta.- hojas, flor, tallo y raíz
- Partes de una flor.- pétalos, estambres, filamentos, estigma y óvulos
- Partes del cuerpo humano.- un analista puede considerar que un ser humano está formado por cabeza, tronco y extremidades; otro a su vez, estimar que los componentes son sistema digestivo, sistema circulatorio, sistema endocrino, sistema nervioso, etc.
- Fichas de un rompecabezas.- el rompecabezas sólo tendrá sentido en la medida en que las fichas que lo componen, se ubiquen en el sitio que corresponde y tengan relación con la forma y el color de las demás que están a su alrededor
- Profesor y estudiantes del curso de Análisis de Sistemas

De otro lado se define como *Relación* a la situación que se da entre dos cosas, ideas o hechos cuando por alguna circunstancia están unidas de manera real o imaginaria.

También se puede hacer referencia a la relación utilizando los términos unión, conexión, interacción o enlace.

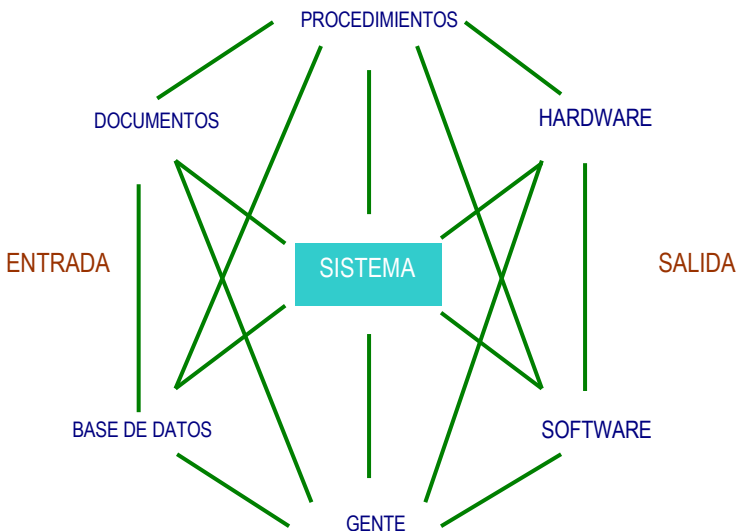


Fig. 1.1 Esquema de un Sistema de Información

SOFTWARE. Son todos los programas de computadoras, las estructuras de datos y la documentación asociada, que sirve para realizar el método lógico.

HARDWARE: Son todos los dispositivos electrónicos que proporcionan la capacidad de computación y que proporcionan las funciones del mundo exterior.

GENTE: Los individuos que son usuarios y operadores del software y del hardware.

BASES DE DATOS: Una colección grande y organizada de información a la que se accede mediante el software y que es una parte integral del funcionamiento del sistema.

DOCUMENTACION: Los manuales, los impresos y otra información descriptiva que explica el uso y / o la operación.

PROCESAMIENTOS: Los pasos que definen el uso específico de cada elemento del sistema o el contexto procedimental en que reside el sistema.

CONTROL: Los sistemas trabajan mejor cuando operan dentro de niveles de control tolerables de rendimiento por ejemplo: el sistema de control de un calentador de agua.

CARACTERISTICAS DE SISTEMA DE INFORMACION

Sus principales características son:

- Suelen lograrse ahorros significativos de mano de obra.
- Son el primer tipo de sistemas de información que se implanta en las organizaciones.
- Son intensivos en entradas y salidas de información; sus cálculos y procesos suelen ser simples y copo sofisticados, requieren mucho manejo de datos para poder realizar sus operaciones y como resultado generan también grandes volúmenes de información.
- Tiene la propiedad de ser recolectores de información.
- Son adaptables de aplicación que se encuentran en el mercado.

Ejemplos: Sistemas de facturación, planillas de sueldos, cuentas por cobrar, cuentas por pagar, contabilidad general, etc.

TIPOS DE SISTEMAS DE INFORMACIÓN

SISTEMAS DE APOYO PARA LA TOMA DE DECISIONES (DSS: Decision Support Systems)

- Apoyar la toma de decisiones mediante la generación y evaluación sistemática de diferentes alternativas o escenarios de decisión.
- Un DSS no soluciona problemas, ya que solo apoya al proceso de toma de decisiones. La responsabilidad de tomar una decisión, de adoptar y de realizarla es de los administradores, no del DSS. Puede emplearse para obtener información que revele los elementos clave de los problemas y las relaciones entre ellos. También puede usarse para identificar, crear y comunicar cursos de acción disponibles y alternativas de decisión.

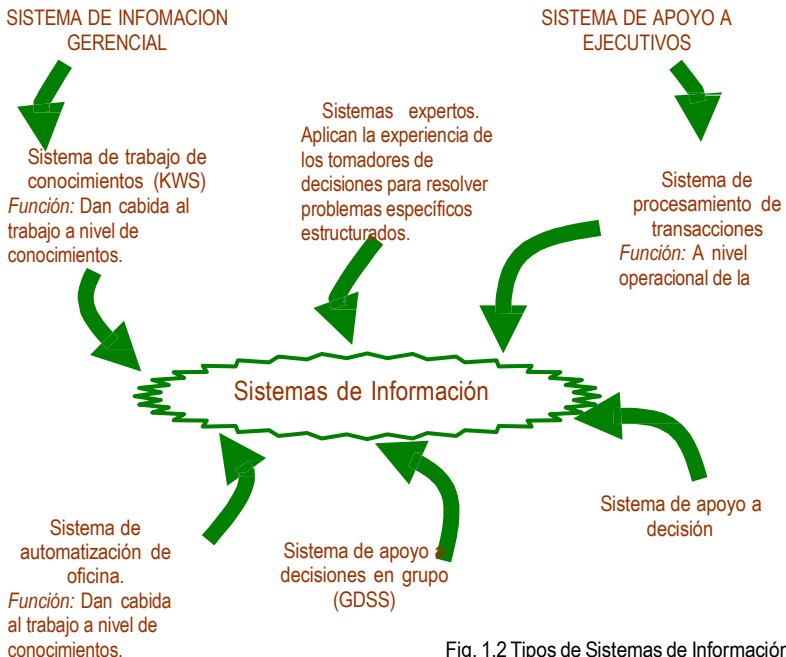


Fig. 1.2 Tipos de Sistemas de Información

Sistemas de Información para Ejecutivos (EIS - Executive information Systems). Están dirigidos a apoyar el proceso de toma de decisiones de los altos ejecutivos de una organización, presentado información relevante y usando recursos visuales de fácil interpretación, con el ejecutivo de mantenerlos informados.

Las principales características de estos sistemas son las siguientes:

- La Información que generan sirve de apoyo a los mandos intermedios y a la alta administración en el proceso de toma de decisiones.
- Suelen ser intensivos en cálculos y escasos en entrada y salidas de información.
- Así, por ejemplo, un modelo de planeación financiera requiere poca información de entrada, genera poca información como resultado pero puede realizar muchos cálculos durante su proceso.
- No suelen ahorrar mano de obra.
- Suelen ser interactivos y amigable, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.
- Apoyan la toma de decisiones que por su misma naturaleza son estructuradas y no estructuradas.

- Estos sistemas pueden ser desarrollados directamente por el usuario final sin la participación operativa de los analistas y programadores del área de informática.

Características de un EISS

- Interactividad. Interactuar en forma amigable y con el cargado de tomar decisiones.
- Tipo de decisiones. Apoya el proceso de toma de decisiones estructuradas y no estructuradas.
- Frecuencia de uso. Tiene una utilización frecuente por parte de la administración.
- Variedad de usuario. Puede emplearse por usuarios de diferentes áreas funcionales.
- Flexibilidad. Permite acoplarse a una variedad determinada de estilos administrativos participativos.
- Desarrollo que el usuario desarrolle de manera directa modelos de decisión sin la participación operativa de profesionales en informática.
- Interacción ambiental. Permite la posibilidad de interactuar con información externa como parte de los modelos de decisión.
- Comunicación ínter organizacional. Facilita la comunicación de información relevante de los niveles altos a los niveles operativos y viceversa, a través de gráficas.
- Acceso a bases de datos. Tiene la capacidad de acceder información de las bases de datos corporativas.
- Simplicidad. Simple y fácil de aprender y utilizar por el usuario final.

DSS integran en su mayoría un conjunto de modelos que apoyan las diferentes decisiones a las que se enfrenta el tomador de decisiones.

Ventajas del uso de los EISS.

- Menores costos.
- Disponibilidad de una gran variedad de herramientas en el mercado que operan en el ambiente de microcomputadoras.
- Muy baja dependencia de personas que se encuentran fuera del control de tomador de decisiones.

Desventajas pueden ser:

- Falta de integridad y consolidación en la administración de la información.
- Problemas de seguridad de la información.
- Pérdida del control administrativa por parte del área de informática.

Las diferentes opciones para la implantación de los DSS

- Implantación aislada en microcomputadoras.

- Implantación en microcomputadoras interconectadas y que constituyen una red local.
- Microcomputadoras conectadas a mini computadoras o servidores.

Sistemas de Soporte para la Toma de Decisiones de Grupo (GDSS: Group Decision Support Systems)

Cubren el objetivo de lograr la participación de un grupo de personas durante la toma de decisiones en ambientes de anonimato y consenso, apoyando decisiones simultaneas.

Los sistemas de Soporte para la Toma de Decisiones de Grupo (GDSS) para considerarse como tal deben reunir un conjunto de características; las principales son las siguientes:

- GDSS. Sistemas diseñados especialmente para apoyar las decisiones en grupo.
- La meta de GDSS. Es apoyar a los tomadores de decisiones en su trabajo.
- GDSS. Es fácil de aprender y de usar. Accesible para usuarios con diferentes niveles de conocimiento computacional y de soporte a la decisión. Tales como ventas, producción, recursos humanos, administración y finanzas.
- Un GDSS. Contiene mecanismo para evitar el desarrollo de conductas negativas en el grupo, como son los problemas de comunicación.
- Un GDSS debe motivar a todos los miembros del grupo a participar de manera activa. Es importante que pueda existir anonimato de la participación.

Las principales ventajas de GDSS son:

- Motiva a los miembros del grupo a trabajar juntos
- Da la misma oportunidad de participación a todos los miembros del grupo.
- Se optimiza el uso de la información que aporta cada miembro del grupo.
- Proporciona un mecanismo para enfocar a grupo en problemas clave.
- Apoya el desarrollo de una memoria organizacional.
- Mejora la calidad de toma de decisiones.
- Incrementa la creatividad en la toma de decisiones.

Las principales desventajas de GDSS son:

- Falta de costumbre al utilizar un sistema para soportar el proceso de toma de decisiones.
- Resistencia al cambiar por parte de los administradores.
- La responsabilidad al tomar una decisión puede diluirse.

APLICACIONES DE LOS DGSS

- Establecimiento de la misión de una empresa.

- Formulación de estrategias que ayudarán a que la misión se cumpla.
- Evaluación de administradores. Para incrementar el sueldo de un administrador o para verificar que esté cumpliendo con su deber.
- Planeación de sistemas de información. Cuando se requiere introducir nueva tecnología de sistemas de información es necesario modificar el plan de sistemas
- Soporte en negociaciones.
- Apoyar los trabajos que visuales, como la selección de un empaque para un nuevo producto.
- Apoyar los trabajos que involucran diseño y revisiones de control de calidad.
- Apoyar una decisión en particular.

Sistemas Expertos de Soporte para la Toma de Decisiones (EDSS: Expert Decision Support Systems)

Los sistemas expertos constituyen el área de la inteligencia artificial que quizá en este momento tiene más relación con el apoyo al proceso de la toma de decisiones en las organizaciones.

Beneficios de la utilización de un sistema EDSS

1. Reducción en la dependencia de personal clave se debe a tener los conocimientos del personal especializado son detenidos durante el proceso de aprendizaje y están listos para ser utilizados por diferentes personas.
2. Facilitar el entrenamiento del personal. Capacitación y adiestramiento del personal sin experiencia.
3. Mejora en la calidad y eficiencia en el proceso de la toma de decisiones. Las decisiones podrán tomarse de una forma más ágil con el apoyo de un sistema experto.
4. Transferencia de la capacidad de decisiones. Un sistema experto puede facilitar la descentralización de datos en el proceso de la toma de decisiones en aquellos casos que se consideren convenientes.

COSTO QUE INVOLUCRA

- El Shell o paquete generador del sistema experto.
- El equipo computacional o hardware que se requiera.
- Consultorio especializado.
- Contratación o pago a los ingenieros especialistas.
- El tiempo de los expertos.
- Costos de implantación.
- Costos involucrados con el mantenimiento y seguimiento del sistema.

Sistemas de Información para Ejecutivos (EIS - Executive Information Systems)

- Están diseñados para cubrir las necesidades específicas y particulares de la alta administración de la empresa.
- Extraen, filtran, comprimen y dan seguimiento a formación crítica del negocio.
- Pueden acceder información que se encuentra en línea, extrayéndose en forma directa de las bases de datos de la organización
- El sistema está soportado por elementos especializados de hardware, tales como monitores o videos de alta resolución y sensibles al tacto.

Las siguientes características adicionales deben estar presentes para considerar a un EIS:

- Contempla las facilidades de comunicación electrónica.
- Capacidad de análisis de datos, tales como hoja electrónica de cálculo.
- Herramientas para la organización personal del ejecutivo, tales como calendario.

SISTEMAS GERENCIALES

Una herramienta para soportar las funciones operativas. La perspectiva actual y futura tiende a cambiar este enfoque radicalmente, los sistemas de información son vistos además como áreas de oportunidad para lograr ventajas en el terreno de los negocios, y éstas representan un diferencial o valor agregado con respecto a los competidores.

La perspectiva estratégica considera a los sistemas de información como una herramienta para mejorar la estructura competitiva del negocio, por lo que tienen su área de influencia en el medio ambiente de la organización, a través de nuevos servicios a clientes, nuevos negocios y oportunidades de inversión.

Wiseman define la visión gerencial o estrategia como <<la necesidad de entender de qué forma la tecnología de la información es utilizada para soportar o dar forma a la estrategia competitiva de la empresa>>. Esta habilidad de ver y entender el nuevo rol de los sistemas de información constituye la esencia de la visión de los sistemas de información estratégica.

Sus principales características son:

- Proporcionar información para apoyar la toma de decisiones.
- No pueden adaptarse fácilmente a paquetes disponibles en el mercado.
- Típicamente su forma de desarrollo es a base de incrementos y a través de su evolución dentro de la organización.
- Su función es lograr ventajas que los competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores. En este contexto, los sistemas estratégicos son creados de barreras de entrada al negocio.
- Apoyan los procesos de innovación de productos y proceso dentro de la empresa. Una forma de hacerlo es innovando o creando productos y procesos.

Wiseman utilizan el término impulsos estratégicos para connotar los movimientos que hace una empresa con el fin de ganar o mantener algún tipo de ventaja competitiva.

Las cinco categorías que contempla Wiseman en cuanto a los impulsos estratégicos.

DIFERENCIACIÓN

Este impulso estratégico se refiere a la diferenciación de los productos o servicios a través de precios, plazos o promociones. El proceso de diferenciación puede trabajar en dos direcciones. La primera de ellas se refiere a lograr ventajas de diferenciación sobre los competidores utilizando la tecnología de la información; la segunda consiste en identificar oportunidades para reducir las ventajas de diferenciación de los competidores, clientes o proveedores.

COSTO

Se refiere a los movimientos que puede hacer la empresa para reducir sus costos o bien provocar la reducción de costos a proveedores o clientes, con el fin de obtener un trato preferencial.

Las economías de escala se logran cuando se aumenta el volumen de las ventas de productos o servicios para reducir los costos unitarios, a través de mejores negociaciones con proveedores de servicio debidas a mayor volumen de compra.

CRECIMIENTO

El impulso estratégico del crecimiento permite la consecución de ventas competitivas, mediante el incremento del volumen de operaciones en el negocio.

El crecimiento de producto o mercado se refiere a la expansión de mercados, satisfacción de nuevas necesidades o la incorporación de nuevas tecnologías asociadas al producto. El crecimiento puede darse funcionalmente, es decir, sustituyendo los servicios que proporcionan los proveedores, las funciones que llevan a cabo los clientes (hacia adelante).

Pueden lograrse ventajas competitivas, el impulso estratégico de la globalización es, según Wiseman, un impulso de crecimiento que involucra elementos foráneos al producto neto de la compañía.

ALIANZAS

Las alianzas son definidas por Wiseman como la combinación de dos más grupos o individuos que se unen para lograr un objetivo común.

INNOVACIÓN

Otro de los impulsos estratégicos que puede ser apoyando a través de la tecnología de información, ya sea en productos o en tecnología de información, ya sea en productos o en procesos nuevos. Para que un proceso de innovación tenga éxito requiere respuestas rápidas a las oportunidades que se representan, sin embargo, existen

riesgos inherentes debido a la naturaleza del proceso, ya que es difícil innovar sin correr riesgos.

El proceso de innovación consta de las siguientes fases: nacimiento de una idea, venta de la idea a una persona con poder de decisión, desarrollo de la idea y lanzamiento al mercado de la idea desarrollada.

Alcanzar al mercado la idea puede tenerse éxito o fracaso en el proceso. Si se tiene éxito deben construirse barreras de entrada a esta innovación para protegerse de los competidores.

¿PORQUÉ ES IMPORTANTE EL ANÁLISIS Y DISEÑO DE SISTEMAS?

El análisis y diseño de sistemas pretende estudiar la operación de ingreso de los datos, el flujo de los mismos y la salida de la información, todo ello dentro del contexto de una organización. Es decir, sirve para analizar, diseñar y formular mejoras en la operación de la organización, la cual puede realizarse mediante el uso de sistemas de información computarizado.

El diseño y análisis se conforman por una serie de procesos, que al ejecutarse sistemáticamente mejoran la operación de un negocio, mediante el uso de los sistemas de información computarizado.



TIPOS DE USUARIOS DE SISTEMAS

Todo aquel que se relacione con un sistema de información dentro del contexto de la organización puede definirse como Usuario. Siempre que sea posible, el analista debiera tratar de establecer contacto directo con el usuario.

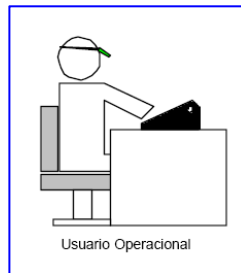
Uno de los errores más frecuente que se cometen es suponer que todos los usuarios son iguales. Evidentemente tienen diferentes personalidades, diferentes intereses, diferente preparación, etc. Los usuarios se pueden clasificar por:

- Usuarios Operacionales
- Usuarios Supervisores
- Usuarios Ejecutivos

USUARIOS OPERACIONALES

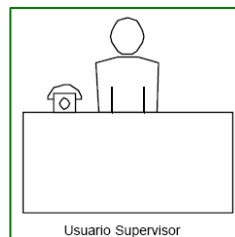
Son todos aquellos que laboran en las oficinas, administradores y operadores que son los que tendrán contacto con el sistema. Los usuarios de este nivel se preocupan por las funciones que tendrá el sistema y por la interface humana.

Son conocedores del trabajo específico que hacen y de las personas con las que tienen comunicación inmediata (clientes, colegas, etc.). Normalmente no están familiarizados con el panorama general, es decir como encaja la actividad en cuestión dentro de la organización global. Suelen pensar en los sistemas en términos físicos, es decir, en términos de la tecnología de puesta en práctica que comúnmente se utiliza para implementar o hacer uso del sistema.

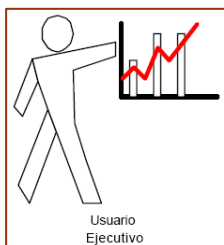


USUARIOS SUPERVISORES

Son los que administran a un grupo de usuarios operacionales y son responsables de sus logros y/o fracasos. Están familiarizados con el trabajo. Se interesa en sistemas de información por a posibilidad de incrementar el volumen y la supervisión del trabajo. Actúan como intermediarios entre el analista y los usuarios operacionales. Piensa en los mismos términos físicos que los usuarios operacionales. Es el que definirá los requerimientos y las políticas de la empresa que su sistema deberá realizar. Puede ser un miembro pasivo (actúa cuando se lo solicita), o bien un miembro de tiempo completo o incluso, el gerente del proyecto.



USUARIOS EJECUTIVOS O FUNCIONARIOS



Generalmente nunca se involucran directamente con el proyecto de desarrollo de sistema, a menos que el proyecto sea muy amplio e importante que tenga un impacto de primer orden en la empresa. Proporcionan la iniciativa para el proyecto y financian el mismo. Se preocupan por los detalles

estratégicos y las ganancias y pérdidas a largo plazo. Se interesan por el panorama global de la organización.

EL ANALISTA DE SISTEMAS

Este será usted en un futuro no muy lejano. Es el personaje clave en cualquier proyecto de desarrollo de sistemas. En un sentido amplio, el analista desempeña papeles de descubrir detalles y documentar la política de un negocio. Debe distinguir entre síntomas, problemas de usuarios y causas. Con un amplio conocimiento en ordenadores. Se encuentra en el medio de usuarios, programadores, administradores, auditores y otros participantes.

Se requiere facilidad en el manejo de personas. Se requiere habilidad en computación para entender los usos potenciales de hard y soft y se necesita de una mente lógica y organizada.

ROLES DEL ANALISTA DE SISTEMAS

Los tres roles importantes que el analista debe cubrir son: el de consultor; el de especialista de apoyo o soporte y el de agente de cambio.

EL ANALISTA DE SISTEMAS COMO CONSULTOR

Un analista puede contratarse solo para canalizar a la empresa ciertos tópicos de la informática. Esto ofrece una ventaja, en el sentido de que el consultor externo trae consigo perspectivas frescas, que no poseen otros miembros de la empresa. Por otra parte para el analista externo implica una desventaja, pues no tiene fácil acceso a la cultura organizacional auténtica..

Como consultor externo deberá conocer e implementar metodología para el modelo de sistema y contará con la ayuda de los usuarios para entender la cultura de la organización desde sus propios puntos de vistas.-

EL ANALISTA DE SISTEMA COMO ESPECIALISTA DE APOYO

Otro papel es como especialista de apoyo o de staff. En esta posición, el analista dispone de una experiencia profesional respecto al hardware y al software y a sus aplicaciones en la organización. Con frecuencia estas tareas no se asocian a un proyecto ambicioso de sistemas, sino más bien implican decisiones o modificaciones menores que se dan en un área individual.

Como especialista de apoyo, no dirigirá un proyecto, solo será un recurso humano de apoyo para quienes lo dirigen. Si es un analista de sistemas contratado por una organización de servicios o de manufactura, muchas de sus actividades diarias se ajustarán a este papel.

EL ANALISTA DE SISTEMA COMO AGENTE DE CAMBIO

El papel que le confiere un alto grado de responsabilidad, es el de agente de cambio; sin importar si es o no de la organización. Como analista, será un agente de cambio cada vez que realice alguna de las actividades del ciclo de desarrollo del sistema. Un agente de cambio es una persona que sirve de catalizador para el cambio, que desarrolla un plan para el mismo y que colabora con otros para llevarlo a cabo y agilizarlo.

Debe aceptar el desafío del cambio y tomarlo como punto de partida de su análisis. Como analista de sistema, al actuar como agente de cambio, apoya una corriente particular de cambio, que involucra el uso de los sistemas de información. Además, transmite a los usuarios el proceso de cambio, ya que esta convencido de que tales cambios no ocurren de manera independiente en los sistemas de información, sino más bien, éstos ocasionan cambios a lo largo de las organizaciones.

CUALIDADES DEL ANALISTA DE SISTEMAS

El analista de sistema con éxito deberá contar con una amplia gama de cualidades.

Ante todo un analista de sistemas es un solucionador de problemas. Ve el análisis de los problemas como un reto y se deleita encontrando soluciones posibles y factibles. Cuando es necesario tiene que ser capaz de abordar de manera sistemática la situación, mediante la aplicación de herramientas, técnicas y experiencia. Debe ser un buen interlocutor, mantener una relación cordial con todos los usuarios. Necesita contar con suficiente experiencia en computación para programar, entender las capacidades de las computadoras, recoger las necesidades de información de los usuarios y llegar a transmitir a los programadores lo necesario.

Debe ser auto disciplinado y auto motivado como individuo. Debe ser capaz de administrar y coordinar innumerables recursos del proyecto, incluyendo a otras personas. El análisis de sistema exige demasiado, pero se compensa por la naturaleza siempre cambiante de los problemas, así como por el continuo enfrentamiento al reto.

CICLO DE VIDA DEL DESARROLLO DE SISTEMAS

CONTENIDOS

- 2.1 Concepto
- 2.2 Objetivos del ciclo de vida del desarrollo de sistemas:
- 2.3 Etapas del ciclo de vida del desarrollo de sistemas:
- 2.4 Identificación de problemas, oportunidades y objetivos
- 2.5 Determinación de los requerimientos de información
- 2.6 Análisis de las necesidades del sistema
- 2.7 Diseño del Sistema Recomendado
- 2.8 Desarrollo y documentación del Software
- 2.9 Prueba y mantenimiento del sistema
- 2.10 Implantación y evaluación del sistema

2.1 Concepto

El ciclo de vida del desarrollo de sistemas (Systems Development Life Cycle - SDLC) es un enfoque por etapas del análisis y de diseño, que postula que el desarrollo de los sistemas mejora cuando existe un ciclo específico de actividades del analista y de los usuarios.

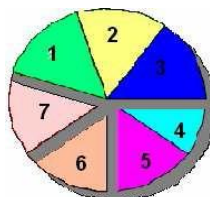
No existe un acuerdo tácito en el número de etapas; sin embargo, por lo general se reconoce la importancia de su enfoque sistemático.

OBJETIVOS DEL CICLO DE VIDA DEL DESARROLLO DE SISTEMAS:

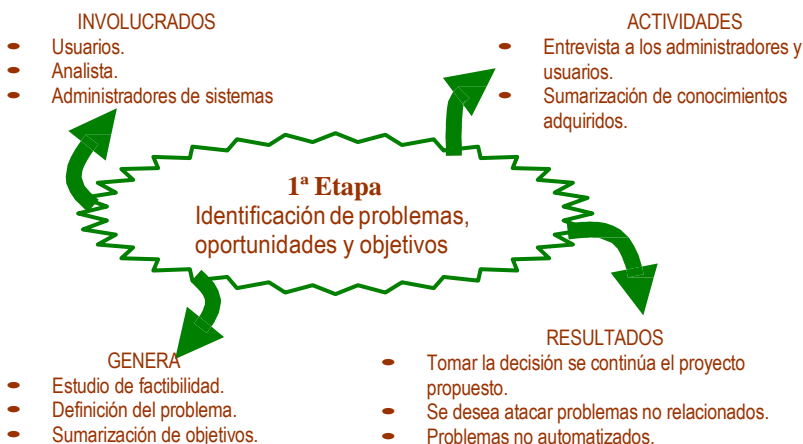
- Definir las actividades a ser ejecutadas en un proyecto de Procesamiento Electrónico de Datos (PED)
- Introducir coherencia en muchos proyectos de PED de la misma organización
- Establecer punto de control para control de gerencia y puntos de control para tomar la decisión de "continuar o no".

2.2 ETAPAS DEL CICLO DE VIDA DEL DESARROLLO DE SISTEMAS:

Por convención, vamos a dividir el ciclo en siete etapas, como se muestra en la figura siguiente. Aunque cada etapa se muestra de manera discreta, nunca se lleva a cabo como un elemento independiente. En lugar de ello, se realizan al mismo tiempo diversas actividades, y estas llegan a repetirse. Por ello es de mayor utilidad suponer que el ciclo de desarrollo de los sistemas transcurre en etapas y no como elementos separados.



2.2.1 Identificación de problemas, oportunidades y objetivos



Esta primera etapa involucra al analista en la identificación de los problemas, oportunidades y objetivos. Esta fase es la más importante para el éxito del resto

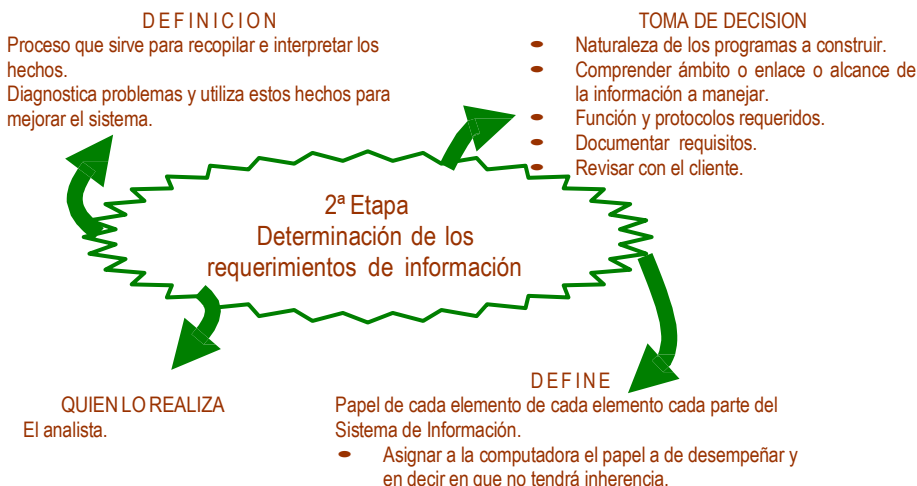
del proyecto, ya que nadie estará dispuesto a desperdiciar su tiempo dedicándolo al problema equivocado.-

Es necesario observar en forma objetiva lo que ocurre en la empresa, luego hacer notar los problemas.

Las oportunidades son aquellas situaciones que el analista considera que pueden perfeccionarse mediante el uso de los sistemas de información. Al aprovechar estas oportunidades, la empresa puede lograr una ventaja competitiva o llegar a establecer un estándar industrial.

La identificación de objetivos también es un componente importante de la primera fase. En primera instancia, el analista deberá descubrir lo que la empresa intenta realizar. Y luego, estará en posibilidad de determinar si el uso de los sistemas de información apoyaría a la empresa para alcanzar sus metas, el encaminarla a problemas u oportunidades específicas.

2.2.2. Determinación de los requerimientos de información



La segunda etapa determinará los requerimientos de información a partir de los usuarios involucrados. Para identificar los requerimientos dentro de la empresa, pueden utilizarse diversos instrumentos, los cuales incluyen: el muestreo; la entrevista; los cuestionarios; la observación de quienes toman las decisiones, técnicas para recolectar información: JAD, Brasstorming, PIECES, etc.; y el uso de prototipos. Se trata de identificar que información requiere el usuario para desempeñar sus tareas. Se relaciona directamente con los usuarios.

3.- Análisis de las necesidades del sistema



En esta tercera etapa habrá que determinar que herramientas y técnicas especiales facilitaran al analista la realización de las determinaciones requeridas.

Durante esta etapa, el analista de sistema también analiza las decisiones estructuradas por realizar, que son decisiones donde las condiciones, condiciones alternativas, acciones y reglas de acción podrán determinarse.

A esta altura del ciclo de desarrollo del sistema, el analista prepara una propuesta del sistema que resume todo lo que ha encontrado, puede incluso presentar un análisis costo / beneficio de las alternativas y plantea las recomendaciones de lo que deberá realizarse.

2.2.4. Diseño del Sistema Recomendado

En esta cuarta etapa se utiliza la información que se recolectó con anterioridad y elabora el diseño lógico del sistema de información. El analista diseña procedimientos precisos de captura de datos, con el fin de que los datos que se introducen al sistema sean correctos. Diseña accesos efectivos al sistema de información, mediante el uso de técnicas de diseño de formas y de pantallas.

Una parte del diseño lógico es el diseño de las interfaces o pantallas de usuario, esta conecta al usuario con el sistema y es de suma importancia.

La etapa del diseño también incluye el diseño de los archivos o las bases de datos que almacenarán aquellos datos requeridos por quien toma las decisiones en la organización.

2.2.5. Desarrollo y documentación del Software

En la quinta etapa el analista trabaja con los programadores para desarrollar o dicho de otra forma construir todo el software original que sea necesario. Dentro de las técnicas estructuradas para el diseño y documentación del software se

tienen método HIPO, los diagramas de flujo, Pseudocódigo, etc. Aquí es donde el analista de sistemas transmite al programador los requerimientos de programación.

Durante esta etapa, el analista también colabora con los usuarios para desarrollar la documentación indispensable del software, incluyendo los manuales de procedimientos. La documentación le dirá al usuario cómo manejar el software, y así también, que hacer en caso de presentarse algún problema.

2.2.6. Prueba y mantenimiento del sistema

En la penúltima etapa el sistema de información debe probarse antes de utilizarlo. El costo es menor si se detectan los problemas antes de la entrega del sistema.

El programador puede realizar algunas pruebas por su cuenta, otras se llevan a cabo con colaboración del analista de sistema. En un principio, se hace una serie de pruebas, con datos tipo, para identificar las posibles fallas del sistema; más adelante, se utilizarán los datos del sistema real.



El mantenimiento del sistema y de su documentación empiezan en esta etapa; y después, esta función se realizara de forma rutinaria a lo largo de toda la vida del sistema. Las actividades de mantenimiento integran buena parte de la rutina del programador, que para las empresas llega a implicar importantes sumas de dinero. Sin embargo, el costo del mantenimiento disminuye de manera importante cuando el analista aplica procedimientos sistemáticos en el desarrollo de los sistemas.

2.2.7. Implantación y evaluación del sistema

En esta ultima etapa, el analista ayuda a implantar el sistema de información. Esto incluye el adiestramiento que el usuario requerirá. Es responsabilidad del analista la supervisión de la capacitación de todos y cada uno de los usuarios en el manejo y operación del sistema. El analista necesita planear la transición que trae consigo un cambio de sistema.



Aunque la evaluación del sistema se plantea como parte integrante de la última etapa del ciclo de desarrollo de sistema; realmente, la evaluación toma parte en cada una de las etapas. Uno de los criterios fundamentales que debe satisfacerse, es que el futuro usuario *utilice* el sistema desarrollado. En la realidad, todas las etapas mantienen una dinámica de carácter espiral, hasta que el sistema finalmente se concluye.

INVESTIGACIÓN PRELIMINAR

CONTENIDOS

- 3.11. Fundamento de los Proyectos
- 3.12. Determinación de la Factibilidad
- 3.13. Factibilidad técnica
- 3.14. Factibilidad Económica
- 3.15. Factibilidad Operativa
- 3.16. Planeación y control de actividades
- 3.17. Administración de las actividades de Análisis y de Diseño
- 3.18. Productividad: El retraso en las aplicaciones.
- 3.19. Confiabilidad.
- 3.20. Mantenibilidad.

3.1 Fundamento de los Proyectos

Un proyecto de sistemas que se inicia en una empresa, tiene problemas y oportunidades para mejorarlos, y con frecuencia estos surgen cuando la organización procura adaptarse al cambio.

Los proyectos surgen de numerosas fuentes que la gente de negocios sugiere por dos razones: la experimentación de problemas que los conduzcan a soluciones con sistemas informatizados y la identificación de oportunidades para mejorar que eventualmente llegaran a presentarse. Ambas, surgen conforme las organizaciones se van adaptando o enfrenando a los cambios evolutivos.

Los buenos administradores reconocen la existencia de problemas dentro de una organización y estos son necesarios diagnosticarlos y al final enfrentarlos. Mas aún cuando se dan cuenta que las metas no se cumplen o las condiciones normales del desempeño no se están cumpliendo, y ambas traen como consecuencia la persistencia y cantidad de errores, desarrollo lento, incompleto o incorrecto del trabajo o la no realización del mismo.

El analista de sistemas sirve de catalizador y de especialista en apoyo, ya que es capaz de identificar el sitio donde los procesos pueden mejorar.

Algunas oportunidades de mejoría son:

- La aceleración de un proceso
- Eliminación de pasos innecesarios
- Reducción de errores
- Eliminación de salidas redundantes
- Mejorar la relaciones proveedor/cliente/vendedor con el sistema

Los proyectos deben examinarse desde una perspectiva de sistemas, de tal forma que se considere el impacto del proyecto propuesto sobre toda la organización.

Existen cinco criterios para la selección de proyectos:

- Contar con el respaldo de la dirección
- Disponibilidad de tiempo
- Posibilidad de mejorar la consecución de las metas de la organización
- Viabilidad en función de recursos y de capacidades
- Ventajas sobre otras alternativas

3.2. Determinación de la Factibilidad

Para los proyectos de sistemas, la factibilidad se apoya en tres principios básicos: operativo, técnico y operativo. Este estudio, no es un estudio de sistemas, más bien es una recopilación de datos relevantes para la alta dirección, quien deberá tomar la decisión si procede al estudio de sistemas.

Parte de la indagación de la factibilidad consiste en descubrir cuales son objetivos que serán necesarios abordar.

Algunos de esos objetivos son:

- Reducción de errores, mayor precisión en la captura y aceleración en la misma
- Reducción del costo de las salidas
- La integración de los subsistemas del negocio
- La actualización del servicio al cliente (finalidad competitiva)
- Reducción del tiempo de procesamiento
- Automatización de procedimientos manuales

Para determinar los recursos se sigue el mismo patrón de factibilidad.

3.3. Factibilidad técnica

El analista debe indagar si los recursos técnicos usuales pueden actualizarse o complementarse, de tal manera que satisfagan la necesidad considerada. Si no puede ser actualizada, tenemos que considerar la existencia de algún tipo de tecnología que pueda satisfacer el requisito. Para ello se necesita mucha experiencia y conocimiento.

3.4 Factibilidad Económica

En este estudio deben considerarse los siguientes recursos básicos: el tiempo del analista y del equipo de trabajo, el costo de la realización integral de un estudio de sistemas, el costo del tiempo del empleado para la empresa, el costo estimado del equipo y el costo estimado del software comercial o en su defecto, su desarrollo.

3.5. Factibilidad Operativa

Depende de los recursos humanos que participan durante la operación del proyecto. Esto se refiere al pronóstico de si, una vez instalado, el sistema llegará a funcionar o a usarse.

Requiere de una imaginación creativa y habilidad de persuasión.

3.6. Planeación y control de actividades

El análisis y diseño de sistemas involucra actividades de naturaleza muy diferentes, que al integrarse constituyen un proyecto. El analista de sistema debe administrar con cuidado el proyecto, lo que implica realizar tareas de planeación y control.

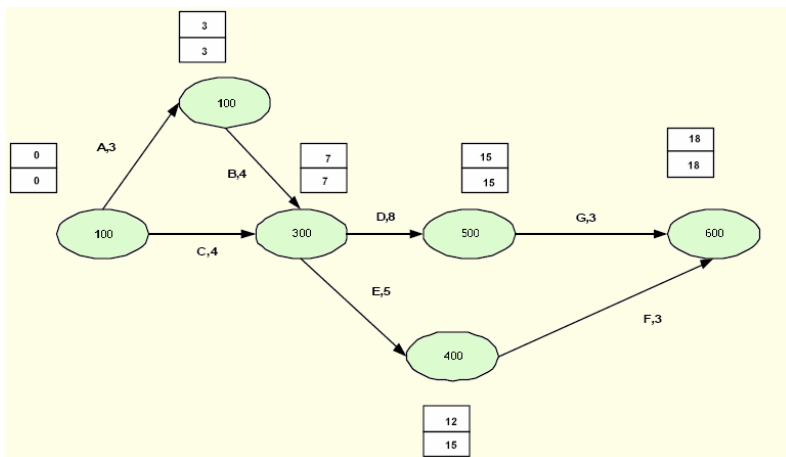
La planeación incluye a todas las actividades que se requieren para la selección del equipo de análisis de sistemas, la asignación de proyectos apropiados, la estimación de tiempo que cada tarea requiere para su ejecución, y la programación del proyecto, de tal forma que las tareas se concluyan oportunamente.

El control denota el uso de la retroalimentación para darle seguimiento al proyecto, esto incluye comparar lo planificado con lo realizado, tomar acciones adecuadas para

Las actividades se representan por la flechas, la longitud de la flecha no esta asociado a la duración de la actividad. Los nodos circulares, también llamados eventos, sirven para:

- Reconocer que una actividad ha concluido
- Indicar que actividades necesitan concluirse antes de iniciar una nueva (precedencia)

Los nodos, para su identificación, pueden tener número, letras u otra cualquier identificación.



Ejemplo de diagrama de PERT

Todo proyecto tiene un principio, una parte media y un fin. A la trayectoria mas larga se le denomina ruta critica, es el trayecto más largo y un atraso en dicho trayecto implicara un retraso en todo el proyecto.

Para mantener la lógica y caridad de los diagramas es necesario la utilización de actividades ficticia (duración cero) o dummy.

El tiempo de holgura puede obtenerse para cada uno de los eventos y con ello se sabrá el del tiempo que se dispone para concluir una actividad antes de que se arriesgue la duración global de proyecto. Para obtener la holgura, primero deberá determinarse el fin temprano (FT) y el inicio tardío (IT). El fin temprano de un evento es el momento en el que más tempranamente es posible concluir todas las actividades que convergen a tal nodo. El inicio tardío es el último momento disponible para iniciar aquellas actividades que se originan en tal nodo. La diferencia entre el FT y IT es la holgura particular del evento. El diagrama de PERT nos permite:

- La identificación fácil del orden de la precedencia
- La identificación fácil de la ruta critica; y consecuentemente, de las actividades criticas

- El calculo sencillo de la duración de la holgura

3.7. Administración de las actividades de Análisis y de Diseño

Además de administrar tiempo y recursos, los analistas tienen que administrar recursos humanos. Esto se logra estableciendo una certera comunicación entre los integrantes de los grupos, quienes fueron elegidos por su competitividad y su compatibilidad. Se deben establecer metas de productividad y lograr la motivación de los miembros del equipo de análisis.

Una forma de organizar las ideas acerca del trabajo de grupo es concibiéndolo como si siempre se buscara un balance entre el cumplimiento de las tareas y el manejo de las relaciones de los integrantes del grupo.

De hecho, los grupos, normalmente tienen dos líderes; el líder que dirige a los integrantes para el desempeño de las actividades y la otra persona (líder) se encarga de las relaciones sociales de los integrantes.

Otras áreas que se deben considerar son:

Identificación con lo que el grupo produce: Aparte de hacer un esfuerzo individual en el trabajo, los integrantes deberán atribuir al equipo los éxitos y los fracasos

Responsabilidad de la supervisión del desempeño del grupo: responsabilizar a uno de los integrantes la supervisión periódica del desempeño del equipo. Es una manera de verificar si se cumplen los objetivos.

Integración del grupo en la organización: Los grupos deben tener el compromiso primordial hacia la empresa, desmotivando parcialmente, la cohesión que los aleja de los otros grupos.

Motivar la protagonización de múltiples papeles: Deben ser motivados para aceptar múltiples responsabilidades en su trabajo como analistas de sistemas.

Aspectos importantes en el desarrollo de sistemas

Para desarrollar sistemas de información útil y de alta calidad que cubran las necesidades de los usuarios, se verán influenciadas por cuestiones de:

- PRODUCTIVIDAD
- MANTEBILIDAD
- CONFIABILIDAD

3.8. Productividad: El retraso en las aplicaciones.

Nos encontramos en una etapa de máxima exigencia. Más sistemas, más calidad, más complejidad y todo mucho más rápido. Dos aspectos de este problema son el retraso en los nuevos sistemas que se necesitan desarrollar, y el tiempo que se requiere para construir un sistema individual nuevo.

El retraso se presenta en tres tipos diferentes:

- Retraso visible. Sistemas nuevos que los usuarios han pedido (aprobados y financiados) pero no se han iniciado por no contar con recursos adecuados.
- Retraso invisible. Sistemas que se necesitan pero no se han pedido.
- Retraso desconocido. Sistemas que ni si quiera se sabe que se necesitan

Un segundo aspecto de la productividad es el tiempo necesario para desarrollar un sistema individual. Se preocupan por si, para cuando el nuevo sistema esté listo, habrán cambiado las condiciones de negocios tan drásticamente que los requerimientos originales ya no sean relevantes, es decir, un nuevo sistema se asocia con una oportunidad de negocios que el usuario percibe, y esa oportunidad a menudo tiene un margen de oportunidad, es decir, un periodo tras el cual ésta habrá desaparecido y ya no se necesitara el sistema nuevo.

Un tercer aspecto: algunos proyectos resultan ser útiles y la administración los cancela antes de que terminen ya sea por problemas técnicos, problemas administrativos, personal sin experiencia, falta de tiempo, etc.

Para evitar el retraso en las aplicaciones algunas de las técnicas más comunes son:

- *Contratar más programadores y analistas*
- *Permitir a los usuarios desarrollar sus propios sistemas*
- *Mejores lenguajes de programación*
- *Atacar los problemas de mantenimiento*
- *Disciplinar la Ingeniería de Software*
- *Uso de herramientas automatizadas para el desarrollo de sistemas*

El hecho es que no se puede elaborar un sistema que de buen resultado, de alta calidad y mantenible si no se sabe precisamente, y con suficiente detalla, que se supone que debe poder hacer.

3.9. Confiabilidad.

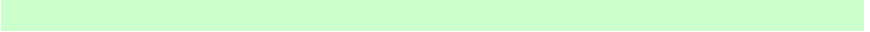
El largo tiempo que se invierte en la prueba y la corrección de errores, y la muy poca productividad pudieran ser aceptables si el resultado fuesen sistemas altamente confiables y fácilmente mantenibles. La evidencia de los últimos años es justo lo contrario: los sistemas producidos por las organizaciones en todo el mundo están llenos de errores y son casi imposibles de modificar.

En muchos casos, nadie esta muy seguro respecto a cuantos errores tiene un sistema, pues algunos errores jamás se encuentran antes de que caduque el sistema y el proceso de documentación y registro de errores están descuidado que la mitad de los errores encontrados no se reportan, aun dentro de la organización misma de desarrollo de sistemas. El fenómeno típico de descubrir errores, es un periodo de varios años de utilidad de un sistema de software.

3.10. Mantenibilidad.

La corrección de errores sobre la marcha es un aspecto del mantenimiento. El mantenimiento del software es un problema primordial en muchas organizaciones, entre el 50 y 80 por ciento del trabajo que se hace en la mayoría de las organizaciones de desarrollo de sistemas se asocia con la revisión, modificación, conversión, mejoramiento o corrección de errores en algún programa escrito con anterioridad.

Si fuera simplemente un caso de que el software fuese malo, se podría considerar desecharlo o remplazarlo. Pero muchas organizaciones jamás han capitalizado su software, y sus políticas de administración y de negocios hacen que se vuelva prohibitivamente caro remplazar los sistemas antiguos. Y existe un problema aún más fundamental: en la mayoría de las organizaciones, no existe una descripción coherente de lo que se supone que los sistemas deben hacer. La documentación existente suele ser obsoleta y confusa. Por eso, aunque se pueda argumentar que la mantenibilidad es enteramente función de la implantación, es imposible mantener un sistema si no existe un modelo preciso y actualizado de sus requerimientos y esto es tarea del analista.



LOS REQUERIMIENTOS DEL SISTEMA

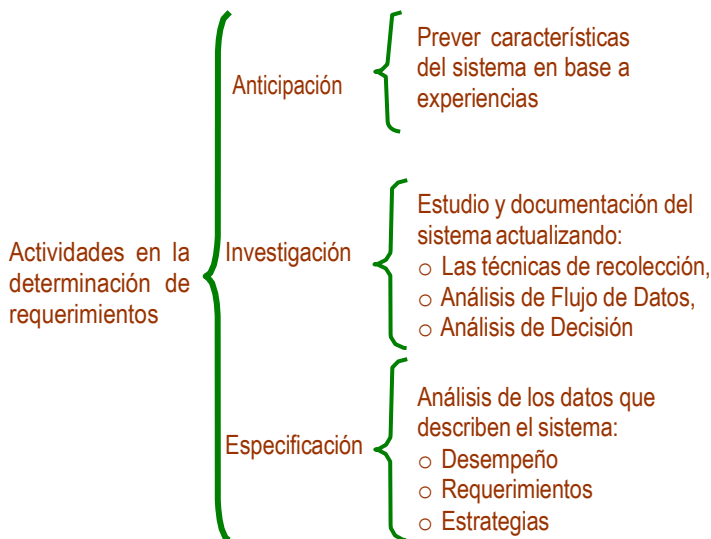
CONTENIDOS

- 4.8 Acciones para determinar los requerimientos
- 4.9 Requerimientos básicos
- 4.10 Técnicas para relevar datos
 - 4.3.1. La Entrevista
 - 4.3.2. El Cuestionario
 - 4.3.3. La Observación
 - 4.3.4. El Muestreo

4.1. ACCIONES PARA DETERMINAR LOS REQUERIMIENTOS

Determinación de requerimientos. Se refiere al estudio de un sistema para conocer como trabaja y donde es necesario efectuar mejoras.

Requerimientos. Es una característica que debe incluirse en un nuevo sistema.



4.2. REQUERIMIENTOS BASICOS

Los analistas estructuran sus investigaciones al buscar respuestas a las siguientes cuatro importantes preguntas:

- ¿Cuál es el proceso básico de la empresa?
- ¿Qué datos utiliza y produce este proceso?
- ¿Cuáles son los límites impuestos por el tiempo y la carga de trabajo?
- ¿Qué controles de desempeño utiliza?

Comprensión del proceso

Los analistas hacen preguntas que, cuando reciben respuestas, proporcionan antecedentes sobre detalles fundamentales relacionados con el sistema y que sirve para describirlo. Las siguientes preguntas son de utilidad para adquirir la comprensión necesaria:

- ¿Cuál es la finalidad d esta actividad dentro de la empresa?
- ¿Qué paso se siguen para llevarla a cabo?
- ¿Dónde se realizan estos pasos?
- ¿Quiénes lo realizan?
- ¿Cuánto tiempo tardan en efectuarlo?
- ¿Con cuanta frecuencia lo hacen?

- ¿Quiénes emplean la información resultante?

Frecuencia y volumen del proceso

La frecuencia con la que se presentan las actividades en una empresa cambia mucho. El volumen de artículos manejados, puede aumentar el tiempo necesario para completar la actividad.

Aunque la frecuencia de esta actividad es muy baja cuando el calendario inicia la actividad al finalizar cada trimestre, el volumen de trabajo es muy grande ya que, en ocasiones, se necesitan prepara decenas de miles de estados de cuenta. La cantidad total de pasos de que consta una actividad, puede generar problemas especiales para el estudio que efectúa el analista, aun cuando la actividad ocurra con poca frecuencia.

Identificación de controles

La falta o debilidad de los controles es un descubrimiento importante en cualquier investigación de sistemas.

Las dos secciones siguientes muestran como utilizar las preguntas básicas para comprender sistemas hacia transacciones y hacia decisiones.

Requerimientos de las transacciones de los usuarios

Los sistemas a nivel de transacciones, capturan, procesan y almacenan datos por alguna razón.

Los analistas seleccionados para trabajar en un sistema de procesamiento de pedidos, deben conocer todo lo relacionado con la forma en que se procesan estas transacciones. Para entender *los requerimientos de transacciones*, los analistas sin lugar a dudas formularan preguntas como las siguientes:

- ¿Qué es lo que forma parte de la transacción que esta siendo procesada?
- ¿Qué es lo que inicia la transacción?
- ¿Quién inicia los pedidos? ¿Con que propósito?
- ¿Con que frecuencia ocurren los pedidos?
- ¿Qué volumen esta asociado con cada pedido?
- ¿Existen diferentes condiciones que pueden afectar la forma en que se procesan los pedidos?
- ¿Qué detalles son necesarios para procesar las transacciones?
- ¿Qué información se genera? ¿Qué datos se guardan?

Requerimientos de decisión de usuarios

A diferencia de las actividades de transacción, las relacionadas con decisiones no siguen un procedimiento específico. Las rutinas no son muy claras y es posible que los controladores sean vagos. Las decisiones se toman al integrar la información en forma tal que los gerentes puedan saber que acciones emprender.

Los analistas que investigan sistemas para el soporte de decisiones deben formularse las mismas preguntas sobre frecuencia y volumen, mencionadas anteriormente, pero también hacerse otras para determinar *los requerimientos de las decisiones*:

- *¿Qué información se utiliza para la toma de decisiones?*
- *¿Cuál es la fuente de información?*
- *¿Qué sistemas de transacciones producen los datos utilizados en el proceso de decisión?*
- *¿Qué otros datos son necesarios y no es posible obtener del procesamiento de transacciones?*
- *¿Qué datos se originan en fuentes externas de la organización?*
- *¿Cómo se deben procesar los datos para producir la información necesaria?*
- *¿Cómo debe presentarse la información?*

Requerimientos de toda la organización

En las empresas, los departamentos dependen unos de otros para brindar servicios, fabricar productos y satisfacer a los clientes. Por consiguiente, el trabajo hecho en un departamento afecta al de los otros. Cuando los analistas estudian sistemas para un departamento también deben evaluar las implicaciones para los demás departamentos con los que interactúa el sistema bajo investigación. Algunas veces los sistemas fabrican el trabajo de varios departamentos. Es responsabilidad del analista identificar las dependencias entre departamentos y determinar como les afecta un proyecto de un sistema.

4.3. TÉCNICAS PARA RELEVAR DATOS

4.3.1. LA ENTREVISTA

Antes de que entreviste a alguien, primero se debe entrevistar uno mismo. Se necesita conocer sus referencias y la manera que afectará sus preferencias.

Educación, intelecto, contexto cultural y emociones directa en como filtros poderosos para lo que estará oyendo en las entrevistas.

TIPOS DE INFORMACION BUSCADA

Una entrevista para recolección de información es una conversación dirigida en un propósito específico que usa formato de preguntas y respuestas. En ella se quiere obtener la opinión del entrevistado y sus sentimientos acerca del estado actual del sistema, los objetivos de la organización, los personales y los procedimientos informales.

Antes que todo, observa las opiniones de la persona a la que está entrevistando. Las opiniones pueden ser más importantes y más reveladoras que los hechos. Además de las opiniones, se debe tratar de capturar los sentimientos del entrevistado. Recuerda aquel entrevistado conocer la organización mejor que uno.

Los sentimientos expresados ayudan a capturar la emoción y las actitudes.

Los objetivos son información importante que puede ser recogida de las entrevistas. Los hechos que se obtienen de los datos relevantes pueden explicar el desempeño pasado, pero los objetivos proyectan el futuro de la organización.

PLANEACION DE LA ENTREVISTA

LECTURA DE MATERIAL DE FONDO. Lea y comprenda tanta información de fondo acerca del entrevistado y su organización, le sea posible. Este material puede ser tenido mediante una llamada rápida a la persona de contacto para pedirle un reporte anual o de cualquier publicación que haya sido enviada para explicar la organización ante el público.

ESTABLECIMIENTO DE LOS OBJETIVOS DE LA ENTREVISTA. Usa la información de fondo que recopiló, así como su propia experiencia, para establecer los objetivos de la entrevista. Deben de hacer de 4 a 6 áreas que se relacionan con el procesamiento de la información y con el comportamiento de las tomas de decisiones: fuentes de información, formatos de la información, frecuencia de la toma de decisiones, cualidades de la información y estilo de la toma de decisiones.

DECIDIRA QUIEN ENTREVISTAR. Cuando esté decidiendo a quién entrevistar incluye agentes clave de todos los niveles que serán afectadas por el sistema en alguna forma. Es importante muestrear a los miembros organizacionales. Tratando de obtener balance para que sean tratadas tantas necesidades de los usuarios como sean posible.

PREPARE AL ENTREVISTADO. Prepare la persona a ser entrevistada, llamando lea con anticipación y permitiendo a que el entrevistado tenga tiempo para pensar acerca de la entrevista; las entrevistas deben durar 45 minutos a una hora, a lo mucho.

DECIDIRA SOBRE TIPOS DE PREGUNTAS Y ESTRUCTURAS. Este era preguntas para tratar las áreas principales de la toma de decisiones descubiertas cuando se averiguaron los objetivos de la entrevista.

TIPOS DE PREGUNTAS

Pregunta abiertas.

Describe por cierto la opción de alternativas para responder, la respuesta puede ser de dos palabras o dos párrafos.

Ventajas:

1. Pone confortable al entrevistado.
2. Proporciona riqueza de detalle.
3. Permite más espontaneidad.
4. Hace la construcción de frases sea más fácil para el entrevistador.

Desventajas:

1. La posibilidad de perder el control de la entrevista.

2. Permite respuestas que pueden llevarse demasiado tiempo para la cantidad de información obtenida.
3. Puede demostrar potencialmente que el entrevistador no está preparando.

Preguntas cerradas.

Las respuestas posibles están cerradas al entrevistado, debido a que solamente puede responder con un número finito, tal como “ninguno”, “uno”, o “quince”.

Un tipo de especie de pregunta cerrada es la pregunta bipolar, esto limita también más al entrevistado, permitiéndole solamente una selección de cualquier extremo, tal como sí o no, cierto o falso, acuerdo o desacuerdo.

Ventajas:

1. Se ahorra tiempo.
2. Se llega al punto.
3. Se mantiene control sobre la entrevista.
4. Se tratan muchos temas rápidamente.

Desventajas:

1. Ser aburridas para el entrevistado.
2. No llegan a entender grandes detalles (debido que el entrevistador proporciona el marco de referencia para el entrevistado).
3. Se pierden ideas principales por la razón anterior.

Averiguaciones.

El objetivo de averiguar es ir más allá de la respuesta inicial para obtener más significado para aclararlo y para obtener y expandir el punto de vista del entrevistado. Las averiguaciones pueden ser preguntas abiertas o cerradas.

ELUSION DE PREGUNTAS CONDUCENTES

Las preguntas conducentes tienden a dirigir al entrevistado hacia la respuesta que no parece querer. La respuesta es entonces sugerida, debido a que se está poniendo un tipo de trampa.

ELUSION DE PREGUNTAS NOBLES

Las preguntas nobles son aquellas en las se usa una sola pregunta para lo que de hecho son dos preguntas separadas. Una pregunta tal como “¿Qué decisiones toman durante el día típico y como las toman?”, este es un ejemplo de pregunta doble.

Una pregunta doble es una mala alternativa, debido a que el entrevistado puede responder solo una pregunta y la otra reservársela.

USO DE UNA ESTRUCTURA DE PIRÁMIDE

Se debe usar una estructura de pirámide si considera que el entrevistador necesita ambientarse en el tema. También es útil si el entrevistador parece ser que se resiste a entrar en el tema.

La estructura de pirámide para las entrevistas va de preguntas específicas a generales.

USO DE ESTRUCTURA DE EMBUDO

En este tipo de estructura el entrevistador toma un enfoque deductivo, comenzando con preguntas generales y abiertas y estrechando las respuestas posiblemente utilizando preguntas cerradas.

La estructura de embudo para la entrevista comienza con preguntas amplias y luego se estrecha hacia preguntas específicas.

USO DE LA ESTRUCTURA DE ROMBO

Frecuentemente es mejor una combinación de dos estructuras anteriores, dando como resultado la estructura de entrevista con forma de rombo, esto conlleva a comenzar en forma muy específica, luego examinar temas generales y por ultimo llegar a una conclusión muy específica.

ENTREVISTA ESTRUCTURADA CONTRA NO ESTRUCTURADA

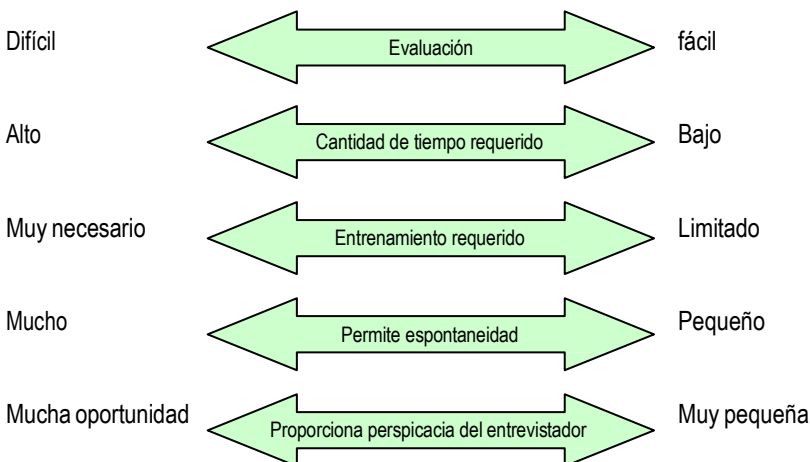
En una entrevista completamente estructurada todo esta planteado y el plan es seguido estrictamente. Las preguntas cerradas son la parte medular de una entrevista completamente estructurada.

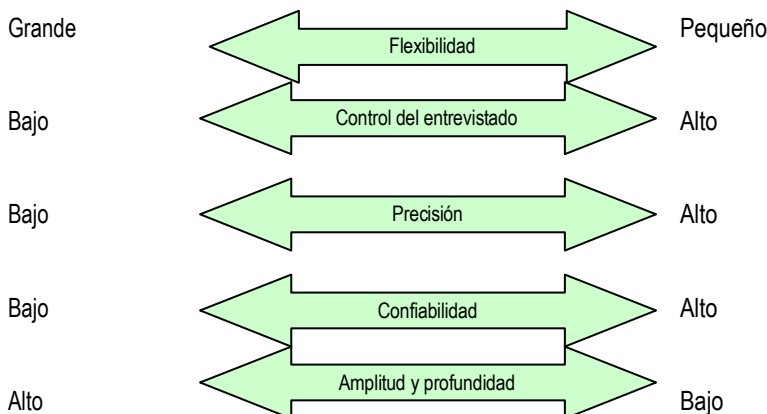
En el caso de las entrevistas no estructuradas es importante seguir un orden aunque no de sentido dentro de la entrevista.

Recuerde que lo no estructurado se refiere simplemente al orden en que se realizan las preguntas y no implica la falta de otra preparación. Es necesario que se proyecte que la otra persona puede decir. Esto lleva bastante tiempo y pensamientos, y es, por si mismo, un argumento excelente para preparar varios posibles camino dentro de la entrevista. Este enfoque es hecho un programa de ramificaciones, si la respuesta es si, sé continua de una manera, y si se no se toma la otra ruta de preguntas.

NO ESTRUCTURADA

ESTRUCTURADA





Atributos de entrevista estructurada y no estructurada a considerar cuando decide un formato de entrevista.

REGISTRO DE LA ENTREVISTA

Registre los aspectos más importantes de la entrevista. Puede usar una grabadora de cinta o un lápiz y papel para tomar notas.

Uso de una grabadora de cinta.

Es conveniente que antes de que se realice la entrevista tenga conocimiento de la existencia de la grabadora y el uso que se hará con ella después de la entrevista.

El registro en cintas tiene desventajas y ventajas.

Las ventajas son que la grabación en cinta logra lo siguiente:

1. Proporcionan un registro completo de lo que se dijo durante la entrevista.
2. Libera al entrevistador para escuchar y responder más rápido.
3. Permite mejor contacto visual, por consecuencia mayor desarrollo de una relación armónica entre el entrevistador y el entrevistado.
4. Permite la reproducción de la entrevista para otros miembros del equipo.

Las desventajas son:

1. El hacer posiblemente inquietante la entrevista y menos apta para responder libremente.
2. Posiblemente hacer que el entrevistador sea menos apto para escuchar, debido a que toda esta siendo grabado.
3. La dificultad de localizar pasajes importantes dentro de una cinta larga.
4. El incremento de costo de relación de datos, debido a la necesidad de transcribir a cinta.

Toma de notas

La toma de notas puede ser la única manera para registrar la entrevista en caso de que el entrevistado no quiera ser grabado.

Algunas de sus ventajas son:

1. Mantiene alerta al entrevistador.
2. Ayuda a recordar preguntas importantes.
3. Ayuda a recordar las ascendencias importantes de la entrevista.
4. Muestra el interés del entrevistador en la entrevista.
5. Demuestra la preparación del entrevistador.

Algunas de sus desventajas son:

1. La pérdida de contacto visual entre el entrevistador y el entrevistado.
2. La pérdida del hilo de la conversación.
3. Se hace que el entrevistador tema hablar cuando sé esta tomando nota.
4. Hace que se ponga excesiva atención a lo hechos y poca atención a los sentimientos y opiniones.

Antes de la entrevista

Es importante preparar cuando menos con un día antes de anticipación, y llegar temprano a la cita para la entrevista, vestir adecuadamente según la ocasión.

Inicio de la entrevista

Es importante dar un saludo de apretón de mano de manera firme esto se aplica a mujeres también, debemos cerciorarnos de que los micrófonos estén preparadas y funcionando adecuadamente.

Atrapando la impresión de la primera pregunta se puede atrapar el vocablo del entrevistado. También es importante empezar la entrevista con una buena pregunta y captar aptitudes y creencias del entrevistado sobre el tema a desarrollar.

El material debe ser cubierto en 45 min. O en una hora para evitar el aburrimiento del entrevistado y caída de rutina dentro de la entrevista. Fije citas futuras para entrevistas de seguimiento, déle las gracias al entrevistado por haberle dado parte de su tiempo y despídase con un apretón de mano.

Escritura del reporte de la entrevista

Es imperativo que se escriba el reporte de la entrevista tan pronto como sea posible. Esta es otra forma con la cual se aseguran los datos logrados. Entre más se demore en prepara su reporte más sospechoso se convertirá la calidad de los datos.

Recuerde hacer notar los puntos más importantes y trascendentales para un mayor entendimiento de los objetivos en la entrevista.

Entrevistada: Rosa Castillo	Fecha 3 de marzo
-----------------------------	------------------

Entrevistador: G.. Sánchez		Tema: Uso de computadora	
Objetivo de la entrevista: Encontrar la actitud acerca del uso de la computadora;		Obtener la estimación del usuario sobre el uso; encontrar	
		Opiniones de un nuevo sistema propuesto.	
¿Se lograron los objetivos?			
Objetivos para entrevista de seguimiento:			
Saber de que manera Rosa (enfoca) el soporte del departamento de sistemas.			
Encontrar opiniones sobre con quien platicar después.			
Puntos principales de la entrevista		Opiniones del entrevistador	
Rosa dijo “La computadora es mi amiga”		Esta interesada en aprender más acerca	
Usa la computadora todo el tiempo		de como puede el sistema ayudarle con su trabajo.	

Un reporte de seguimiento del entrevistador documentando los datos recopilados así como la reacción del entrevistador ante ellos.

Las entrevistas personales son consumidoras de tiempo, propensas a error y sus datos son propensas a malas interpretaciones: Un enfoque alternativo a la (JAD) fue desarrollado por IBM. La motivación del uso del JAD es reducir el tiempo (y por lo tanto el costo) requerido por las entrevistas personales, mejorar la calidad de los resultados de la valoración de los requerimiento de información y la creación de más identificación del usuario con el nuevo sistema de información a consecuencia del proceso participativo.

CONDICIONES QUE DAN SOPORTE AL USO DE “JAD”

Considere el diseño conjunto de aplicaciones cuando

1. Los grupos de usuarios están impacientes y quieren algo nuevo, y no una solución estándar a un problema típico.
2. la cultura organizacional da soporte a los comportamientos de la solución de problemas en conjunto entre varios niveles de empleados.
3. las analistas predicen que la cantidad de ideas generadas por medio de entrevistas persona a persona no será tan abundante como la cantidad de ideas posibles del ejercicio de un grupo amplio.
4. el flujo de trabajo organizacional permite la ausencia de personas importantes durante un bloque de tiempo de dos a cuatro días.

Quienes están involucrados

Las sesiones de diseño conjunto de aplicaciones incluyendo una variedad de participantes, analistas, ejecutivos, etc., que contribuirán con sus diferentes apariencias

y aptitudes a la sesión. El interés principal de usted es que todos los miembros del equipo del proyecto estén abiertos al enfoque JAD y lleguen a estar involucrados.

Trate de seleccionar usuarios que estén arriba del nivel de empleado de oficina que puede decir claramente que información necesita para ejecutar el trabajo, así como que es lo que se desea en un sistema de computadora nuevo o mejorado.

El líder de la sesión no debe ser un experto en análisis y diseño de sistemas, sino que alguien que tenga habilidades excelentes de comunicación para facilitar las iteraciones adecuadas.

Plantación de la sesión JAD

Una de las claves para trabajar en grupo JAD exitoso es el poner las bases por medio de estudio y planeación avanzadas.

Donde efectuar las reuniones de JAD

Se recomienda que se realicen las sesiones de dos a cuatro días en un lugar aparte, fuera de la organización, en ambientes agradables. Algunos grupos usan centros ejecutivos o hasta instalaciones con apoyo a decisiones de grupo en que se encuentran disponibles en las universidades principales.

El cuarto debe alojar confortablemente como máximo 20 personas invitadas. El equipo de apoyo a presentaciones mínimo debe incluir dos proyectos de transparencia, un pizarrón blanco, un rota folio y fácil acceso a una copiadora.

No realice las sesiones a menos que puedan todos aquellos que hayan sido invitados.

Logro de un análisis estructurado de las actividades del proyecto

IBM recomienda que las sesiones JAD examinen estos puntos en los proyectos de sistemas propuestos: Planeación, recepción, procesamiento y seguimiento de lo recibido, monitores y asignación, procesamiento, riesgo envío y evaluación. Para cada tema deben ser preguntadas y respondidas las preguntas sobre quién, qué, cómo, dónde y por qué.

Beneficios potenciales del uso de JAD en vez de la entrevista tradicional

Hay cuatro potencial principales que usted, los usuarios y el equipo de analistas de sistemas deben considerar cuando caloren las posibilidades del uso del diseño conjunto de aplicaciones.

1. Beneficio potencial es el ahorro de sobre las entrevistas tradicionales persona a persona.
2. El desarrollo puede continuar mucho más rápido.
3. Posibilidad de una propiedad del sistema de información.
4. Beneficio es el desarrollo creativo de los diseños.

Desventajas potenciales del uso de JAD

Las cuatro desventajas son:

1. Es que JAD requiere la dedicación de un gran bloque de tiempo por parte de los 18 a 20 participantes.
2. Sucede cuando la preparación de las sesiones JAD es inadecuada en cualquier aspecto o cuando el reporte de seguimiento y la documentación de especificaciones es incompleto.
3. El éxito del diseño resultante de sesiones JAD es menos predecible que el logro por medio de entrevistas estándar.
4. Las habilidades organizacionales necesarias y la cultura organizacional puede no estar desarrollada lo suficiente para el esfuerzo concertado que se quiere para ser productivo en u ambiente JAD.

EJEMPLO DE UNA ENTREVISTA

¿Cuál es el problema que se presenta en la empresa?

¿Cree usted que un analista o persona capacitada pueda ayudarlo?

SI

NO_

¿Cuál es el tipo de información que se maneja?

¿Usted es quien maneja el sistema? ó ¿Hay más personas que la utilizan? Especifique.

¿Durante qué tiempo utilizan el sistema?

¿Existen otras áreas que utilizan la misma información?

SI

NO_

Mencione los tipos de dificultades que se encuentran en su área de trabajo.

4.3.2. EL CUESTIONARIO

Los cuestionarios son técnicas de recopilación de información que permite que los analistas de sistemas estudien actitudes, comportamientos y características de varias personas principales en la organización que pueden ser afectadas por los sistemas actuales y propuestos.

Las actitudes son lo que la gente de la organización dice que quiere, las creencias son lo que la gente piensa que es, de hecho cierto, el comportamiento es lo que hacen los miembros de la organización y las características son propiedades de las personas o cosas.

Los cuestionarios pueden ser usados para determinar que tan amplio limitado es en realidad un sentimiento expresado en una entrevista. En forma inversa, los cuestionarios pueden ser usados para investigar a una gran muestra de usuarios de sistemas, para tratar de encontrar problemas o recoger cosas importantes antes de que las entrevistas sean realizadas.

PLANEACION PARA EL USO DE CUESTIONARIO

Primero se debe decidir lo que se está tratando de obtener mediante el uso del cuestionario. Considere el uso de cuestionario si:

1. Las personas a quienes necesita preguntarles están ampliamente dispersas.
2. En el proyecto de sistemas están involucrados una gran cantidad de personas y tiene sentido saber qué proporción de u grupo dado, aprueba o desaprueba una característica particulares del sistema propuesto.
3. Se está haciendo un estudio exploratorio y se quiere medir la opinión general antes de darle al proyecto de sistema una dirección específica.

USO DE ESCALAS EN CUESTIONARIOS

El escalamiento es el proceso de asignar números u otro símbolo a un atributo. Y estas escalas pueden ser arbitrarias.

FUNDAMENTOS DE LAS ESCALAS

Razones para las escalas:

1. Medir las actitudes o características de las personas que responden el cuestionario. En esta la respuesta pueden ser combinadas o agrupadas para reflejar esta información.
2. Hacer que los interlocutores juzguen los temas del cuestionario. Si el analista de sistemas está interesado en la manera en que es cotizado cada uno de los reportes de muestra.

Medición:

- *Nominal.* Los analistas lo que pueden hacer con ellas es obtener totales de cada clasificación.
- *Ordinal.* Permiten clasificación, implica también ordenamiento de rango.

- *De intervalo.* Poseen las características de que los intervalos entre cada uno de los números son iguales. Debido a esto se puede realizar operaciones matemáticas sobre los datos del cuestionario.
- *De relación.* Cuando los intervalos son iguales y hay un cero absoluto.

Validez y confiabilidad:

Validez es el rango con el cual la pregunta mide lo que el analista trata de medir.

Confiabilidad mide consistencia:

Si el cuestionario fue administrado una vez y luego nuevamente bajo las mismas condiciones y se obtuvieron los mismos resultados, se dice que el instrumento tiene consistencia externa. Si el cuestionario contiene subpartes y estas partes tienen resultados equivalentes, se dice que el instrumento tiene consistencia interna.

CONSTRUCCION DE ESCALAS

PROBLEMAS:

- *Lenidad (blandura).* Este es causado por interlocutores que califican a la ligera. Para evitar esto el analista mueve la categoría “promedio” a la izquierda o derecha del centro.
- *Tendencia central.* Cuando los interlocutores califican todo como promedio. Se puede mejorar:
 1. Haciendo que las diferencias sean más pequeñas a ambos extremos.
 2. Ajustando las fuerzas de los descriptores.
 3. Creando una escala con más puntos.
- *Efecto de Halo.* Cuando la impresión formada en una pregunta se transporta a la siguiente pregunta. ¿Los reportes desempeño son fáciles de leer?

DISEÑO Y ADMINISTRACION DEL CUESTIONARIO

Diseño del cuestionario

Muchos de los mismos principios que son relevantes para el diseño de formas para capturar datos son también importantes aquí. Aunque el objetivo del cuestionario es recolectar información sobre actitudes, creencias, comportamiento y características, cuyo impacto puede alterar sustancialmente el trabajo de los usuarios, los interlocutores no siempre están motivados para responder.

Formato del cuestionario

Para la buena presentación de un cuestionario manejamos los siguientes puntos que son muy necesarios para la elaboración de dicho formato:

1. Deje bastante espacio en blanco.
2. Deje suficiente espacio para las respuestas.
3. Pida al interlocutor que cierre las respuestas con un círculo.
4. Use objetivos que le ayuden a determinar el formato.
5. Sea consistente en estilo.
6. Orden de las preguntas.
7. Las preguntas importantes para los interlocutores van primero.
8. Agrupe conceptos de contenido similares.

Emplear tendencias asociativas de los interlocutores

El analista emplea los tipos de asociaciones que hacen los interlocutores y luego usa estas asociaciones para el ordenamiento de las preguntas.

Poner primero los conceptos menos controvertidos

En la valoración de lo que esta sucediendo en el negocio se tendrá que entrar en algunos temas que son divisores para grupos particulares.

Administración del cuestionario

Interlocutores

La decisión de quien recibirá el cuestionario se toma en conjunto con la asignación de objetivos para su resultado. El muestreo ayudará al nazista de sistemas a determinar que tipo de interlocutores deberán recibir el cuestionario.

Métodos para la administración del cuestionario

El analista de sistema cuenta con vario métodos para la realización del cuestionario, pero la selección del método a utilizar va a estar determinado por la situación de la problemática que existe en el negocio.

Las opciones incluyen:

- Reunir a todos los interlocutores involucrados a la vez.
- Manejar personalmente cuestionarios en blanco y recoger los llenos.
- Permitir que los interlocutores administren el cuestionario por sí mismo en el trabajo y lo depositen en una caja ubicada en el punto central.
- Enviar por correo los cuestionarios a los empleados de sucursales.

El permitir que los interlocutores administren por sí mismo el cuestionario se realiza frecuentemente. Las tasas de respuestas de este método son un poco menores que las de otras, debido a que las gentes pueden olvidar las formas, perderla o ignorarla a propósito. Sin embargo, la auto administración permite que la gente sienta su anonimato.

Una forma para incrementar la tasa de respuesta en forma auto administrado es poner un buzón central en el escritorio de algún empleado.

4.3.3. LA OBSERVACION

OBSERVACIÓN DE LA TOMA DE DECISIONES DEL GERENTE TIPICO

Los días de trabajo de los gerentes han sido descritos como una serie de interrupciones acompañadas de pequeñas ráfagas de trabajo, es por esto que para que los analistas de sistemas aprecien su trabajo, se usan las entrevistas y pos cuestionarios; a través de la observación los gerentes recopilan procesos, comparten y usan información para hacer que el trabajo se realice. Los pasos de la observación de las actividades de las decisiones del gerente son:

1. Decidir lo que va a ser observado.
2. Decidir a que nivel de concreción van a ser observadas las actividades.
3. Crear categorías que capturen adecuadamente las actividades principales.
4. Preparar escalas, listas de verificación y otros materiales adecuados para la observación.
5. Decidir cuando observar (muestreo de tiempos y eventos).

Muestreo de tiempos

Permite que el analista ponga intervalos específicos en los cuales pueda observar las actividades del gerente.

Ventajas:

1. Elimina la observación que pueda hacerse en cualquier memento.
2. Permite una vista representativa de las actividades frecuentes.

Desventajas:

1. Recolecta datos en forma fragmentada.
2. Se pierden decisiones importantes que son poco frecuentes.

Muestreo de eventos

Proporciona observaciones sobre un comportamiento íntegro en su contexto natural.

Ventajas:

1. Permite la observación de comportamientos conforme suceden.
2. Permite la observación de un evento considerado importante.

Desventajas:

1. Se lleva gran cantidad de tiempo del analista.
2. Se pierde una muestra representativa de decisiones frecuentes.

OBSERVACION DEL LENGUAJE CORPORAL DEL TOMADOR DE DECISIONES

A través del lenguaje corporal el analista debe ser capaz de comprender los requerimientos de información del tomador de decisiones, añadiendo dimensión a lo que está haciendo, dicho y llevando a la percepción extremadamente difícil y varía entre las culturas.

Pares de adjetivos

Sirven para registrar el comportamiento del tomador de decisiones.

Sistema de categorías

El analista determina categorías de actividad antes de que sean tomadas las decisiones.

Al guión del analista

Es una técnica para registrar el comportamiento observado.

El "ACTOR" es el tomador de decisiones que es observado "actuando" o tomando decisiones. El guión contiene el actor en la columna izquierda y sus opciones son listadas a la derecha. Las actividades empiezan con un verbo. El guión es un enfoque organizado y sistemático que demanda que el analista sea capaz de comprender y articular la acción tomada por cada tomador de decisiones. Este guión ayuda eventualmente en la determinación de información requerida en las decisiones principales o frecuentes realizadas por la persona observada.

ONSERVACION DEL AMBIENTE FISICO

Esto revela mucho acerca de sus requerimientos de información. Significa examinar las oficinas de los tomadores de decisiones.

OBSERVACION ESTRUCTURADA DEL AMBIENTE (STROBE)

Confirma o niega la narración organizacional de entrevistas o cuestionarios.

La observación es sistemática:

1. Sigue una metodología estándar y una clasificación estándar para el análisis d los elementos organizacionales que influncian la toma de decisiones.
2. Permite a otros analistas apliquen el mismo marco de trabajo analítico a la misma organización.
3. Limita el análisis a la organización a como existe durante la etapa de su ciclo actual de vida.

Elemento STROBE

1. *Ubicación de la oficina.* La ubicación d la oficia de un tomador de decisiones particular con respecto a las demás oficinas.
2. *Ubicación del escritorio del tomador de decisiones.* Proporciona pistas sobre el ejercicio de poder por el tomador de decisiones.
3. *Equipo de oficina fijo.* Se conforma de archiveros, libreros, etc.

4. *Propiedades.* Todo el equipo pequeño que se usa para procesar información (calculadoras, pantallas de video, lápices, etc.).
5. *Revistas y periódicos del negocio.* Estas revelan si el tomador de decisiones busca información externa o se apoya más en información interna.
6. *Iluminación y color de la oficina.* Nos indica la manera en que el tomador de decisiones recopila información.
7. *Vestimenta usada por los tomadores de decisiones.* El analista de sistemas puede obtener una comprensión de la credibilidad exhibida por los gerentes de la organización observando la vestimenta que usan en el trabajo.

Mediante el uso de STROBE el analista de sistemas puede obtener una mejor comprensión sobre la manera en que los gerentes recopilan, procesan, almacenan y usan información.

Alternativas de aplicación

Existen estrategias muy estructuradas, hasta sin estructura cuando se usa el enfoque STROBE.

Análisis de fotografías. Consiste en fotografiar el ambiente de los tomadores de decisiones y análisis posterior de las mismas sobre los elementos STROBE.

Ventajas:

1. Se puede hacer un documento al que se pueda hacer referencia repetidamente.
2. el fotógrafo se enfoca a los elementos STROBE.
3. Permite una comparación lado a lado de las organizaciones.
4. La fotografía proporciona detalles que se descuidan durante el contacto personal.

Desventajas:

1. El decidir que fotografiar.
2. A la larga puede probarse no obstruyente, inicialmente si lo es.

Enfoque de las listas de verificación/escala LIKERT

Es menos estructurada. Son escalas tipo LIKERT de cinco puntos en relación con siete características del tomador de decisiones que fueron observables por medio de elementos físicos en los ambientes organizacionales de los tomadores de decisiones.

Lista anecdótica con símbolos

Es menos estructurada. Consiste en usar analistas de verificación anecdótica con símbolos y abreviaturas significativas. Sirve para evaluar los elementos STROBE en comparación con la narración organizacional generada por medio de entrevistas.

Para usar esta técnica:

1. Determinar los temas organizacionales principales que se desprenden de las entrevistas.
2. se construye una matriz, que liste las ideas principales a partir de la narrativa organizacional, acerca de la recopilación, procesamiento, almacenamiento y compartición de la información (los renglones) y elementos STROBE (columnas).
3. Se compara la narrativa y las acciones, se usa uno de los cinco símbolos adecuados para caracterizar la relación entre la narración y el elemento relevante.
4. El analista crea una tabla que primero documenta y luego ayuda en el análisis de las observaciones.

COMPARACION DE LA OBSERVACION / NARRACION

Es menos estructurada. Mientras el analista de sistemas esté conciente de los elementos del mise-en-scene y éstos sean observados a conciencia se puede obtener apreciaciones valiosas, incluso sin la ayuda de analista de verificación, lo que lleva a realizar observaciones estructuradas.

4.3.4. EL MUESTREO

INTRODUCCION

Subyacente a todos los métodos de investigación, de entrevista y de observación, están las decisiones cruciales en relación con qué examinar y a quienes preguntar u observar. El analista de sistema puede tomar estas decisiones con base en un enfoque estructural llamado muestreo.

DEFINICION

El muestreo es el proceso de seleccionar sistemáticamente elementos representativos de una población. Cuando estos elementos seleccionados son examinados de cerca, se supone que el análisis revelará información útil acerca de la población como un todo.

El analista de sistemas tiene que tomar una decisión sobre dos puntos importantes:

- Primero, hay muchos reportes, formas, documentos de salida y memorándums que han sido generados por los miembros de la organización. ¿A cuáles de éstos debe prestar atención el analista de sistemas y cuáles deben ser ignorados?
- Segundo, una gran cantidad de empleados pueden ser afectados por el sistema de información propuesto. ¿A qué personas debe entrevistar el analista de sistemas de información, buscar información por medio de cuestionarios y observar en el proceso de llevar a cabo sus papeles de toma de decisión?

LA NECESIDAD DE MUESTREO

Hay muchas razones por las que un analista de sistemas quiera seleccionar una muestra representativa de los datos a examinar, o personas representativas, aplicar cuestionarios u observar.

Ellas incluyendo:

1. *Los costos contenidos.* El examen de cada papel y el hablar con todas las personas de la organización podría ser demasiado costoso para el analista de sistemas. La fotocopia de reportes, el pedir tiempo valioso de los empleados y el duplicar cuestionarios daría como resultado gastos innecesarios.
2. *La agilización de la recolección de datos.* El muestreo ayuda a acelerar el proceso, recolectando datos seleccionados en vez de todos los datos de la población completa. Además, al analista de sistemas le es enviado el problema de analizar todos estos datos de la población.
3. *La mejora de efectividad.* El muestro puede ayudar a mejorar la efectividad que si se obtuviera información más precisa. Adicionalmente, si menos gentes son entrevistadas el analista de sistemas puede tomarse el tiempo para completar datos incompletos o faltantes, mejorando así la efectividad de la recolección de éstos.
4. *La reducción de la ascendencia.* La ascendencia en la recolección de datos puede ser reducida por el muestreo. Cuando el analista de sistemas pide una opinión acerca de una característica permanente del sistema de información instalado, el ejecutivo entrevistado puede proporcionar una evaluación ascendente, debido a que hay muy poca posibilidad de cambiarlo.

DISEÑO DE MUESTREO

Los cuatro pasos que deben seguir un analista para diseñar una buena muestra son:

1. *Determinar los datos a ser recolectados o descritos.* El analista de sistemas necesita un plan realista sobre lo que hará con los datos una vez que los recolecte. Si se recolectan datos irrelevantes, se desperdicia tiempo en la recolección, almacenamiento y análisis de datos inútiles.
2. *Determinar la población a ser muestreada.* El analista de sistemas debe determinar cuál es la población. En el caso de datos relevantes, necesita decidir, por ejemplo, si son suficientes los dos últimos meses o si se necesitan los reportes de todo el año para ese análisis. En forma similar, cuando se decide a quien entrevistar, el analista de sistemas tiene que determinar si la población debe incluir un nivel de la organización o todos los niveles, o tal vez ir hasta el exterior del sistema para incluir la reacción de los clientes.
3. *Selección del tipo de muestra.* El analista de sistemas tiene cuatro tipos principales de muestra:

	No basadas en probabilidades	Basadas en probabilidades
Los elementos de muestra son seleccionados directamente sin restricciones.	Conveniencia	Aleatoria Simple

Los elementos de muestra son seleccionados de acuerdo con un criterio específico.	Intencionada	Aleatoria Compuesta
---	--------------	---------------------

En las muestras Aleatorias Complejas frecuentemente se pueden lograr objetivos del muestreo seleccionando uno de los enfoques más adecuados para el analista de sistemas los cuales son:

- *Muestreo Sistemático.* El analista de sistemas podría escoger entrevistar a cada enésima persona de una lista de empleados de la compañía. Sin embargo, este método tiene ciertas desventajas.
- *Muestreo Estratificado.* Es el más importante para el analista de sistemas. La estratificación es el proceso de identificación de subpoblaciones, o estratos, y luego la selección de objetos o personas a muestrear dentro de estas subpoblaciones. La estratificación también se aplica cuando el analista quiere usar diferentes subgrupos.
- *Muestreo Aglomerado.* Algunas veces el analista de sistemas debe seleccionar un grupo de documentos o personas a estudiar.
- *Decisión del tamaño de la muestra.* Es importante recordar que la cantidad absoluta es más importantes en el muestreo que en el porcentaje de la población. Podemos obtener resultados satisfactorios muestreando 20 personas en 200 ó 20 personas en 2, 000, 000.

El tamaño de la muestra depende de muchas cosas, algunas puestas por el analista de sistemas, algunas determinadas por lo que se sabe acerca de la población misma y otros factores imperantes. El analista de sistemas puede escoger la estimación de intervalo aceptables (esto es el grado de precisión deseado) y el error estándar (seleccionando el grado de precisión y el grado de confianza).

Lo que es más, las características de la población pueden cambiar el tamaño de la muestra. Si las cifras de ventas de un reporte van de \$10,000 a \$15,000, un tamaño de muestra pequeño seria suficiente para dar una estimación de las ventas promedio. Sin embargo, si las ventas van de \$1,000 a \$ 100,000, entonces se requerirá un tamaño de muestra mucho más grande.

DISEÑO DEL TAMAÑO DE LA MUESTRA

El tamaño de la muestra depende frecuentemente del costo involucrado en el tiempo requerido por el analista de sistemas o hasta del tiempo disponible de las personas de la organización. Esta sección le da al analista de sistemas algunos lineamientos para determinar el tamaño de muestra requerido bajo condiciones lineales.

Determinación del tamaño de muestra cuando se muestrean datos de atributos:
Algunas veces el analista de sistemas querrá encontrar qué proporción de personas de

una organización piensan de determinada manera o tienen determinadas características. Otras veces el analista puede necesitar saber qué porcentaje de las formas de entrada tienen errores. Este tipo de datos pueden ser mencionados como datos de atributos.

El analista de sistemas necesita seguir una serie de pasos, donde algunos son juicios subjetivos, para determinar el tamaño de muestra requerido. La siguiente es la lista de los siete pasos:

1. Determinar el atributo que se muestreará.
2. Localizar la base de datos o reportes donde se puede encontrar el atributo.
3. Examinar el atributo. Estimar p , la proporción de la población que tiene el atributo.
4. Tomar la decisión objetivo en relación con el intervalo estimado aceptable, i .
5. Seleccionar el nivel de confianza y buscar el coeficiente de confianza (valor z) en una tabla.
6. Calcular op , el error estándar de la proporción, de la manera siguiente:

$$Op = i/z$$

7. Determinar el tamaño de muestra necesario, n , usando la siguiente formula:

$$N = p(1-p)/op^2$$

Determinación del tamaño de muestra cuando se muestrean datos en variables:

Un analista de sistemas puede algunas veces necesitar recolectar información sobre cifras actuales, tales como ventas brutas, cantidad de artículos regresados o cantidades de error tecleados. Los datos de este tipo son mencionados como variables.

Los pasos para la determinación del tamaño de muestra necesario para las variables es similar a los pasos para los datos de atributos. Son los siguientes:

1. Determinar la variable que se muestreará.
2. Localizar la base de datos o reportes donde se puede encontrar la variable.
3. Examinar la variable para obtener alguna idea acerca de su magnitud y dispersión. Idealmente, sería útil saber la media para determinar un estimado de intervalo aceptable más adecuado, y la desviación estándar, s , para determinar el tamaño de muestra.
4. Tomar la decisión subjetiva en relación con el estimado de intervalo aceptable, i .
5. Seleccionar el nivel de confianza y buscar el coeficiente de confianza (valor z) en una tabla.
6. Calcular Ox , el error estándar de la media, de la manera siguiente:

$$Ox = i/z$$

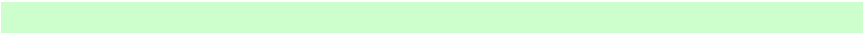
- 7.- Determinar el tamaño de muestra necesario, n , usando la siguiente formula:

$$n = (s/ox)^2 + 1$$

Determinación del tamaño de muestra cuando se muestrean datos cualitativos:
Una gran cantidad de información no puede ser obtenida buscando en archivos. Esa información puede ser obtenida en mejor forma entrevistando personas de la organización.

No hay fórmulas mágicas que ayuden al analista a determinar el tamaño de muestra para las entrevistas. La variable relevante que determina a qué tanta gente debe entrevistar el analista de sistemas a fondo es el tiempo que se lleva una entrevista. Una verdadera entrevista a fondo, y el seguimiento de ésta, consume demasiado tiempo, tanta para el entrevistador como para el participante.

Una buena regla empírica es entrevistar al menos a tres personas de cada nivel de la organización y al menos a una de cada una de las áreas funcionales de la organización.



ESTRATEGIAS PARA DETERMINAR LOS REQUERIMIENTOS DE SISTEMAS

CONTENIDOS

- | | |
|---|--|
| 5.31. Introducción | 5.47. Deducción de panorama lógico: |
| 5.32. Tareas del análisis | 5.48. Usos de diagramas físicos y lógicos de flujo de datos: |
| 5.33. Principios del análisis | 5.49. Reglas generales para el dibujo de diagramas lógicos de flujo de datos |
| 5.34. El dominio de la Información | 5.50. Diagramas de Entidad-Relación |
| 5.35. Partición | 5.51. Características del diccionario de datos |
| 5.36. Visiones Lógicas y Físicas | 5.52. Creación del diccionario de datos |
| 5.37. Métodos de Análisis de Requerimientos | 5.53. Uso del diccionario de datos |
| 5.38. Metodologías de Análisis de Requerimientos | 5.54. Descripción de los elementos dato |
| 5.39. Métodos de Análisis Orientados al Flujo de Datos | 5.55. Descripción de los datos |
| 5.40. Diagramas de Flujos de Datos | 5.56. Descripciones funcionales |
| 5.41. Diagramas físicos de flujo de datos | 5.57. Métodos Orientados a la Estructura de Datos |
| 5.42. Dibujo del diagrama de contexto | 5.58. Desarrollo de sistemas de Jackson |
| 5.43. Desarrollo de graficas de proceso | 5.59. Requerimientos de las bases de datos |
| 5.44. Desarrollo del primer nivel o diagrama cero de un diagrama físico de flujo de datos | 5.60. Características de las bases de datos |
| 5.45. Descripción del panorama lógico. | |
| 5.46. Diagrama lógico de flujo de datos – DFD | |

5.1. Introducción

El análisis de requerimientos es la tarea que plantea la asignación de software a nivel de sistema y el diseño de programas (Figura 1). El análisis de requerimientos facilita al ingeniero de sistemas especificar la función y comportamiento de los programas, indicar la interfaz con otros elementos del sistema y establecer las ligaduras de diseño que debe cumplir el programa. El análisis de requerimientos permite al ingeniero refinar la asignación de software y representar el dominio de la información que será tratada por el programa. El

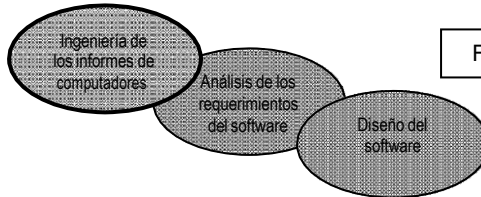


Figura 1

análisis de requerimientos de al diseñador la representación de la información y las funciones que pueden ser traducidas en datos, arquitectura y diseño procedimental. Finalmente, la especificación de requerimientos suministra al técnico y al cliente, los medios para valorar la calidad de los programas, una vez que se haya construido.

5.2. Tareas del Análisis

El análisis de requerimientos puede dividirse en cuatro áreas:

- 1.- Reconocimiento del problema
- 2.- Evaluación y síntesis
- 3.- Especificación
- 4.- Revisión.

Inicialmente, el analista estudia la especificación del sistema (si existe) y el plan de proyecto. Es importante comprender el contexto del sistema y revisar el ámbito de los programas que se usó para generar las estimaciones de la planificación. A continuación, debe establecerse la comunicación necesaria para el análisis, de forma que se asegure el reconocimiento del problema.

Las formas de comunicación requeridas para el análisis se ilustran en la Figura 2. El analista debe establecer contacto con el equipo técnico y de gestión del usuario/cliente y con la empresa que vaya a desarrollar el software. El gestor del programa puede servir como coordinador para facilitar el establecimiento de los caminos de comunicación. El objetivo del analista es reconocer los elementos básicos del programa tal como lo percibe el usuario/cliente.

La evaluación del problema y la síntesis de la solución es la siguiente área principal de trabajo del análisis. El analista debe evaluar el flujo y estructura de la información, refinar en detalle todas las funciones del programa, establecer las características de la interfase del sistema y descubrir las ligaduras del diseño, Cada una de las tareas sirve para descubrir el problema de forma que pueda sintetizarse un enfoque o solución global.

Las tareas asociadas con el análisis y especificación existen para dar una representación del programa que pueda ser revisada y aprobada por el cliente. En un mundo ideal el cliente desarrolla una especificación de requerimientos del software completamente por sí mismo. Esto se presenta raramente en el mundo real. En el mejor de los casos, la especificación se desarrolla conjuntamente entre el cliente y el técnico.

Una vez que se hayan descrito las funcionalidades básicas, comportamiento, interfase e información, se especifican los criterios de validación para demostrar una comprensión de una correcta implementación de los programas. Estos criterios sirven como base para hacer una prueba durante el desarrollo de los programas. Para definir las características y atributos del software se escribe una especificación de requerimientos formal. Además, para los casos en los que se desarrolle un prototipo se realiza un manual de usuario preliminar.

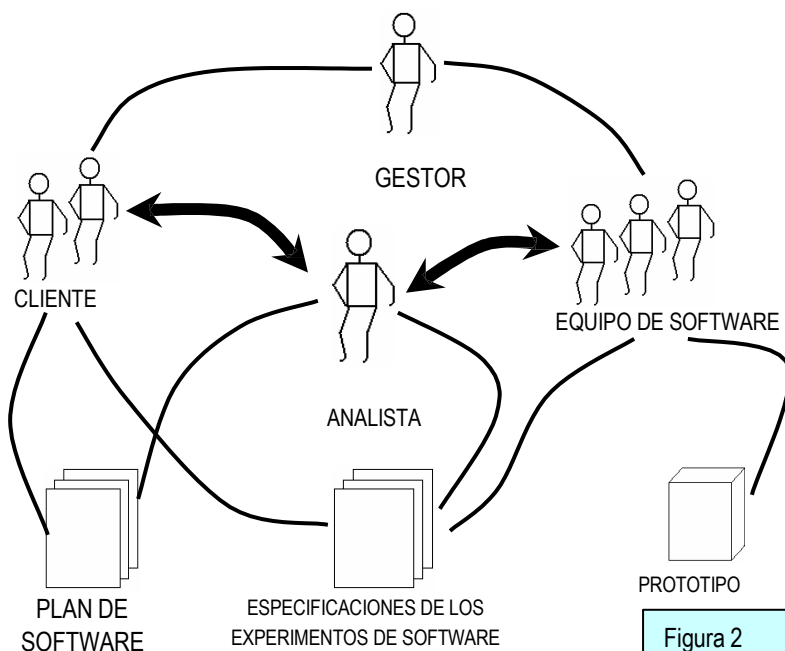


Figura 2

Puede parecer innecesario realizar un manual de usuario en una etapa tan temprana del proceso de desarrollo, Pero de hecho, este borrador del manual de usuario fuerza al analista a tomar el punto de vista del usuario del software. El manual permite al usuario / cliente revisar el software desde una perspectiva de ingeniería humana y frecuentemente produce el comentario: "La idea es correcta pero esta no es la forma en que pensé que se podría hacer esto". Es mejor descubrir tales comentarios lo mas tempranamente posible en el proceso.

Los documentos del análisis de requerimiento (especificación y manual de usuario) sirven como base para una revisión conducida por el cliente y el técnico. La revisión de los requerimientos casi siempre produce modificaciones en la función, comportamiento, representación de la información, ligaduras o criterios de validación. Además, se realiza una nueva apreciación del plan del proyecto de software para determinar si las primeras estimaciones siguen siendo validas después del conocimiento adicional obtenido durante el análisis.

5.3. Principios del Análisis

En el siglo pasado, se desarrollaron varios métodos de análisis y especificación del software. Los investigadores han identificado los problemas y sus causas y desarrollando reglas y procedimientos para resolverlos. Cada método de análisis tiene una única notación y punto de vista. Sin embargo, todos los métodos de análisis están relacionados por un conjunto de principios fundamentales:

El dominio de la información, así como el dominio funcional de un problema debe ser representado y comprendido.

El problema debe subdividirse de forma que se descubran los detalles de una manera progresiva (o jerárquica)

Deben desarrollarse las representaciones lógicas y físicas del sistema.

Aplicando estos principios, el analista enfoca el problema sistemáticamente. Se examina el dominio de la información de forma que pueda comprenderse su función mas completamente. La partición se aplica para reducir la complejidad. La visión lógica y física del software, es necesaria para acomodar las ligaduras lógicas impuestas por los requerimientos de procesamiento, y las ligaduras físicas impuestas por otros elementos del sistema.

5.4. El dominio de la Información

Todas las aplicaciones del software pueden colectivamente llamarse procesamiento de datos. Este término contiene la clave de lo que entendemos por requerimientos del software.

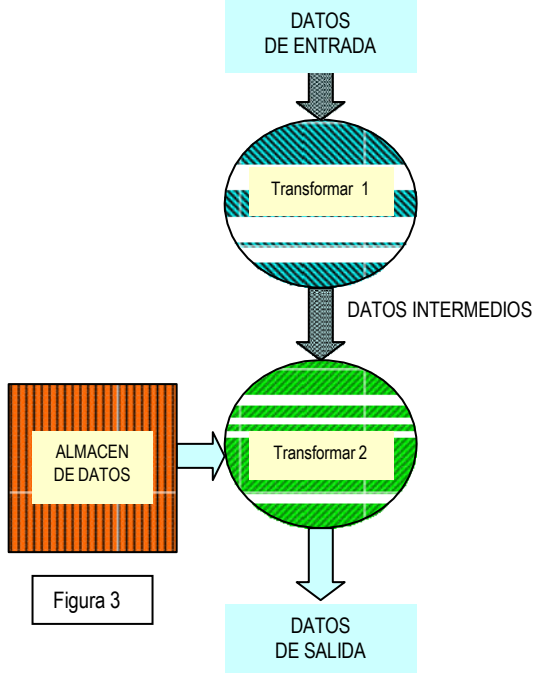


Figura 3

El software se construye para procesar datos; para transformar datos de una forma a otra; esto es, para aceptar entrada, manipularla de alguna forma y producir una salida. Este establecimiento fundamental de los objetivos es verdad tanto si construimos software por lotes para un sistema de planillas, como software empotrado en tiempo real para controlar el flujo de la gasolina de un motor de automóvil; el dominio de la información contiene tres visiones diferentes de los datos que se procesan por

los programas de computadoras:

- 1) el flujo de información;
- 2) el contenido de la información y
- 3) la estructura de la información.

Para comprender completamente el dominio de la información, deben considerarse cada una de estas tres partes.

El flujo de la información representa la manera en la que los datos cambian conforme pasan a través de un sistema. Refiriéndonos a la Figura 3, la entrada se transforma en datos intermedios y más adelante se transforma en la salida.

5.5. Partición

Normalmente los problemas son demasiado grandes y complejos para ser comprendidos como un todo. Por esta razón, tendemos a particionar (dividir) tales problemas en partes que puedan ser fácilmente comprendidas, y establecer interfases entre las partes, de forma que se realice la función global. Durante el análisis de requerimientos, el dominio funcional y el dominio de la información del software pueden ser particionados.

En esencia la partición descompone un problema en sus partes constituyentes. Conceptualmente, establecemos una representación jerárquica de la función o información y luego partimos el elemento superior mediante: 1) incrementando los detalles, moviéndonos verticalmente en la jerarquía, o 2) descomponiendo funcionalmente el problema, moviéndonos horizontalmente en la jerarquía.

5.6. Visiones Lógicas y Físicas

La visión lógica de los requerimientos del software presenta las funciones que han de realizarse y la información que ha de procesarse independientemente de los detalles de implementación.

La visión física de los requerimientos del software presenta una manifestación del mundo real de las funciones de procesamiento y las estructuras de información. En algunos casos se desarrolla una representación física como el primer paso del diseño del software. Sin embargo la mayoría de los sistemas basados en computador, se especifican de forma que se dictan ciertas recomendaciones físicas.

5.7. Métodos de Análisis de Requerimientos

Las metodologías de análisis de requerimientos combinan procedimientos sistemáticos con una notación única para analizar los dominios de información y funcional de un problema de software; suministra un conjunto de heurísticas para subdividir el problema y define una forma de representación para las visiones lógicas y físicas. En esencia, los métodos de análisis de requerimientos del software, facilitan al ingeniero de software aplicar principios de análisis fundamentales, dentro del contexto de un método bien definido.

La mayoría de los métodos de análisis de requerimientos son conducidos por la información. Estos es, el método suministra un mecanismo para representar el dominio de la información. Desde esta representación, se deriva la función y se desarrollan otras características de los programas.

El papel de los métodos de análisis de requerimientos, es asistir al analista en la construcción de "una descripción precisa e independiente" del elemento software de un sistema basado en computadora.

5.8. Metodologías de Análisis de Requerimientos

Las metodologías de análisis de requerimientos facilitan al analista la aplicación de los principios fundamentales del análisis de una manera sistemática.

Características Comunes

Aunque cada método introduce nueva notación y heurística de análisis, todos los métodos pueden ser evaluados en el contexto de las siguientes características comunes:

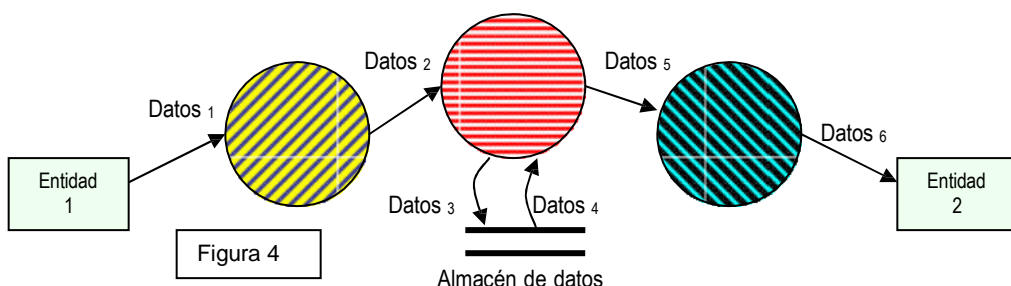
1. Mecanismos para el análisis del dominio de la información
2. Método de representación funcional
3. Definición de interfaces
4. Mecanismos para subdividir el problema
5. Soporte de la abstracción
6. Representación de visiones físicas y lógicas

Aunque el análisis del dominio de la información se conduce de forma diferente en cada metodología, pueden reconocerse algunas guías comunes. Todos los métodos se enfocan (directa o indirectamente) al flujo de datos y al contenido o estructura de datos. En la mayoría de los casos el flujo se caracteriza en el contexto de las transformaciones (funciones) que se aplican para cambiar la entrada en la salida. El contenido de los datos puede representarse explícitamente usando un mecanismo de diccionario o, implícitamente, enfocando primero la estructura jerárquica de los datos.

Las funciones se describen normalmente como transformaciones o procesos de la información. Cada función puede ser representada usando una notación específica. Una descripción de la función puede desarrollarse usando el lenguaje natural, un lenguaje procedimental con reglas sintácticas informales o un lenguaje de especificación forma.

5.9. Métodos de Análisis Orientados al Flujo de Datos

La información se transforma como un flujo a través de un sistema basado en computadora. El sistema acepta entrada de distintas formas; aplica un hardware, software y elementos humanos para transformar la entrada en salida; y produce una salida en distintas formas. La entrada puede ser una señal de control transmitida por un transductor, una serie de números escritos por un operador humano, un paquete de información transmitido por un enlace a red, o un voluminoso archivo de datos almacenado en memoria secundaria. La transformación puede comprender una sencilla comparación lógica, un complejo algoritmo numérico, o un método de inferencia basado en regla de un sistema experto. La salida puede encender un sencillo led o producir un informe de 200 paginas. En efecto, un modelo de flujo de datos puede aplicarse a cualquier sistema basado en computadora independientemente del tamaño o complejidad.



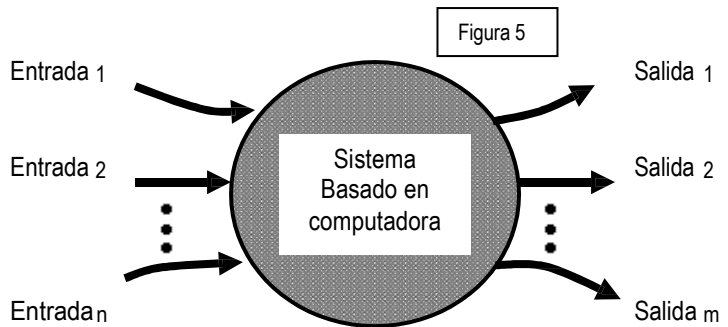
Una técnica para representar el flujo de información a través del sistema basado en computadora se ilustra en la figura 4. La función global del sistema se representa como una transformación sencilla de la información, representada en la figura como una burbuja. Una o más entradas. Representadas como flechas con etiqueta, conducen la transformación para producir la información de salida. Puede observarse que el modelo

puede aplicarse a todo el sistema o solo a un elemento de software. La clave es representar la información dada y producida por la transformación.

5.10. Diagramas de Flujos de Datos

Conforme con la información se mueve a través del software, se modifica mediante una serie de transformaciones. Un diagrama de flujos de datos (DFD), es una técnica grafica que describe el flujo de información y las transformaciones que se aplican a los datos, conforme se mueven de la entrada a la salida. La forma básica de un DFD se ilustra en la figura 5. El diagrama es similar en la forma a otros diagramas de flujo de actividades, y ha sido incorporado en técnicas de análisis y diseños propuesto por

Yourdon y Constantine, DeMarco y Gane y Sarson. También se le conoce como un grafo de flujo de datos o un diagrama de burbujas.



5.11. Diagramas físicos de flujo de datos

Proporcionan un panorama del sistema en uso, que es dependiente de la implantación, que muestra que tareas se llevan a cabo y cómo. Las características físicas incluyen:

- Nombres de personas.
- Nombres con números de formatos y documentos.
- Nombres de departamentos.
- Archivos maestros y de transacciones.
- Equipo y dispositivos utilizados.
- Ubicaciones.
- Nombres de procedimientos.

El enfoque más amplio y útil para desarrollar una descripción exacta y completa del sistema en uso, comienza con el desarrollo de los diagramas físicos de flujo de datos. El empleo de estos diagramas especiales por tres razones.

Primera, es común que los analistas de sistemas encuentren mucho más fácil escribir la interacción entre los componentes físicos que comprender la política empleada para administrar la aplicación.

Segunda, los diagrama físicos de flujo de datos son de utilidad para comunicarse con los usuarios. Están relacionados con facilidad a las personas, las localidades y los documentos ya que trabajan todos los días en cada entidad.

Tercera, los diagramas físicos de flujo de datos proporcionan un camino para validar y verificar el punto de vista del usuario sobre la forma en que el sistema en uso.




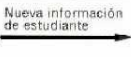



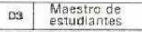
5.12. DIBUJO DEL DIAGRAMA DE CONTEXTO

Como ya se indico, los primeros pasos para determinar los requisitos tiene como finalidad conocer las características generales del proceso bajo investigación. Conforme los analistas comprenden mejor los detalles, ahondan con mayor profundidad para recopilar información más precisa y detallada. Cada mes se formulan preguntas más específicas utilizando para ello el análisis descendente.

A menudo al diagrama de alto nivel se le denomina DIAGRAMA DE CONTEXTO define el sistema que va ser estudiado en el sentido de que determinan las fronteras.

5.13. DESARROLLO DE GRAFICAS DE PROCESO

Un sistema está formado por varias actividades o procesos. En la programación de computadoras, los programadores con frecuencia desarrollan el software como una colección de módulos independientes pero interactúan entre si. A menudo estos módulos encuentran en los diagramas de jerarquía.

Simbolos	Significado	Ejemplo
	ENTIDAD	
	FLUJO DE DATOS	
	PROCESO	
	ARCHIVO o ALMACEN DE DATOS	

Estos diagramas son similares a los desarrollados por los programadores. Los diagramas de jerarquía de procesos continúan hasta los niveles que sean necesarios para identificar las actividades que forman parte del sistema.

5.14. DESARROLLO DEL PRIMER NIVEL O DIAGRAMA CERO DE UN DIAGRAMA FISICO DE FLUJO DE DATOS

Algunos analistas encuentran ventajoso trabajar primero con todos los flujos de datos o diagramas de detalle y asimilar nombres que sean descriptivos y útiles. Se identifican todos los procesos pero no se les da nombre hasta que están bien comprendidos todos los flujos de datos.

Después, cuando se les ha asignado nombre de los procesos, si el analista tiene dificultad para unir los flujos de datos con los nombres apropiados entonces esta situación indica que es necesario dividir aún más el proceso. Para algunos analistas lo anterior da buenos resultados, para describir el sistema.

El diagrama físico de flujo de datos, emplea sólo símbolos estándar para describir el sistema de soporte automatizado para preparar de diagramas de flujo de datos.

5.15. DESCRIPCION DEL PANORAMA LOGICO.

Los diagramas físicos de flujo de datos son un medio para alcanzar un fin, no un fin en si mismo.

Recuerde que se elaboran para describir la implantación del sistema existente, aspecto que este interés por dos razones:

- Estar seguro de tener la comprensión correcta de la implantación real del sistema existente.
- La propia implantación puede ser un problema un factor limitante; cambiar la implantación, más que el concepto del sistema; proporcionan los resultados deseado.

5.16. DIAGRAMA LOGICO DE FLUJO DE DATOS - DFD

Proporcionan un panorama del sistema independiente de la implantación, que se centra el flujo de datos entre los procesos sin considerar los dispositivos específicos y la localización de almacenes de datos o personas en el sistema. En este tipo de diagramas no se indican las características físicas, las cuales sí suceden con los diagramas físicos de flujo.

5.17. DEDUCCIÓN DE PANORAMA LÓGICO:

El panorama lógico es una visión retrospectiva de la implantación actual y proporciona la base para examinar la combinación de procesos, flujo de datos, almacenes de datos, entradas y salidas sin tomar en cuenta dispositivos físicos, personas a los aspectos de control que caracterizan la implantación.

El diagrama lógico de flujo de datos se obtiene del diagrama físico al llevar a cabo lo siguiente:

- Señalar los datos necesarios en este momento para un proceso, no los documentos que los contiene.
- Promover la información relacionada con las rutas de datos; esto es, indicar el flujo entre posprocedimientos y no entre personas, oficinas o localidades.
- Remover las herramientas y dispositivos (por ejemplo: fólder y gabinetes de archivos).
- Remover la información de control.
- Consolidar los almacenes de datos redundantes.

- Remover los procesos innecesarios, como los que no cambian los datos o flujo de datos (por ejemplo: de itinerario, de almacenamiento y de copiado), y que son independiente de los dispositivos donde ocurre (preparación o actividades de entrada de datos), o que representan un proceso único dentro del sistema (si existen procesos duplicados entonces deben considerarse en un comentario al margen).

5.18. USOS DE DIAGRAMAS FÍSICOS Y LÓGICOS DE FLUJO DE DATOS:

Cuando se inicia el estudio de sistemas en un área poco familiar, el analista necesita obtener un panorama del terreno.

Una vez que se tiene el panorama del terreno se puede estudiar con mayor cuidado los aspectos esenciales de una tarea. Para esto es necesario, por lo decirlo de algún modo, ir debajo de la superficie. Los diagramas lógicos de flujo son los que permiten hacer lo anterior.

Los diagramas lógicos de flujo de datos describen datos, procesos y eventos en forma diferente. Son más abstractos que sus contrapartes físicas, pero esta diferencia es importante.

Al centrarse en los elementos de fondo, aspectos lógicos y no físicos, es cuando el analista comprende la estructura del sistema. Solo entonces el analista puede desarrollar una comprensión completa y centrar las bases para diseñar el sistema correcto.

5.19. REGLAS GENERALES PARA EL DIBUJO DE DIAGRAMAS LÓGICOS DE FLUJO DE DATOS

1. Cualquier flujo de datos que abandone un proceso debe estar basado en los datos que entran al proceso.
2. Todos los flujos de datos reciben un nombre, el nombre refleja los datos que influyen entre procesos, almacenes de datos, fuente o destinos.
3. Solo deben entrar al proceso los datos necesarios para llevarlo a cabo.
4. un proceso no debe saber nada de ningún otro en el sistema.
5. Los procesos siempre están en continua ejecución; no se inician y tampoco se detienen (los sistemas nunca son estáticos).
6. La salida de los procesos pueden tomar una de las siguientes formas:
 - a. Flujo de datos con información añadida por el proceso (por ejemplo una anotación en la factura).
 - b. Una respuesta o cambio en la forma de datos (como un cambio en la forma de expresar las utilidades de dólares a porcentajes).
 - c. Un cambio de condición (de no autorizado a autorizado).
 - d. Un cambio de contenido (integración o separación de la información contenida en uno o más flujos entrantes de datos).

- e. Cambio en la organización (por ejemplo separación física o reacomodo de datos).
- En los diagramas de flujo lógicos es innecesaria cierta información física sobre los controles.
- Los elementos importantes para comprender un proceso durante el análisis lógico de flujo lógico de datos, no son los números de copias de los documentos y sino las descripciones de los datos necesarios para llevar a cabo el proceso.

5.20. Diagramas de Entidad-Relación

El diagrama de entidad-relación (también conocido como DER, o diagrama E-R) es un modelo de red que describe con un alto nivel de abstracción la distribución de datos almacenados en un sistema.

Para el analista, el DER representa un gran beneficio: porque enfatiza las relaciones entre almacenes de datos en el DFD que de otra forma se hubiera visto sólo en la especificación de procesos. Por ejemplo, un DER típico se muestra en la figura 3.6. Cada una de las cajas rectangulares corresponden a un almacén de datos en DFD y puede verse que hay relaciones que normalmente no se aprecian en un DFD.



Figura 3.6: Diagrama de entidad-relación típico.

5.21. Diccionario de Datos

Es un documento de referencia de datos compilados por los analistas de sistemas para guiarse a través del análisis y diseño. El diccionario de datos recolecta, coordina y confirma lo que significa un término de datos específicos para diferentes personas de la organización.

En un diccionario de datos se encuentra la lista de todos los elementos que forman parte del flujo de datos de todo el sistema.

Los elementos más importantes son flujos de datos, almacenes de datos (archivos o datos almacenados) y datos usados en los procesos [transformaciones]. El diccionario guarda detalles y descripciones de todos estos elementos.

Características del diccionario de datos:

- Volumen de datos.
- Proporciona información relacionada con el sistema.

- Recolecta, coordina y confirma lo que significa un término de datos específico para diferentes personas de la organización.
- Puede servir como estándar consistente para los elementos de datos.
- Los diccionarios de datos automatizados son valiosos por su capacidad para hacer referencias cruzadas de conceptos de datos.

5.22. Creación del diccionario de datos

Un paso importante en la creación de diccionarios de datos es identificar y categorizar el flujo de datos de entrada y salida del sistema.

La información puede ser guardada en numerosos lugares y en cada lugar el almacén de datos puede ser diferente. Mientras los flujos de datos representan los datos en movimiento, los almacenes de datos representan a los datos en reposo.

5.23. Uso del diccionario de datos

El diccionario de datos ideal es automatizado, en línea y revolucionario. Estos permiten mejoras en el mantenimiento de documentación. Con la creación de un diccionario de datos se pueden ahorrar muchas horas en las fases del análisis y diseño del sistema.

Contenido de un registro del diccionario:

El diccionario contiene dos tipos de descripciones para el flujo de datos dentro del sistema:

- Elemento de dato: Nivel más importante de datos. Los elementos de dato son los bloques básicos para todos los demás datos del sistema. Por sí mismo no conllevan suficiente significado para ningún usuario.
- Estructura de datos: Es un grupo de datos elementales que están relacionados con otros y que en conjunto describen un componente del sistema. Los flujos y los almacenes de datos son estructuras de datos. Están formados por elementos relevantes que describen la actividad o entidad bajo estudio.

5.24. Descripción de los elementos de dato:

Cada entrada en el diccionario de datos consiste de un conjunto de detalles que describen los datos utilizados o producidos por el sistema, cada uno está identificado con un nombre, descripción, alias y longitud, junto con el intervalo de valores específicos para el dato permitidos por el sistema bajo estudio.

Nombre de los datos: Sirven para distinguir un dato de otro. Los nombres se emplean para hacer referencia a cada elemento durante todo el proceso de desarrollo de sistemas. Debe tenerse cuidado al seleccionar nombres para que estos sean comprensibles y significativos.

Algunas organizaciones imponen estándares para el desarrollo de nombres de datos. Un estándar común específico que los nombres de los datos no deben ser mayores de

treinta caracteres (letras mayúsculas desde la A hasta la Z; números de 0 hasta 9 y el guión) y tampoco debe tener espacios en blanco.

5.25. Descripción de los datos:

Indica de manera breve lo que este representa en el sistema. Las descripciones de datos deben escribirse con la suposición que cada persona de que la persona que las leerá no sabe nada con respecto al sistema.

Deben evitarse la jerga del campo o los términos especiales; todas las palabras deben ser comprensibles para el lector.

Alias: Con frecuencia el mismo dato recibe varios nombres, mismos que dependen de quien haga uso del dato. Estos nombres se denominan alias. Un diccionario de datos significativo debe incluir todos los alias.

Longitud: Identifica el numero de espacios (para letras, números o símbolos) necesarios para cada dato pero sin considerar la forma en que serán almacenados. En otras palabras, si el nombre consta de 30 caracteres cuando sea escrito sobre una forma de pedido, entonces la entrada correspondiente en el diccionario de datos debe señalar una longitud igual a 30.

Valores de los datos: A menudo los números de orden de compra en muchas organizaciones tienen como prefijo una letra que indica el departamento que dio origen a la orden.

5.26. Descripciones Funcionales

Una vez que ha sido representado el dominio de la información (usando un DFD y un diccionario de datos), el analista describe cada función (transformación) representada, usando el lenguaje natural o alguna otra notación estilizada. Una de tales notaciones se llama ingles estructurado (también llamado lenguaje de diseño del programa o proceso (LDP)). El ingles estructurado incorpora construcciones procedimentales básicas – secuencia, selección y repetición- junto con frases del lenguaje natural, de forma que pueden desarrollarse descripciones procedimentales precisas de las funciones representadas dentro de un DFD.

5.27. Métodos Orientados a la Estructura de Datos

Hemos ya observado que el dominio de la información para un problema de software comprende el flujo de datos, el contenido de datos y la estructura de datos. Los métodos de análisis orientados a la estructura de datos representan los requerimientos del software enfocándose hacia la estructura de datos en vez de al flujo de datos. Aunque cada método orientado a la estructura de datos tiene un enfoque y notación distinta, todos tienen algunas características en común:

- 1) Todos asisten al analista en la identificación de los objetos de información clave (también llamados entidades o ítems) y operaciones (también llamadas acciones o procesos);
- 2) Todos suponen que la estructura de la información es jerárquica;
- 3) Todos requieren que la estructura de datos se represente usando la secuencia, selección y repetición; y
- 4) Todos dan un conjunto de pasos para transformar una estructura de datos jerárquica en una estructura de programa.

Como los métodos orientados al flujo de datos, los métodos de análisis orientados a la estructura de datos proporcionan la base para el diseño de software. Siempre puede extenderse un método de análisis para que abarque el diseño arquitectural y procedimental del software.

5.28. Desarrollo de Sistemas de Jackson

El desarrollo de sistema de Jackson (DSJ) se obtuvo a partir del trabajo de M.A. Jackson sobre el análisis del dominio de la información y sus relaciones con el diseño de programas y sistemas. En palabras de Jackson: "El que desarrolla el software comienza creando un modelo de la realidad a la que se refiere el sistema, la realidad que proporciona su materia objeto [del sistema]..."

Para construir un DSJ el analista aplica los siguientes pasos:

Paso de las acciones y entidades. Usando un método muy similar a la técnica de análisis orientada al objeto, en este paso se identifican las entidades (persona, objetos u organizaciones que necesita un sistema para producir o usar información) y acciones (los sucesos que ocurren en el mundo real que afectan a las entidades).

Paso de estructuración de las entidades. Las acciones que afectan a cada entidad son ordenadas en el tiempo y representadas mediante diagramas de Jackson (una notación similar a un árbol).

Paso de modelación inicial. Las entidades y acciones se representan como un modelo del proceso; se definen las conexiones entre el modelo y el mundo real.

Paso de las funciones. Se especifican las funciones que corresponden a las acciones definidas.

Paso de temporización del sistema. Se establecen y especifican las características de planificación del proceso.

Paso de implementación. Se especifica el hardware y software como un diseño.

Los últimos tres pasos del DSJ están muy relacionados con el diseño de sistemas.

5.29. Requerimientos de las Bases de Datos

El análisis de requerimientos para una base de datos incorpora las mismas tareas que el análisis de requerimientos del software. Es necesario un contacto estrecho con el cliente; es esencial la identificación de las funciones e interfaces; se requiere la especificación del flujo, estructura y asociatividad de la información y debe desarrollarse un documento formal de los requerimientos.

5.30. Características de las bases de datos

El termino base de datos se ha convertido en uno de los muchos tópicos del campo de las computadoras. Aunque existen muchas definiciones elegantes, definiremos una base de datos como: una colección de información organizada de forma que facilita el acceso, análisis y creación de informes. Una base de datos contiene entidades de información que están relacionadas vía organización y asociación. La arquitectura lógica de una base de datos se define mediante un esquema que representa las definiciones de las relaciones entre las entidades de información. La arquitectura física de una base de datos depende de la configuración del hardware residente. Sin embargo, tanto el esquema (descripción lógica) como la organización (descripción física) deben adecuarse para satisfacer los requerimientos funcionales y de comportamiento para el acceso al análisis y creación de informes.

ESTRATEGIAS PARA EL ANÁLISIS DE DECISIONES

CONTENIDOS

- | | |
|---|--|
| 6.22. Tablas de decisión | 6.36. Conceptos básicos sobre decisiones |
| 6.23. Características de las tablas de decisión | 6.37. Acciones |
| 6.24. Como construir tablas de decisión | 6.38. Árboles de decisión. |
| 6.25. Verificación de tablas | 6.39. Características de los árboles de decisión |
| 6.26. Tipos de entradas en tabla | 6.40. Uso de árboles de decisión |
| 6.27. Tablas múltiples | 6.41. Identificación de los requerimientos de datos |
| 6.28. Procesadores de tabla de dedición | 6.42. Como evitar los problemas que se generan al utilizar árboles de decisión |
| 6.29. Español estructurado | |
| 6.30. Desarrollo de declaraciones estructuradas | |
| 6.31. <i>Estructuras de secuencia.</i> | |
| 6.32. <i>Estructuras de decisión.</i> | |
| 6.33. <i>Estructuras de iteración:</i> | |
| 6.34. Herramientas para documentar procedimientos de decisiones | |
| 6.35. Herramientas | |

6.1. TABLAS DE DECISION

Más que un árbol, una tabla de decisión es una matriz de renglones y columnas que indican condiciones y acciones. Las reglas de decisión, incluidas en una tabla de decisión, establecen el procedimiento a seguir cuando existan ciertas condiciones.

6.2. CARACTERISTICAS DE LS TABLAS DE DECISION

La tabla de decisión esta integrada por cuatro secciones:

1. *Identificación de condiciones.* Señala aquellas que son relevantes.
2. *Entrada de condiciones.* Indican que valor, si es que lo hay, se debe asociar para una determinada condición.
3. *Identificación de acciones.* Enlista el conjunto de todos los pasos que deben seguir cuando se presenta cierta condición.
4. *Las entradas de acción.* Muestran las acciones específicas del conjunto que deben emprenderse cuando ciertas condiciones o combinaciones de éstas son verdaderas.

Las columnas del lado derecho de la tabla enlazan condiciones y acciones, forman reglas de decisión que establecen la condiciones que deben satisfacer para emprender un determinad conjunto de acciones.

CONDICION	REGLAS DE DECISION
Identificación de condiciones	Entradas de acciones
Identificación de acciones	Entradas de condiciones

6.3. COMO CONSTRUIR TABLAS DE DECISIÓN

Para desarrollar tablas de decisión, los analistas deben seguir los siguientes pasos:

1. Determinar los factores considerados como más relevantes en la toma de decisiones. Esto permite identificar las condiciones en la decisión. Cada condición seleccionada debe tener la característica de ocurrir o no ocurrir.
2. Determinar los pasos o actividades más factibles bajo condiciones que cambian. Esto permite identificar acciones.
3. Estudiar las diferentes posibilidades de combinaciones de condiciones. Para cualquier número N de condiciones, existen 2^N combinaciones a considerar.
4. Llenar la tabla con la reglas de decisión. Primero es llenar los renglones de condiciones con valores si o no para cada combinación posible de condiciones, esto es llenar la primera mitad de renglones con *si* y la segunda con *no*. el siguiente renglón se llena alternando con S y N cada renglón.

6.4. VERIFICACION DE TABLAS

- Eliminación de la redundancia. Las tablas de decisión pueden volverse muy grandes y difíciles de manejar si se permite que crezcan sin ningún control. Remover las entradas redundantes puede ser de ayuda para manejar el tamaño de la tabla. La redundancia se presenta cuando las siguientes condiciones son verdaderas al mismo tiempo 1) dos reglas de decisión son idénticas salvo para una condición del renglón, 2) las acciones para las dos reglas son idénticas.
- Supresión de contradicciones. Las reglas de decisión son contradictorias entre si cuando dos o más reglas tienen el mismo conjunto de contradicciones pero sus acciones son diferentes.

En la siguiente tabla existe una contradicción entre las reglas de decisión 5 y 7, las dos tienen los mismos valores para las condiciones pero sus acciones son diferentes el analista se equivocó al hacer la tabla o los datos que le proporcionaron las diferentes personas de la organización están mal.

CONDICION

[illegible]

80

CONDICION	REGLAS DE DECISION				
	1	2	3	4	5
Suficiente efectivo	Y	-	y	n	n
Crédito bueno	Y	y	n	n	n
Desea “hacerse a un lado”	-	n	y	y	n
Seleccionar el articulo a comprar	x	x	x	x	
No seleccionar ningún articulo					x

6.5. TIPOS DE ENTRADAS EN TABLA

- 1.- *Forma de entrada limitada.* En la estructura básica de la tabla se dice que es limitada porque se llena con un si o un no. este es uno de los formatos más comunes, ejemplo:
- 2.- *Forma de entrada extendida.* Esta forma reemplaza las S y N con acciones que le indican al lector cómo decidir. En este formato, los identificadores de condición y acción no están complejos y es la razón por la que las entradas contienen más detalle que una S y N. la forma de entrada extendida tiene sólo una identificación de acción: *ACCION*. Para cada regla, se coloca una frase breve en la sección de identificación de acciones: descontar 3%, descontar 2%, pagar el monto total de la factura. Muchos analistas emplean este formato sobre el método de entradas limitadas porque es más explícito para señalar las acciones, ejemplo:
- 3.- *Forma de entrada mixta.* En ocasiones los analistas prefieren combinar en la misma tabla las características de los dos métodos anteriores.
- 4.- *Forma ELSE.* Esta es otra variante en las tablas de decisión que tiene como finalidad omitir la repetición por medio de reglas *ELSE*. Para construir una tabla de decisión en la forma *ELSE*, se especifican las reglas, junto con las entradas de decisiones, que cubren todo el conjunto de acciones con excepción de una que se convierte en la regla a seguir cuando ninguna de las demás condiciones explícitas es verdadera. Esta regla se encuentra en la columna *ELSE*. Si ninguna de las otras condiciones es valida, entonces se sigue la regla de decisión *ELSE*, esta regla elimina la necesidad de repetir condiciones que conducen a las mismas acciones.

6.6. TABLAS MULTIPLES

La forma *ELSE* es una alternativa para controlar el tamaño de las tablas de decisión. Otra manera de alcanzar este mismo objetivo es enlazando varias tablas de decisión. De acuerdo con las acciones seleccionadas en la primera tabla, otras se explican en una o más tablas adicionales, cada tabla proporciona mayores detalles relacionados

con las acciones a emprender, por otro lado las tablas de múltiples permiten al analista establecer las acciones repetitivas que deben realizarse después de tomar la decisiones y que continúan hasta que se alcanza determinada condición.

Para utilizar este método los analistas construyen, por separado, tablas de decisión que satisfacen todos los requerimientos normales y que están relacionados con una decisión específica. Las tablas se enlazan en forma jerárquica: una tabla de nivel alto contiene las condiciones principales que, cuando son seleccionadas, determinan las tablas y acciones adicionales donde se encuentran otros detalles. Una declaración de transferencia como GO TO o PERFORM.

Existen dos tipos de transferencias:

- *Transferencia directa.* La directa se emplea una sola vez; la tabla que es seleccionada de esta manera no vuelve a referirse a la tabla original. La proposición "GO TO" indica cual es la siguiente tabla que se va a examinar.
- *Transferencia temporal.* En contraste con la anterior, la tabla 1, se enlaza con la tabla 2 por medio de la proposición *PERFORM* tabla 2, al final de la tabla 2 la proposición *RETURN* regresa de nuevo el control a la proposición que sigue al GO TO en la tabla.

6.7. PROCESADORES DE TABLA DE DECISION

Las tablas de decisión han sido parcialmente automatizadas. Los procesadores de tablas de decisión son programas para computadoras que manejan la formulación actual de una tabla con base a la información de entrada proporcionada por el analista. Estos procesadores también emprenden todas las verificaciones necesarias para detectar inconsistencias y redundancias.

La utilidad de los procesadores de tablas de decisiones radica en el ahorro d tiempo de programación y detección de errores.

6.8. ESPAÑOL ESTRUCTURADO

Es otro método para evitar los problemas de ambigüedad del lenguaje al establecer condiciones y acciones, tanto en procedimientos como en decisiones. Este no hace uso de árboles o tablas; en su lugar utiliza declaraciones para describir el proceso.

El método no muestra las reglas de decisión; las declara.

En el español estructurado requieren que el analista identifique las condiciones que se pregunta en un proceso y las condiciones que se deben tomar cuando esto suceda, junto con las decisiones que se deben tomar cuando esto suceda, junto con las acciones correspondientes. Este método permite hacer una lista de todos los pasos en el orden en que se llevan a cabo.

6.9. DESARROLLO DE DECLARACIONES ESTRUCTURADAS

Emplea tres tipos básicos de declaraciones para describir un proceso: estructuras de secuencias, *ESTRUCTURAS DE DECISIÓN* y *ESTRUCTURA DE ITERACIÓN*. Estas estructuras son adecuadas para el análisis de decisión y pueden trasladarse al desarrollo de software y programación.

6.10. *ESTRUCTURAS DE SECUENCIA*: una estructura de secuencia es un solo paso o acción incluido en un proceso. Este no depende de la existencia de ninguna condición y, cuando se encuentra, siempre se lleva a cabo. En general, se emplean varias instrucciones en secuencia para describir un proceso.

Por ejemplo, es probable que la compra de un libro siga un procedimiento similar al siguiente:

1. Escoger el libro deseado.
2. Llevar el libro al mostrador de salida.
3. Pagar el libro.
4. Obtener el recibo.
5. Abandonar la librería.

Este ejemplo muestra una secuencia de 5 pasos. Ninguno contiene alguna decisión o condición para determinar la realización del siguiente paso.

6.11. *ESTRUCTURAS DE DECISIÓN*: El español estructurado es otro camino para mostrar el análisis de decisión. A menudo se incluyen las secuencias de acciones entre estructuras de decisión que sirven para identificar condiciones. Es así como las estructuras de decisión aparecen cuando se pueden emprender dos o más acciones, lo que depende del valor de una condición específica. Primero se evalúa la condición y después se toma la decisión de emprender las acciones asociadas con esta condición. Una vez determinada la condición las acciones son incondicionales.

Estas condiciones junto con las acciones pueden indicarse de la siguiente manera:

Si se encuentra el libro deseado ENTONCES.

Llevar el libro al mostrador de salida para libro.

Pagar el libro

Asegurarse de obtener el recibo de compras

Abandonar la librería

DE OTRO MODO

No llevar los libros al mostrador

Abandonar la librería

La estructura de decisión que emplea las frases SI / ENTONCES / DE OTRO MODO, señala con bastante claridad las alternativas de proceso de decisión. En este caso se indican dos condiciones y dos acciones.

Las estructuras de decisión no esta limitada a pares de combinaciones condición – acción. Pueden existir muchas condiciones.

6.12. *ESTRUCTURAS DE ITERACIÓN*: En las actividades rutinarias de operación, es común encontrar que algunas de ellas se repiten mientras existan ciertas condiciones o hasta que estas se representan. Las instrucciones de iteración permiten al analista describir estos casos.

La búsqueda de un libro en la librería puede realizarse repitiendo los siguientes pasos:

EJECUTAR MIENTRAS se examinan más libros:

Leer el título del libro

Si el título suena interesante

ENTONCES tomar el libro y hojearlo

Buscar el precio

Si la decisión es llevar el libro

Colocarlo en la filas de LIBROS PARA LLEVAR

OTRO regresar el libro al instante

FIN DE SI

OTRO continuar

FIN DE EJECUTAR

Si se encuentra en los libros deseados ENTONCES

Llevar los libros al mostrador de salida

Pagar los libros

Asegurarse de obtener el recibo

Abandonar la librería

OTRO

No llevar al libro al mostrador de salida

Abandonar la librería

FIN DE SI

El español estructurado puede ser de utilidad para describir con claridad condiciones y acciones.

6.13. HERRAMIENTAS PARA DOCUMENTAR PROCEDIMIENTOS DE DECISIONES

Seguir procedimientos y tomar decisiones son aspectos importantes de cualquier empresa. Algunas como, aceptar o no ofertas afectan a todas las organizaciones, otras como saber cuando pedir materiales del almacén. Dependen de pocas personas y sigue los procedimientos pasos por paso. Sin embargo las decisiones y procedimientos son de importancia para el analista cuando este conduce una investigación de sistemas dentro de la empresa.

6.14. HERRAMIENTAS

Es cualquier dispositivo, objetivo, objetos u operación utilizada para ejecutar una tarea específica. El analista de sistema depende de las herramientas para realizar su trabajo de la misma manera que otras personas de sus actividades cotidianas.

Las herramientas ayudan al analista a recopilar los datos por los diversos métodos en la sección anterior.

6.15. CONCEPTOS BÁSICOS SOBRE DECISIONES

Condiciones y variables de decisión.

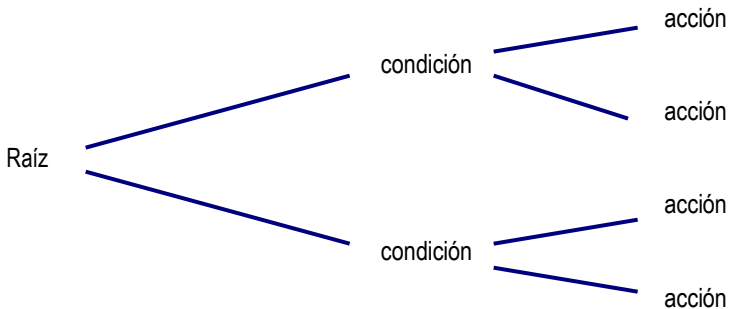
Cuando se observa un sistema y se pregunta ¿Cuáles son las posibilidades ó que puede suceder? En realidad se esta preparando por las condiciones, que son los posibles estados (personas, lugar, objeto o eventos) las variables de cabían y es por esto que el analista se refiere a ellas como variables de decisiones.

6.16. ACCIONES

Son alternativas, pasos, actividades o procedimientos que deben emprenderse cuando se toman una decisión específica.

6.17. ÁRBOLES DE DECISION.

Son uno de los tres métodos que se emplean para describir decisiones y que evita dificultades en la comunicación.



6.18. CARACTERISTICAS DE LOS ÁRBOLES DE DECISION

El árbol de decisión es un diagrama que representa en forma secuencial condiciones y acciones.

La raíz del árbol es decir, la secuencia de decisión es de izquierda a la derecha.

6.19. USO DE ÁRBOLES DE DECISIÓN

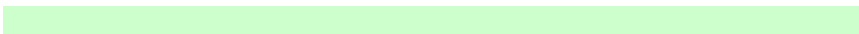
El desarrollo de árboles de decisión beneficia al analista, primero que todo, la necesidad de describir condiciones y acciones lleven a los analistas a identificar de manera formal las decisiones que actualmente deben tomarse. De esta forma es difícil para ellos, pasar por alto cualquier etapa del proceso de decisión.

6.20. IDENTIFICACIÓN DE LOS REQUERIMIENTOS DE DATOS

Se ha demostrado que los árboles de decisión son eficaces cuando es necesario describir problemas con más de una dimensión o condición. Sin embargo también son útiles para identificar los requisitos de datos críticos que rodean al proceso de decisión; es decir, los árboles indican los conjuntos de datos que la gerencia requiere para formular decisiones o tomar acciones

6.21. COMO EVITAR LOS PROBLEMAS QUE SE GENERAN AL UTILIZAR ÁRBOLES DE DECISIÓN

Los árboles de decisión no siempre son una buena herramienta, que sí tiene varias ramas ocasiona problemas al analista; lo cual corre el riesgo de no determinar que política o estrategia de la empresa es la específica.



DESARROLLO DE LA PROPUESTA DE SISTEMA

CONTENIDOS

- 7.18. Introducción
- 7.19. Evaluación del equipo y del software
- 7.20. Identificación de costos y beneficios
- 7.21. Análisis de punto de equilibrio
- 7.22. Retorno de la inversión
- 7.23. Análisis del flujo efectivo
- 7.24. Valor presente
- 7.25. Lineamientos para el uso de los métodos
- 7.26. El valor presente
- 7.27. Redacción y presentación de la Propuesta
- 7.28. Recomendaciones del Analista de Sistema
- 7.29. Elección de un estilo de redacción
- 7.30. Uso eficaz de las tablas
- 7.31. Uso eficaz de las gráficas
- 7.32. Adopción de un estilo único para la propuesta
- 7.33. Presentación de la propuesta de sistema
- 7.34. Organización de la propuesta de sistema

7.1. Introducción

La propuesta de sistema es un resumen de lo que se sugiere para mejorar su desempeño.- Para abordar de manera adecuada la necesidad de información:

Método sistemático para la adquisición de Hard y Soft.

Identifica Costo/beneficio futuros

Análisis de costos.

7.2. Evaluación del Equipo y del Software

Esta actividad se compone de cuatro etapas:

- a) Inventario del equipo de cómputo la relación siguiente nos muestra un ejemplo de la forma de inventario que puede presentarse a algunas personas.

Algunas alternativas son la expansión o la resignación del equipo actual, de tal forma que es importante conocer lo que se tiene a mano.(básicamente en el inventario se desea conocer):

- El tipo de equipo, numero de modelo y fabricante.
- El estado de la operación del equipo (por instalar, en operación, en almacenamiento y con necesidad de reparación).
- La estimación del tiempo de uso del equipo.
- La vida proyectada del equipo.
- La localización física del equipo.
- La persona, departamento responsable del equipo.
- Estado financiero del equipo: propio, rentado, o arrendamiento con opción a compra.

- b) Estimación de la carga de trabajo Estimar la carga actual y proyectada, de manera que cualquier equipo que se adquiera cuente con la posibilidad de manejar las cargas de trabajo actuales y futuras. Si las estimaciones son correctas no habrá necesidad del cambio de equipo.

- c) Evaluación del Hardware: esta responsabilidad es compartida entre la dirección, los usuarios y los analistas.

Para la evaluación se debe considerar: el tiempo requerido para las operaciones típicas, la capacidad total del sistema, los tiempos muertos de CPU y el tamaño de la memoria.

Dentro de esta etapa se evaluará:

- Tamaño y uso de computadoras

- Dispositivos de almacenamientos
 - Adquisición del Equipo
 - Renta,
 - Comprar
 - Renta con opción a compra.
- d) La evaluación del Software. los paquetes de Software se han vuelto muy accesibles. Es una alternativa valida, ahorran muchas horas de programación. Deben ser evaluados junto con el usuario.-

Existen seis categorías principales para ubicar a un Soft.-

- 1.- Eficacia en el desempeño
- 2.- Eficiencia operativa
- 3.- Facilidad del uso
- 4.- Flexibilidad
- 5.- Calidad de documentación
- 6.- Soporte del fabricante

7.3. Identificación de costos y beneficios

Deben considerarse en conjunto. antes de presentar la propuesta se debe predecir ciertas variables, para ello cuenta con modelos de pronósticos:

Estos costos y beneficios también pueden ser de naturaleza tangible e intangible.

Beneficios tangibles: ventajas económicas cuantificables que obtiene la Organización a través del uso del Sistema de Información (Ej.: Velocidad de proceso).

Beneficios intangibles: no se pueden cuantificar. También son importantes.- (Ej.: mejora el proceso de toma de decisión).

Costos tangibles: son aquellos que se proyectan con precisión por el analista y el personal de contabilidad (Ej.: costo del equipo).

Costo intangible: son difíciles de estimar y pueden no conocerse.- (Ej.: pérdida de prestigio de la Empresa, debido a descuidos).-

Comparaciones de los costos y los beneficios

Existen técnicas para la comparación de costos y los beneficios propuestos.-

- Análisis de punto de equilibrio
- El retorno a la inversión

- EL análisis del flujo efectivo
- El valor presente

7.4. Análisis de punto de equilibrio

Compara exclusivamente costos, permite determinar el momento en el cual se alcanza el equilibrio del sistema de información propuesto.- El punto en el cual se interceptan los costos total del sistema actual y del sistema propuesto representa el punto de equilibrio a partir del cual la empresa obtendrá utilidades si contara con el nuevo sistema.- Se incluyen únicamente costo tangibles. Es útil cuando una empresa esta creciendo y el costo asociado al volumen se vuelve de gran relevancia en el costo global de la operación. La desventaja es que supone que los beneficios permanecen constantes. (y esto no refleja la realidad)

7.5. Retorno de la inversión

Es una manera sencilla de establecer si una empresa debe invertir en un Sistema de información, con base en cuanto tiempo requiera obtener beneficios del sistema que amorticen el costo de su desarrollo. A manera de resumen, el método de retorno de la inversión determina el número de años de operación del sistema de información que se requieren para comenzar el costo de su inversión.

Los inconvenientes que limitan su utilidad son:

- a) No deja de ser un enfoque a corto plazo de decisiones de inversión y reemplazo.
- b) No le da importancia a la forma que serán programados los ingresos.-
- c) No considera el retorno total del proyecto, que pudiera ir más allá del año de amortización.

7.6. Análisis del flujo efectivo

Examina la dirección, la magnitud y el patrón del flujo efectivo, que se asocia con el sistema de información propuesto.

Se utiliza para determinar cuando una compañía empezara a obtener utilidades y cuando pasara de los números rojos, esto se lograra cuando la utilidad supere la inversión inicial.

7.7. Valor presente

Auxilia al analista para presentarle a los tomadores de decisiones de la empresa el valor de la inversión en el tiempo, así como el flujo de efectivo asociado con el sistema.- Es la manera de valorar todos los recursos económicos gasta dos y generados a lo largo de la vida útil del sistema y de comparar los costos actuales en el futuro y los beneficios actuales en el futuro.

7.8. Lineamientos para el uso de los métodos

Use punto de equilibrio si el proyecto necesita justificarse con base a su costo y no en sus beneficios.

Use retorno a la inversión cuando los beneficios tangibles mejoren y sean un argumento convincente para el sistema propuesto.-

Use análisis de flujo efectivo cuando el proyecto sea costoso en relación con el tamaño de la empresa o cuando el negocio se vea afectado de manera significativa por una gran sangría de fondos.- - Use el valor presente cuando el periodo de inversión sea largo o cuando el costo del financiamiento sea elevado.

7.9. El Valor Presente

El análisis del valor presente auxilia al analista para presentarle a los tomadores de decisiones el valor de la inversión en el tiempo, así como el flujo efectivo asociado con el sistema de información. El valor presente es la manera de valorar todos los recursos económicos gastados y generados a lo largo de la vida útil del sistema y de comparar los costos actuales con los futuros, y los beneficios actuales con los futuros.-

7.10. Redacción y presentación de la Propuesta

La propuesta escrita es el resumen del trabajo del analista y esta propuesta puede ser eficaz se usa algunas de los siguientes métodos:

- La organización eficaz del contenido
- Un estilo profesional de redacción
- Presentación oral de la propuesta
- Organización eficaz del contenido

Una vez recopilado toda la información, necesita integrarla, esto implica incluir diez elementos fundamentales, un estilo eficaz de redacción, el uso complementario de figuras.

Que incluir en la propuesta de sistema:

1. Carta de presentación: Para la Dirección y para el grupo de trabajo del MIS, de incluir: quienes participaron en el estudio y un resumen de los objetivos, puede incluir también la fecha y hora de la presentación oral de la propuesta. Debe ser concisa.
2. Página de título: en esta página incluya: nombre del proyecto, nombre de los miembros del grupo y la fecha que se presenta la propuesta. Debe ser clara.- (tiene mas de arte que de ciencia).
3. Índice de contenido: Es de utilidad para propuestas largas. En propuestas cortas es aconsejable omitir el índice. Debe ser conciso y debe ser preparado durante el desarrollo de Sistema.

4. Resumen Ejecutivo: Destinado a jerárquicos con poco tiempo. Debe incluir una descripción somera de las actividades. Quien, cuando, cual, donde y el como de la propuesta.- (Ej.: como ocurre en la 1ra pagina del periódico).
5. Descripción del estudio de sistemas. Esta sección proporciona información acerca de todos los métodos utilizados en el estudio y quien o que fue objeto de estudio.
6. Resultados detallados del estudio de sistema. Esta sección detalla lo que el analista ha encontrado acerca del sistema a través de los métodos descritos en la sección anterior.

Incluye conclusión acerca de problemas, tipos y frecuencias de errores, las cargas de trabajos actuales y proyectados y la manera en que las enfrenta el sistema actual. Se incluyen problemas

7. Alternativas del Sistema. En esta parte del sistema el analista presenta dos o tres soluciones alternativas que se dirigen de manera directa a los problemas anteriormente señalados.

Cada una de las alternativas deben analizarse por separado. Describa los costos y beneficios de cada situación. Incluya gráficas de comparación de alternativas.

8. Recomendaciones del analista.- Una vez que se han ponderado las alternativas, el analista tendrá una opinión definida acerca de cual sería la solución mas adecuada.

En esta sección se presenta la solución recomendada. Incluye motivos que apoyan a la recomendación.

9. Resumen de la propuesta.- Es un planteamiento breve que refleja lo contenido en el resumen ejecutivo. Permite que el analista refleje una vez mas la relevancia del Proyecto.

10. Apéndices.- Es la ultima parte, puede incluir cualquier información que considere de interés para individuos específicos, pero sin ser indispensable para la comprensión del estudio.

Aunque tiene que ser completo, no lo sature con información irrelevante.-

Una vez que redacte la propuesta, seleccione a quien enviara las copias (No todos recibirán una copia), entregue personalmente a quien haya seleccionado, esto puede causar una buena impresión, al contagiar su entusiasmo e involucrarlos en el Proyecto.

7.11.Recomendaciones del Analista de Sistema

Una vez que se ha ponderado las alternativas se tendrá una opinión bien definida de cual sería la solución mas adecuada.-

En esta sección se presenta la solución recomendada.

Resumen de la Propuesta: Es un planteamiento breve, de los objetivos del estudio y las soluciones recomendadas. Se enfatiza la relevancia del proyecto y su factibilidad.

Apéndices: Puede incluir cualquier información que se considere de interés para individuos específicos, pero sin ser indispensable para la comprensión del estudio de sistemas y de lo que se propone.-

Puede incluir: resumen de las etapas concluidas, gráficos con mayor detalle, correspondencia pertinente.

7.12. Elección de un estilo de redacción

El más adecuado es un estilo de redacción comercial, pero si a la audiencia a quien se dirige favorece cierto estilo, no deje de utilizarlo.-

Mantenga las referencias en un mínimo y no utilice pie de página. Cuando sea necesario haga uso de ejemplos, ilustraciones, diagramas, tablas, figuras, gráficos, etc.

El uso de figuras para una comunicación eficaz: la integración de figuras es un indicio de que esta al tanto de las diferentes maneras en que la gente asimila la información.

Las figuras complementan la información escrita y siempre se deben interpretar con palabras, nunca se deben presentar solas.

7.13. Uso eficaz de las Tablas

No son consideradas como ayuda visuales, pero permiten agrupar y presentar los datos que el analista desea comunicar al lector en forma diferente.-

Las tablas deben contar con encabezados de columnas y renglones, título y deben enumerarse. Incluya una sola tabla por página y de ser posible ajústela verticalmente. Utilice marcos o recuadros.

7.14. Uso eficaz de las gráficas

Las gráficas de líneas, de columnas y de barras comparan variables, mientras que las circulares ilustran la integración porcentual de una entidad.- Lineamientos a seguir: Dibuje una sola grafica por página, a menos que haga una comparación crítica entre los gráficos.

Integre los gráficos en el cuerpo de la propuesta. Enumérelos y póngale títulos.

Identifique cada columna cada eje, barra o segmento. Incluya referencias de los distintos tipos de barras, columnas y/o líneas.

Uso eficaz de gráficas y diagramas de programación

No deje de incluir las gráficas de Gantt en su propuesta. Estas gráficas muestran cuando concluirá el Proyecto y el tiempo requerido para cada actividad.

Los diagramas de PERT son otro de tipo de gráficas para audiencia más técnica, si tiene que incluirlas, hágalo.

7.15. Adopción de un estilo único para la propuesta

Consideraciones visuales y de formato. La propuesta son documentos persuasivos, en consecuencia la propuesta debe ser persuasiva desde el punto de vista visual.

Lineamientos: Uso de espacios en blancos: los espacios en blanco distribuidos a lo largo del texto permiten destacar las ideas, asegurando que no pasaran desapercibidas.

Deje márgenes superior, inferior, izquierda y derecha de cada página. Si el documento se va a encuadernar deje más margen izquierdo.

Escriba a doble espacio en todo el documento.

Uso de títulos y subtítulos. Es decisivo su uso, si la propuesta es larga. Los títulos separan las distintas secciones y ubican al lector en el contexto principal. Si los títulos se redactan con imaginación permiten que el lector siga la lógica de la redacción y no pierda el interés. Destaque los títulos y subtítulos, use letras mayúsculas, minúsculas, negritas, subrayadas, sangrías, cualquier combinación, etc. Ubíquelos en distintos lugares dentro del texto.

Numeración de las páginas. Es importante numerar las páginas, es la forma más rápida de ubicación. Los números puede ubicarlos en: la esquina superior derecha, de manera alternada en las esquinas izquierda y derecha, si es que el material se imprime por ambos lados, al pie o la parte superior de la página.-

Relacione el número de página de cada título con la tabla de contenido.

Referencias y apéndices. La mayoría de los lectores aprecian que las referencias de material externo sean mínimas, si es necesario incluya al final de su informe las referencias a doble espacio y con formato consistente.

Las referencias incluyen: especificaciones técnicas, costo y tiempo involucrados.

7.16. Presentación de la propuesta de Sistema

Las siguientes consideraciones incluyen quienes deben conformar la audiencia, como organizar, apoyar y realizar la presentación oral.

Compresión de la audiencia: Es necesario conocer la audiencia a fin de determinar que tan formal ser, que presentar, que tipo de ayudas visuales debe incluir, en fin, es imperativo conocer a quien se dirigirá la presentación oral.

A continuación se presentan diferentes tipos de audiencias y la acción correspondiente.

Básicamente los tipos de audiencias son:

- a) ejecutivos
- b) usuarios principales

Ejecutivos: Alta dirección, tomadores de decisiones. No es necesario en abundar en detalles técnicos, si es necesario la información costo / beneficio.

Pueden ser útiles algunos ejemplos o ilustraciones. Use estadísticas contenidas en ayudas visuales (gráficas o diagramas).- Sea respetuoso del tiempo (es el recurso mas valioso de los tomadores de decisiones). Ensaye para controlar su duración. Deje tiempo abundante para preguntas y comentarios. No este a la defensiva.

Usuarios principales: Difieren de los ejecutivos en usaran la salida del Sistema.

Debe discutir los cambios que ocurrirán, detalles operativos.- Utilice ejemplos e ilustraciones, comentarios sobre las bondades del sistema nuevo, la relación costo/beneficio también puede servir, pero no es el aspecto principal a cubrir.-

Utilice ayudas visuales de la ayuda o cualquier ayuda de que permita al usuario concebir el uso de Sistemas (Prototipos). -

7.17. Organización de la Propuesta de Sistema:

De todos los datos reunidos en la propuesta, tome 4 o 5 principales (resumen ejecutivo, recomendaciones resumen de la propuesta), si el tiempo de presentación es mayor a media hora, puede llegar a tomar hasta 9 puntos, sin embargo en la presentación oral, el asistente no llega a retener mas de cuatro puntos. Cada uno de los puntos debe contar con algún tipo de soporte.

Planeación de la Introducción y la conclusión

Una vez que se ha trabajado en los puntos principales puede redactarse una introducción y una conclusión. Tal presentación prepara a la audiencia para escuchar que es lo que va a oír.

La introducción debe incluir un aliciente, algo que mantenga intrigado a la audiencia respecto a lo que vendar después. Tiene que tener un enfoque creativo:

Ej. Un poema, una anécdota, analogía, una broma.

La duración de la introducción debe ser proporcional al tiempo disponible (entre el 5 y el 10 % del tiempo total).

Las conclusiones deben reflejarse en la introducción, por lo tanto deben redactarse juntas.- Deben revisar los puntos principales de la propuesta. No debe repetir las ideas principales expuestas en la introducción sino que debe reiterarlas.

Preguntas del momento

Las preguntas pueden hacerse durante y después de la presentación. La contestación de las preguntas durante la `presentación, hace mas informal y relajada la presentación. Sin embargo, puede desviar la atención. Es conveniente dejar las preguntas para el final de la presentación, con el fin de mantener el control y comunicar los puntos de vista.

Uso de ayudas visuales

Los elementos visuales deben tener una apariencia profesional. Los elementos impresos deben ser uniformes y visibles de cualquier sitio del auditorio. Deben prepararse de antemano. Utilice paquetes de Soft. Practique con antelación a la presentación a fin de evitar inconvenientes que puedan surgir. Asegúrese que el equipo este disponible para la presentación. Cuando exponga no deje de mirar a la audiencia en lugar de mirar los elementos que este utilizando.

Uso de computadora personal en la presentación

Uno de los elementos más eficaz es una pantalla de video conectada a una PC portátil.

Si lleva una PC a su presentación deberá considerar:

- a) Si es posible trate de utilizar un sistema de proyecciones de pantalla gigante, si esto no fuera posible, utilice varias computadoras.
- b) Utilice planillas de calculo, permite que el tomador de decisiones considere numerosos escenarios de simulación durante la presentación.
- c) Si su sistema involucra computadoras, en este caso actúan como prototipos, y la audiencia contara con una idea clara y concreta de lo que será el sistema resultante.
- d) Deje que la audiencia opere la computadora.-
- e) No haga la presentación solo en la PC, e incluso si la utilizara, deberá explicar que es lo que realizara
- f) Planee alternativas por si tiene algún tipo de inconveniente en la utilización del equipo.

7.18. Principios de exposición

Si la audiencia esta integrada por Jerárquicos, la exposición deberá ser formal, en otro caso, tal vez convenga una presentación del tipo taller.

Los lineamientos a seguir son:

- 1. Mantenga un nivel de voz suficientemente fuerte
- 2. Mire a cada una de las personas, conforme hable.
- 3. Elabore ayudas visuales, lo bastante grande para que toda la audiencia la pueda ver.
- 4. Use gestos que sean naturales a su estilo de conversación.
- 5. Inicie y concluya su plática de manera segura.

Para vencer la ansiedad existen cuatro lineamientos:

- a) Sea usted mismo; la personalidad del expositor es muy importante para persuadir a la audiencia. Debe desarrollarse intelectual, emocional y espiritualmente.

- b) Prepárese; cuando más sepa de su tema, más fácil le será exponerlo. Esto significa que deberá prepararse de antemano en la exposición. Son muy pocos los que tienen el don de improvisar un buen discurso.
- c) Hable de manera natural; no memorice ni lea, parecería poco ético.- La lectura evita la oportunidad del contacto visual. Memorizar una plática disminuye la posibilidad de adaptarse de manera adecuada a una audiencia en particular.

Confíe en que tiene un mensaje que comunicar y que sabe cual es.- En liste los puntos principales, indique el momento de presentar algún tipo de elemento que utilice.

- d) Recuerde respirar; esto suena curioso, ya que respiramos sin darnos cuenta. Sin embargo, una excelente forma de alcanzar un adecuado volumen de voz es acordarse del proceso de respiración. Aspire profunda y largamente, justo ante de dirigirse al grupo. Permítase respirar entre frases y durante las pausas normales de su exposición. Recuerde que fisiológicamente es imposible respirar con profundidad y seguir nervioso.

CASO DE EJEMPLO SENCILLO DE ANÁLISIS DE SISTEMAS

CONTENIDOS

Descripción de la organización
Empresa:
División Funcional:
Organigrama de la empresa
Funcionamiento de la empresa:
Selección del sub-sistema:
Sistema de ventas
Análisis de procesos
Diagrama de contexto (nivel 0)
Diagrama intermedio (nivel 1)
Diagrama detalle (nivel 2)
DFD de Detalle 1: Registrar Pedidos

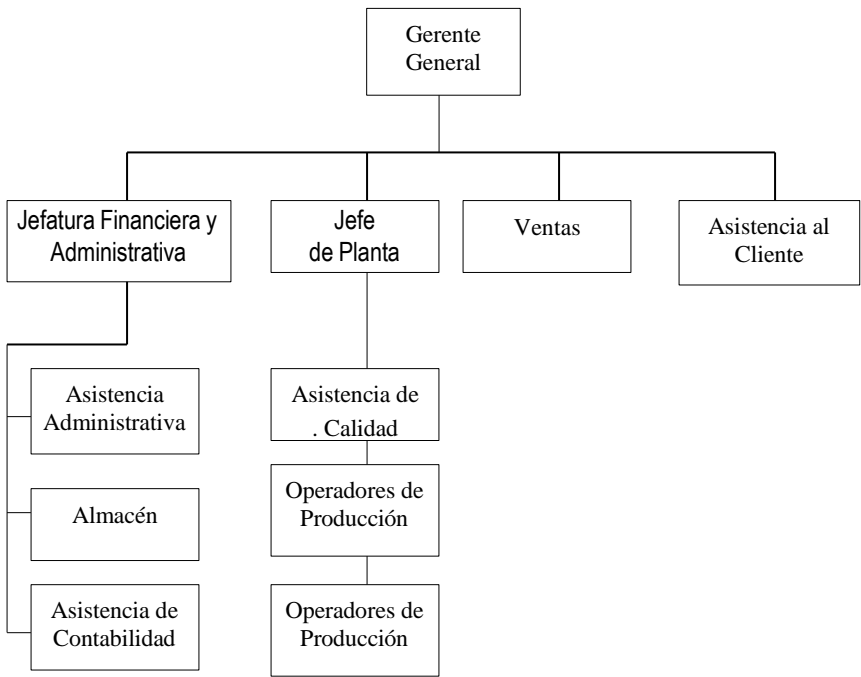
DFD de Detalle 3: Almacenar:
DFD de Detalle 4: Asistir al Cliente
Análisis de datos
Entidades
Diagrama entidad-relación del sistema de
ventas
Diccionario de datos
Análisis de procesos del Diagrama Nivel 1
(Intermedio)
Análisis de procesos del Diagrama Nivel 2
(Detalles)
Análisis de procesos: entidad-relación

DESCRIPCIÓN DE LA ORGANIZACIÓN

Empresa:
ENORSA - Envases del Norte S.A
Ubicada en la Carretera Panamericana Norte Km. 560, en la provincia de Trujillo.
Formada en 1998. De capitales extranjeros. Su casa matriz se encuentra en Ecuador, en donde cubre un 80% del mercado ecuatoriano. El mercado que abarca en la localidad es de aproximadamente del 20%.
Su asesoría técnica es ecuatoriana.

División Funcional:
Su actividad principal es la fabricación de envases de hojalata para elementos de conserva; y en menos grado, la fabricación de tapas-coronas (chapas para envases de vidrio).

ORGANIGRAMA DE LA EMPRESA



Usuario Clave: Debido a que las políticas y normas que posee la empresa, el obtener información directa de esta es muy difícil. La realización del trabajo se limitó a entrevistas a operadores de planta.

El tamaño de la empresa no es muy grande, por lo cual los datos que se obtuvieron de las entrevistas a los operadores fueron suficiente para poder desarrollar el Análisis del funcionamiento del Sistema.

FUNCIONAMIENTO DE LA EMPRESA:

ENORSA se dedica primordialmente a la producción de Latas para conserva. Esta producción la realiza en función a los pedidos que realizan los clientes.

Los pedidos son recepcionados en su departamento de Ventas, el cual toma las características del pedido (pedidos especiales o regulares). La empresa, por política, verifica los pedidos con los clientes, para evitar posibles errores.

Una vez registrado el pedido, se dispone a determinar los requerimientos para iniciar la producción del pedido. Una vez determinados estos, conjuntamente con la Jefatura Administrativa y la Jefatura de Planta (Producción), se emite una solicitud de compra a los Proveedores. Los pedidos son de Láminas Revestidas (bultos de láminas).

La empresa debe manejar un tiempo de espera, hasta la llegada del pedido con los datos que dan los Proveedores. Así la empresa proyecta el tiempo que le tomará producir el pedido y entregarlo, cumpliendo en el compromiso adquirido con el cliente.

La dificultad que se presenta, es ante las demoras de los proveedores en entregar los insumos, ya que estos son importados. La empresa maneja un plan de contingencia, pero en ciertas oportunidades las demoras son difíciles de manejar.

Una vez que los Bultos de láminas llegan, se inicia con la producción de los pedidos. La empresa produce bajo un constante control de calidad, ya que siempre busca ampliar su lista de clientes (un mayor mercado), y para esto debe entregar un producto óptimo.

Una vez terminada la producción, estos se guarda en Almacén (lotes de producción). Dependiendo del tipo de pedido, la empresa realiza el embalaje de las latas o tapas coronas (primordialmente las primeras) con las características pedidas por el cliente.

Todos los procesos de producción, son supervisados por los respectivos Jefe de cada área. Estos realizan informes, los cuales son presentados a la Gerencia Administrativa, quien es el representante de la Casa Matriz de ENORSA. Esta, es la encargada de que todos los procesos se realizan cumpliendo las normas y políticas bajo las cuales funciona la empresa.

SELECCIÓN DEL SUB-SISTEMA:

SISTEMA DE VENTAS

Se eligió el sistema de Ventas por presentar un funcionamiento que podía mejorarse. La mejora propuesta sería que la empresa deje de usar proveedores como

intermediarios para abastecerse de su materia prima (láminas revestidas), y que implemente una nueva área encargada de las importaciones de insumos, pero de láminas sin revestimiento.

El proceso de revestido estaría a cargo de la misma empresa, por lo cual, ellos no tendrían demoras esperando la llegada de insumos, ya que dependerían de sí mismos. Esto implicaría un crecimiento físico de la empresa, y por tal un mejor servicio a los clientes, concluyendo en una mejora en las rentas de la empresa.

ANÁLISIS DE PROCESOS

El Análisis de Procesos se realizará implementando la mejora propuesta en los diagramas de Contexto, Intermedio y de Detalle.

DIAGRAMA DE CONTEXTO (NIVEL 0)

Ventas es el encargado de recepcionar los pedidos de los Clientes, así como de determinar cuales serán los requerimientos de producción para el pedido solicitado. La empresa, por política, verifica los pedidos con los clientes.

Toda operación, se realiza bajo las normas y políticas que dicta la Gerencia General, en cuanto a los montos, calidad, cantidades, etc.; las cuales requieren de informes y reportes, los cuales indiquen los movimientos. Para cumplir con los pedidos, emiten solicitudes de compra a su área de Importaciones.

Luego emite una orden de producción la cual es recepcionada por Planta. Una vez finalizada la producción, se recibe el producto final emitido por Planta, y se dispone a entregar el pedido.

Los reclamos por productos defectuosos, o que no cumplan con los pedidos específicos del cliente, también son tratados por ventas.

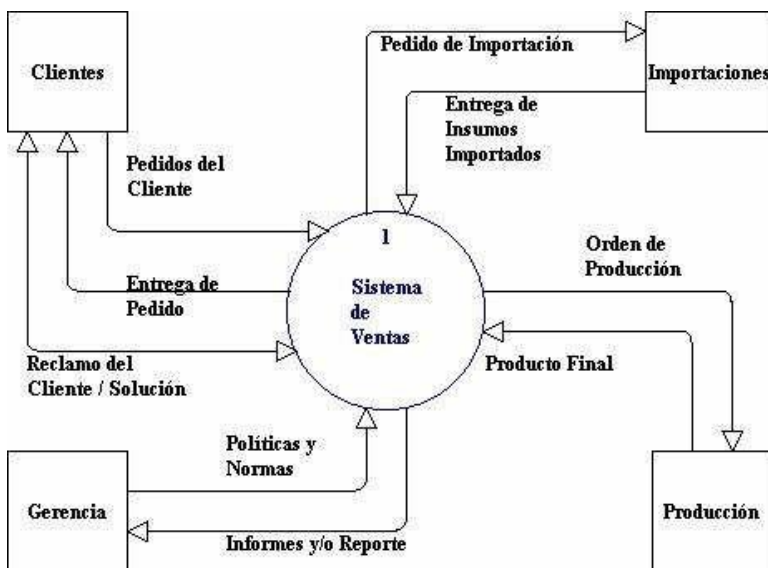


DIAGRAMA INTERMEDIO (NIVEL 1)

Los Pedidos de los clientes, son registrados y almacenados, para llevar cuentas de crecimiento de la empresa. Los detalles de los pedidos son enviados hacia la Asistencia Administrativa para que determine los requerimientos necesarios para cumplir con el pedido.

Asistencia Administrativa emiten los respectivos pedido de importación y recibe los insumos. Inmediatamente se envían los insumos a la Planta para iniciar la Producción, en el tiempo estimado. Finalizada la producción, se almacena el producto terminado en almacén hasta el momento de la entrega.

Asistencia al Cliente se encarga de entregar a los clientes sus respectivos pedidos, así como atender sus posibles reclamos.

Toda operación se realiza bajo las normas y políticas que dicta la Gerencia General, las cuales requieren de informes y reportes.

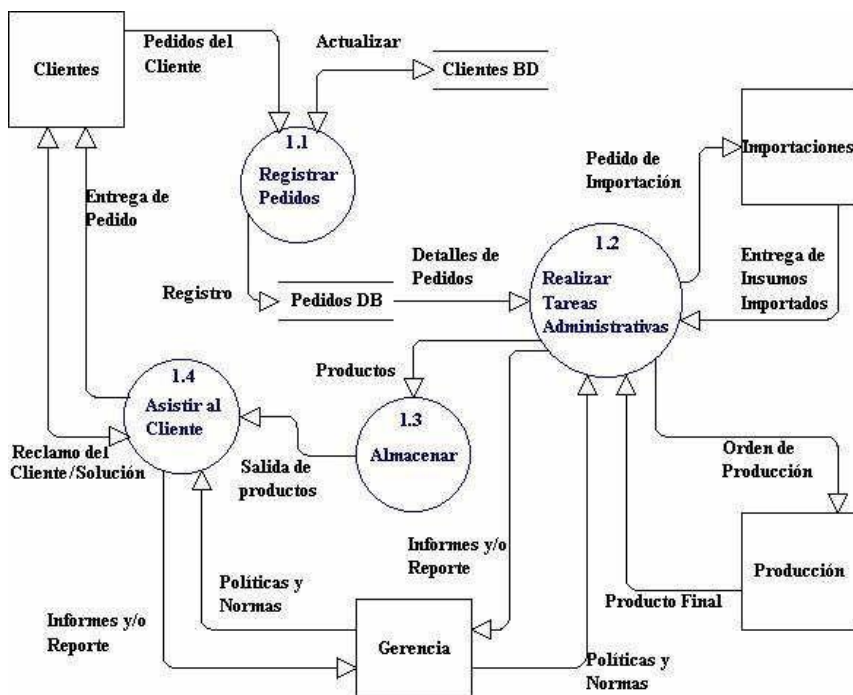


DIAGRAMA DETALLE (NIVEL 2)

DFD de Detalle 1: Registrar Pedidos

Para registrar los pedidos, estos primero se evalúan, para saber si la empresa está en capacidad de cumplir con este. Una vez evaluados los pedidos, son clasificados según el tipo y la prioridad de estos, así como así com: o por las políticas de empresa.

Luego de procede a procesar los pedidos y determinar los detalles de estos para almacenarlos y para su posterior elaboración.

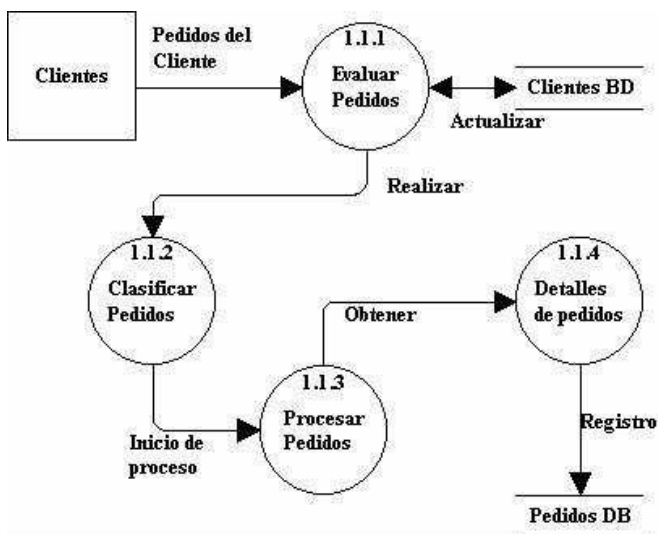


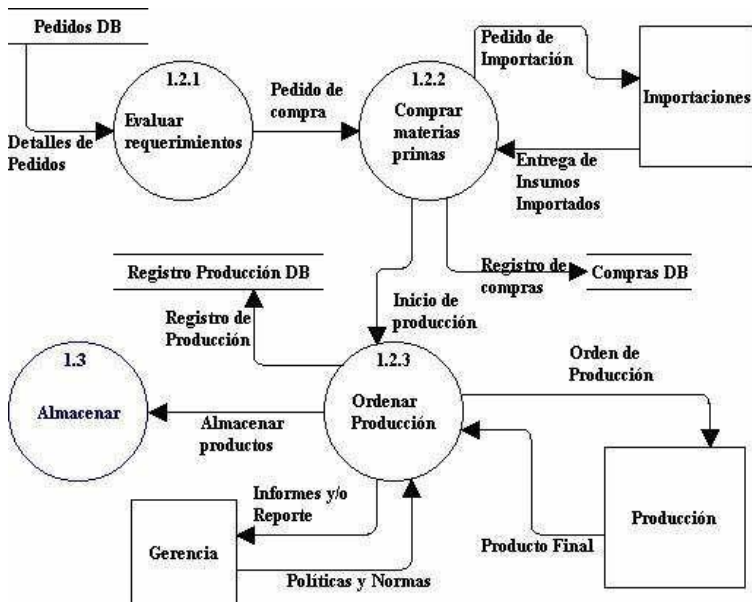
Fig. DFD 1.1. Registrar Pedidos

DFD de Detalle 2: Realizar tareas Administrativas

Luego de registrados los pedidos, las especificaciones de estos son usadas para determinar los requerimientos necesarios para poder cumplir con el pedido respectivo. Una vez determinados los requerimientos, se procede a realizar el pedido de compra de materias primas. Se procede a realizar las respectivas importaciones. Una vez recibidos los insumos, se registran la compras para llevar las cuentas de egresos. Luego se procede a iniciar la producción, emitiendo la respectiva orden de producción a planta.

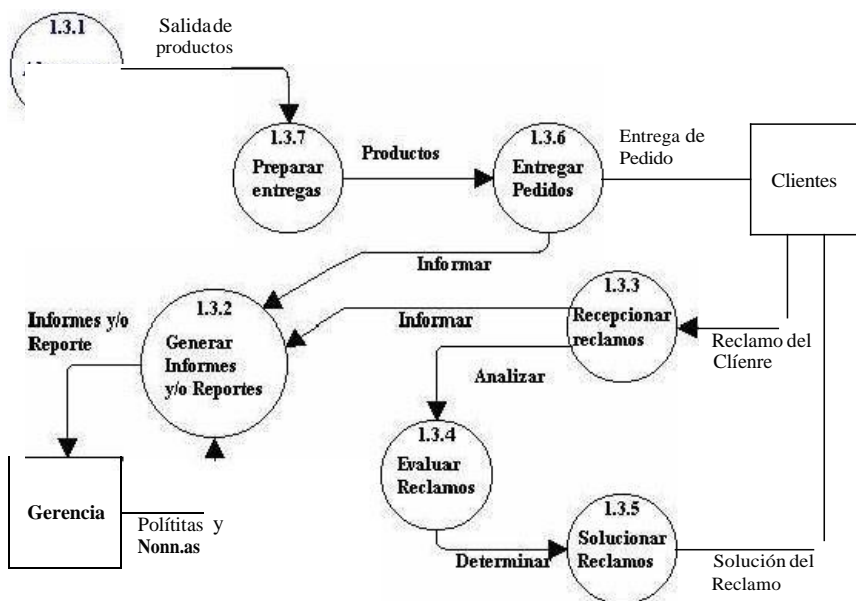
Una vez recibido el producto final de planta, se registra la cantidad producida, para las estadísticas de la empresa. Luego se procede a guardar los productos en Almacén, para su posterior entrega.

Toda operación se realiza bajo las normas y políticas que dicta la Gerencia General, las cuales requieren de informes y reportes.



DFD de Detalle 3: Almacén:

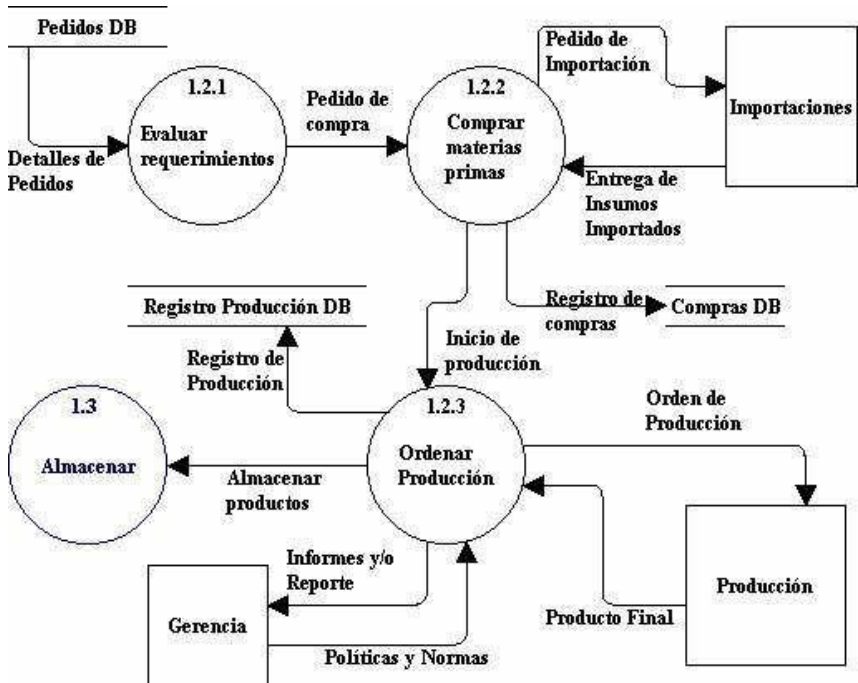
Se clasifican los productos, dependiendo del tipo de producto final que llega desde Producción. Luego se almacenan y esto es registrado en una base de datos (Ingresos). Los productos se mantienen en almacén hasta la fecha de entrega de estos. Al realizarse el retiro de los productos, esto también se almacena en una Base de Datos (Salidas)



DFD de Detalle 4: Asistir al Cliente

Retirados los productos de Almacén, estos son preparados y entregados a los clientes. La entrega de los productos genera un informe hacia Gerencia.

Si existe algún reclamo por parte del Cliente, estos son recepcionados, para evaluarlos y brindar una solución que satisfaga al Cliente. Esto también genera un Informe de Reclamos hacia Gerencia.



ANÁLISIS DE DATOS

El Análisis de Datos nos permite modelar las interrelaciones que existen en la organización y los datos que se requieren en el proceso.

ENTIDADES

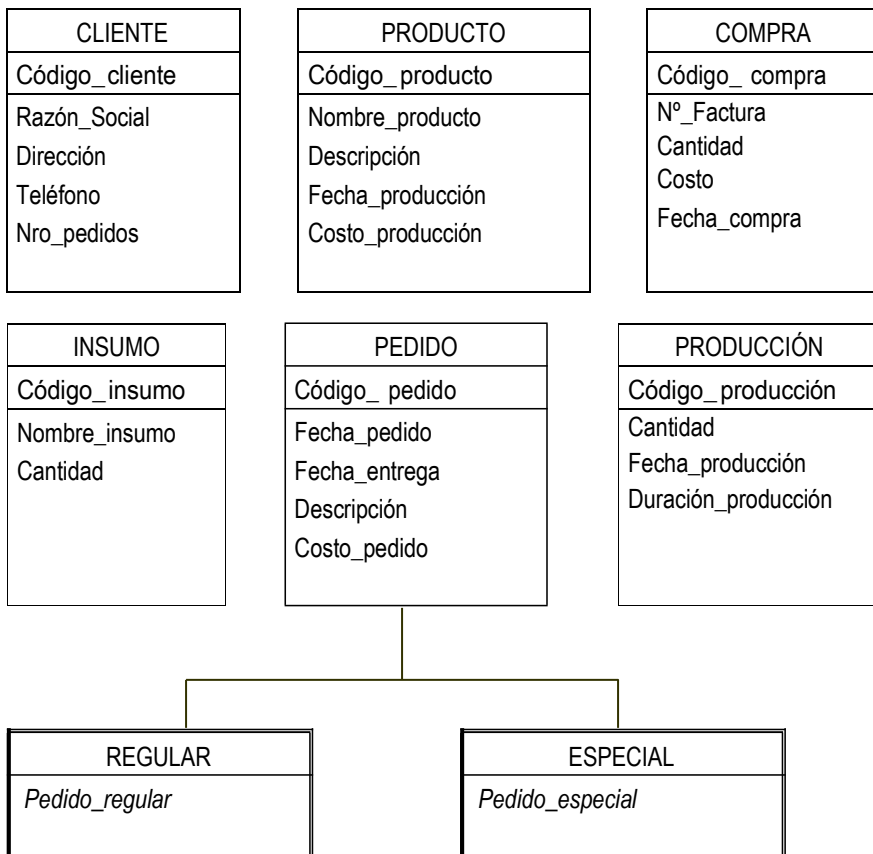
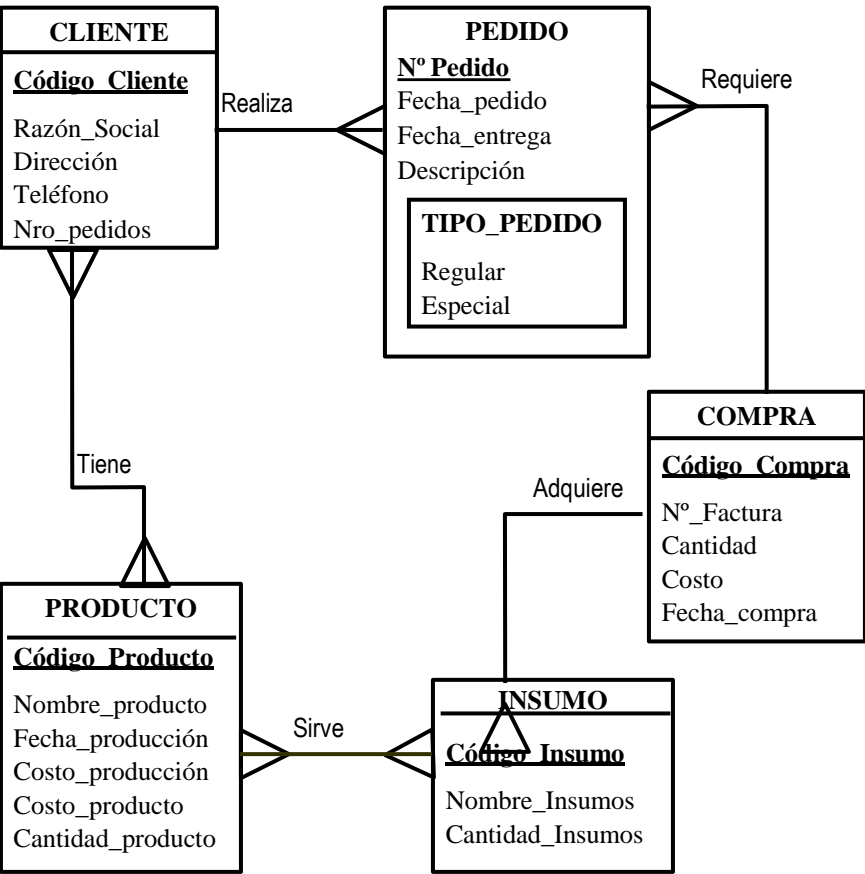


DIAGRAMA ENTIDAD-RELACIÓN DEL SISTEMA DE VENTAS



DICCIONARIO DE DATOS

ANÁLISIS DE PROCESOS: Diagrama Nivel 1 (Intermedio)

ENTIDAD	VERBO/ACCION	RECIBE	EMITE
Cientes	Realiza Recibe Realiza	Pedido	Pedido Reclamo
Importaciones	Recibe Entrega	Pedido de compra	Insumos comprados
Producción	Recibe Entrega	Orden de producción	Producto pedido
Gerencia	Da Recibe	Informes/Reportes	Normas/Políticas
Ventas	Recibe Emite Recibe Emite Recibe Entrega Recibe Emite Recibe Emite	Pedidos de Clientes Insumos comprados Producto pedido Normas/Políticas Reclamos de Clientes	Orden Importación Orden Producción Pedidos Informes/Reportes Solución de reclamos

PROCESO	DESCRIPCIÓN
Registrar Pedidos	Se encarga de Registrar todos los pedidos que los clientes realizan, así como registrar los datos de los clientes
Realizar Tareas Administrativas	Realiza las funciones administrativas del departamento, que tengan que ver con compras y ordenes de producción.
Almacenar	Recibe los productos finales y los guarda en almacén hasta la fecha de la entrega
Asistir Cliente	Se encarga de atender al cliente en la entrega de los pedidos, así como de los reclamos que estos puedan tener.

ALMACEN-DATOS	RECUPERAR	GRABAR	EXPLICACIÓN
Cientes BD	Datos_Clientes regulares	Datos_Clientes nuevos	Informes de clientes Actualizar/Informes
Pedidos BD		Pedidos	Determinar requerimientos

FLUJO DE DATO	DESCRIPCIÓN	EMITE	RECIBE
---------------	-------------	-------	--------

Pedido del Cliente	Información de Pedidos	Cliente	Registrar pedidos
Detalles de Pedidos	Especificaciones de los pedidos.	Registrar Pedidos	Realizar Tareas Admin.
Pedido de Importación	Orden de compra de insumos.	Realizar Tareas Admin.	Importaciones
Entrega de Insumos	Entrega de Insumos importados.	Importaciones	Realizar Tareas Admin.
Orden Producción	Orden de producción de pedidos.	Realizar Tareas Admin.	Producción
Producto final	Presentación del producto terminado	Producción	Realizar Tareas Admin.
Productos	Entrada de productos para almacenar.	Realizar Tareas Admin.	Almacén
Salidas de productos	Salida de Productos para entrega	Almacenar	Asistir al Cliente
Entrega de Pedidos	Entrega de producto final a los clientes.	Asistir al Cliente	Cliente
Reclamo de Cliente	Pedidos que no cumplen con las características requeridas por los clientes.	Cliente	Asistir al Cliente
Solución de Reclamo	Solución a los reclamos de los clientes.	Asistir al Cliente	Cliente
Informes y/o Reportes	Emisión de Informes y/o Reportes de procesos.	- Realizar Tareas Admin. - Asistir al Cliente	Gerencia
Normas y Políticas	Supervisión de Gerencia	Gerencia	- Realizar Tareas Admin. - Asistir al Cliente

ANÁLISIS DE PROCESOS: Diagrama Nivel 2 (Detalles)

Registrar Pedidos:

PROCESO	DESCRIPCIÓN
Evaluar Pedidos	Evaluación preliminar de los pedidos realizados por los clientes
Clasificar Pedidos	Clasificación de los pedidos factibles de realizar.
Procesar Pedidos	Iniciar procesos para cumplir pedidos según sus prioridades.
Detalles de pedidos	Determinación de los detalles y requerimientos de los pedidos.

ALMACÉN-DATOS	RECUPERAR	GRABAR	EXPLICACIÓN
---------------	-----------	--------	-------------

Cientes BD	Datos_Cientes regulares	Datos_Cientes nuevos	Informes de clientes Actualizar/Informes
Pedidos BD		Pedidos	Determinar requerimientos

FLUJO DE DATO	DESCRIPCIÓN	EMITE	RECIBE
Realizar	Realizar clasificación de los pedidos	Evaluar Pedidos	Clasificar Pedidos
Inicio de proceso	Iniciar procesamiento de los pedidos	Clasificar Pedidos	Compras_DB
Obtener	Analizar y determinar los detalles de los pedidos	Procesar Pedidos	Procesar Pedidos

Realizar tareas Administrativas:

PROCESO	DESCRIPCIÓN
Evaluar Requerimientos	Estudia los insumos se necesitaran para cumplir con los pedidos.
Comprar Materias Primas	Emite las respectivas ordenes de compras de insumos que son necesarios para cumplir con los pedidos.
Ordenar Producción	Emite las ordenes de producción, una vez que se tiene los insumos para elaborar los productos.

ALMACÉN-DATOS	RECUPERAR	GRABAR	EXPLICACIÓN
Compras BD		Insumos Comprados	Determinar gastos por compras.
Registro Producción DB		Registro de Producción	Determinar índices de producción de la empresa.

FLUJO DE DATO	DESCRIPCIÓN	EMITE	RECIBE
Pedido de Compra	Pedido de compra de insumos	Determinar Requerimientos	Comprar Materias Primas
Registro de Compras	Información de compras de insumos	Comprar Materias Primas	Compras_DB
Inicio de Producción	Emisión de	Comprar Materias Primas	Ordenar Producción

Almacenar:

PROCESO	DESCRIPCIÓN
Clasificar productos	Evaluación preliminar de los pedidos realizados por los clientes
Almacenar productos	Clasificación de los pedidos factibles de realizar.

Retirar Productos	Iniciar procesos para cumplir pedidos según sus prioridades.
-------------------	--

ALMACÉN-DATOS	RECUPERAR	GRABAR	EXPLICACIÓN
Ingresos		Ingresos en almacén	Actualizar/Informes
Salidas		Salidas de almacén	Actualizar/Informes

FLUJO DE DATO	DESCRIPCIÓN	EMITE	RECIBE
Almacén	Almacenar productor que ingresan	Clasificar productos	Almacenar productos
Retiros	Retiros de productos del almacén	Almacenar productos	Retirar productos

Asistir al Cliente:

PROCESO	DESCRIPCIÓN
Preparar entregas	Encargado de realizar la preparación de las entregas.
Entregar pedidos	Responsable de la entrega de los productos.
Generar Informes y/o Reportes	Emisión de los informes y/o reportes de los pedidos entregados y/o de los reclamos existentes.
Recepcionar reclamos	Recepcionar los reclamos que pueden realizar los clientes.
Evaluar reclamos	Evaluación de los reclamos, para determinar fallas.
Solución reclamos	Encargado de solucionar los reclamos para cumplir con el cliente.

FLUJO DE DATO	DESCRIPCIÓN	EMITE	RECIBE
Productos	Envía los productos listos para ser entregados	Prepara pedidos	Entregar pedidos
Informar	Envía datos del proceso de entrega. Envía datos del proceso de reclamos.	- Entregar pedidos. - Recepcionar reclamos	Generar Informes y/o reportes
Analizar	Iniciar evaluación de los reclamos	Recepcionar reclamos	Evaluar reclamos
Determinar	Determinar fallas y dar una solución.	Evaluar reclamos	Solucionar reclamos

ANÁLISIS DE PROCESOS: ENTIDAD-RELACIÓN

ENTIDAD	DESCRIPCIÓN	COMPONENTES
Cliente	Datos Comerciales de los Clientes	Código_Cliente Razón_Social Dirección Teléfono Nº_pedidos
Pedido	Datos de los Pedidos que realizan los clientes	Nº_Pedido Fecha_Pedido Fecha_Entrega Descripción
Compra	Datos de los Insumos que se compran	Código_Compra Nº_Factura Cantidad Costo Fecha_compra
Insumo	Elementos requeridos para elaborar productos pedidos	Código_Insumo Nombre_Insumo Cantidad_Insumo
Producto	Tipo de producto a elaborar	Código_producto Nombre_producto Fecha_producción Costo_producción Costo_producto Cantidad_producto

RELACIÓN	DESCRIPCIÓN	ENTIDADES	CARDINALIDAD	RESTRICCIONES
Realiza	Pedidos realizados por los Clientes	Cliente Pedido	1 : M	N/A
Requiere	Los pedidos requieren compras	Pedido Compra	M : M	N/A
Adquieren	Las compras adquieren insumos	Compra Insumo	1 : M	N/A
Sirve	Los insumos sirven para elaborar productos	Insumo Producto	M : M	N/A
Tiene	Los Clientes obtienen productos	Cliente Producto	M : M	N/A

GENERALIZACIONES	DESCRIPCIÓN	SUPER-ENTIDAD	SUB-ENTIDAD
Pedidos	Tipos de Pedidos que realiza el cliente	PEDIDO	- Regular - Especial

CONSTRUYENDO DIAGRAMAS DE FLUJO DE DATOS

CONTENIDOS

1. DIAGRAMAS DE FLUJOS DE DATOS

Proceso.

Flujo.

Almacén.

Terminador.

Guía para la construcción de DFD.

1.1.- Escoger nombres con significado para los procesos, flujos, almacenes y terminadores.

1.2.- Numerar los procesos.

1.3.- Evitar los DFD excesivamente complejos.

1.4.- Redibujar el DFD tantas veces como sea necesario estéticamente.

1.5.- Asegúrese de que el DFD sea lógicamente consistente.

1.6.- Extensiones del DFD para sistemas de tiempo real.

2 DIAGRAMAS DE ENTIDAD -RELACIÓN

Tipos de objetos

Relaciones

Indicadores asociativos de tipo de objeto

Indicadores de subtipo/supertipo

Reglas para la construcción de diagramas de Entidad- Relación.

ANEXO 1

CONSTRUYENDO DIAGRAMAS DE FLUJO DE DATOS

1. DIAGRAMAS DE FLUJOS DE DATOS

Como ya sabemos el diagrama de flujo de datos (DFD), es una herramienta que permite visualizar un sistema como una red de procesos funcionales, conectados entre sí por "conductos" y "tanques de almacenamiento" de datos. Siendo éste, una de las herramientas más comúnmente usadas, sobre todo por sistemas operacionales en los cuales las funciones del sistema son de gran importancia y son más complejos que los datos que éste maneja.

Es importante tener en mente: los DFD no sólo se pueden utilizar para modelar sistemas de sistemas de proceso de información, sino también como manera de modelar organizaciones enteras, es decir, como una herramienta para la planeación estratégica y de negocios.

Los componentes de un diagrama típico de flujo de datos:

- Proceso.
- Flujo.
- Almacén.
- Terminador.

Proceso.

El primer componente del DFD se conoce como proceso. Los sinónimos comunes son burbuja, función, transformación. El proceso muestra una parte del sistema que transforma entradas en salidas. El proceso se representa gráficamente como un círculo, como se muestra en figura 1.1. Algunos analistas prefieren usar un óvalo o un rectángulo con esquinas redondeadas, como se muestra en la figura 1.2. Y otros prefieren usar un rectángulo, como se muestra en la figura 1.3. Las diferencias entre estas tres formas son puramente cosméticas, aunque obviamente es importante usar la misma forma de manera consistente para representar todas las funciones de un sistema.



Fig: 1.1



Fig: 1.2



Fig: 1.3

Figuras 1.1, 1.2, 1.3: Ejemplos de procesos.

Nótese que el proceso se nombra o describe con una sola palabra, frase u oración sencilla. Un buen nombre para un proceso generalmente consiste en una frase verbo-objeto tal como validar entradas o calcular impuesto. En algunos casos, el proceso contendrá el nombre de una persona o un grupo (por ejemplo, un departamento o una división de una organización), o de una computadora o un aparato mecánico.

Flujo.

Un flujo se representa gráficamente por medio de una flecha que entra o sale de un proceso; un ejemplo se muestra en la figura 1.4. El flujo se usa para describir el movimiento de bloques o paquetes de información de una parte del sistema a otra.



Figura 1.4 Ejemplo de un flujo.

En la mayoría de los sistemas que modele como analista, los flujos realmente representan datos, es decir, bits, caracteres, mensajes, números de punto flotante y los diversos tipos de información con los que las computadoras pueden tratar.

Nótese que el flujo de la figura 1.4 tiene nombre. El nombre representa el significado del paquete que se mueve a lo largo del flujo. Un corolario de esto es que el flujo sólo lleva un tipo de paquete, como lo indica su nombre.

Los flujos muestran también la dirección: una cabeza de flecha en cualquier extremo (o posiblemente ambos) del flujo indica si los datos (o el material) se está moviendo hacia adentro o hacia fuera de un proceso (o ambas cosas). El flujo que se muestra en la figura 1.5 por ejemplo, indica claramente que el número se está mandando hacia el proceso denominado Validar números telefónicos. Y el flujo denominado honorarios de entrega de chóferes de la figura 1.6 claramente indica que es una salida generada por el proceso Generar honorarios de entrega de chóferes. Los datos que se mueven a lo largo de dicho flujo viajarán ya sea a otro proceso (como entrada) o a un almacén o a un terminador. El flujo de dos cabezas que se muestra en la figura 1.7 es un diálogo, es decir, un empaquetado conveniente de dos paquetes de datos (una pregunta y una respuesta) el mismo flujo. En el caso de un diálogo, los paquetes de cada extremo de la flecha deben nombrarse, como se ilustra en la figura 4.1.4 (c).



Figuras 1.5: Flujo de entrada.



Figura 1.6; Flujo de salida.



Figura 1.7: Flujo de diálogo.

Almacén.

El almacén se utiliza para modelar una colección de paquetes de datos en reposo. Se denota por dos líneas paralelas, como lo muestra la figura 1.8. De modo característico el nombre que se utiliza para identificar al almacén es el plural del que se utiliza para los paquetes que entran y salen del almacén por medio de flujos.

PEDIDOS

Figura 1.8: Representación gráfica de un almacén.

Para el analista con conocimiento de proceso de datos es tentador referirse a los almacenes como archivos o base de datos; pero un almacén también pudiera consistir en datos almacenados en tarjetas perforadas, microfilm, microfichas, discos ópticos, etc. y un almacén también puede ser un conjunto de fichas de papel en una caja de cartón, nombres y domicilios en un directorio, diversos archivos en un archivero, o varias formas no computarizadas.

Aparte de la forma física que toma el almacén, también existe la cuestión de su propósito: ¿Existe el sistema por causa de un requerimiento fundamental del usuario o por algún aspecto conveniente de la realización del sistema?. En el primer caso, la

base de datos existe como un área de almacenamiento diferida en el tiempo, necesaria entre dos procesos que ocurren en momentos diferentes.

Los almacenes se conectan por flujos a los procesos. Así, el contexto en el que se muestra en un DFD es uno de los siguientes (o ambos):

- Un flujo desde un almacén.
- Un flujo hacia un almacén.

Terminador.

El terminador gráficamente se representa como un rectángulo, como se muestra en la figura 1.9. Los terminadores representan entidades externas con las cuales el sistema se comunica. Comúnmente, puede ser una persona, o un grupo, por ejemplo, una organización externa o una agencia gubernamental, o un grupo o departamento que esté dentro de la misma compañía u organización, pero fuera del control del sistema que se está modelando. En algunos casos, un terminador puede ser otro sistema, como algún otro sistema computacional con el cual se comunica éste.



Figura 1.9: Representación gráfica de un terminador.

Existen tres cosas importantes que debemos recordar acerca de los terminadores:

- Son externos al sistema que se está modelando.
- Es evidente que ni el analista ni el diseñador del sistema están en posibilidades de cambiar los contenidos de un terminador o la manera en que trabaja.
- Las relaciones que existan entre los terminadores no se muestran en el modelo de DFD.

Guía para la construcción de DFD.

Además de la regla básica que existen para la elaboración de DFD tal como, los componentes básicos de DFD son: proceso(burbuja) flujo, almacenes y terminadores. Existen otras reglas adicionales que nos permitirán no elaborar DFD erróneos y gratos a la vista de los usuarios.

Las reglas incluyen las siguientes:

1. Escoger nombres con significado para los procesos, flujos, almacenes y terminadores.
2. Numerar los procesos.

3. Evitar los DFD excesivamente complejos
 4. Redibujar el DFD tantas veces como sea necesario estéticamente.
 5. Asegurarse de que el DFD sea lógicamente consistente y que también sea con cualesquiera DFD relacionados con él.
 6. Extensiones del DFD para sistemas de tiempo real
- 1.1.- Escoger nombres con significado para los procesos, flujos, almacenes y terminadores.

Un proceso en un DFD puede representar una función que se está llevando a cabo, o pudiera indicar cómo se está llevando a cabo, identificando a la persona, grupo o mecanismo involucrado.

Un buen sistema que se puede utilizar para nombrar procesos es usar un verbo y un objeto. Es decir, escoja un verbo activo (un verbo transitivo que tenga objeto) y un objeto apropiado para formar una frase descriptiva para el proceso. Los siguientes son ejemplos de nombres de procesos:

- Calcular trayectoria del proyectil.
- Producir informe de inventario.
- Validar número telefónico.
- Asignar estudiante a la clase.

Los nombres de los procesos (al igual que los nombres de flujos y de terminadores) deben provenir de un vocabulario que tenga algún significado para el usuario.

1.2.- Numerar los procesos.

Como una forma conveniente de referirse a los procesos en un DFD, muchos analistas numeran cada burbuja. No importa mucho como sea haga esto, de izquierda a derecha, de arriba abajo o de cualquier otra manera servirá, mientras haya constancia en la forma de aplicar los números.

La única cosa que se debe tener en mente es que el sistema de numeración implicará, para algunos lectores casuales de su DFD, una cierta secuencia de ejecución. Esto es, cuando se muestre el DFD a un usuario, él pudiera preguntar: ¿Acaso la burbuja número 1 sucede primero, luego la 2 y luego la 3? Y esto no es así en absoluto. El modelo de DFD es una red de procesos asincrónicos que se intercomunican, lo cual es, de hecho, una representación precisa de la manera en la que en realidad muchos sistemas operan.

Un ejemplo de la funcionalidad de enumerar las burbujas es el siguiente: Es más fácil en una discusión sobre un DFD decir " burbuja 1" en lugar de "Editar transacción y reportar errores". Pero de mayor importancia aún es el hecho de que los números se convierten en base para la numeración jerárquica.

1.3.- Evitar los DFD excesivamente complejos.

El propósito de un DFD es modelar de manera precisa las funciones que deben llevar a cabo un sistema y las interacciones entre ellas. Pero otro propósito del DFD es ser leído y comprendido, no sólo por el analista que construyó el modelo, sino por los usuarios que sean los expertos en la materia de aplicación.

Existe una regla principal para la elaboración de un DFD, que se debe tener en mente: no cree un DFD con demasiados procesos, flujos, almacenes y terminadores. En la mayoría de los casos, esto significa que no debería haber más de media docena de procesos y almacenes, flujos y terminadores relacionados en un solo diagrama.

Existe una excepción importante a esto, un diagrama especial conocido como diagrama de contexto, que representa el sistema entero como un solo proceso y destaca las interfaces entre el sistema y los terminadores externos.

1.4.- Redibujar el DFD tantas veces como sea necesario estéticamente.

En un proyecto real de análisis de sistemas el DFD debe dibujarse y volver a dibujar a menudo hasta 10 veces o más, antes de 1) ser técnicamente correcto, 2) ser aceptable para el usuario y 3) estar lo suficientemente bien dibujado como para que no sea embarazoso mostrarlo a las dirección de la organización.

¿Qué hace estéticamente agradable a un DFD?. Esto es obviamente una cuestión de gustos y puede determinarse por normas dispuestas por su organización o por las características particulares de cualquier paquete que utilice de diseño de diagramas basado en una estación de trabajo automatizada. Y la opinión de usuario pudiera ser un tanto diferente de la suya; lógicamente, cualesquiera cosas que el usuario encuentre agradable debe determinar la manera de la que se dibuje el diagrama.

1.5.- Asegúrese de que el DFD sea lógicamente consistente.

Las principales reglas de consistencia son:

- Evite sumideros infinitos, burbujas que tienen entradas pero no salidas.
- Evite las burbujas de generación espontánea, que tienen salidas sin tener entradas, porque son sumamente sospechosas y generalmente incorrectas.
- Tenga cuidado con los flujos y procesos no etiquetados. Esto suele ser un indicio de falta de esmero, pero puede esconder un error aún más grave: a veces el analista no etiqueta un flujo o un proceso porque simplemente no se le ocurre algún nombre razonable.

1.6.- Extensiones del DFD para sistemas de tiempo real.

Para los sistemas de tiempo real necesitamos alguna manera de modelar flujos de control (es decir señales o interrupciones). Y se requiere una manera de mostrar procesos de control (esto es, burbujas cuya única labor es coordinar y sincronizar las

actividades de otras burbujas del DFD). Se muestran gráficamente con líneas punteadas en el DFD.

2 DIAGRAMAS DE ENTIDAD -RELACIÓN

El diagrama de entidad-relación (también conocido como DER, o diagrama E-R) es un modelo de red que describe con un alto nivel de abstracción la distribución de datos almacenados en un sistema.

¿Porque podríamos estar interesados en modelar los datos de un sistema?. Primeramente, porque las estructuras de datos y las relaciones pueden ser tan complejas que se deseará enfatizarlas y examinarlas independientemente del proceso que se llevará a cabo. De hecho, esto se da sobre todo si mostramos el modelo del sistema correspondiente a los usuarios ejecutivos quienes se preocupan más por los datos: ¿Qué dato requerimos para manejar nuestro negocio? ¿Quién lo tiene? ¿Quién tiene acceso a ellos?.

Para el analista, el DER representa un gran beneficio: porque enfatiza las relaciones entre almacenes de datos en el DFD que de otra forma se hubiera visto sólo en la especificación de procesos. Por ejemplo, un DER típico se muestra en la figura 2.1 Cada una de las cajas rectangulares corresponden a un almacén de datos en DFD y puede verse que hay relaciones que normalmente no se aprecian en un DFD.



Figura 2.1: Diagrama de entidad-relación típico.

Hay cuatro componentes principales en un diagrama de entidad-relación:

1. Tipos de objetos.
2. Relaciones.
3. Indicadores asociativos de tipo de objeto.
4. Indicadores de supertipo/subtipo.

Tipos de objetos

El tipo de objeto se representa en un diagrama de entidad-relación por medio de una caja rectangular; en la figura 2.2 se muestra un ejemplo. Representa una colección de objetos (cosas) del mundo real cuyos miembros individuales o instancias tienen las siguientes características:

- Cada una puede identificarse de manera única por algún medio. Por ejemplo, si se tiene un tipo de objeto conocido como cliente, debemos ser capaces de distinguir uno de otro (tal vez por un número de cuenta, por su apellido, o por su número de Seguro Social).



Figura 2.2: Un tipo de objeto

- Cada uno juega un papel necesario en el sistema que se construye. Es decir, para que el tipo de objeto sea legítimo, debe poder decirse que el sistema no puede operar sin tener acceso a esos miembros.
- Cada uno puede describirse por uno o más datos. Es decir, un cliente puede describirse por medio de datos tales como nombre, domicilio, límite de crédito y número telefónico.

En muchos de los sistemas que desarrolle, los tipos de objetos serán la representación del sistema de algo material del mundo real. Esto significa que los clientes, artículos de inventario, empleados, partes manufacturadas, etc., son objetos típicos. El objeto es algo material del mundo real, y el tipo de objeto es su representación en el sistema. Sin embargo, un objeto pudiera ser algo no material: por ejemplo, horarios, planes, estándares, estrategias y mapas.

Una persona (o cualquier cosa material) pudiera ser diversos tipos de objetos distintos en distintos modelos de datos, o incluso en un mismo modelo. Juan Pérez, por ejemplo puede ser empleado en un modelo de datos y cliente en otro. También pudiera ser empleado y cliente dentro del mismo modelo.

Relaciones

Una relación representa un conjunto de conexiones entre objetos, y se representa por medio de un rombo. La figura 2.3 muestra una relación sencilla, que pudiera existir entre dos o más objetos.



Figura 2.3: Una relación

Cada instancia de la relación representa una asociación entre cero o más ocurrencias de un objeto y cero o más ocurrencias del otro. Así, en la figura 2.3, la relación etiquetada como compras puede contener las siguientes instancias individuales:

- Instancia 1: el cliente 1 compra el artículo 1
- Instancia 2: el cliente 2 compra los artículos 2 y 3.
- Instancia 3: el cliente 3 no compra ningún artículo.

La relación representa algo que debe ser recordado por el sistema: algo que no pudo haberse calculado ni derivado mecánicamente. Así, el modelo de datos de la figura 2.3 indica que existe alguna razón relacionada con el usuario para recordar el hecho de que el cliente 1 compra el artículo 1, etc. Y también indica que no existe nada priori que hubiera permitido determinar que el cliente 1 compró el artículo 1 y nada más.

Notación alternativa para relaciones

El diagrama E-R son multidireccionales, esto es, puede leerse siguiendo cualquier dirección. Y no muestran cardinalidad, es decir, no muestran el número de objetos que participan en la relación.

Una notación alternativa utilizada por algunos analistas muestra tanto la cardinalidad como la ordinalidad.

Indicadores asociativos de tipo de objeto

El indicador asociativo de tipo de objeto representa algo que funciona como objeto y como relación. Por ejemplo, un cliente que adquiere un artículo. En donde la relación de compra no hace más que asociar un cliente con uno o más artículos. Pero suponga que existen datos que deseamos recordar acerca de cada instancia de una compra (por ejemplo a qué hora del día se hizo). ¿Dónde se podría almacenar dicha información? "Hora del día" no es un atributo de cliente, ni de artículo. Más bien, se asocia "Hora del día" con la compra misma, y esto se muestra en un diagrama como el que ilustra la figura 2.5.

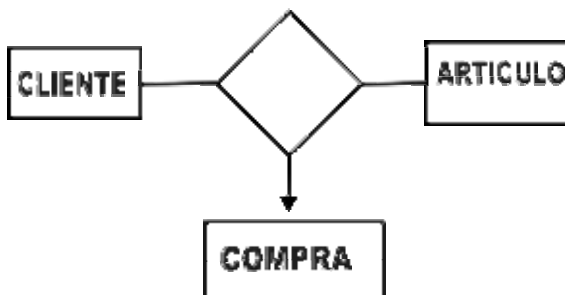


Figura 2.5: Indicador asociativo de tipo objeto

Nótese que compra ahora se escribe dentro de una caja rectangular conectada por medio de líneas dirigidas, a un rombo de relación sin nombre. Esto pretende indicar que compra funciona como:

- Un tipo de objeto, algo acerca de lo cual se desea almacenar información. En este caso la hora en la cual se realizó la compra y el descuento, que se dio al cliente.
- Una relación que conecta los dos tipos de objetos cliente y artículo. Lo que significa aquí es que cliente y artículo se mantienen solo. Existirían con o sin la compra.

Indicadores de subtipo/supertipo

Los tipos de objetos de subtipo/supertipo consisten en tipos de objeto de una o más subcategorías, conectados por una relación. La figura 2.6 muestra un subtipo /supertipo típico: la categoría general es empleado y las subcategorías son empleados asalariados y empleado por horas. Nótese que los subtipos se conectan al supertipo por medio de una relación sin nombre. Note también que el supertipo se conecta con una línea que contiene una barra.

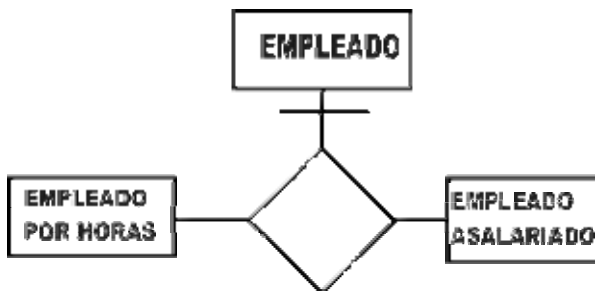


Figura 2.6: Indicador de subtipo/supertipo

Reglas para la construcción de diagramas de Entidad- Relación.

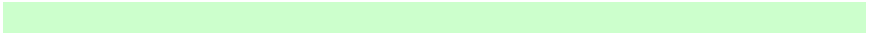
El primer DER típicamente se creará a partir de entrevista iniciales con el usuario, y de su conocimiento de la materia en cuanto al negocio del usuario. Después de desarrollar el primer DER, el siguiente paso es asignar los datos del sistema a los diversos tipos de objetos. Se supone, que se sabe cuales son los datos. Esto puede suceder en cualquier de tres maneras:

1. Si el modelo del proceso (DFD) ya se ha desarrollado o se está desarrollando paralelamente al modelo de datos, entonces el diccionario de datos ya existirá.
2. Si el modelo del proceso no se ha desarrollado o no tiene intención de desarrollar, entonces pudiera tener que empezar por entrevistar a todos los usuarios apropiados para construir una lista exhaustiva de datos y sus definiciones.

3. Si está trabajando con un grupo de administración de datos, hay una buena probabilidad de que ya exista un diccionario de datos, que podría obtener durante el proyecto.

Existe un número de situaciones en las que los refinamientos del DER llevan a la eliminación de tipos de objetos y relaciones redundantes o erróneas. Las más comunes son:

1. Tipos de objetos que consisten en un identificador.
2. Tipos de objetos para los cuales existe una sola instancia.
3. Tipos asociativos de objetos flotantes.
4. Relaciones derivadas.



MOELAMIENTO ORIENTADO A OBJETOS - MOO

CONCEPTOS BÁSICOS DE ORIENTADO A OBJETOS

CONTENIDOS

- 8.1 Definiciones análisis orientado a objetos
- 8.2 ¿Qué es análisis orientado a objeto?
- 8.3 ¿Qué es desarrollo orientado objeto?
- 8.4 Técnicas para modelado
- 8.5 Modelando objetos
- 8.6 Modelo dinámico
- 8.7 Tópicos avanzados del modelado de objetos

DEFINICIONES ANALISIS ORIENTADO A OBJETOS

Objeto combina estructura de datos y comportamiento en la misma entidad.

El Modelamiento Orientado a Objetos es usado para entender problemas, facilitar comunicación entre usuario, programador y expertos del tema.

El primer Modelo de Análisis se hace con el fin de abstraer los elementos esenciales del problema ignorando detalles.

¿QUÉ ES ANÁLISIS ORIENTADO A OBJETO?

Es un método por el cual se organiza al software como combinación de objetos que incorpora estructura de datos y comportamiento.

Un código orientado a objetos debe tener 4 características.

Identificación

Los datos se identifican como entidades únicas llamadas objetos.

Dos o más objetos distintos pueden tener atributos idénticos.

Cada objeto tiene un índice, es referenciado por medio de uno de sus atributos: deberá ser único.

Clasificación

Trabaja con grupo de objetos que tienen mismos atributos y comportamientos (operaciones).

Es una abstracción que describe las propiedades más importantes para el objeto; el resto se ignora.

Cada objeto es una instancia de clase.

Al instanciar una misma clase los valores de atributos son únicos pero nombres de atributos y comportamiento son iguales.

Polimorfismo

Una operación puede tener diferentes comportamientos gracias al polimorfismo.

Operación es transformación o acción que un objeto experimenta.

Método es la implementación de una operación en una determinada clase.

Leng OO se ejecuta un determinado método basándose en nombre de operación y la clase a la cual el obj pertenece.

Herencia

Es tener la ventaja de heredar propiedades de otras clases basado en relaciones jerárquicas.

Una subclase tendrá los mismos atributos y comportamiento que la superclase más los suyos.

Ventaja es no tener que repetir mismas propiedades que la de sus superiores.

¿QUÉ ES DESARROLLO ORIENTADO OBJETOS?

Es una técnica de desarrollo de software basado en la abstracción del mundo real.

Se basa en una porción de desarrollo de ciclo de vida: Análisis, diseño e implementación.

Modelado

Una vía para pensar, NO es una técnica de programación.

Abstracción de algo con el fin de entenderlo antes de construirlo.

Omite aspectos no importantes para la aplicación.

Mejora comunicación entre desarrollador y usuario.

Reduce complejidad:

Se realiza distintas vistas para armar un modelo y se agregan detalles para transformar en implementación.

METODOLOGÍA ORIENTADA A OBJETOS (ESTADOS)

Análisis

Comienza haciendo definición del problema.

Hace una abstracción del mundo real describiendo aspectos importantes.

Qué se hace y no cómo debería hacerse, no se describen detalles de implementación.

Se trabaja con personas que entiendan del problema.

Diseño Sistema

El diseñador toma decisiones sobre arquitectura del sistema.

Decide cuáles serán las características de performance a optimizar.

Elige una estrategia para hacerlo.

Diseño Objeto

Se basa en modelo de análisis agregando detalles de implementación.

Hace foco en estructura datos y algoritmos para implementar cada clase.

Implementación

Los objetos de clase hechos en diseño objeto se pasan a código fuente.

Es la fase menos compleja pues No se toman decisiones de ningún tipo.

TÉCNICAS PARA MODELADO

Modelo Objeto (Diagrama de objeto)

Describe la estructura de un objeto (identidad, relación con otros, atributos y comportamiento)

Describe los conceptos del mundo real más importantes para la aplicación.

Las clases definen los valores de atributos que tendrá en cada instancia y operaciones que realiza.

Modelo dinámico (Diagrama de estados)

Describe aspectos de un sistema que conciernen con el tiempo y secuencia de operaciones.

Captura el control. Muestra qué ocurre al ejecutar operación pero no porqué se implementan.

Cada diagrama de estados muestra el estado y la secuencia de operaciones en el sistema para cada clase de objetos.

Acción = Función en Modelo Funcional.

Evento realiza operaciones sobre un objeto en Modelo de Objeto.

Modelo Funcional (DFD)

Describe aspectos del sistema que conciernen con transformación de valores.

Muestra qué hace el sistema, no importa cómo ni quién.

DFD muestra dependencia entre valores y funciones sin importar si serán ejecutadas.

Funciones = Acción en Modelo Dinámico

Funciones = Operación sobre objeto en Modelo de Objeto.

MODELANDO OBJETOS

Característica es una palabra genérica, hace referencia a un atributo u operación.

Objetos

Algo que toma valor en el contexto de una aplicación.

Sirven para:

Entender al mundo real.

Tener idea básica para implementar en computadora.

Todos los objetos tienen una identidad que los diferencia de los demás.

Un objeto es único por su existencia, y no por el valor de sus propiedades.

Clases

Una clase define a un conjunto de objetos que tienen atributos similares y comportamientos iguales, relaciones con otros objetos y semántica en común.

La pertenencia de un objeto a una clase es una referencia implícita del objeto.

Diagrama de objetos

Proveen notación gráfica para modelar objetos, clases y relación con otras clases.

Diagrama de clases: Describe clases de objeto. Equivale a infinitos diagramas de instancias.

Diagrama de instancia: Muestra relación de algunos objetos con otros.

Diagrama de instancia se usa Para aclarar diagramas de clases complejas.

Atributos

Es un valor de dato que tiene el objeto de una clase.

Cada atributo tiene un valor para cada instancia de objeto.

Nombre de atributo es único en cada clase pero puede aparecer en más de una clase.

Un atributo *NO* puede ser un objeto porque no tiene identidad.

Los LOO utilizan algún atributo para referenciar un objeto determinado.

Operaciones y métodos

Operación: Acción o transformación que se experimenta en un objeto.

Una misma operación puede ser ejecutada en clases diferentes.

Método: Implementación de una operación en una determinada clase.

Todos los métodos realizan misma tarea pero situación distintas.

Es importante que una operación con varios métodos tenga misma firma (número y tipo argumentos)

(Lista de argumento) separados por comas. También se describe el tipo resultado (después de :)

Links y asociaciones

Link: Es la comunicación física o conceptual entre 2 instancias de objeto.

Link: instancia de una asociación.

Asociación: relaciona a un conjunto de links con estructura y semántica en común.

Es bidireccional, aunque se lea en una sola dirección.

En lenguaje de programación un link se implementa por medio de punteros.

Puntero: Atributo de un objeto en el que hace referencia explícita a otro objeto.

Multiplicidad

Describe cuántas instancias de clases se relacionan con otra instancia de clase asociada.

Especifica el número de objetos relacionados.

Generalmente se define con valores de intervalo continuos, pero también pueden discontinuos.

Atributos de Link

Es una propiedad del link en la asociación.

Cada atributo de link tiene valor para cada link.

Nombres de parte

Una parte es uno de los extremos de una asociación.

Una asociación binaria tiene 2 roles.

Se usa para asociaciones entre 2 objetos de una misma clase.

Calificación

Sirve para disminuir el efecto que produce la multiplicidad en una asociación.

MODELO DINÁMICO

Muestra la variación de los estados con el paso del tiempo.

Estructura estática: Estructura de datos de un objeto y relación con otros en un momento dado.

Eventos y estados

Modelo objeto describe posibles características y relación con otros objetos.

Estado: valores de atributo de un objeto y vínculos con otros.

El resultado de un estímulo depende del estado del objeto destino.

Un diagrama dinámico representa a muchos diagramas de estados.

Diagrama de estados: Una red de eventos y estados.

Evento (Ocurrencia)

Vía de transmisión de información que se manda de un objeto a otro.

Es algo que sucede de forma instantánea. No tiene duración.

Dos eventos que no están muy separados y no se afectan entre sí son concurrentes.

Clases de eventos

Reúne a un conjunto de eventos que tienen estructura y comportamiento en común.

Tienen atributos que indican la información que conduce el evento.

El tiempo en que ocurre un evento es un argumento implícito de todos los eventos.

Escenarios

Es Una secuencia de eventos que ocurren en una ejecución determinada.

Un escenario puede incluir todos los eventos de un sistema.

Estado

Abstracción de valores de atributos y links de un objeto.

Especifica respuesta de un objeto a una entrada de un evento.

Eventos = puntos en el tiempo.

Estados = intervalos de tiempos.

Diagrama de estados

Describe secuencia de estados causado por una secuencia de eventos.

Describe comportamiento de una sola clase de objetos.

Todos los objetos de una misma clase comparten mismo diagrama estado.
Cada objeto tiene su propio estado por tener distintos valores de atributo.
Cuando se recibe un evento el prox estado depende del estado actual y del evento.
Transición: Cambio de estado causado por un evento.

TOPICOS AVANZADOS DEL MODELADO DE OBJETOS

Agregación

Es una forma fuerte de asociación donde un objeto está hecho por componentes (obj. menores)

Un agregado es un objeto extendido tratado como unidad en muchas operaciones.

La agregación es transitiva.

Agregación versus asociación

Asociación: Hay un link entre 2 objetos independientes.

Agregación: 2 objetos que tienen una parte relacionada.

Agregación vs. generalización

Clases abstractas

Una clase que no tiene instancia directa.

Sus clases descendentes tienen instancias directas.

Una clase concreta puede tener subclases abstractas si tiene descendencias concretas.

Se usan para definir métodos que serán heredadas por las subclases.

Herencia múltiple (Clase acoplada - join class)

Permite a una subclase heredar características de más de una superclase.

Ventaja: Implementar posibilidad de reuso.

Desventaja: Pierde simplicidad conceptual y de implementación

PERSPECTIVA GENERAL DEL UML Y NATURALEZA Y PROPOSITO DE LOS MODELOS

CONTENIDOS

- 9.16. Introducción al UML
- 9.17. Diagramas: Vistazo general
- 9.18. Diagrama de casos de uso (use case)
- 9.19. Modelado del contexto.
- 9.20. Modelado de requisitos
- 9.21. Diagrama de clases
- 9.22. La Clase
- 9.23. Relaciones entre clases
- 9.24. Ejemplo
- 9.25. Diagrama de objetos
- 9.26. Diagrama de componentes
- 9.27. Ejecutables
- 9.28. Código fuente
- 9.29. Diagramas de despliegue
- 9.30. Diagrama Secuencia

9.1. Introducción al UML



UML (Unified Modeling Language, en español Lenguaje Unificado de Modelado) es una especificación de notación orientada a objetos. Se basa en las anteriores especificaciones BOOCH, RUMBAUGH y COAD-YOURDON. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto.

Es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo.

El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.

UML no es un lenguaje de programación. Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelamiento visual que permite una abstracción del sistema y sus componentes.

Existían diversos métodos y técnicas Orientadas a Objetos, con muchos aspectos en común pero utilizando distintas notaciones, se presentaban inconvenientes para el aprendizaje, aplicación, construcción y uso de herramientas, etc., además de pugnas entre enfoques, lo que generó la creación del UML como estándar para el modelamiento de sistemas de software principalmente, pero con posibilidades de ser aplicado a todo tipo de proyectos.

9.1.1 *Objetivos del UML*

- UML es un lenguaje de modelado de propósito general que pueden usar todos los modeladores. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática.
- UML no pretende ser un método de desarrollo completo. No incluye un proceso de desarrollo paso a paso. UML incluye todos los conceptos que se consideran necesarios para utilizar un proceso moderno iterativo, basado en construir una sólida arquitectura para resolver requisitos dirigidos por casos de uso.
- Ser tan simple como sea posible pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir. UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un

sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son la encapsulación y componentes.

- Debe ser un lenguaje universal, como cualquier lenguaje de propósito general.
- Imponer un estándar mundial.

9.1.2 *Arquitectura del UML*

Arquitectura de cuatro capas, definida a fin de cumplir con la especificación Meta Object Facility del OMG:

- Meta-metamodelo: define el lenguaje para especificar metamodelos.
- Metamodelo: define el lenguaje para especificar modelos.
- Modelo: define el lenguaje para describir un dominio de información.
- Objetos de usuario: define un dominio de información específico.

9.1.3 *Áreas conceptuales de UML*

Los conceptos y modelos de UML pueden agruparse en las siguientes áreas conceptuales:

9.1.4. Estructura estática:

Cualquier modelo preciso debe primero definir su universo, esto es, los conceptos clave de la aplicación, sus propiedades internas, y las relaciones entre cada una de ellas. Este conjunto de construcciones es la estructura estática. Los conceptos de la aplicación son modelados como clases, cada una de las cuales describe un conjunto de objetos que almacenan información y se comunican para implementar un comportamiento. La información que almacena es modelada como atributos; La estructura estática se expresa con diagramas de clases y puede usarse para generar la mayoría de las declaraciones de estructuras de datos en un programa.

9.1.5. Comportamiento dinámico:

Hay dos formas de modelar el comportamiento, una es la historia de la vida de un objeto y la forma como interactúa con el resto del mundo y la otra es por los patrones de comunicación de un conjunto de objetos conectados, es decir la forma en que interactúan entre sí. La visión de un objeto aislado es una máquina de estados, muestra la forma en que el objeto responde a los eventos en función de su estado actual. La visión de la interacción de los objetos se representa con los enlaces entre objetos junto con el flujo de mensajes y los enlaces entre ellos. Este punto de vista unifica la estructura de los datos, el control de flujo y el flujo de datos.

9.1.6. Construcciones de implementación:

Los modelos UML tienen significado para el análisis lógico y para la implementación física. Un componente es una parte física reemplazable de un sistema y es capaz de responder a las peticiones descritas por un conjunto de interfaces. Un nodo es un recurso computacional que define una localización durante la ejecución de un sistema. Puede contener componentes y objetos.

9.1.7 Organización del modelo:

La información del modelo debe ser dividida en piezas coherentes, para que los equipos puedan trabajar en las diferentes partes de forma concurrente. El conocimiento humano requiere que se organice el contenido del modelo en paquetes de tamaño modesto. Los paquetes son unidades organizativas, jerárquicas y de propósito general de los modelos de UML. Pueden usarse para almacenamiento, control de acceso, gestión de la configuración y construcción de bibliotecas que contengan fragmentos de código reutilizable.

9.1.8 Mecanismos de extensión:

UML tiene una limitada capacidad de extensión pero que es suficiente para la mayoría de las extensiones que requiere el ODA sin la necesidad de un cambio en el lenguaje básico. Un estereotipo es una nueva clase de elemento de modelado con la misma estructura que un elemento existente pero con restricciones adicionales.

9.2. Diagramas. Vistazo general

La explicación se basará en los diagramas, en lugar de en vistas o anotación, ya que son estos la esencia de UML. Cada diagrama usa la anotación pertinente y la suma de estos diagramas crean las diferentes vistas.

Las vistas existentes en UML son:

- Vista casos de uso: Se forma con los diagramas de casos de uso, colaboración, estados y actividades.
- Vista de diseño: Se forma con los diagramas de clases, objetos, colaboración, estados y actividades.
- Vista de procesos: Se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.
- Vista de implementación: Se forma con los diagramas de componentes, colaboración, estados y actividades.
- Vista de despliegue: Se forma con los diagramas de despliegue, interacción, estados y actividades.

Se Dispone de dos tipos diferentes de diagramas los que dan una vista estática del sistema y los que dan una visión dinámica.

Diagramas de la Vista Estática:

- Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
- Diagrama de objetos: Es un diagrama de instancias de las clases mostradas en el diagrama de clases. Muestra las instancias y como se relacionan entre ellas. Se da una visión de casos reales.
- Diagrama de componentes: Muestran la organización de los componentes del sistema. Un componente se corresponde con una o varias clases, interfaces o colaboraciones.
- Diagrama de despliegue: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
- Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y relaciones existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

Diagramas de la Vista Dinámica:

- Diagrama de secuencia, Diagrama de colaboración: Muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin perdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.
- Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
- Diagrama de actividades: Es un caso especial del diagrama de estados. Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.
- Como podemos ver el número de diagramas es muy alto, en la mayoría de los casos excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos. En esta unidad se dará una breve explicación de todos.

9.3. Diagrama de casos de uso (use case).

Se emplean para visualizar el comportamiento del sistema, una parte de el o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que solo especifica como deben comportarse y no como están implementadas las partes que define. Por ello es un buen sistema de documentar

partes del código que deban ser reutilizables por otros desarrolladores. El diagrama también puede ser utilizado para que los expertos de dominio se comuniquen con los informáticos sin llegar a niveles de complejidad. Un caso de uso especifica un requerimiento funcional, es decir indica esta parte debe hacer esto cuando pase esto.

En el diagrama nos encontramos con diferentes figuras que pueden mantener diversas relaciones entre ellas:

- Casos de uso: representado por una elipse, cada caso de uso contiene un nombre, que indique su funcionalidad. Los casos de uso pueden tener relaciones con otros caso de uso. Sus relaciones son:
 - Include (incluye): Representado por una flecha, en el diagrama de ejemplo podemos ver como un caso de uso, el de totalizar el coste incluye a dos casos de uso.
 - Extend (Extiende): Una relación de una caso de Uso A hacia un caso de uso B indica que el caso de uso B implementa la funcionalidad del caso de uso A.
 - Generalización: Es la típica relación de herencia.
- Actores: se representan por un muñeco. Sus relaciones son:
 - Communicates: Comunica un actor con un caso de uso, o con otro actor.
- Parte del sistema (System boundary): Representado por un cuadro, identifica las diferentes partes del sistema y contiene los casos de uso que lo forman.

En este grafico encontramos tres casos de usos Crear producto utiliza Validar producto, y Crear pack productos es una especialización de Crear productos.

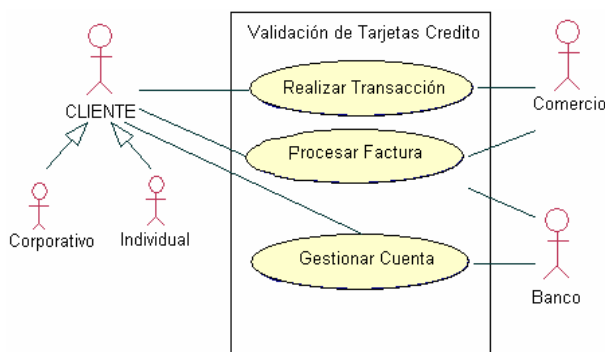
Podemos emplear el diagrama de dos formas diferentes, para modelar el contexto de un sistema, y para modelar los requisitos del sistema.

9.4. Modelado del contexto.

Se debe modelar la relación del sistema con los elementos externos, ya que son estos elementos los que forman el contexto del sistema.

Los pasos a seguir son:

- Identificar los actores que interactúan con el sistema.
- Organizar a los actores.
- Especificar sus vías de comunicación con el sistema.



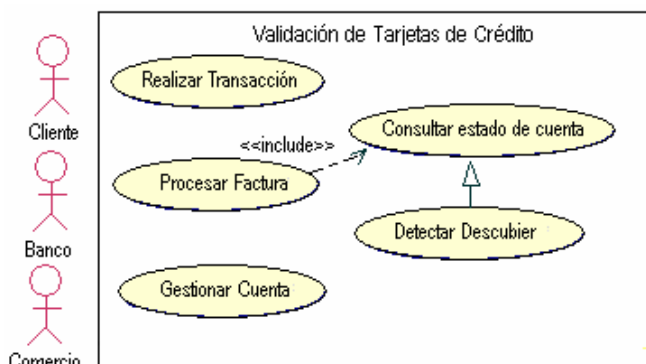
9.5. Modelado de requisitos.

La función principal, o la mas conocida del diagrama de casos de uso es documentar los requisitos del sistema, o de una parte de el.

Los requisitos establecen un contrato entre el sistema y su exterior, definen lo que se espera que realice el sistema, sin definir su funcionamiento interno. Es el paso siguiente al modelado del contexto, no indica relaciones entre autores, tan solo indica cuales deben ser las funcionalidades (requisitos) del sistema. Se incorporan los casos de uso necesarios que no son visibles desde los usuarios del sistema.

Para modelar los requisitos es recomendable:

- Establecer su contexto, para lo que también podemos usar un diagrama de casos de uso.
- Identificar las necesidades de los elementos del contexto (Actores).
- Nombrar esas necesidades, y darles forma de caso de uso.
- Identificar que casos de uso pueden ser especializaciones de otros, o buscar especializaciones comunes para los casos de uso ya encontrados.



Como podemos ver se incluyen nuevos casos de uso que no son visibles por ninguno de los actores del sistema, pero que son necesarios para el correcto funcionamiento.

9.6. Diagrama de clases.

Forma parte de la vista estática del sistema. En el diagrama de clases como ya hemos comentado será donde definiremos las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización. Es decir, es donde daremos rienda suelta a nuestros conocimientos de diseño orientado a objetos, definiendo las clases e implementando las ya típicas relaciones de herencia y agregación.

En el diagrama de clases debemos definir a estas y a sus relaciones.

9.7. La Clase.

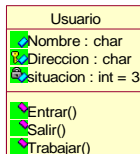
Una clase esta representada por un rectángulo que dispone de tres apartados, el primero para indicar el nombre, el segundo para los atributos y el tercero para los métodos.

Cada clase debe tener un nombre único, que las diferencie de las otras.

Un atributo representa alguna propiedad de la clase que se encuentra en todas las instancias de la clase. Los atributos pueden representarse solo mostrando su nombre, mostrando su nombre y su tipo, e incluso su valor por defecto.

Un método o operación es la implementación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen es una función que le indica a las instancias de la clase que hagan algo.

Para separar las grandes listas de atributos y de métodos se pueden utilizar



estereotipos.

Aquí vemos un ejemplo. La clase usuario contiene tres atributos. Nombre que es public, dirección que es protected y situación que es private. Situación empieza con el valor 3. También dispone de tres métodos Entrar, Salir y Trabajar.

9.8. Relaciones entre clases.

Existen tres relaciones diferentes entre clases, Dependencias, Generalización y Asociación. En las relaciones se habla de una clase destino y de una clase origen. La origen es desde la que se realiza la acción de relacionar. Es decir desde la que parte la flecha, la destino es la que recibe la flecha. Las relaciones se pueden modificar con estereotipos o con restricciones.

9.8.1. Dependencias.

Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua va desde la clase utilizadora a la clase utilizada. Con la dependencia mostramos que un cambio en la clase utilizada puede afectar al funcionamiento de la clase utilizadora, pero no al contrario. Aunque las dependencias se pueden crear tal cual, es decir sin ningún estereotipo (palabreja que aparece al lado de la línea que representa la dependencia) UML permite dar mas significado a las dependencias, es decir concretar mas, mediante el uso de estereotipos.

- Estereotipos de relación Clase-objeto.
 - Bind: La clase utilizada es una plantilla, y necesita de parámetros para ser utilizada, con Bind se indica que la clase se instancia con los parámetros pasándole datos reales para sus parámetros.
 - Derive: Se utiliza al indicar relaciones entre dos atributos, indica que el valor de un atributo depende directamente del valor de otro. Es decir el atributo edad depende directamente del atributo Fecha nacimiento.
 - Friend: Especifica una visibilidad especial sobre la clase relacionada. Es decir podrá ver las interioridades de la clase destino.
 - InstanceOF: Indica que el objeto origen es una instancia del destino.
 - Instantiate: indica que el origen crea instancias del destino.
 - Powertype: indica que el destino es un contenedor de objetos del origen, o de sus hijos.
 - Refine: se utiliza para indicar que una clase es la misma que otra, pero mas refinada, es decir dos vistas de la misma clase, la destino con mayor detalle.

9.8.2. Generalización.

Pues es la herencia, donde tenemos una o varias clases padre o superclase o madre, y una clase hija o subclase. UML soporta tanto herencia simple como herencia múltiple. Aunque la representación común es suficiente en el 99.73% de los casos UML nos permite modificar la relación de Generalización con un estereotipo y dos restricciones.

- Estereotipo de generalización.

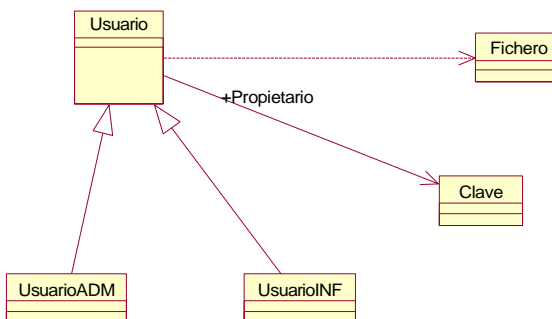
- Implementation: El hijo hereda la implementación del padre, sin publicar ni soportar sus interfaces.
- Restricciones de generalización.
 - Complete: La generalización ya no permite mas hijos.
 - Incomplete: Podemos incorporar mas hijos a la generalización.
 - Disjoint: solo puede tener un tipo en tiempo de ejecución, una instancia del padre solo podrá ser de un tipo de hijo.
 - Overlapping: puede cambiar de tipo durante su vida, una instancia del padre puede ir cambiando de tipo entre los de sus hijos.

9.8.3. Asociación.

Especifica que los objetos de una clase están relacionados con los elementos de otra clase. Se representa mediante una línea continua, que une las dos clases. Podemos indicar el nombre, multiplicidad en los extremos, su rol, y agregación.

9.9. Ejemplo.

En este diagrama se han creado cuatro clases. La clase principal es Usuario, que tiene dos clases hijas UsuarioADM y UsuarioINF. El usuario mantiene una relación de asociación con la clase Clave, se indica que es propietario de una clave, o de un número indeterminado de ellas. Se le crea también una relación de dependencia con la clase Perfil, es decir las instancias de usuario contendrán como miembro una instancia de Perfil.



9.10. Diagrama de objetos.

Forma parte de la vista estática del sistema. En este diagrama se modelan las instancias de las clases del diagrama de clases. Muestra a los objetos y sus relaciones, pero en un momento concreto del sistema. Estos diagramas contienen objetos y enlaces. En los diagramas de objetos también se pueden incorporar clases, para mostrar la clase de la que es un objeto representado.

En este diagrama se muestra un estado del diagrama de eventos. Para realizar el diagrama de objetos primero se debe decidir que situación queremos representar del sistema. Es decir si disponemos de un sistema de mensajería, deberemos decidir que representaremos el sistema con dos mensajes entrantes, los dos para diferentes departamentos, dejando un departamento inactivo. Para el siguiente diagrama de clases:

Tendríamos un diagrama de objetos con dos instancias de Mensaje, mas concretamente con una instancia de MensajeDIR y otra de MensajeADM, con todos sus atributos valorados. También tendríamos una instancia de cada una de las otras clases que deban tener instancia. Como CanalEnt, INS, Distr, y el Buzon correspondiente a la instancia de mensaje que se este instanciando. En la instancia de la clase INS se deberá mostrar en su miembro Estado, que esta ocupado realizando una inserción.

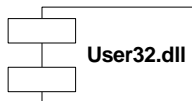
En un diseño no podemos encontrar con multitud de diagramas de objetos, cada uno de ellos representando diferentes estados del sistema.

9.11. Diagrama de componentes.

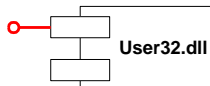
Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

En el situaremos librerías, tablas archivos, ejecutables y documentos que formen parte del sistema.

Uno de los usos principales es que puede servir para ver que componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.



Aquí tenemos un componente del sistema de Windows. En el diagrama de componentes de Windows debe salir este componente, ya que sin el sistema no funcionaría.



En esta otra figura tenemos el mismo componente, pero indicamos que dispone de un interface. Al ser una DLL el interface nos da acceso a su contenido. Esto nos hace pensar que la representación anterior es incorrecta, pero no es así solo corresponde a un nivel diferente de detalle.

Como ya hemos indicado antes todo objeto UML puede ser modificado mediante estereotipos, los estándares que define UML son:

- Executable
- Library
- Table
- File
- Document.

Aunque por suerte no estamos limitados a estas especificaciones. Que pasa si queremos modelar un proyecto de Internet donde nuestros componentes son ASP, HTML, y Scripts, y queremos marcarlo en el modelo. Pues utilizamos un estereotipo. Existe ya unos definidos WAE (Web Applications Extensión).

Podemos modelar diferentes partes de nuestro sistema, y modelar diferentes entidades que no tiene nada que ver entre ellas.

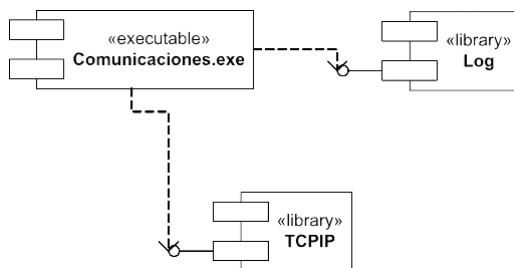
- Ejecutables y bibliotecas.
- Tablas.
- API
- Código fuente.
- Hojas HTML.

9.12. *Ejecutables.*

Nos facilita la distribución de ejecutables a los clientes. Documenta sus necesidades y dependencias. Si disponemos de un ejecutable que solo se necesita a el mismo para funcionar no necesitaremos el diagrama de componentes.

Los pasos a seguir para modelar, a priori no a posteriori, son:

- Identificar los componentes, las particiones del sistema, cuales son factibles de ser reutilizadas. Agruparlos por nodos y realizar un diagrama por cada nodo que se quiera modelar.
- Identificar cada componente con su estereotipo correspondiente.
- Considerar las relaciones entre componentes.

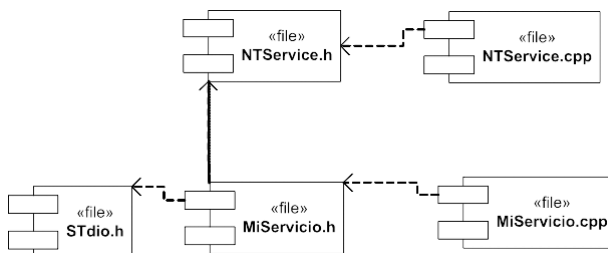


En este grafico se muestra un ejecutable que utiliza dos librerías, estas dos librerías disponen de su interface con el que ofrecen el acceso a sus servicios. Se puede ver que estas librerías son componentes que pueden ser reutilizados en otras partes del sistema.

9.13. Código fuente.

Se utiliza para documentar las dependencias de los diferentes ficheros de código fuente. Un ejecutable, o librería es una combinación de estos ficheros, y al mostrar la dependencia entre ellos obtenemos una visión de las partes necesarias para la creación del ejecutable o librería.

Al tener documentadas las relaciones se pueden realizar cambios en el código de un archivo teniendo en cuenta donde se utiliza, y que otros ficheros pueden verse afectados por su modificación.



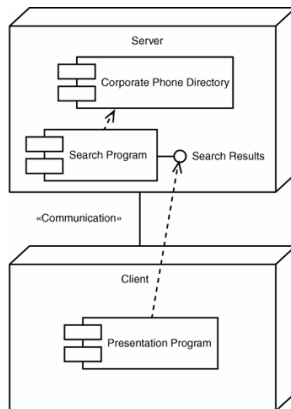
Aquí tenemos la relación entre los diferentes ficheros de un sistema. Cada fichero Cpp utiliza su fichero .h correspondiente, y MiServicio.h utiliza NTServicio.h u Stdio.h.

9.14. Diagramas de despliegue.

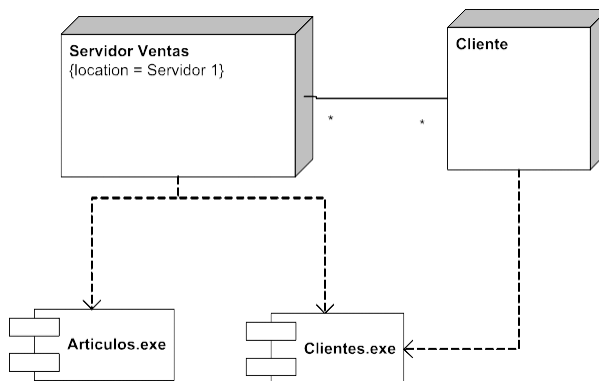
En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo.

Un nodo se representa como un cubo, un nodo es un elemento donde se ejecutan los componentes, representan el despliegue físico de estos componentes.

Aquí tenemos dos nodos, el cliente y el servidor, cada uno de ellos contiene componentes. El componente del cliente utiliza un interface de uno de los componentes del servidor. Se muestra la relación existente entre los dos Nodos. Esta relación podríamos asociarle un estereotipo para indicar que tipo de conexión disponemos entre el cliente y el servidor, así como modificar su cardinalidad, para indicar que soportamos diversos clientes.



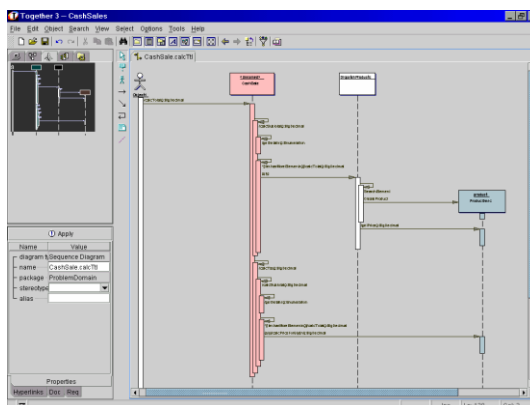
Como los componentes pueden residir en mas de un nodo podemos situar el componente de forma independiente, sin que pertenezca a ningún nodo, y relacionarlo con los nodos en los que se sitúa.



9.15. Diagrama Secuencia.

El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema.

En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su



vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando el esta activo.

En esta pantalla tenemos tres objetos, y un actor, situado a la izquierda que es el que inicia la acción. Como podemos ver el objeto de mas a la derecha aparece mas abajo que los otros existentes. Esto se debe a que recibe una llamada de creación. Es decir el objeto no existe en el sistema hasta que recibe la primera petición.

LA VISTA ESTÁTICA

CONTENIDOS

- 10.1 Descripción
- 10.2 Relaciones
- 10.3 Asociaciones
- 10.4 Generalización
- 10.5 Realización
- 10.6 Dependencias
- 10.7 Restricción
- 10.8 Instancias

10.1. Diagramas de Estructura Estática

Con el nombre de Diagramas de Estructura Estática se engloba tanto al Modelo Conceptual de la fase de Análisis como al Diagrama de Clases de la fase de Diseño. Ambos son distintos conceptualmente, mientras el primero modela elementos del dominio el segundo presenta los elementos de la solución software. Sin embargo, ambos comparten la misma notación para los elementos que los forman (clases y objetos) y las relaciones que existen entre los mismos (asociaciones).

10.1.1 Clases

Una clase se representa mediante una caja subdividida en tres partes: En la superior se muestra el nombre de la clase, en la media los atributos y en la inferior las operaciones. Una clase puede representarse de forma esquemática (plegada), con los detalles como atributos y operaciones suprimidos, siendo entonces tan solo un rectángulo con el nombre de la clase. En la Figura 10.1 se ve cómo una misma clase puede representarse a distinto nivel de detalle según interese, y según la fase en la que se esté.

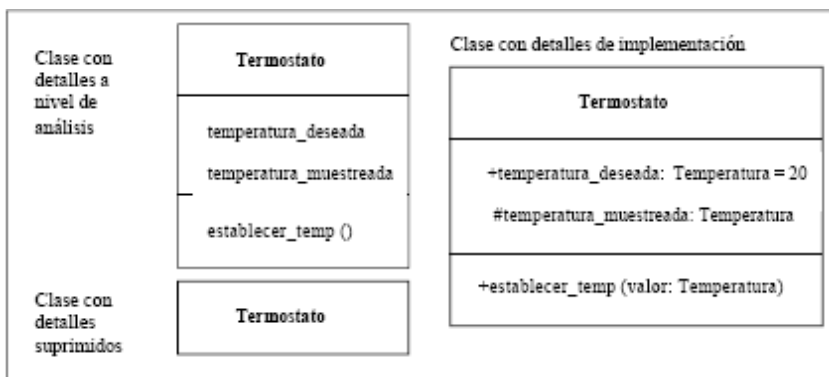


Figura 10.1 Notación para clases a distintos niveles de detalle

10.1.2 Objetos

Un objeto se representa de la misma forma que una clase. En el compartimento superior aparece el nombre del objeto junto con el nombre de la clase subrayados, según la siguiente sintaxis:

nombre_del_objeto:nombre_de_la_clase

Puede representarse un objeto sin un nombre específico, entonces sólo aparece el nombre de la clase.

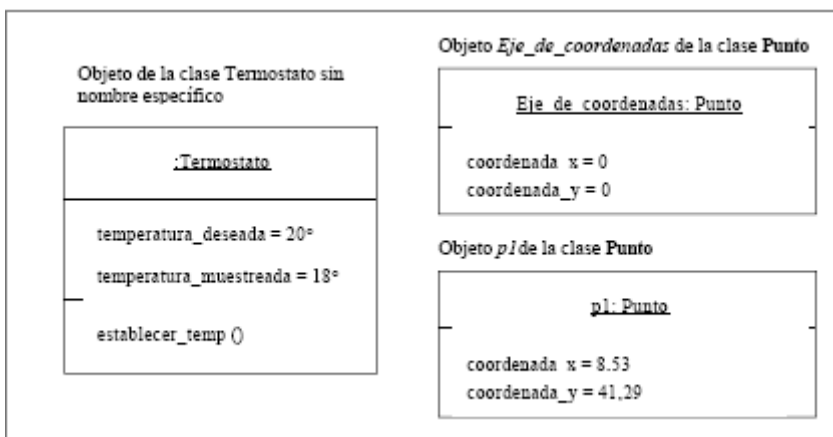


Figura 10.2 Ejemplos de objetos

10.1.3 Asociaciones

Las asociaciones entre dos clases se representan mediante una línea que las une. La línea puede tener una serie de elementos gráficos que expresan características particulares de la asociación. A continuación se verán los más importantes de entre dichos elementos gráficos.

10.1.3.1 Nombre de la Asociación y Dirección

El nombre de la asociación es opcional y se muestra como un texto que está próximo a la línea. Se puede añadir un pequeño triángulo negro sólido que indique la dirección en la cual leer el nombre de la asociación. En el ejemplo de la Figura 10.3 se puede leer la asociación como “Director manda sobre Empleado”.

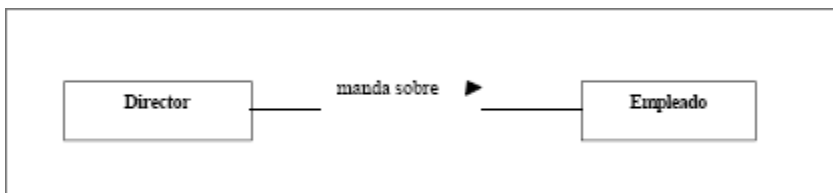


Figura 10.3 Ejemplo de asociación con nombre y dirección

Los nombres de las asociaciones normalmente se incluyen en los modelos para aumentar la legibilidad. Sin embargo, en ocasiones pueden hacer demasiado abundante la información que se presenta, con el consiguiente riesgo de saturación. En ese caso se puede suprimir el nombre de las asociaciones consideradas como suficientemente conocidas.

En las asociaciones de tipo agregación y de herencia no se suele poner el nombre.

10.1.3.2 Multiplicidad

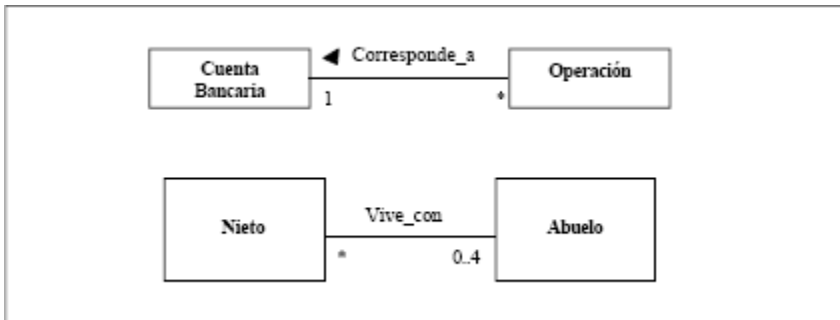


Figura 10.4 Ejemplos de multiplicidad en asociaciones

La multiplicidad es una restricción que se pone a una asociación, que limita el número de instancias de una clase que pueden tener esa asociación con una instancia de la otra clase. Puede expresarse de las siguientes formas:

- Con un número fijo: 1.
- Con un intervalo de valores: 2..5.
- Con un rango en el cual uno de los extremos es un asterisco. Significa que es un intervalo abierto. Por ejemplo, 2..* significa 2 o más.
- Con una combinación de elementos como los anteriores separados por comas: 1, 3..5, 7, 15..*.
- Con un asterisco: *. En este caso indica que puede tomar cualquier valor (cero o más).

10.1.3.3 Roles

Para indicar el papel que juega una clase en una asociación se puede especificar un nombre de rol. Se representa en el extremo de la asociación junto a la clase que desempeña dicho rol.

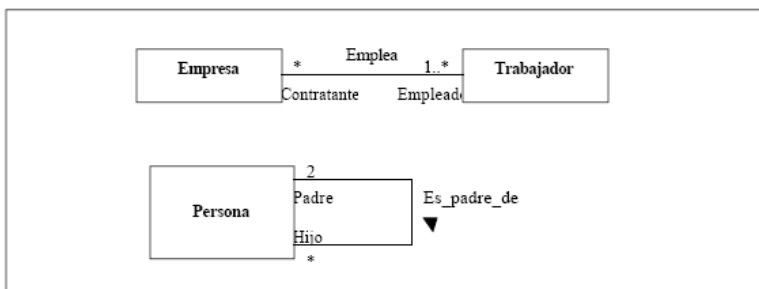


Figura 10.5 Ejemplo de roles en una asociación

10.1.3.4 Agregación

El símbolo de agregación es un diamante colocado en el extremo en el que está la clase que representa el “todo”.

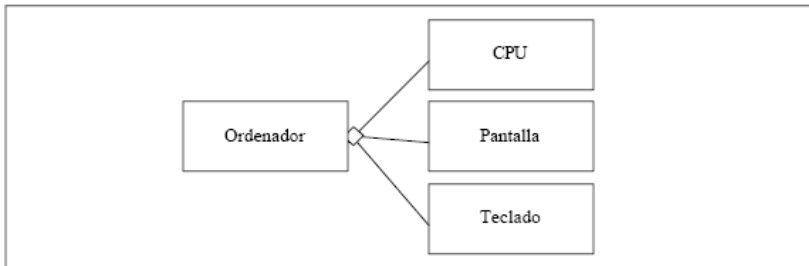


Figura 10.6 Ejemplo de agregación

10.1.3.5 Clases Asociación

Cuando una asociación tiene propiedades propias se representa como una clase unida a la línea de la asociación por medio de una línea a trazos. Tanto la línea como el rectángulo de clase representan el mismo elemento conceptual: la asociación. Por tanto ambos tienen el mismo nombre, el de la asociación. Cuando la clase asociación sólo tiene atributos el nombre suele ponerse sobre la línea (como ocurre en el ejemplo de la Figura 10.7). Por el contrario, cuando la clase asociación tiene alguna operación o asociación propia, entonces se pone el nombre en la clase asociación y se puede quitar de la línea.

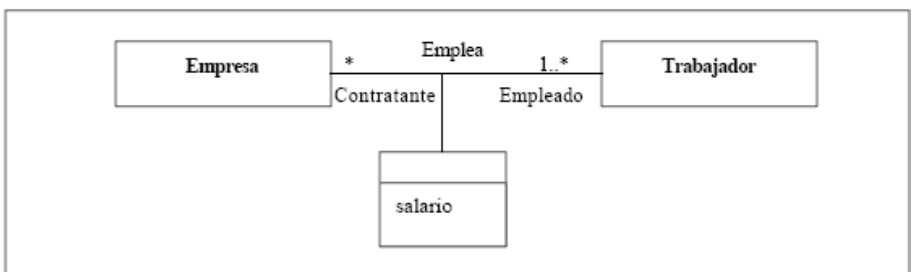


Figura 10.7 Ejemplo de clase asociación

10.1.3.6 Asociaciones N-Arias

En el caso de una asociación en la que participan más de dos clases, las clases se unen con una línea a un diamante central. Si se muestra multiplicidad en un rol, representa el número potencial de tuplas de instancias en la asociación cuando el resto de los N-1 valores están fijos. En la Figura 12 se ha impuesto la restricción de que un jugador no puede jugar en dos equipos distintos a lo largo de una temporada, porque la multiplicidad de “Equipo” es 1 en la asociación ternaria.

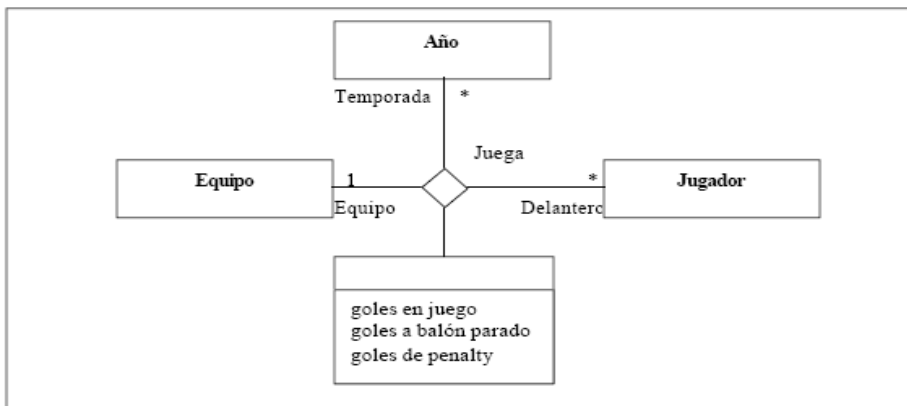


Figura 10.8 Ejemplo de asociación ternaria

10.1.3.7 Navegabilidad

En un extremo de una asociación se puede indicar la navegabilidad mediante una flecha. Significa que es posible "navegar" desde el objeto de la clase origen hasta el objeto de la clase destino. Se trata de un concepto de diseño, que indica que un objeto de la clase origen conoce al (los) objeto(s) de la clase destino, y por tanto puede llamar a alguna de sus operaciones.

10.1.4 Herencia

La relación de herencia se representa mediante un triángulo en el extremo de la relación que corresponde a la clase más general o clase "padre".

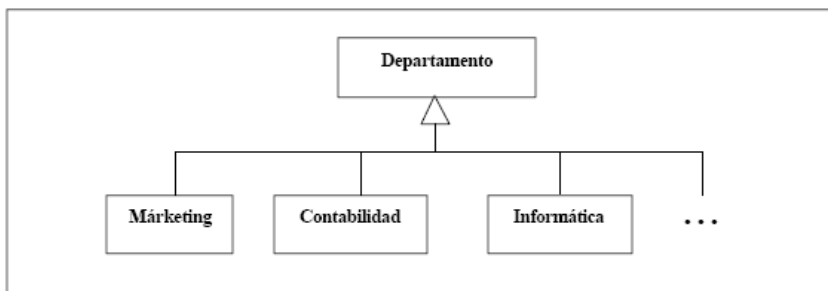


Figura 10.9 Ejemplo de herencia

Si se tiene una relación de herencia con varias clases subordinadas, pero en un diagrama concreto no se quieren poner todas, esto se representa mediante puntos suspensivos. En el ejemplo de la Figura 13, sólo aparecen en el diagrama 3 tipos de departamentos, pero con los puntos suspensivos se indica que en el modelo completo

(el formado por todos los diagramas) la clase “Departamento” tiene subclases adicionales, como podrían ser “Recursos Humanos” y “Producción”.

10.1.5 Elementos Derivados

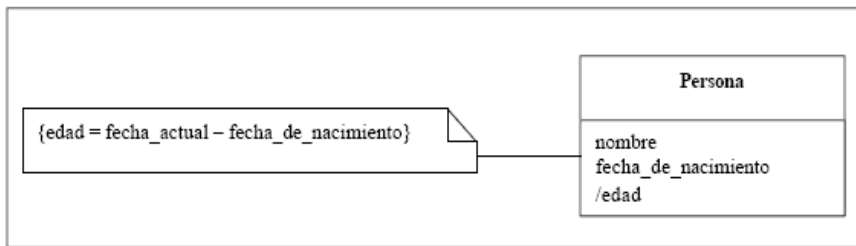
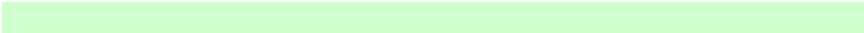


Figura 10.10 Ejemplo de atributo derivado

Un elemento derivado es aquel cuyo valor se puede calcular a partir de otros elementos presentes en el modelo, pero que se incluye en el modelo por motivos de claridad o como decisión de diseño. Se representa con una barra “/” precediendo al nombre del elemento derivado.



LA VISTA DE CASOS DE USO

CONTENIDOS

11.6 Diagrama de Casos de Uso

11.7 Actor

11.8 Casos de Uso

11.9 UML define cuatro tipos de relación en los Diagramas de Casos de Uso

11.10 La descripción del Caso de Uso comprende:

11.1. Diagrama de Casos de Uso

Los Casos de Uso es una técnica para capturar información de cómo un sistema o negocio trabaja, o de cómo se desea que trabaje. No pertenece estrictamente al enfoque orientado a objeto, es una técnica para captura de requisitos.

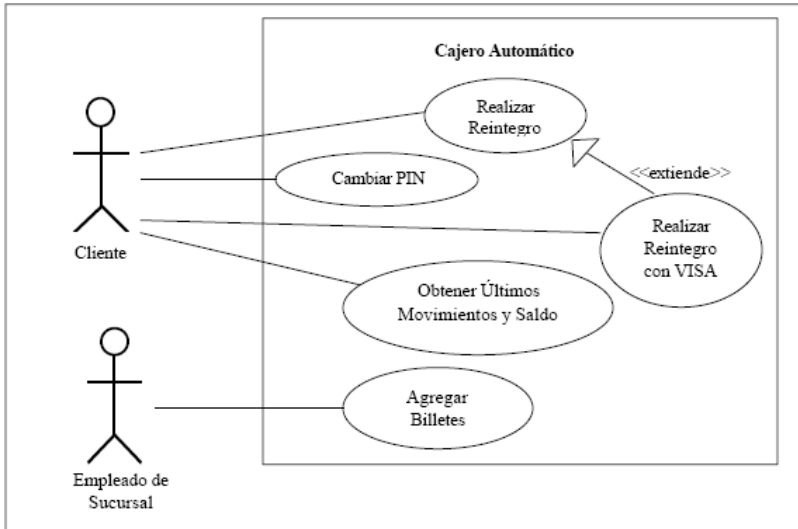


Figura 11.1 Diagrama de Casos de Uso

- Los Casos de Uso (Ivar Jacobson) describen bajo la forma de acciones y reacciones el comportamiento de un sistema desde el punto de vista del usuario.
- Permiten definir los límites del sistema y las relaciones entre el sistema y el entorno.
- Los Casos de Uso son descripciones de la funcionalidad del sistema independientes de la implementación.
- Comparación con respecto a los Diagramas de Flujo de Datos del Enfoque Estructurado.
- Los Casos de Uso cubren la carencia existente en métodos previos (OMT, Booch) en cuanto a la determinación de requisitos.
- Los Casos de Uso particionan el conjunto de necesidades atendiendo a la categoría de usuarios que participan en el mismo.
- Están basados en el lenguaje natural, es decir, es accesible por los usuarios.

11.2 Actor(es)

- Principales: personas que usan el sistema.

- Secundarios: personas que mantienen o administran el sistema.
- Material externo: dispositivos materiales imprescindibles que forman parte del ámbito de la aplicación y deben ser utilizados.
- Otros sistemas: sistemas con los que el sistema interactúa.

La misma persona física puede interpretar varios papeles como actores distintos, el nombre del actor describe el papel desempeñado.

Los Casos de Uso se determinan observando y precisando, actor por actor, las secuencias de interacción, los escenarios, desde el punto de vista del usuario. Los casos de uso intervienen durante todo el ciclo de vida. El proceso de desarrollo estará dirigido por los casos de uso. Un escenario es una instancia de un caso de uso.

11.3 UML define cuatro tipos de relación en los Diagramas de Casos de Uso:

- Comunicación
- Inclusión: una instancia del Caso de Uso origen incluye también el comportamiento descrito por el Caso de Uso destino. «include» reemplazó al denominado «uses»
- Extensión: el Caso de Uso origen extiende el comportamiento del Caso de Uso destino. «extend»
- Herencia: el Caso de Uso origen hereda la especificación del Caso de Uso destino y posiblemente la modifica y/o amplía.

11.4 Parámetros para la construcción de un caso de uso:

Un caso de uso debe ser simple, inteligible, claro y conciso. Generalmente hay pocos actores asociados a cada Caso de Uso. Preguntas clave:

1. ¿Cuáles son las tareas del actor?
2. ¿Qué información crea, guarda, modifica, destruye o lee el actor?
3. ¿Debe el actor notificar al sistema los cambios externos?
4. ¿Debe el sistema informar al actor de los cambios internos?

11.5 La descripción del Caso de Uso comprende:

1. El inicio: ¿cuándo y qué actor lo produce?
2. El fin: ¿cuándo se produce y qué valor devuelve?
3. La interacción actor-caso de uso: ¿qué mensajes intercambian ambos?
4. Objetivo del caso de uso: ¿qué lleva a cabo o intenta?
5. Cronología y origen de las interacciones
6. Repeticiones de comportamiento: ¿qué operaciones son iteradas?
7. Situaciones opcionales: ¿qué ejecuciones alternativas se presentan en el caso de uso?

LA VISTA DE ACTIVIDADES

CONTENIDOS

12 Vista de Actividades

12.1 Descripción

Es una forma especial de maquina de estado, prevista para modelar cómputos y flujos de trabajo. Los estados del grafo de actividad representan los estados de ejecución del cómputo, no los estados del objeto ordinario.

Un grafo de actividades contiene un estado de actividad. Un estado de actividad presenta la ejecución de una sentencia en un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo.

El Diagrama de Actividad es una especialización del Diagrama de Estado, organizado respecto de las acciones y usado para especificar:

- Un método
- Un caso de uso
- Un proceso de negocio (Workflow)

Un estado de actividad representa una actividad: un paso en el flujo de trabajo o la ejecución de una operación. Un grafo de actividades describe grupos secuenciales y concurrentes de actividades. Los grafos de actividades se muestran en diagramas de actividades. Las actividades se enlazan por transiciones automáticas. Cuando una actividad termina se desencadena el paso a la siguiente actividad.

Un diagrama de actividades es provechoso para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos de los mensajes. Los parámetros de entrada y salida de una acción se pueden mostrar usando las relaciones de flujo que conectan la acción y un estado de flujo de objeto.

Un grafo de actividades contiene estados de actividad que representa la ejecución de una secuencia en un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. En vez de esperar un evento, como en un estado de espera normal, un estado de actividad espera la terminación de su cómputo. Cuando la actividad termina, entonces la ejecución procede al siguiente estado de actividad dentro del diagrama. una transición de terminación es activada en un diagrama de actividades cuando se completa la actividad precedente. Los estados de actividad no tienen transiciones con eventos explícitos, peor pueden ser abortados por transiciones en estados que los incluyen.

Un grafo de actividades puede contener también estados de acción, que son similares a los de actividad pero son atómicos y no permiten transiciones mientras están activos. Los estados de acción se deben utilizar para las operaciones cortas de mantenimiento.

Un diagrama de actividades puede contener bifurcaciones, así como divisiones de control en hilos concurrentes. los hilos concurrentes representan actividades que se

pueden realizar concurrentemente por los diversos objetos o personas. La concurrencia se representa a partir de la agregación, en la cual cada objeto tiene su propio hilo. Las actividades concurrentes se pueden realizar simultáneamente o en cualquier orden. Un diagrama de actividades es como un organigrama tradicional, excepto que permite el control de concurrencia además del control secuencial.

12.2 Diagrama de actividades

Describen cómo se coordinan las actividades, es útil en operaciones que alcanzan números de cosas distintas y modela cual es la dependencia fundamental entre ellas antes de ver en que orden se van a realizar

Sus representaciones graficas son.

12.2.6 Actividad

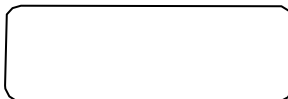
Representa las acciones que serán realizadas por el sistema y se representa con el siguiente grafico:

Un estado de actividad se representa como una caja con los extremos redondeados que contiene una descripción de actividad. Las transacciones simples de terminación se muestran como flechas. Las ramas se muestran como condiciones de guarda en transiciones o como diamantes con múltiples flechas de salida etiquetadas. Una división o una unión de control se representan con múltiples flechas que entran o salen de la barra gruesa de sincronización.

Cuando es necesario incluir eventos externos, la recepción de un evento se puede mostrar como un disparador en una transición, o como un símbolo especial que denota la espera de una señal.

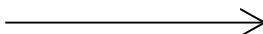
A menudo es útil organizar las actividades en un modelo según su responsabilidad. Esta clase de asignación puede mostrarse organizando las actividades en regiones distintas separadas por líneas en el diagrama. Debido a su aspecto, esto es conocido como Calles.

Un diagrama de actividades puede mostrar el flujo de objetos como valores. Para un valor de salida, se dibuja una flecha con línea discontinua desde la actividad al objeto. Para un valor de entrada, se dibuja una flecha con línea discontinua desde el objeto a una actividad.



12.2.7 Transición

Es el cambio de una actividad a otra y normalmente no se etiquetan. Se representa por el siguiente grafico



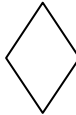
12.2.8 Barra de sincronización

Describe la coordinación entre actividades y se representa con el siguiente grafico



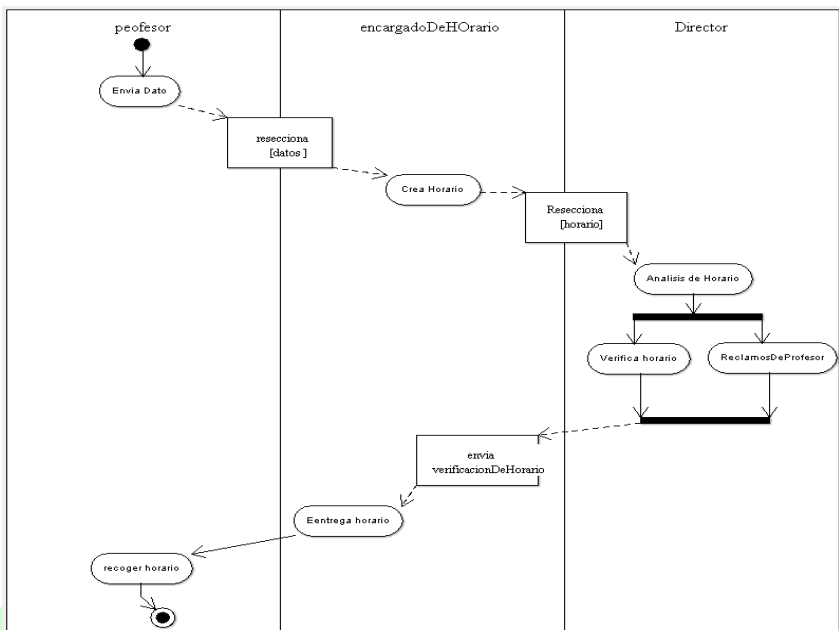
12.2.9 Diamante de decisión

Se utiliza para representar las decisiones y se representa con el siguiente grafico



12.2.10 Calles

Usados para dividir los estados de las actividades en grupo y se representa de la siguiente manera




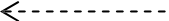


LA VISTA DE INTERACCION

CONTENIDOS

- 13.1 Colaboración
- 13.2 Interacción
- 13.3 Diagrama de Secuencia
- 13.4 Vista Física
 - 13.4.1 Descripción
 - 13.4.2 Componente
 - 13.4.3 Nodo

13 Vista de interacción

Proporciona una visión más integral del comportamiento de un sistema de objetos. Los objetos interactúan para realizar los servicios ofrecidos por las aplicaciones. Describe las secuencias de intercambios de mensajes entre roles que implementan el comportamiento de un sistema. Un rol describe el comportamiento del objeto en el contexto de una actividad.

Tipo de interacción	Símbolo	Significado
Síncrono o llamada		El emisor pierde el control hasta que el receptor termina de mantener el mensaje, entonces recupera el control
Retorno		Es el retorno del mensaje anterior
Plano		El mensaje no espera una respuesta
Asíncrono		El mensaje no espera respuesta. El emisor permanece activo

13.5 Colaboración

Es la interacción que tiene una colección de objetos para implementar un comportamiento dentro de un contexto.

Un rol clasificador representa una descripción de los objetos que pueden participar en una ejecución de la colaboración, un rol de asociación representa una descripción de los enlaces que pueden participar en una ejecución de colaboración. Un rol de clasificador es una asociación que está limitada por tomar parte en la colaboración. Las relaciones entre roles de clasificador y asociación dentro de una colaboración sólo tienen sentido en ese contexto.

Una Colaboración tiene un aspecto estructural el cual es similar a una vista estática ya que contiene un conjunto de roles y relaciones que definen el contexto para su comportamiento.

El comportamiento es el conjunto de mensajes intercambiados por los objetos ligados a los roles. El conjunto de mensajes en una colaboración se llama Interacción. Una colaboración puede incluir una o más interacciones.

13.6 Interacción

Es el conjunto de mensajes intercambiados por los roles de clasificador a través de los roles de asociación. Un mensaje puede ser una señal (comunicación explícita entre objetos, con nombre y asíncrona) o una llamada (la invocación síncrona de una operación con un mecanismo para el control, que retorna posteriormente al remitente).

Un patrón de intercambios de mensajes que se realizan para lograr un propósito específico es lo que se denomina una interacción.

13.7 Diagrama de Colaboración

La distribución de los objetos en este diagrama permite observar adecuadamente la interacción entre ellos. Solo se representan los objetos que están implicados en la colaboración. Los objetos se conectan por medio de enlaces, cada enlace representa una instancia de una asociación entre las clases implicadas. La estructura estática viene dada por los enlaces; la dinámica por el envío de mensajes por los enlaces.

Se pueden utilizar estereotipos para indicar si el objeto que recibe el mensaje es un atributo, un parámetro de un mensaje anterior, si es un objeto local o global.

Al igual que en los diagramas de secuencia se pueden crear o destruir objetos en este caso se añade new o delete respectivamente.

13.8 Diagrama de Secuencia

Muestra las interacciones cronológicamente que se dan entre objetos. Cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical que representa el rol durante cierto plazo de tiempo, con la interacción completa. Durante el tiempo que existe un objeto, el rol se muestra por una línea discontinua. Durante el tiempo que dura una activación de un procedimiento en el objeto, la línea se dibuja como una línea doble. Los diagramas de secuencia Son mejores para especificaciones de tiempo real y para escenarios complejos.

13.9 Vista Física

13.9.1 Descripción

Muestra el empaquetado físico de las partes reutilizables del sistema en unidades sustanciales, llamadas componentes, además Muestra las disposiciones físicas de los recursos de ejecución computacional, tales como computadores y sus interacciones

13.9.2 Componente

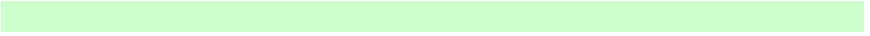
Es una unidad física de implementación con interfaces bien definidas pensada para ser utilizada como parte reemplazable de un sistema.

Los componentes soportan más interfaces y requieren interfaces de otros componentes. Una interfaz es una lista de operaciones que una pieza de software ó de hardware ofrece y puede realizar. Un componente se puede dibujar como un rectángulo, con dos rectángulos a un lado.



13.9.3 Nodo

Es un objeto físico de ejecución que representa un curso computacional, que generalmente tiene por lo menos memoria y a menudo también capacidad de proceso.



LA VISTA DE GESTIÓN DE MODELO

CONTENIDOS

- 14.1 Descripción
- 14.2 Paquete
- 14.3 Identificación de paquetes
- 14.4 Dependencias en los paquetes
- 14.5 Dependencia de acceso e importación
- 14.6 Modelo y subsistema

14.1 DESCRIPCIÓN

Cualquier sistema grande se debe dividir en unidades más pequeñas, de modo que las personas puedan trabajar con una cantidad de información limitada, a la vez y de modo que los equipos de trabajo no interfieran con el trabajo de los otros.

14.2 PAQUETE

Un paquete es una parte de un modelo. Cada parte del modelo debe pertenecer a un paquete. Pero para ser funcional, la asignación debe seguir un cierto principio racional, tal como funcionalidad común, implementación relacionada y punto de vista común. UML no impone una regla para componer los paquetes.

Los paquetes ofrecen un mecanismo general para la organización de los modelos/subsistemas agrupando elementos de modelado. Cada paquete corresponde a un submodelo (subsistema) del modelo (sistema).

Los paquetes son unidades de organización jerárquica de uso general de los modelos de UML. Pueden ser utilizados para el almacenamiento, el control de acceso, la gestión de la configuración y la construcción de bibliotecas que contengan fragmentos reutilizables del modelo.

Un paquete puede contener otros paquetes, sin límite de anidamiento pero cada elemento pertenece a (está definido en) sólo un paquete

Los paquetes contienen elementos del modelo al más alto nivel, tales como clases y sus relaciones, máquinas de estado, diagramas de casos de uso, interacciones y colaboraciones; atributos, operaciones, estados, líneas de vida y mensajes están contenidos en otros elementos y no aparecen como contenido directo de los paquetes.

La forma que tiene UML de agrupar elementos en subsistemas es a través del uso de Paquetes, pudiéndose anidar los paquetes formando jerarquías de paquetes. De hecho un sistema que no tenga necesidad de ser descompuesto en subsistemas se puede considerar como con un único paquete que lo abarca todo.

Gráficamente un paquete viene representado como se indica:

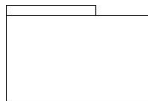


Figura 14.1 Paquete de un negocio



Figura 14.2 Arquitectura de un sistema utilizando paquetes

14.3 IDENTIFICACIÓN DE PAQUETES

Vamos a definir una serie de reglas que nos pueden ser de utilidad a la hora de agrupar los diferentes elementos en paquetes.

- Conviene agrupar elementos que proporcionen un mismo servicio. Los elementos que se agrupen en un mismo paquete han de presentar un alto grado de cohesión, es decir deben estar muy relacionados.
- Los elementos que estén en diferentes paquetes deben tener poca relación, es decir debe colaborar lo menos posible.

14.4 DEPENDENCIAS EN LOS PAQUETES

Las dependencias que se presentan entre elementos individuales, pero en un sistema de cualquier tamaño, deben ser vistas en un nivel más alto. las dependencias entre paquetes resumen dependencias entre los elementos internos a ellos, es decir, las dependencias del paquete son derivables a partir de las dependencias entre los elementos individuales.

La presencia de una dependencia entre paquetes implica que existe en un enfoque ascendente (una declaración de existencia), o que se permite que exista más adelante en un enfoque descendente (una restricción que limita cualquier otra relación), por lo menos un elemento de relación con el tipo de dependencia indicado entre elementos individuales dentro de los paquetes correspondientes.

Las dependencias múltiples del mismo tipo entre elementos individuales se agregan a una sola dependencia entre los paquetes que contienen los elementos. Si las dependencias entre elementos contienen estereotipos, éste puede ser omitido en la dependencia del paquete, para dar una sola dependencia de alto nivel.



Figura 14.3 Diagrama de despliegue

Una clase de un paquete puede aparecer en otro paquete por la importación a través de una relación de dependencia entre paquetes. Todas las clases no son necesariamente visibles desde el exterior del paquete, es decir, un paquete encapsula a la vez que agrupa.

14.5 DEPENDENCIA DE ACCESO E IMPORTACIÓN

En general, un paquete no puede tener acceso al contenido de otro paquete. Los paquetes son opacos, a menos que sean abiertos por una dependencia de acceso o de importación. La dependencia de acceso indica que el contenido del paquete del proveedor puede aparecer en referencias efectuadas por los elementos del paquete cliente. En general, un paquete puede ver solamente los elementos de otros paquetes que tienen visibilidad pública. Los elementos con visibilidad protegida pueden ser vistos únicamente por los paquetes que son descendientes del paquete contenedor de dichos elementos. Los elementos con visibilidad privada sólo son vistos por su paquete contenedor y anidados. La visibilidad también se aplica a las clases. El permiso de acceso y visibilidad son necesarios para hacer referencia a un elemento.

La dependencia de acceso no modifica el espacio de nombres del cliente no crea las referencias automáticamente, simplemente concede permiso para establecer referencias. La dependencia de importación se utiliza para agregar nombres al espacio de nombres del paquete del cliente como sinónimos de los caminos completos.

Los paquetes se dibujan como rectángulos con pestañas (similar al icono "carpeta"), las dependencias se muestran como flechas con líneas discontinuas. El operador "::" permite designar una clase definida en un contexto distinto del actual.

14.6 MODELO Y SUBSISTEMA

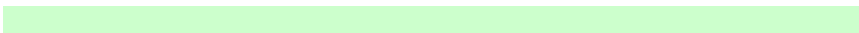
Un modelo es un paquete que abarca una descripción completa de una vista particular de un sistema. Proporciona una descripción cerrada de un sistema a partir de un punto

de vista. No tiene dependencias fuertes en otros paquetes, tales como dependencias de implementación o dependencia de herencia.

La relación detrás es una forma débil de dependencia entre elementos que observan la presencia de una cierta conexión sin implicaciones semánticas específicas.

Un modelo se estructura en forma de un árbol. El paquete raíz contiene anidados en si mismo paquetes que constituyen el detalle completo del sistema. Desde un punto de vista dado.

Un subsistema es un paquete que tiene piezas separadas de especificación y de realización, representa una unidad coherente del modelo con interfaces limpias al resto del sistema.



Etapas y actividades en el desarrollo Orientado a Objetos basado en UML

CONTENIDOS

Etapas

Análisis de Requerimientos

Diseño del Sistema

Diseño detallado

Implementación y pruebas

Complemento

Conceptos básicos en un Diagrama de Secuencia

Mensaje

Conceptos avanzados en un Diagrama de Secuencia

Conceptos básicos en un Diagrama de Colaboración

Objeto

Conceptos avanzados en un Diagrama de Colaboración

Elementos básicos en un Diagrama de Estructura Estática

Conceptos avanzados en un Diagrama de Estructura Estática

Conceptos básicos en un Diagrama de Estados

Conceptos avanzados en un Diagrama de Estados

Etapas y actividades en el desarrollo Orientado a Objetos basado en UML

Se pueden tener en cuenta las siguientes etapas

Etapas

- Análisis de Requerimientos
- Diseño del sistema
- Diseño detallado
- Implementación y pruebas

En cada etapa se resaltan las actividades que le dan cuerpo y los documentos que se esperan al final de cada una de ellas.

Análisis de Requerimientos

Descripción

En esta etapa se logra claridad sobre lo que desea el usuario y la forma en la cual se le va a presentar la solución que está buscando

Actividades técnicas	Documentos Entregables
1. Identificar Casos de Uso del sistema	Casos de Uso iniciales
2. Dar detalle a los casos de uso descritos	Borradores de Interfaz
3. Definir una interfaz inicial del sistema (si es aplicable)	Modelo del mundo inicial
4. Desarrollar el modelo del mundo	
5. Validar los modelos	

Actividades Técnicas

1. Identificar Casos de Uso del sistema

Esta información se representa en un diagrama de casos de uso.

¿Cómo encontrar un actor?

- Identifique los usuarios del sistema
 - ¿Por qué se diseña el sistema?
 - ¿Cuáles son los actores que el sistema va a beneficiar?

- ¿Qué actores van a interactuar directamente con el sistema? (actores primarios)
- ¿Qué actores van a supervisar, mantener, recibir información del sistema? (actores secundarios)
- Identifique los roles que juegan esos usuarios desde el punto de vista del sistema
- Identifique otros sistemas con los cuales exista comunicación

¿Cómo encontrar un caso de uso?

Identifique las operaciones importantes del sistema a construir

¿Cuáles son las principales tareas de un actor?

¿Qué información tiene el actor que consultar, actualizar, modificar? ¿Cómo?

¿Qué cambios del exterior debe informar el actor al sistema?

¿Qué información debe informársele al actor, con respecto a los cambios del sistema?

¿Cómo encontrar relaciones entre actores y casos de uso?

Identifique los casos de uso en los cuales se ve implicado un actor

Busque relaciones *extends* entre casos de uso

- ¿Qué casos de uso son similares, diferenciándose en la forma en la cual hacen algunas operaciones?
- ¿Qué caso de uso redefine la forma en la cual se realiza una transacción dentro de otro caso de uso?

Busque relaciones *uses* entre casos de uso

¿Que casos de uso son usados como transacciones de otros?

2. Dar detalle a los casos de uso descritos

Describir la información de entrada y salida de cada caso de uso

Descripción detallada del caso de uso

- Descripción textual de su objetivo
- Variantes posibles para realizar este caso de uso. Diagramas de interacción de detalle (de secuencia o colaboración)
- Errores y excepciones posibles en el caso de uso

Relacionar el caso de uso con la interfaz a usuario que lo representa

Especificar el diálogo que da solución al caso de uso (ver definición de interfaz)

3. Definir una interfaz inicial del sistema (si es aplicable)

- Dibujar las pantallas de interacción para los distintos actores-usuarios
- Copiar el modelo mental del usuario

- Revisar los elementos del modelo del mundo interesantes para el actor-usuario (Ver Modelo del Mundo)
- Visualización típica de los elementos del modelo del mundo
- Información relevante para el actor
- Metáforas de interacción válidas
- Especificar el diálogo que da solución a cada caso de uso que se soluciona con la interacción con esta interfaz. Puede especificarse este diálogo de varias maneras, dependiendo de la complejidad de la interfaz definida (en esta etapa se sugiere escoger el mínimo nivel de detalle posible, para dar más libertad de diseño en las etapas posteriores):
 1. Por medio de una descripción textual de su funcionamiento
 2. Por medio de diagramas de interacción que muestren la secuencia de operaciones entre los objetos de interfaz y los actores involucrados
 3. Por medio de diagramas de estados, donde se muestre claramente los estados de la interfaz
- Por medio de un prototipo funcional, en términos de la interacción con el usuario
- Definir restricciones para la comunicación con actores y sistemas
- Describir en el detalle del actor o de la relación con el caso de uso particular

4. Desarrollar el modelo del mundo

Esta información se representa en un diagrama de estructura estática de clases.

Identificar Clases

- Elementos físicos y lógicos dentro del sistema a modelar
- Top-Down: Comenzar por la clase del objeto más general (el mundo). Encontrar sus componentes hasta llegar a clases de tipos básicos
- Identificar los sustantivos del enunciado del problema y determinar si son clases del modelo del mundo
- Identificar clases desde el punto de vista de la información
 - Identifique los elementos del espacio del problema
 - Identifique otros sistemas relacionados como objetos externos
 - Identifique dispositivos relacionados
 - Identifique los eventos que el sistema debe recordar y manipular
 - Identifique los roles de los elementos del mundo
 - Identifique sitios
 - Identifique unidades organizacionales importantes en el problema
- Identificar clases desde el punto de vista funcional (casos de uso)
 - Identifique los objetos que participan en un caso de uso particular

- Continué con los mensajes de cada objeto, dejando para el final los atributos.
- Identificar clases desde el punto de vista de sus estados
 - En qué estados está en sistema? Cuáles objetos determinan estos estados?
 - Cómo es el ciclo de vida de estos objetos?

Posibles errores:

- Una clase HACE en vez de una clase ES
- Solo se requiere un objeto de la clase
- Dificultad para encontrar atributos
- Objetos con iniciativa propia
- Es un objeto y un usuario a la vez
- Solo se encuentra un servicio
- Varias clases tienen los mismos atributos o servicios
- Solo tienen información o mensajes no relevantes para el problema
- Vista Funcional: Dividir un sistema de la manera clásica

Identificar atributos y asociaciones.

- ¿Cuáles son las características determinantes del objeto en el dominio del problema?
- ¿Con qué objetos esta relacionado?
- ¿Con qué objetos debe estar relacionado para realizar sus mensajes?
- Identificar el nombre, los roles y cardinalidad de las asociaciones
- ¿Qué asociaciones hay de tipo partes y un todo (composición)?
- ¿Qué información se requiere en una clase para realizar su comportamiento?

Posibles errores

- Identificar atributos o relaciones no relevantes a los casos de uso identificados
- Las relaciones no reflejan directamente la realidad

Identificar mensajes

- Punto de vista funcional
 - ¿Qué mensajes debe tener un objeto para colaborar en un caso de uso?
- Punto de vista de comportamiento
 - ¿Qué comportamiento se espera de un objeto dado en el modelo del mundo?
 - ¿Qué mensajes se requieren para manipular la información que contienen?

- ¿Qué mensajes requieren para manipular las relaciones que tiene?
- ¿Qué mensajes hacen que el objeto cambie de un estado a otro?

Posibles errores

- Identificar servicios no relevantes a los casos de uso identificados
- Identificar servicios que no puede realizar la clase por falta de información

Identificar relaciones de herencia

- ¿Qué clases son abstracciones naturales de clases ya existentes?
- ¿Qué clases comparten atributos o servicios?
- ¿Qué clases extienden atributos o servicios de otras?

Posibles Errores

- No tener una relación *Es Un* entre las clases

Identificar restricciones del modelo

- Identificar valores posibles y no posibles de los atributos. Describirlos como restricciones de las clases
- Identificar valores permitidos para las asociaciones. Describirlos como restricciones de la asociación
- Identificar restricciones que relaciones dos o más atributos o relaciones. Describirlas dentro de la clase correspondiente

Posibles errores

- Hay estados en el modelo imposibles en el mundo real
- Hay estados en el mundo real no considerados en el modelo

Identificar paquetes

- ¿Qué subdivisiones lógicas pueden tener las clases identificadas?
- ¿Que subconjunto de clases y casos de uso pueden ser reutilizados en otros dominios?
- Combinar clases fuertemente relacionadas en un paquete
- Combinar clases que tienen que ver con los mismos casos de uso en un paquete

Consideraciones de reutilización

- Reutilizar modelos de dominio existentes
- Identificar posibles variantes en el futuro tenerlas en cuenta para diseño (patrones)

5. Validar los modelos

Validar las restricciones descritas para las clases

- Para cada clase evaluar la completitud de las restricciones

- Desarrollar objetos ejemplo que cumplan con las restricciones y que no sean válidos en el mundo real

Validar atributos y mensajes

- ¿La clase tiene toda la información necesaria para desarrollar la tarea?
- ¿La clase tiene las relaciones necesarias para propagar el mensaje y cumplir con la tarea?
- ¿Los mensajes si son utilizados dentro del contexto del problema?
- ¿Los mensajes obligan la conservación de las restricciones del modelo?

Desarrollar diagramas de interacción (diagramas de secuencia o de colaboración) para la variante por defecto de cada caso de uso, usando los objetos del modelo del mundo encontrados y sus mensajes.

- Escoger la opción por defecto de cada caso de uso
- Identificar los objetos involucrados
- Desarrollar el diagrama de secuencia o el de colaboración para la interacción

Validar los diagramas de Interacción

- Todo mensaje de un objeto a otro implica una asociación y un rol en el diagrama de clases
- Todo mensaje está definido en su correspondiente clase
- Opcional: Completar el diagrama de clases con asociaciones de dependencia a las clases de los argumentos de los mensajes

Validar con un experto del dominio

- Validar estructura del mundo
- Validar funcionalidad esperada del sistema
- Validar los diagramas de interacción descritos como detalle de los casos de uso

Validar con un usuario representativo de cada actor

- Validar la funcionalidad esperada para el actor en particular: completitud, relevancia
- Validar los diagramas de interacción descritos como detalle de los casos de uso del actor
- Validar la interfaz diseñada y el diálogo descrito

Iterar si es necesaria más información

Documentos Entregables

Casos de Uso iniciales	Requerimientos más importantes del sistema Usuarios y sistemas externos en comunicación Especificación de requerimientos
------------------------	--

Borradores de Interfaz	Presentaciones iniciales para los distintos usuarios de la forma de solucionar sus requerimientos
Modelo del mundo inicial. Versión de requerimientos	Clases, relaciones entre clases y especificación

Diseño del Sistema

Descripción

En esta etapa se define una subdivisión en aplicaciones del sistema (si es lo suficientemente grande) y la forma de comunicación con los sistemas ya existentes con los cuales debe interactuar

Actividades técnicas	Documentos Entregables
1. Identificar la arquitectura del sistema	Diagramas de Ejecución, versión inicial

Actividades Técnicas

1. Identificar la arquitectura del sistema

- Definir componentes del sistema, las aplicaciones y su ubicación. Representarlos por medio de nodos, componentes y objetos activos (representando las aplicaciones) dentro de los nodos.
- Definir mecanismos de comunicación. Expresarlos por medio de asociaciones de dependencia entre los nodos, componentes o aplicaciones y, si es conocido, agregar un estereotipo para definir el protocolo de comunicación requerido. Agregar notas con restricciones, rendimiento esperado y demás detalles de las conexiones.
- Particularizar los casos de uso a la arquitectura planteada. Refinar los casos de uso ya existentes de la etapa anterior para adecuarse a la arquitectura planteada.
- Validar arquitectura. Comprobar la validez técnica, económica y organizacional de la propuesta.

Documentos Entregables

Diagramas de Ejecución, versión inicial

- Procesadores
- Procesos
- Mecanismos de comunicación
- Descripción detallada

Diseño detallado

Descripción

En esta etapa se adecua el análisis a las características específicas del ambiente de implementación y se completan las distintas aplicaciones del sistema con los modelos de control, interfaz o comunicaciones, según sea el caso.

Actividades técnicas	Documentos Entregables
1. Agregar detalles de implementación al modelo del mundo	Diagramas de clases y paquetes, con el detalle de la implementación
2. Desarrollar el modelo de interfaz	Diagramas de interacción con el detalle de las operaciones más importantes del sistema
3. Desarrollar los modelos de control, persistencia y comunicaciones	Diagramas de estados y/o actividades para las clases concurrentes o complejas

Actividades Técnicas

1. Detalles de implementación del modelo del mundo

Completar el detalle de las clases:

- Tipos de los atributos

- Atributos y métodos de clase

- Diseño de asociaciones

- Completar los métodos

Enriquecer el modelo con el framework de base en el ambiente de implementación escogido

Incorporar patrones de diseño

Subdividir en paquetes

Definir excepciones

Completar comportamiento de las clases: Constructores, destructores, modificadores, consultores

Adecuar el modelo a las características del lenguaje de programación

Evaluar eficiencia

Validar el sistema

2. Desarrollar el modelo de interfaz

Conocer el framework de base

Enlazar las clases de interfaz con las clases del modelo del mundo

3. Desarrollar los modelos de control, persistencia y comunicaciones

Conocer los frameworks de base

Enlazar las clases del framework con las demás clases del sistema

Documentos Entregables

Diagramas de clases y paquetes, con el detalle de la implementación

Diagramas de interacción con el detalle de las operaciones más importantes del sistema

Diagramas de estados y/o actividades para las clases concurrentes o complejas

Implementación y pruebas

Descripción

Se desarrolla el código de una manera certificada.

Actividades técnicas	Documentos Entregables
1. Definir estándares de programación	Código fuente
2. Codificación y pruebas unitarias	Soporte de pruebas unitarias
3. Pruebas de módulos y de sistema	Documentación del código

Actividades Técnicas

1. Definir estándares de programación

Asimilar los idiomas aplicables al lenguaje

Conocer y adecuar estándares de programación al lenguaje

Definir estructura de directorios

Diseñar makefiles

2. Codificación y pruebas unitarias

Revisiones de código

Tratamiento de Trace y Log

3. Pruebas de módulos y de sistema

Casos de prueba

Procedimiento de instalación

Documentos Entregables

Código fuente

Soporte de pruebas unitarias

Documentación del código

COMPLEMENTO

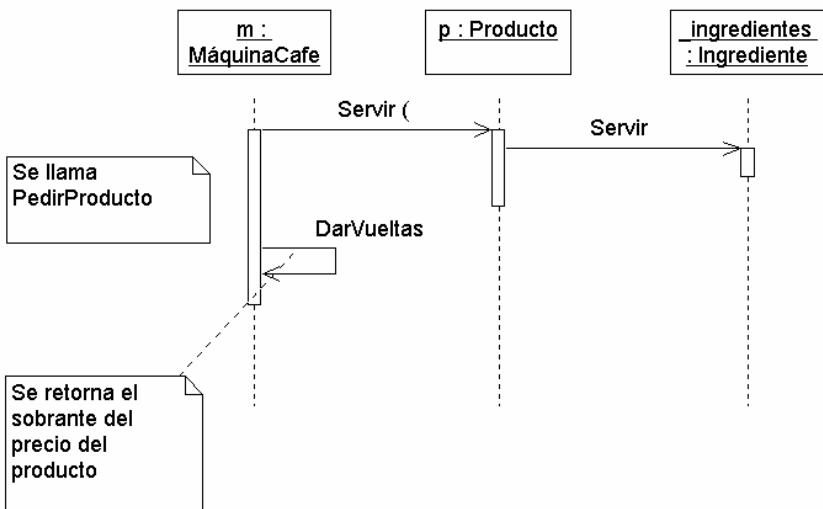
Conceptos básicos en un Diagrama de Secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Esta descripción es importante porque puede dar detalle a los casos de uso, aclarándolos al nivel de mensajes de los objetos existentes, como también muestra el uso de los mensajes de las clases diseñadas en el contexto de una operación.

A continuación se muestra un ejemplo de diagrama de secuencia, que da detalle al caso de uso PedirProducto del ejemplo de la cafetera.

Línea de vida de un objeto

Un objeto se representa como una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (véase activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato *nombreObjeto: nombreClase*. Por ejemplo, el objeto m, instancia de la clase MáquinaCafe envía dos mensajes seguidos para dar respuesta a la operación PedirProducto: Servir al objeto p de la clase Producto y DarVueltas a sí mismo.



Activación

Muestra el periodo de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto. En el ejemplo anterior el objeto _ingredientes se encuentra activado mientras ejecuta el método correspondiente al mensaje Servir; el objeto p se encuentra activo mientras se ejecuta

su método Servir (que ejecuta `_ingredientes.Servir`) y el objeto `m` se encuentra activo mientras se ejecuta `p.Servir` y `DarVueltas`.

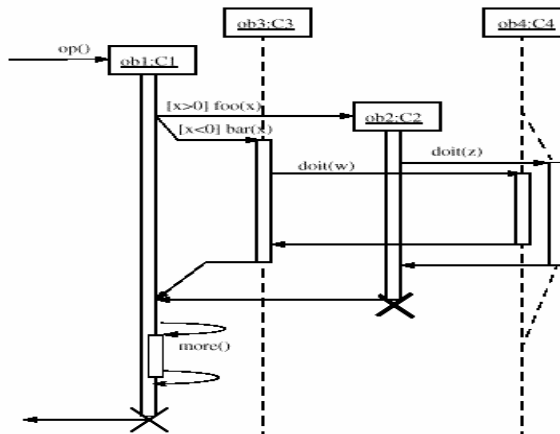
Mensaje

El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. En el ejemplo anterior el objeto `m` envía el mensaje `Servir` al objeto `p` y un poco más adelante en el tiempo el objeto `m` se envía a sí mismo el mensaje `DarVueltas`.

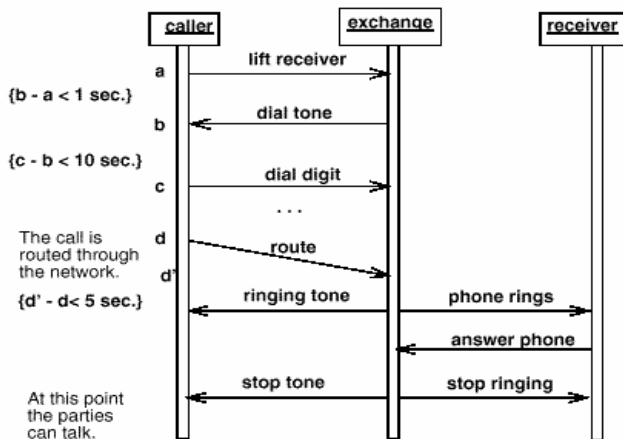
Conceptos avanzados en un Diagrama de Secuencia

Tiempos de transición

En un ambiente de objetos concurrentes o de demoras en la recepción de mensajes, es útil agregar nombres a los tiempos de salida y llegada de mensajes. Analizando la recepción de una llamada telefónica puede tenerse un diagrama como el siguiente:



En este diagrama se tienen tres objetos concurrentes, el que hace la llamada, la central telefónica y el que recibe la llamada. se nombran los tiempos de los mensajes que envía o recibe el caller (a para descolgar, b para el tono de la llamada, c para la marcación, d para el inicio del enrutamiento de la llamada, d' para la finalización del enrutamiento). Estos nombres o tiempos de transición permiten describir restricciones de tiempo (por ejemplo $b - a < 1\text{sec.}$) o demoras entre el envío y la recepción (entre d y d').



Condiciones caminos alternativos de ejecución. Concurrency

En algunos casos sencillos pueden expresarse en un diagrama de secuencia alternativas de ejecución. Estas alternativas pueden representar condiciones en la ejecución o diferentes hilos de ejecución (threads).

En el diagrama anterior se muestran dos casos. ob1 muestra una condición al enviar un mensaje a ob3 o a ob2, dependiendo de si $x > 0$ o $x < 0$. Estas dos líneas de ejecución se vuelven a unir más adelante, indicando el fin del condicional. Por otra parte ob4 muestra dos posibles operaciones dependiendo de si se siguió la condición $x > 0$ o $x < 0$. Ya que se presentan en el mismo instante de tiempo, se requiere dividir la línea del objeto en dos (esta misma representación se utiliza para el caso de dos hilos de ejecución).

Dstrucción de un objeto

Se representa como una X al final de la línea de ejecución del objeto. Por ejemplo, en el diagrama anterior se muestra el final de ob2 y de ob1.

Métodos recursivos

Un ejemplo de un método recursivo es el método more en ob1. Es un rectángulo un poco salido de la activación principal y con líneas de llamado de mensajes, que indican la entrada y salida de la recursión.

Conceptos básicos en un Diagrama de Colaboración

Un diagrama de colaboración es una forma de representar interacción entre objetos, alterna al diagrama de secuencia. A diferencia de los diagramas de secuencia, pueden mostrar el contexto de la operación (cuáles objetos son atributos, cuáles temporales, ..)

y ciclos en la ejecución. Se toma como ejemplo el caso de uso PedirProducto ya descrito como diagrama de secuencia.

Objeto

Un objeto se representa con un rectángulo, que contiene el nombre y la clase del objeto en un formato *nombreObjeto: nombreClase*.

Enlaces

Un enlace es una instancia de una asociación en un diagrama de clases. Se representa como una línea continua que une a dos objetos. Esta acompañada por un número que indica el orden dentro de la interacción y por un estereotipo que indica que tipo de objeto recibe el mensaje. Pueden darse varios niveles de subíndices para indicar anidamiento de operaciones. Los estereotipos indican si el objeto que recibe el mensaje es un atributo (association y se asume por defecto), un parámetro de un mensaje anterior, si es un objeto local o global.

Flujo de mensajes

Expresa el envío de un mensaje. Se representa mediante una flecha dirigida cercana a un enlace.

Marcadores de creación y destrucción de objetos

Puede mostrarse en la gráfica cuáles objetos son creados y destruidos, agregando una restricción con la palabra *new* o *delete*, respectivamente, cercana al rectángulo del objeto

Objeto compuesto

Es una representación alternativa de un objeto y sus atributos. En esta representación se muestran los objetos contenidos dentro del rectángulo que representa al objeto que los contiene. Un ejemplo es el siguiente objeto ventana

Conceptos avanzados en un Diagrama de Colaboración

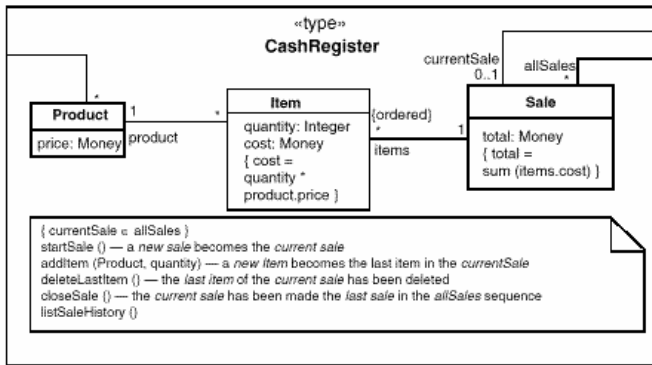
Patrón de diseño

Un diagrama de colaboración puede especificar un contrato entre objetos, parte esencial para la descripción de un patrón de diseño. Este diagrama contiene todos los elementos citados de un diagrama de colaboración, dejando libres posiblemente los tipos exactos de algunos objetos o con nombres genéricos para los mensajes. Una "instanciación" del patrón se representa como una elipse unida mediante flechas puenteadas a los objetos o clases que participan realmente en el patrón. Estas flechas pueden tener roles, indicando cuál es el papel de cada elemento dentro del patrón. Por ejemplo, una instanciación del patrón de observador puede verse como

Contexto

Un contexto es una vista de uno o más elementos dentro del modelo que colaboran en el desarrollo de una acción. Se usa para separar los demás elementos en el modelo de este problema en particular y darle énfasis. Puede mostrar solo los detalles relevantes de las clases u objetos que contiene, para resaltar su utilidad. Un ejemplo es la definición del siguiente tipo:

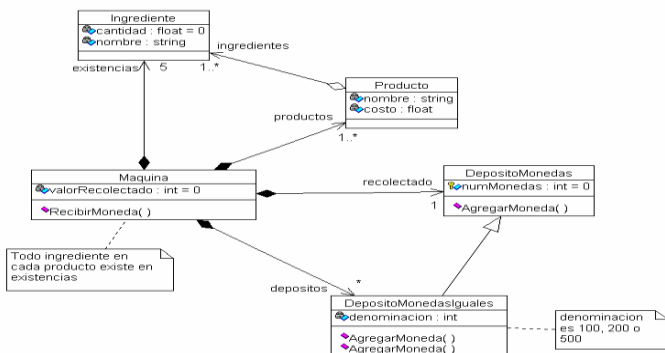
Se representa como un contexto un tipo Registro de Dinero y se muestran los detalles relevantes de Producto, Ítem y Venta para este tipo. Las relaciones de las clases con otras no visibles dentro del contexto pueden omitirse o conectarse al borde del contexto.



Objeto activo

Un objeto activo es el que contiene su propio flujo de control, a diferencia de un objeto pasivo que encapsula datos y solo reacciona al enviarle mensajes. Un objeto activo se representa con un rectángulo de bordes gruesos. Puede contener otros objetos pasivos o activos. Se presenta a continuación un ejemplo en el contexto de una producción en línea robotizada. Se tiene un ente administrador, un robot y un horno (tres objetos activos) que interactúan para desarrollar su tarea.

Los mensajes entre objetos pasivos se denotan mediante una flecha completa, mientras que los mensajes entre objetos activos se denotan con una media flecha. Los threads de ejecución se denotan con las letras A y B antes del número de orden del mensaje. La sincronización entre threads se muestra mediante un `'/'` y el nuevo número de orden. Por ejemplo en `A2, B2 / 2: completed(job)`.



Elementos básicos en un diagrama de estructura estática

Un diagrama de estructura estática muestra el conjunto de clases y objetos importantes que hacen parte de un sistema, junto con las relaciones existentes entre estas clases y objetos. Muestra de una manera estática la estructura de información del sistema y la visibilidad que tiene cada una de las clases, dada por sus relaciones con las demás en el modelo.

Supongamos el modelamiento de una máquina de café. Un diagrama de estructura estática inicial podría ser:

Clase

Representada por un rectángulo con tres divisiones internas, son los elementos fundamentales del diagrama. Una clase describe un conjunto de objetos con características y comportamiento idéntico. En el ejemplo se encuentran las clases Ingrediente, Producto, Maquina, DepositoMonedas y DepositoMonedasiguales.

Los tres compartimientos estándares alojan el nombre de la clase, sus atributos y sus mensajes, respectivamente.

Atributo

Identifican las características propias de cada clase. Generalmente son de tipos simples, ya que los atributos de tipos compuestos se representan mediante asociaciones de composición con otras clases. La sintaxis de un atributo es

visibility name : type-expression = initial-value { property-string }

Donde *visibility* es uno de los siguientes:

+ *public visibility*

protected visibility

- *private visibility*

type-expression es el tipo del atributo con nombre *name*. Puede especificarse como se ve un valor inicial y un conjunto de propiedades del atributo.

En el caso del ejemplo, la clase Ingrediente tiene dos atributos: uno denominado cantidad, de tipo float y con valor inicial 0; y el atributo nombre de tipo string sin valor inicial. En este caso, la herramienta utilizada ha cambiado la representación de la visibilidad, utilizando el símbolo ¡Error! Marcador no definido. para indicar visibilidad privada, el símbolo en Rose para visibilidad protegida y el símbolo en Rose para indicar visibilidad pública.

Operacion

El conjunto de operaciones describen el comportamiento de los objetos de una clase. La sintaxis de una operación en UML es

visibility name (parameter-list) : return-type-expression { property-string }

Cada uno de los parámetros en *parameter-list* se denota igual que un atributo. Los demás elementos son los mismos encontrados en la notación de un atributo.

Asociación (rol, multiplicidad, cualificador)

Una asociación es una línea que une dos o más símbolos. Pueden tener varios tipos de adornos que definen su semántica y características. Los tipos de asociaciones entre clases presentes en un diagrama estático son:

1. Asociación binaria
2. Asociación n aria
3. Composición
4. Generalización
5. Refinamiento
6. Cada asociación presenta algunos elementos que dan detalle a la relación como son:

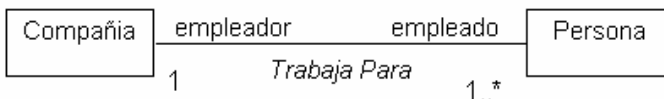
Rol: Identificado como un nombre al final de la línea. Describe la semántica de la relación en el sentido indicado. Por ejemplo, la asociación de composición entre Maquina e Ingrediente recibe el nombre de *existencias*, como rol en ese sentido.

Multiplicidad: Describe la cardinalidad de la relación. En el ejemplo anterior se utilizan 1, 1 ..*, 5, *, como indicadores de multiplicidad.

Asociación binaria

Se identifica como una línea sólida que une dos clases. Representa una relación de algún tipo entre las dos clases, no muy fuerte (es decir, no se exige dependencia existencial ni encapsulamiento).

En posible ejemplo es la relación entre una compañía y sus empleados



En este caso la relación recibe el nombre genérico *Trabaja Para*, la compañía tiene uno o más instancias de la clase *Persona* denominadas *empleado* y cada empleado conoce su *empleador* (en este caso único). Ejemplo de asociación binaria

Composición

Es una asociación fuerte, que implica tres cosas

- Dependencia existencial. El elemento dependiente desaparece al destruirse el que lo contiene y, si es de cardinalidad 1, es creado al mismo tiempo.
- Hay una pertenencia fuerte. Se puede decir que el objeto contenido es parte constitutiva y vital del que lo contiene
- Los objetos contenidos no son compartidos, esto es, no hacen parte del estado de otro objeto.

Se denota dibujando un rombo relleno del lado de la clase que contiene a la otra en la relación. En el ejemplo inicial de esta hoja se presentan varios ejemplos de relaciones de composición entre *Maquina* y *Producto*, *Maquina* y *DepositoMonedas* y *Maquina* y *DepositoMonedasIguales*.

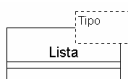
Existe también una relación de composición menos fuerte (no se exige dependencia existencial, por ejemplo) que es denotada por una un rombo sin rellenar en uno de los extremos. Un ejemplo puede encontrarse entre *Producto* e *Ingrediente*.

Generalización

La relación de generalización denota una relación de herencia entre clases. Se representa dibujando un triángulo sin rellenar en el lado de la superclase. La subclase hereda todos los atributos y mensajes descritos en la superclase. En el ejemplo se encuentra una generalización entre *DepositoMonedas* (superclase) y *DepositoMonedasIguales* (subclase).

Clase paramétrica

Una clase paramétrica representa el concepto de clase genérica en los conceptos básicos OO o de template en C++. Se dibuja como una clase acompañada de un rectángulo en la esquina superior derecha, con los parámetros del caso.

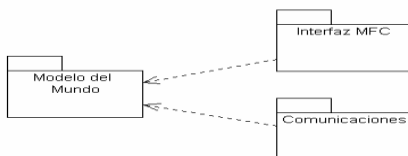


Por ejemplo, la clase Lista que utiliza un parámetro formal *Tipo* se vería de la siguiente manera

Paquete

Un paquete es una forma de agrupar clases (u otros elementos en otro tipo de diagramas) en modelos grandes. Pueden tener asociaciones de dependencia o de generalización entre ellos. Un ejemplo puede ser el siguiente

En este caso existen tres paquetes (que se muestran vacíos en este caso, con su contenido encapsulado), con dos de ellos dependiendo del Modelo del Mundo



Dependencia

Denota una relación semántica entre dos elementos (clases o paquetes, por el momento) del modelo. Indica que cambiar el elemento independiente puede requerir cambios en los dependientes. Se muestra como una línea punteada direccional, indicando el sentido de la dependencia. Puede tener por medio de estereotipos una explicación del tipo de dependencia presentada.

En el ejemplo anterior pueden verse dos relaciones de dependencia hacia el paquete Modelo del Mundo.

Nota

Es un comentario dentro de un diagrama. Puede estar relacionado con uno o más elementos en el diagrama mediante líneas punteadas. Pueden representar aclaraciones al diagrama o restricciones sobre los elementos relacionados (cuando el texto se encuentra entre '[y]'). Se representa mediante un rectángulo con su borde superior derecho doblado.

En el ejemplo inicial de esta hoja se encuentran dos notas: Una relacionada con la clase máquina y otra con el depósito de monedas iguales.

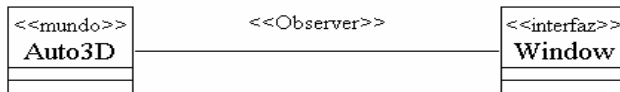
Conceptos avanzados en un diagrama de estructura estática

Los conceptos que a continuación se presentan muestran posibilidades adicionales de descripción en un modelo orientado por objetos. Aunque los conceptos fundamentales y más frecuentemente usados se describieron en los conceptos básicos, éstos completan el poder de expresión.

Estereotipo

Las clases (y demás elementos notacionales en los diagramas) pueden estar clasificados de acuerdo a varios criterios, como por ejemplo su objetivo dentro de un programa. Esta clasificación adicional se expresa mediante un estereotipo.

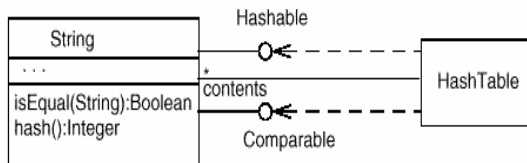
En el ejemplo anterior Auto3D está clasificado con el estereotipo *Mundo*, y la clase *Window* con el de interfaz. Nótese que las relaciones pueden tener esta clasificación también. En este caso la relación se identifica como *Observer*.



Interfaz

Una interfaz es un protocolo exigible a una clase. La representación de este concepto es por medio de una línea terminada en un círculo

En el ejemplo anterior, la clase *String* se utiliza dentro de un hashtable, gracias a que implementa la interfaz *Hashable* (el método *hash*) y la interfaz *Comparable* (el método *isEqual*).

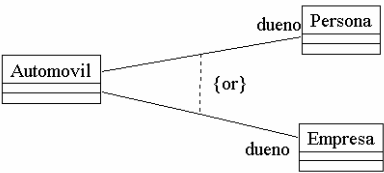


Asociación OR

En algunas ocasiones es necesario describir que una clase está relacionada con un objeto de una u otra clase. Esto se denota por medio de una relación *or exclusiva*. Su representación es una línea punteada que une dos asociaciones, junto con la aclaración (por medio de una propiedad) del tipo de asociación.

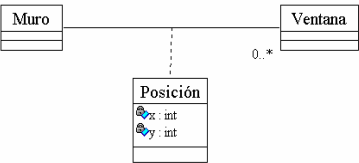
En el ejemplo anterior un automóvil puede tener como dueño una persona natural o una empresa (pero no ambos).

Clase de asociación



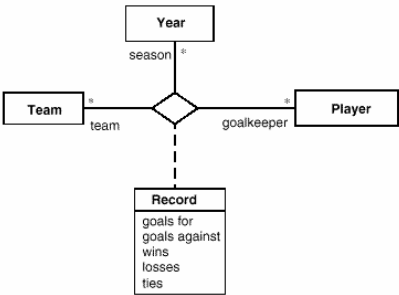
Una clase de asociación es información de detalle. Se denota como una clase relacionada por una línea punteada a una asociación.

En este caso existe una relación entre Muro y Ventana, la cual tiene como detalle un objeto de la clase Posición. Cabe notar que este objeto no podría tomarse como atributo de Muro o Ventana, ya que el contexto de su existencia esta dado precisamente por la relación entre las dos clases.



Asociación N-ARIA

Es una forma de expresar una relación entre tres o más clases. Se representa como un diamante del cual salen líneas de asociación a las clases.



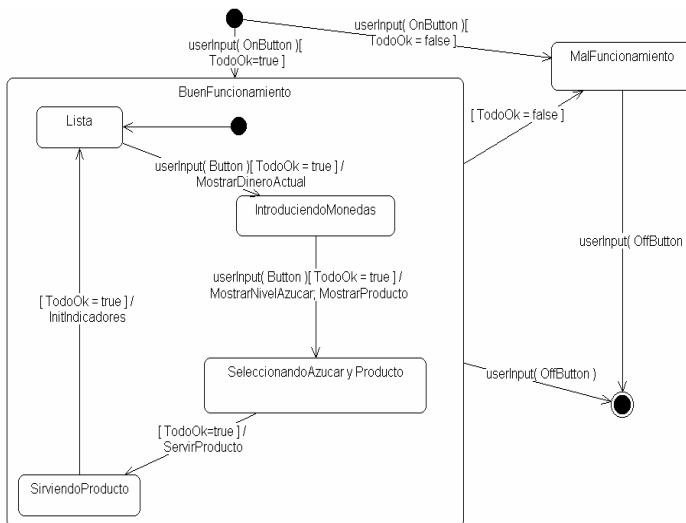
En este caso se tiene una relación ternaria entre las clases Year, Team y Player. A cada terna de objetos <año, equipo, jugador> corresponde un objeto de tipo Record (clase de asociación).

Otros conceptos

Aunque la metodología los incluye, se dejan fuera del alcance de este resumen los siguientes conceptos de UML: Calificador, Compartimiento de Nombre y de Lista, Propiedad, Expresión de Tipo, Elemento acotado, Tipo, Utilidad, Metaclase, Caminos de composición de clases, Relación de refinamiento, Elemento derivado y Expresión de navegación.

Conceptos básicos en un Diagrama de Estados

Muestra el conjunto de estados por los cuales pasa un objeto durante su vida en una aplicación, junto con los cambios que permiten pasar de un estado a otro. Un ejemplo en el caso de la cafetera son los estados posibles para la clase MaquinaCafe:



Estado

Identifica un periodo de tiempo del objeto (no instantáneo) en el cual el objeto esta esperando alguna operación, tiene cierto estado característico o puede recibir cierto tipo de estímulos. Se representa mediante un rectángulo con los bordes redondeados, que puede tener tres compartimentos: uno para el nombre, otro para el valor característico de los atributos del objeto en ese estado y otro para las acciones que se realizan al entrar, salir o estar en un estado (entry, exit o do, respectivamente). En el

caso del ejemplo anterior, se tienen cuatro estados (EnFuncionamiento, SinCambio, SinIngredientes, MalFuncionamiento) , en los cuales se desarrollan ciertas acciones al entrar; por ejemplo, al entrar al estado SinIngredientes se debe realizar la acción "Indicador SinIngredientes en On".

Se marcan también los estados iniciales y finales mediante los símbolos inicial y final , respectivamente.

Eventos

Es una ocurrencia que puede causar la transición de un estado a otro de un objeto. Esta ocurrencia puede ser una de varias cosas:

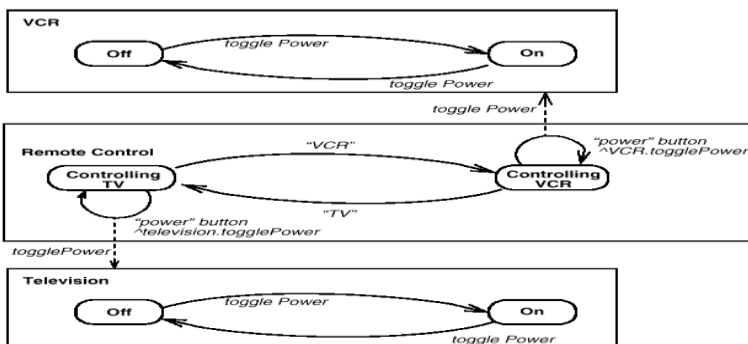
- Condición que toma el valor de verdadero o falso
- Recepción de una señal de otro objeto en el modelo
- Recepción de un mensaje
- Paso de cierto periodo de tiempo, después de entrar al estado o de cierta hora y fecha particular

El nombre de un evento tiene alcance dentro del paquete en el cual está definido, no es local a la clase que lo nombre.

En el caso del ejemplo anterior se encuentra nombrado en varias transiciones el evento userInput, que recibe como parámetro un Button , para indicar el botón que ha sido presionado por el usuario de la máquina de café.

Envío de mensajes

Además de mostrar y transición de estados por medio de eventos, puede representarse el momento en el cual se envían mensajes a otros objetos. Esto se realiza mediante una línea punteada dirigida al diagrama de estados del objeto receptor del mensaje. Si tomamos como ejemplo un control remoto que puede enviar órdenes de encender o apagar al televisor o a la videograbadora se puede obtener un diagrama de estados como el siguiente:



Los tres aparatos tienen diagramas de estados separados y algunas de las transiciones del control remoto causan el envío de mensajes (*togglePower*) a los otros aparatos.

Transición simple

Una transición simple es una relación entre dos estados que indica que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones, cuando un evento ocurre y si ciertas condiciones son satisfechas. Se representa como una línea sólida entre dos estados, que puede venir acompañada de un texto con el siguiente formato:

event-signature '[' guard-condition] '/' action-expression '^' send-clause

event-signature es la descripción del evento que da a lugar la transición, guard-condition son las condiciones adicionales al evento necesarias para que la transición ocurra, action-expression es un mensaje al objeto o a otro objeto que se ejecuta como resultado de la transición y el cambio de estado y send-clause son acciones adicionales que se ejecutan con el cambio de estado, por ejemplo, el envío de eventos a otros paquetes o clases.

En el caso del ejemplo inicial de esta hoja se tiene una transición entre los estados *IntroduciendoMoneda* y *SeleccionadoAzucaryProducto* que tiene una transición con el siguiente detalle:

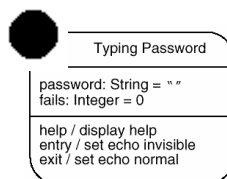
`userInput(Button) | [TodoOk=true] / MostrarNivelAzucar, MostrarProducto`

El evento que dispara el cambio de estado es `userInput(Button)`. Se requiere como condición adicional que no se haya detectado ninguna falla (`TodoOk = true`) y se ejecuta `MostrarNivelAzucar` y `MostrarProducto`, que deberían ser ejecutables por el objeto al cual pertenece el diagrama.

Transición interna

Es una transición que permanece en el mismo estado, en vez de involucrar dos estados distintos. Representa un evento que no causa cambio de estado. Se denota como una cadena adicional en el compartimiento de acciones del estado.

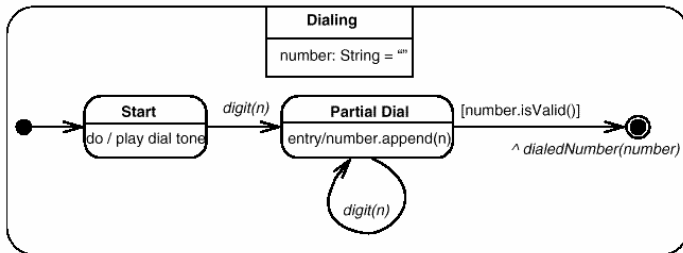
Supongamos el estado de una interfaz pidiendo password al usuario. En este caso puede tenerse una transición interna que muestre una ayuda al usuario. Esta transición se muestra en el siguiente diagrama con la cadena "help / display help " dentro del cuerpo del estado.



Conceptos avanzados en un Diagrama de Estados

Subestados

Un estado puede descomponerse en subestados, con transiciones entre ellos y conexiones al nivel superior. Las conexiones se ven al nivel inferior como estados de inicio o fin, los cuales se suponen conectados a las entradas y salidas del nivel inmediatamente superior. Un ejemplo es el estado *marcando* de un teléfono puede descomponerse en *Inicio* y *marcado parcial*, como lo muestra la siguiente figura.



Transición compleja

Representa la subdivisión en threads del control del objeto o una sincronización. Se representa como una línea vertical del cual salen o entran varias líneas de transición de estado. En el ejemplo que sigue se muestra una transición a dos threads concurrentes que luego se sincronizan.

Transición a estados anidados

Una transición de hacia un estado complejo (descrito mediante estados anidados) significa la entrada al estado inicial del subdiagrama. Las transiciones que salen del estado complejo se entienden como transiciones desde cada uno de los subestados hacia afuera (a cualquier nivel de profundidad).

En la hoja inicial de diagramas de secuencias se encuentran como primer ejemplo los dos casos nombrados: Desde el estado inicial se pasa al estado BuenFuncionamiento (a su estado inicial) y de este estado salen transiciones hacia MalFuncionamiento y hacia el estado final, que deben comprenderse como transiciones de cada uno de los estados internos hacia estos últimos.

Breve guía de ArgoUML

Introducción

ArgoUML, es una herramienta CASE creada para el análisis y diseño de sistemas orientado a objetos utilizando el Lenguaje Unificado de Modelado (UML). ArgoUML, soporta la versión 1.4 de UML y permite modelar los siguientes diagramas:

- Diagrama de clases.
- Diagramas de casos de uso.
- Diagramas de secuencia.
- Diagramas de colaboración.
- Diagramas de estado.
- Diagramas actividad.
- Diagramas de despliegue (Incluye el diagrama de componentes).

ArgoUML permite además la generación de código para Java, C++, C#, PHP4 y PHP5; e ingeniería inversa solo para JAVA.

Para mayor información visitar: <http://argouml.tigris.org/>

Instalación

Requerimientos mínimos para de instalación

- Un sistema operativo que soporte JAVA (Windows 200/XP, Linux).
- 10 MB de espacio en el disco duro.
- Mouse y teclado.
- Tener instalado JAVA Runtime Enviroment (JRE).

Instalación de JAVA Runtime Enviroment (JRE)

Para instalar JRE siga los siguientes pasos:

- Obtenga la última versión de Java Runtime Environment - JRE en: <http://java.com/es/download/manual.jsp>. Descargue el paquete de instalación fuera de línea.
- Una vez descargado, haga doble clic en el paquete de instalación. Siga con los pasos del asistente.
- na vez terminada la instalación, se recomienda reiniciar el sistema.

NOTA: Si ya cuenta con JRE no hay problema si realiza otra instalación, los archivos serán sobrescritos.

Instalación de ArgoUML

- Obtenga la última versión de ArgoUML en: <http://argouml-downloads.tigris.org/>
Ir a la sección "Releases" para elegir la versión y luego descargar los binarios (binary).
- Una vez descargado, descomprimir el archivo.

- En la carpeta descomprimida, localizar “argouml.jar”, hacer doble clic.
- Si todo es correcto, debe aparecer la pantalla de inicio de ArgoUML

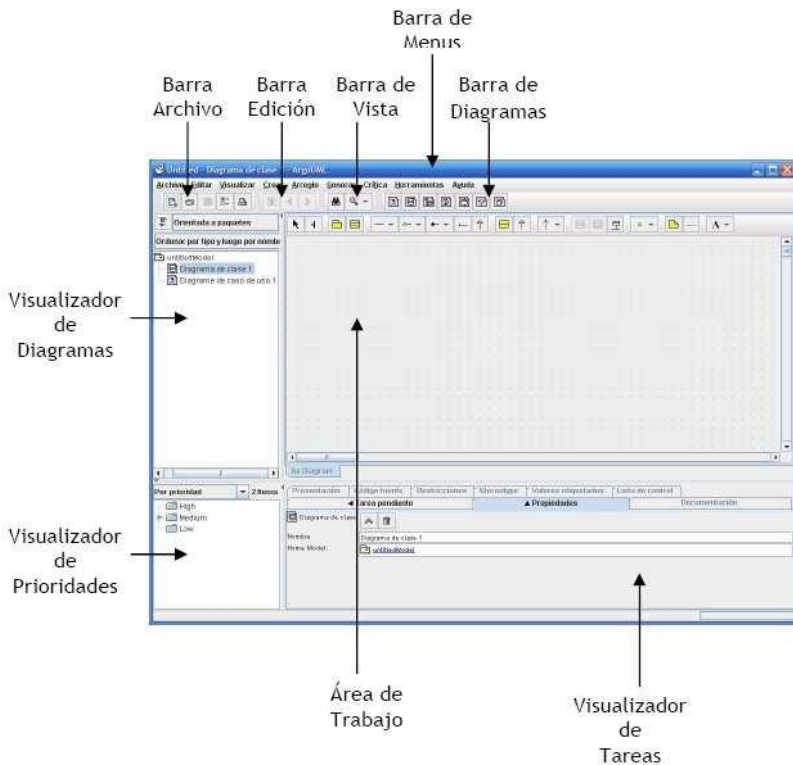


Pantalla de inicio de ArgoUML

NOTA: ArgoUML no posee instalador, funciona solo con los archivos incluidos en el paquete y los recursos de JRE.

Utilización de ArgoUML

Descripción de la ventana



- Barra de menús: Contiene los menús principales
- Barra archivo: Contiene las operaciones básicas para archivos
- Barra edición: Permite avanzar y retroceder en los diagramas también eliminar símbolos.
- Barra de vista: Permite aumentar o reducir la vista de el área de trabajo
- Barra de diagramas: Contiene todos los tipos de diagramas aceptados por el modelador
- Visualizador de prioridades
- Área de trabajo
- oVisualizador de tareas: Contienes todas las tareas referentes al diagrama que se esta realizando.
- Visualizador de Diagramas: Permite visualizar los diagramas que se van realizando.

Modelando en UML con ArgoUML
Consideraciones previas

Sobre la perspectiva: Para hacer más sencillo el manejo de los diagramas, se utilizara la vista orientada a diagramas. Para utilizarla, se debe elegir esta en las opciones del visualizador de diagramas.

Sobre los archivos: ArgoUML maneja tres extensiones de archivos: *.zargo, *.UML, *.zip. Cada archivo al guardarlo, puede contener todos los diagramas utilizados por el modelador. Se utilizara la extensión *.uml para guardar los diagramas, puesto que este tipo es capaz de ser leído por otros modeladores.

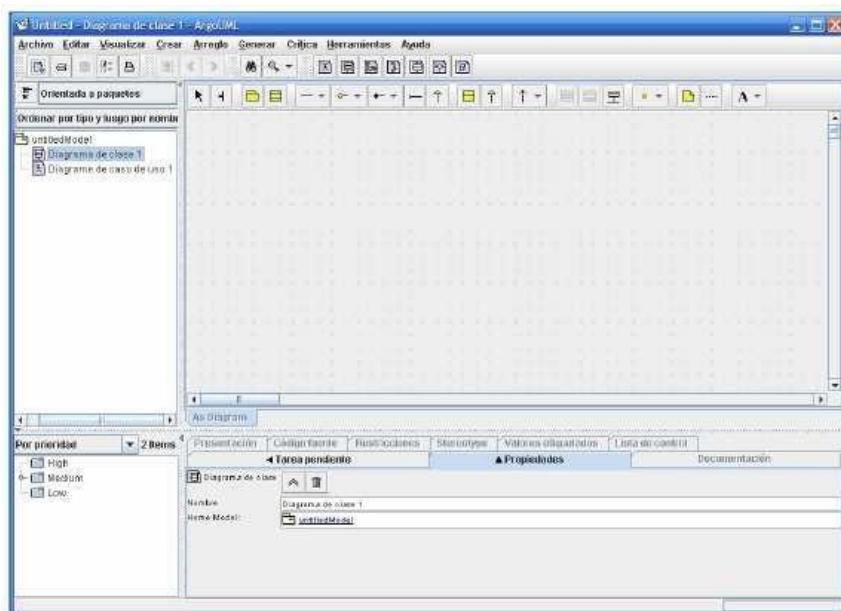
Sobre los graficos: ArgoUML permite guardar los diagramas realizados como imágenes en formato *.png, *.gif entre otros. Para guardar los gráficos como imágenes, dirigirse al menú archivo y hacer clic en la opción "Guardar los gráficos".

¿Como realizar diagramas de UML en ArgoUML?

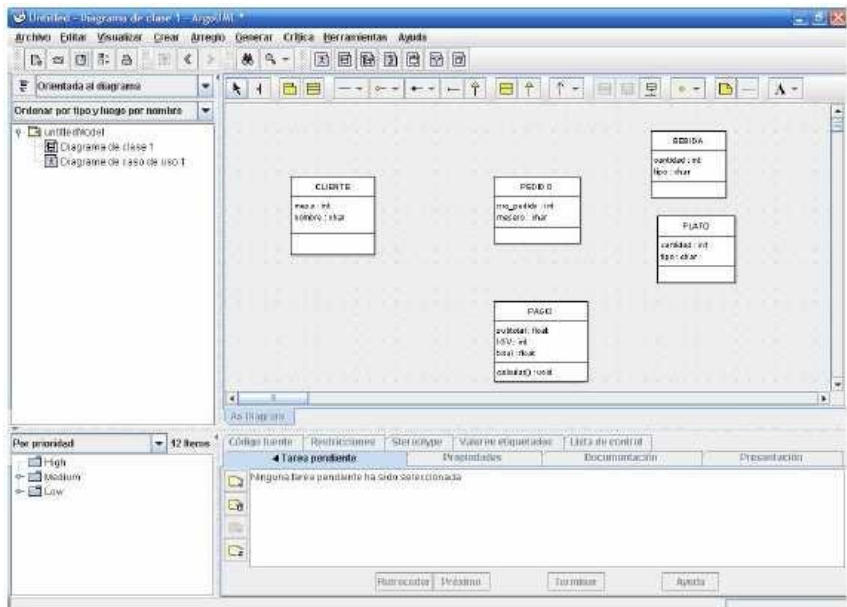
- Primero, hay que definir el diagrama en el cual se esta trabajando. En el visualizador de diagramas, por defecto, se establecen un diagrama de clases y un diagrama de casos de uso. Para empezar pueden seleccionarse los diagramas predefinidos, luego pueden agregarse mas según sea necesario seleccionando el que sea necesario de la barra de diagramas.
- Luego, en la parte superior del área de trabajo, se pueden observar una barra con todos los elementos que pueden ser usados en el diagrama seleccionado (clases, casos de uso, etc). Para utilizar cualquiera de ellos, solo basta con hacer clic y luego dirigirse a la zona inferior para hacer clic nuevamente para insertar el elemento. Para representar las relaciones entre ellos, primero hacer clic en el tipo de relación, luego hacer clic en el elemento de origen y arrastrar hasta elemento de destino. Si la relación que se desea graficar no es correcta, no se podrá visualizar en el diagrama.
- Por último, para editar las propiedades de algún elemento del diagrama (nombre, atributos, operaciones, etc.), hacer clic en el respectivo elemento, dirigirse al panel de tareas, elegir la pestaña propiedades y editarlos.

Ejemplo: Diagrama de clases

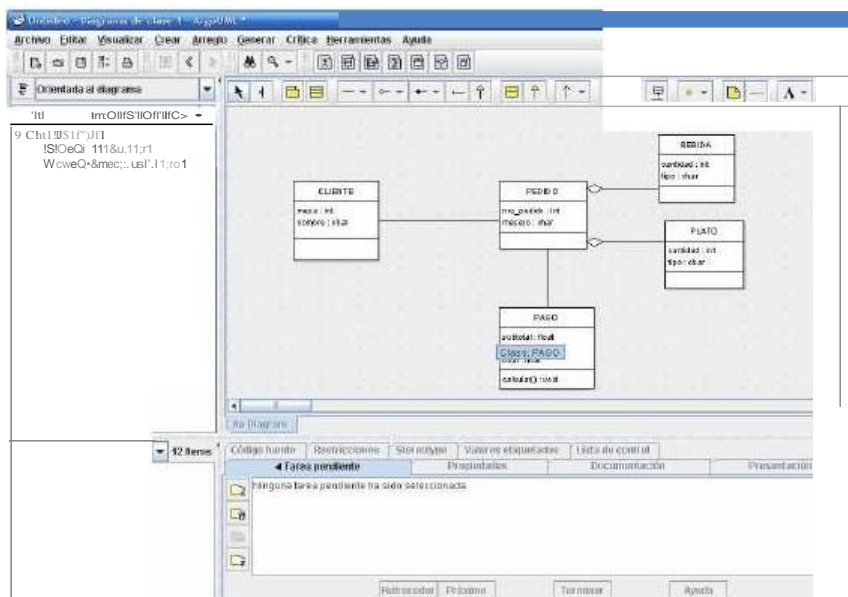
1ero: Elegimos es diagrama por defecto



2do: Colocamos las clases, arrastrándolas de la barra de elementos existente en el área de trabajo. Luego procedemos a colocar sus respectivos nombres, atributos y operaciones.



3ro: Por ultimo graficamos las dependencias. Primero elegimos el tipo de dependencia a graficar y luego arrastramos de una clase origen a una clase destino la dependencia.



Bibliografía

- [Booch, G. ; Jacobson, I. RumbaughT, J. 1999] ***El Lenguaje Unificado de Modelado***. España. Addison Wesley.
- [Booch, G. ; Jacobson, I. RumbaughT, J. 2000] ***El Lenguaje Unificado de Modelado. Manual de Referencia***. España. Addison Wesley.
- [Craing, L. 1999] ***UML Y PATRONES. Introducción al análisis y diseño orientado a objetos***. España. Pearson.
- [Fowler, M ; Scott. 1999] ***UML gota a gota***. España. Pearson.
- [Pressman, R. 1998] ***INGENIERÍA DEL SOFTWARE: Un enfoque práctico***. Cuarta edición. España. McGraw-Hill.