

ANÁLISIS DE REQUISITOS

3.1.- INTRODUCCIÓN AL ANALISIS DE REQUISITOS

Como se dijo en capítulos anteriores, el término **análisis** aplicado a **sistemas** significa descomponer el sistema en sus componentes para estudiar cada uno de ellos tanto como un ente aislado como en interacción con el resto de los componentes.

Para ser útil, al análisis debe seguir un proceso de **síntesis** que consistirá en unir los componentes del sistema para determinar cómo funcionan en conjunto.

Cuando se habla de una fase del ciclo de vida, el análisis consiste en producir un documento de especificación de requisitos que describa lo que el sistema debe hacer, pero no cómo hacerlo. No se trata pues de una actividad sólo de análisis, sino también de síntesis.

Se define el **análisis de requisitos** como "el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, de hardware o de software, así como el proceso de estudio y refinamiento de dichos requisitos" (Estándar IEEE Std. 610 [IEEE 1990]).

El **Requisito** es pues "una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado" (por ejemplo, poder listar rápidamente todos los clientes que deben dinero). Por extensión, el término *Requisito* se aplica también a "las condiciones que debe cumplir o poseer un sistema o uno de sus componentes para satisfacer un contrato, una norma o una especificación".

La definición de los requisitos en un proyecto debe ser fruto del trabajo conjunto de las partes involucradas en su desarrollo: Suministradores de software (analistas), clientes y usuarios. Ningún colectivo ante citado puede redactar la *Especificación de Requisitos Software* (ERS) ya que

- El cliente no suele conocer el proceso de diseño y desarrollo del software.
- Los analistas no entienden completamente el problema del cliente dado que no dominan su área de trabajo.

La fase de análisis de requisitos, según el estándar IEEE 1074 [IEEE, 1991] se desglosa en tres grandes actividades:

- **Definir los requisitos de software.** Tarea iterativa para crear una definición o especificación preliminar de los requisitos que debe cumplir el software a partir de la información obtenida mediante técnicas de recogida de información analizadas en el capítulo anterior.
- **Definir los requisitos de las interfaces del software con el resto del sistema y con el exterior.** Deben definirse las propiedades que se deben satisfacer para obtener una interacción eficaz con otros elementos del sistema (el usuario, el hardware, otras aplicaciones software, ...). En particular la interfaz con el usuario es crítica para la facilidad de uso (y por tanto el éxito) del software.

Los requisitos de interfaz con otras aplicaciones deben describir las características para que el software se relacione con ellas, las cuales pueden estar muy influenciadas por restricciones de trabajo del sistema (S.O. utilizado, SGBD empleado, Compiladores, controladores de red, etc.).

Así mismo deben definirse las características de las interrelaciones con elementos hardware.

- **Integrar los requisitos en un documento de especificación y asignarles prioridades.** La asignación de prioridades debe hacerse en función de su importancia o los beneficios que puede aportar su cumplimiento.

Otra manera de describir las actividades que se realizan en la fase de análisis de requisitos sería la siguiente (Raghavan, et al., 1995):

- **Extracción** o determinación de requisitos. Proceso mediante el cual los clientes o futuros usuarios del software descubren, revelan, articulan y comprenden los requisitos que desean.
- **Análisis de requisitos.** Proceso de razonamiento sobre los requisitos obtenidos en la etapa anterior, detectando y resolviendo posibles inconsistencias o conflictos, coordinando los requisitos relacionados entre sí, etc.
- **Especificación de requisitos.** Proceso de redacción o registro de los requisitos. Suele recurrirse a un lenguaje natural, lenguajes formales, modelos, gráficos, etc.
- **Validación de los requisitos.** Confirmación, por parte del usuario o el cliente de que los requisitos especificados son válidos, consistentes, completos, etc.

Aunque estas actividades no tienen por qué realizarse en secuencia, ya que hay muchas iteraciones y solapamientos entre ellas, sí marcan un proceso general para la fase de análisis.

3.2.- ESPECIFICACIÓN DE REQUISITOS DEL SOFTWARE

3.2-1.- Introducción

Según el estándar IEEE, 1990 se define:

Especificación: Documento que define, de forma completa, precisa y verificable, los requisitos, el diseño, el comportamiento u otras características de un sistema o de un componente de un sistema

Software: Conjunto de programas, procedimientos y documentación asociada a la operación de un sistema informático.

Con estas premisas puede definirse la *Especificación de Requisitos del Software (ERS)* como la documentación de los requisitos esenciales (funciones, rendimiento, diseño, restricciones y atributos) del software y de sus interfaces externas [IEEE,1990].

Las dos características fundamentales de una ERS eficaz son:

Incluir información veraz, es decir, coherente con las necesidades reales del usuario que se desean satisfacer.

Comunicar dicha información de forma veraz, es decir, de tal manera que se pueda comprender perfectamente

Las exigencias para una ERS conducen a no excederse a la hora de definirla y construirla, sino mas bien a abordar la descripción de lo que hay que desarrollar, no el cómo, el cuándo, etc. Se desarrolla el software. Esto implica:

Describir correctamente todos los requisitos de software sin incluir requisitos necesarios.

No describir ningún detalle de diseño de software, de su verificación, de la dirección del proyecto, excepto las restricciones impuestas al diseño que influyen en los requisitos.

3.2-2.- Características de una buena Especificación de Requisitos del Sistema (ERS)

Las características deseables para una buena ERS son las siguientes [IEEE 1984B]:

No ambigua. Cada requisito descrito debe tener una única interpretación.

Completa. Lo será si:

- Incluye todos los requisitos significativos del software
- Define la respuesta del software a todas las posibles clases de datos de entrada y en todas las posibles situaciones, tanto para los datos válidos como para los que no lo son
- Está conforme con el estándar de especificación que se deba cumplir
- Están etiquetadas y referenciadas en el texto todas las figuras, tablas y diagramas
- Si algún término está por determinar, se debe acompañar de una descripción de las condiciones que lo han causado y una posible descripción para eliminarlo

Fácil de verificar. Existencia de algún procedimiento finito y efectivo en coste para que se compruebe que el software satisface dicho requisito

Consistente. Lo será sí y sólo sí ningún conjunto de requisitos entran en conflicto entre ellos. Pueden darse tres tipos de conflictos:

- Dos o más requisitos pueden describir el mismo objeto real pero utilizan términos distintos para designarlo
- Las características especificadas de objetos reales pueden estar en conflicto
- Puede haber conflicto lógico o temporal entre dos acciones determinadas

Fácil de modificar. La estructura y el estilo de la ERS deben permitir que cualquier cambio necesario en los requisitos pueda realizarse de forma fácil, completa y consistente. Esto implica que la ERS debe:

- Tener una organización coherente y manejable (con una tabla de contenidos, un índice y referencias cruzadas)
- No ser redundante, es decir, el mismo requisito no debe aparecer en más de un lugar en la ERS

Facilidad para identificar el origen y las consecuencias de cada requisito (facilidad de traza). Se dice que una ERS facilita las referencias con otros productos del ciclo de vida si establece un origen claro para cada uno de los requisitos y si posibilita la referencia de estos requisitos en desarrollos futuros o en incrementos de la documentación.

Cuando un requisito de la ERS representa un desglose o una derivación de otro requisito, se debe facilitar tanto las *referencias hacia atrás* como las *referencias hacia delante* en el ciclo de vida. Estas últimas son especialmente importantes para el mantenimiento del software. Cuando el código o la documentación son modificados, es esencial poder comprobar el conjunto total de requisitos que pueden verse afectados por estas modificaciones.

Facilidad de utilización durante la fase de explotación y de mantenimiento. La ERS debe considerar las necesidades de mantenimiento, incluyendo una eventual sustitución del software, especialmente debido a:

- El personal que se encarga del mantenimiento no ha estado relacionado con el desarrollo del producto software.
- Gran parte de los conocimientos y de la información necesaria para el mantenimiento se dan por supuestos en la organización del desarrollo, pero suelen estar ausentes en la organización de mantenimiento.

3.2-3.- Evolución de las ERS

Normalmente, la ERS deberá ser cambiada a medida que progresa el producto software ya que es casi imposible especificar algunos detalles en el momento en el que se inicia el proyecto y es casi seguro que se realizarán cambios adicionales como consecuencia de haber encontrado deficiencias, defectos e inexactitudes que se descubren a medida que el producto evoluciona.

En este proceso deben tenerse en cuenta las consideraciones siguientes:

- El requisito debe ser especificado de la forma mas completa posible, aun en el caso en que se prevean de forma inevitable revisiones en el proceso de desarrollo.
- Debe iniciarse un proceso formal de cambio para identificar, controlar, seguir e informar de cambios proyectados tan pronto como sean identificados

Los cambios aprobados en los requisitos deben incluirse en la ERS de forma que permita:

- Suministrar una revisión precisa y completa del rastro de las modificaciones
- Permitir un examen de fragmentos actuales y reemplazados en la ERS.

3.2-4.- Estructura para las ERS

Un modelo propuesto por el estándar IEEE Std. 830 [IEEE, 1984b] es el que se presenta a continuación:

1.- Introducción
1.1 Objetivo
1.2 Ámbito
1.3 Definiciones, Siglas y Abreviaturas
1.4 Referencias
1.5 Visión Global
2.- Descripción General
2.1 Perspectiva del producto
2.2 Funciones del producto
2.3 Características del Usuario
2.4 Limitaciones Generales
2.5 Supuestos y dependencias
3.- Requisitos específicos
(ver tabla II)
Apéndices
Índice

Existen otras normas emitidas por otros organismos que también aportan esquemas para documentar las ERS (DOD, 1988, DORFMAN y THYER, 1990).

TABLA II	
3.-	Requisitos específicos
3.1	Requisitos funcionales
3.1.1	Requisito funcional 1
3.1.1.1	Introducción
3.1.1.2	Entradas
3.1.1.3	Procesamiento
3.1.1.4	Salidas
3.1.2	Requisito funcional 2

3.1.N	Requisito funcional N
3.2	Requisitos de interfaz externa
3.2.1	Interfaces de usuario
3.2.2	Interfaces de hardware
3.2.3	Interfaces de software
3.2.4	Interfaces de comunicaciones
3.3	Requisitos de ejecución
3.4	Restricciones de diseño
3.4.1	Acatamiento de estándares
3.4.2	Limitaciones hardware

3.5	Atributos de calidad
3.5.1	Seguridad
3.5.2	Mantenimiento

3.6	Otros requisitos
3.6.1	Base de datos
3.6.2	Operaciones
3.6.3	Adaptación de situación

3.2-5.- Especificación de requisitos de Interfaces

Las interfaces con el exterior coinciden con lo que tradicionalmente se ha llamado *entradas y salidas (E / S)* del sistema. En el caso del análisis estructurado, pueden identificarse fácilmente observando los flujos que entran y salen del sistema en el diagrama de contexto (del que se hablará posteriormente).

En el caso de las salidas puede hablarse de las pantallas de presentación de la información, listados o salida en papel, ficheros, etc. Las entradas serán pantallas de introducción de datos mediante teclado, introducción de datos mediante sensores, ficheros, etc.

La definición de las interfaces de E / S tiene como objetivo la estabilización del modo en que el sistema va a interactuar con el exterior del sistema.

3.3.- VISIÓN GENERAL DE LAS TÉCNICAS DE ESPECIFICACIÓN

Sobre la clasificación de técnicas de especificación no existe una regla general por lo que posiblemente, la forma mas lógica de hacerlo sea por orden alfabético. Sin embargo, pueden clasificarse las técnicas bajo dos enfoques diferentes:

Por la forma de representación (gráfica, textual, matricial, DFD, DD, etc.)

Por el enfoque de Modelización bajo los que se crean modelos del sistema relativos a su función, información y tiempo.

3.3-1.- Clasificación según la forma de representación

Gráficas. Utilizan iconos que representan un componente particular del modelo. Se usan cuando se quiere resaltar la conexión entre los distintos componentes del modelo.

Textuales. Se utilizan para especificar con mas detalle los componentes definidos en los gráficos mediante una gramática definida mas o menos formal.

Marcos o plantillas. ("Templates") especifican información relativa a un componente de un modelo que ha sido declarado en un diagrama o en otro marco. Se representan mediante un formulario que incluye todas sus características.

ENTIDAD: Estudiante
DESCRIPCIÓN: Cualquier persona que está matriculada en alguna de las asignaturas ofertadas.
ATRIBUTOS: Nº DNT apellido nombre dirección provincia teléfono
IDENTIFICADORES: Nº DNT
RESTRICCIONES:
VOLUMEN MÁXIMO DE OCURRENCIAS: 2000

En la figura se define el contenido de una entidad definida en un diagrama Entidad / Relación.

Matriciales. Son técnicas de comprobación entre modelos que permiten estudiar las referencias cruzadas entre sus componentes.

3.4.- MODELIZACIÓN DE FUNCIONES

3.4-1.- Diagramas de Flujo de Datos

Un DFD es un diagrama en forma de red que representa el flujo de los datos y las transformaciones que se aplican a ellos, al moverse desde la entrada hasta la salida del sistema. Se utiliza para modelizar las funciones del sistema y los datos que fluyen entre ellas a distintos niveles de abstracción.

El sistema se modelizará mediante un conjunto de DFD nivelados en el que los niveles superiores definen las funciones del sistema de forma general y los niveles inferiores definen estas funciones en niveles mas detallados.

3.4-1-1. - Componentes de un DFD

El DFD está compuesto por:

Procesos. Que son los componentes funcionales del sistema

Almacenes. Que representan los datos almacenados o en reposo.

Entidades externas. Que representan la fuente y/o el destino de la información del sistema.

Flujos de datos. Que representan los datos que fluyen entre las funciones.

PROCESOS (funciones o transformaciones)

Es el componente de un diagrama que representa una función que transforma flujos de datos de entrada en flujos de datos de salida.

Un proceso no es exactamente un programa en ejecución sino una función que tiene que realizar el sistema. Debe ser capaz de generar los flujos de datos de salida a partir de los flujos de datos de entrada más una información local (constante o variable) al proceso. (*Regla de conservación de datos*).




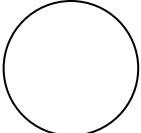

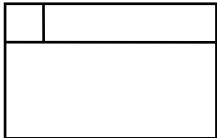
Si al proceso no le llegan todos los datos necesarios para obtener los datos de salida es por que existe un error de conservación de datos debido a que, por lo menos, no se han incluido ciertos datos de entrada.

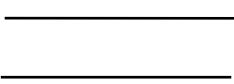
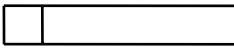
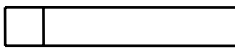

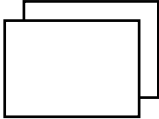

Análogamente, el fenómeno contrario o aquel en el que el flujo de datos o algún componente suyo muere dentro de un proceso y, consecuentemente, no se utiliza para generar los flujos de salida, se denomina *pérdida de información*.

La representación grafica de los procesos mas extendida es la de Yourdon, en la que se representan mediante un círculo que contiene en su interior un número y un nombre con las siguientes características:

- Ser lo mas representativo posible de la función que representa, englobando el contenido total de la función y no sólo parte de ella (Nombres como GESTIONAR ACCION, REALIZAR OPERACION, etc. son poco representativos).
- Ser breve, normalmente formado por un verbo seguido de un sustantivo.
- El nombre y el número del proceso deben ser únicos en el conjunto de DFD que representan el sistema

Los símbolos utilizados en las distintas notaciones de Diagramas de Flujos de Datos (DFD) son :

	YOURDON, DeMARCO	GANE y SARSON	SSADM, METRICA
Flujos de Datos			
Procesos			

Almacenes de Datos			
Entidades Externas			

ALMACENES DE DATOS

El almacén de datos representa la información del sistema almacenada de forma temporal. Si los flujos de datos representan datos en movimiento, los almacenes de datos representan datos en reposo. El almacén es, por tanto, un depósito lógico de almacenamiento y puede representar cualquier dato temporalmente almacenado, independientemente del dispositivo utilizado.

- Los almacenes se representan por dos líneas paralelas y tienen las siguientes características:
- Todos los almacenes de datos deberán llevar un nombre que debe ser lo más representativo posible de los datos que contienen y no debe estar asociado a connotaciones físicas.
- Un almacén de datos se puede representar varias veces en un DFD si con ello mejora su legibilidad.
- Dentro de un conjunto de DFD nivelados, el almacén se situará en el nivel más alto de los que sirven de interconexión entre dos o más procesos en el que se representan todos sus accesos y además se representará en los niveles inferiores.
- Si en un DFD hay un almacén que sólo tiene conexión con un proceso, se dice que el *almacén es local al proceso* y, por tanto, no debe aparecer en ese nivel. Dicho almacén se representará en el DFD en el que se especifique dicho proceso.
- Un almacén se dice que tiene *estructura simple* cuando es de tipo registro, esto es, formado por una sucesión de atributos en el que uno o varios de ellos identifican cada ocurrencia del almacén. El contenido de los almacenes se define en el *Diccionario de datos*.
- El contenido de un almacén con estructura más compleja se puede representar mediante un diagrama entidad / relación.

ENTIDADES EXTERNAS (Terminadores)

Una entidad externa es el componente del DFD que representa un generador o un consumidor de información del sistema y que no pertenece al mismo. Puede representar un subsistema, persona, departamento, organización etc. que proporcione datos al sistema o los reciba de él.

Tienen las siguientes características:

- Son externas al sistema que se está modelizando. Los flujos que parten o llegan a ellas definen la interfaz entre el sistema y el mundo exterior.
- Las relaciones que hay entre las entidades externas no son objeto de estudio del modelo.
- Se representa mediante un cuadrado con un nombre en su interior que debe ser representativo y pueden aparecer varias veces en un DFD con objeto de mejorar la legibilidad del mismo. En este caso suelen marcarse las entidades duplicadas con un

asterisco. En el caso de que exista una interfaz entre el sistema y una entidad externa que tiene varias ocurrencias, se representa con un doble cuadrado superpuesto.

- Normalmente, las entidades externas sólo aparecerán en el DFD de mayor nivel, que se llama **Diagrama de contexto**. No obstante esto es opcional por lo que se pueden incluir en otros diagramas de nivel inferior.

FLUJO DE DATOS

Se define como el camino a través del cual viajan datos de composición conocida de una parte del sistema a otra. Representan los datos en movimiento y con una cardinalidad determinados.

Los flujos de datos son el medio de conexión de los restantes componentes del DFD. Según la persistencia en el tiempo de los datos que fluyen, éstos pueden ser:

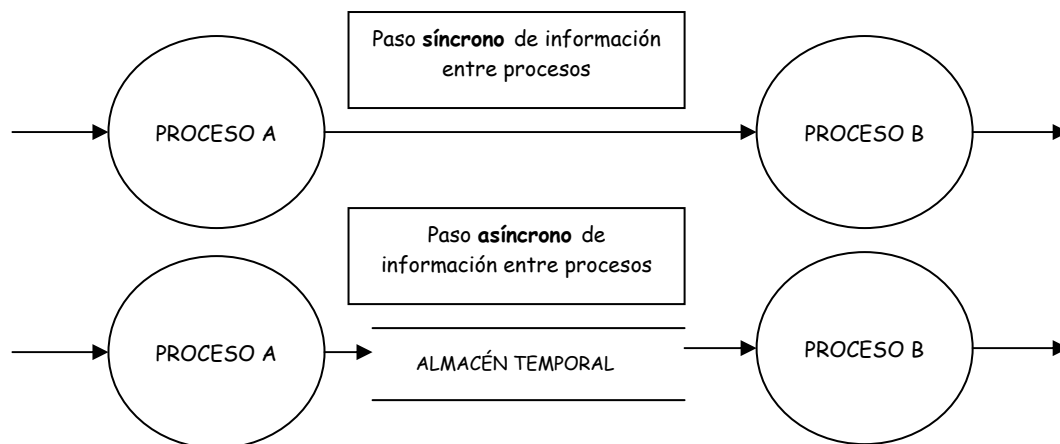
Flujo de datos discreto (—————>) Representan datos en movimiento en un momento determinado.

Flujo de datos continuos (—————>) Representan flujos de datos persistentes en el tiempo.

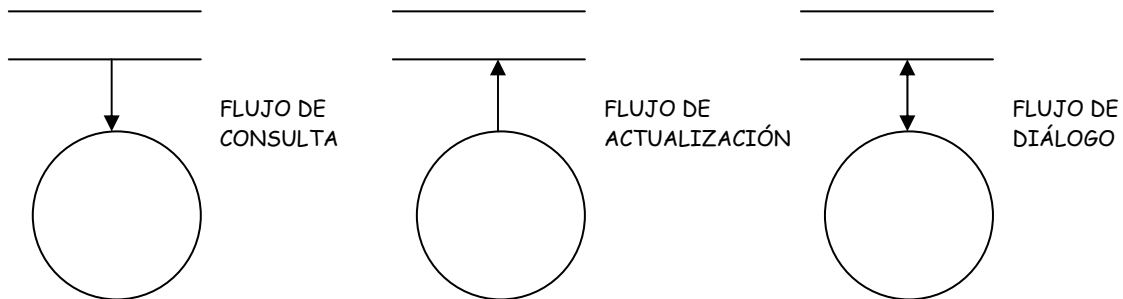
Para realizar la conexión de entre los componentes de un DFD por medio de flujos de datos, hay una serie de restricciones. En la siguiente tabla se establecen las conexiones que están permitidas:

Destino Fuente	PROCESO	ALMACÉN	ENTIDAD EXTERNA
EXTERNA			
PROCESO	SI	SI	SI
ALMACÉN	SI	NO	NO*
ENTIDAD EXTERNA	SI	NO*	NO

La conexión directa entre dos procesos mediante un flujo de datos es posible siempre y cuando la *información sea síncrona*, esto es, que el proceso de destino comienza en el momento en el que el proceso origen finaliza su función. Si no ocurre así es necesario que exista un almacén temporal que guarde los datos del proceso origen, de forma que el proceso destino pueda capturar estos datos cuando los necesite.



Las diferentes conexiones que pueden hacerse entre procesos y almacenes están representadas en la figura siguiente:



El **Flujo de Consulta** muestra la utilización del almacén por el proceso para una de las siguientes acciones:

- Utilizar los valores de uno o más atributos de una ocurrencia de un almacén.
- Comprobar si los valores de los atributos seleccionados cumplen unos determinados criterios.

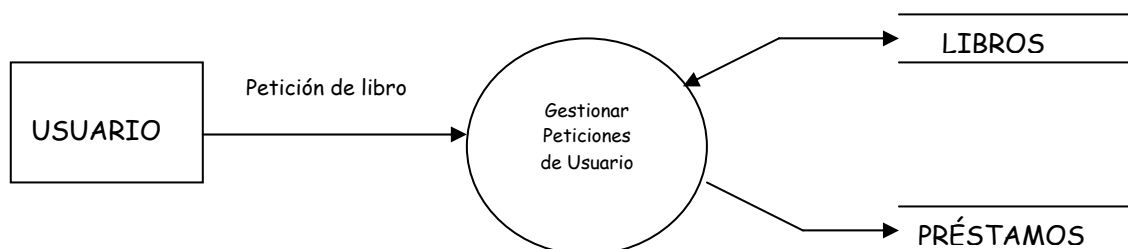
El **Flujo de Actualización** indica el proceso que va a alterar la información mantenida en el almacén para:

- Crear una nueva ocurrencia de una entidad o una relación existente en el almacén.
- Borrar una o más ocurrencias de una entidad o una relación
- Modificar el valor de un atributo

El **Flujo de Diálogo** entre un proceso y un almacén implica, como mínimo, un flujo de consulta y otro de actualización.

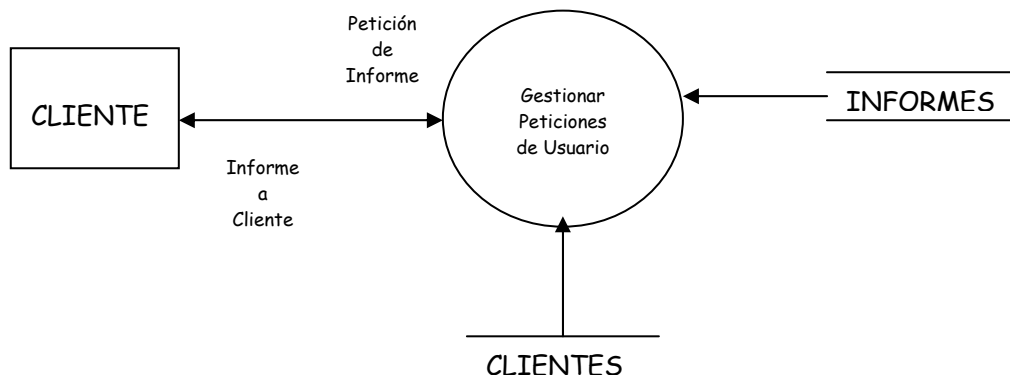
Cualquier actualización lleva asociada una lectura implícita de la ocurrencia sobre la que se va a realizar. Esta lectura supone un flujo de consulta por lo que existe un flujo de diálogo.

Un ejemplo de flujo de diálogo es el de la figura es la petición de un libro prestado en una biblioteca:



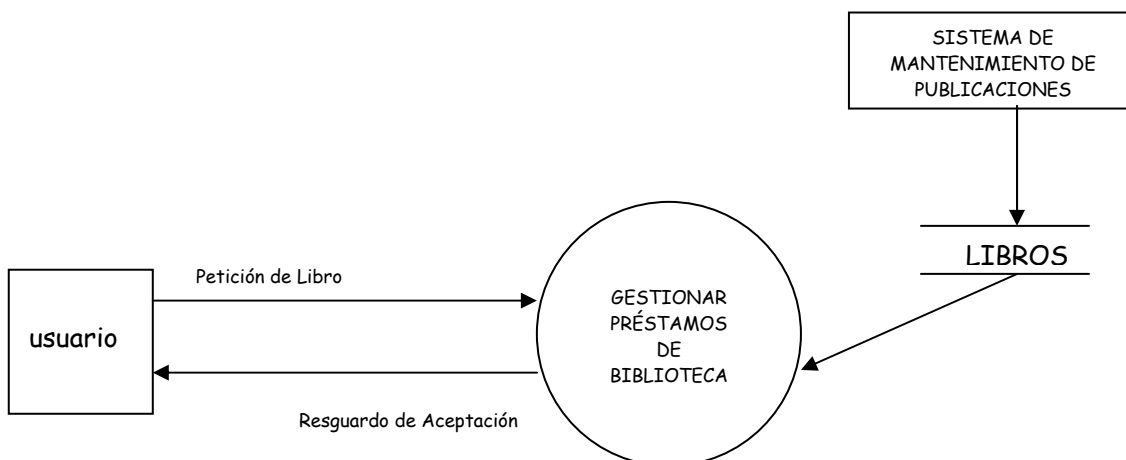
El flujo de diálogo puede aparecer también en un DFD para enfocarse en la relación existente entre dos flujos de datos (denominándose entonces el flujo *par de diálogo*). Un

ejemplo es la petición de un informe por un cliente que lleva asociada la entrega de dicho informe:



Existen dos casos particulares (que en la tabla anterior se marcaron con un asterisco) correspondientes a la conexión entre una entidad externa y un almacén y viceversa. En este caso, la conexión puede ser posible con almacenes externos que sirven de interfaz entre el sistema y una entidad externa.

Un ejemplo es el de la figura en el que se considera la gestión de préstamos de una biblioteca, utilizando un almacén, llamado LIBROS, que es utilizado por otro sistema llamado SISTEMA DE MANTENIMIENTO DE PUBLICACIONES:



Como resumen, los flujos de datos deben tener las siguientes características:

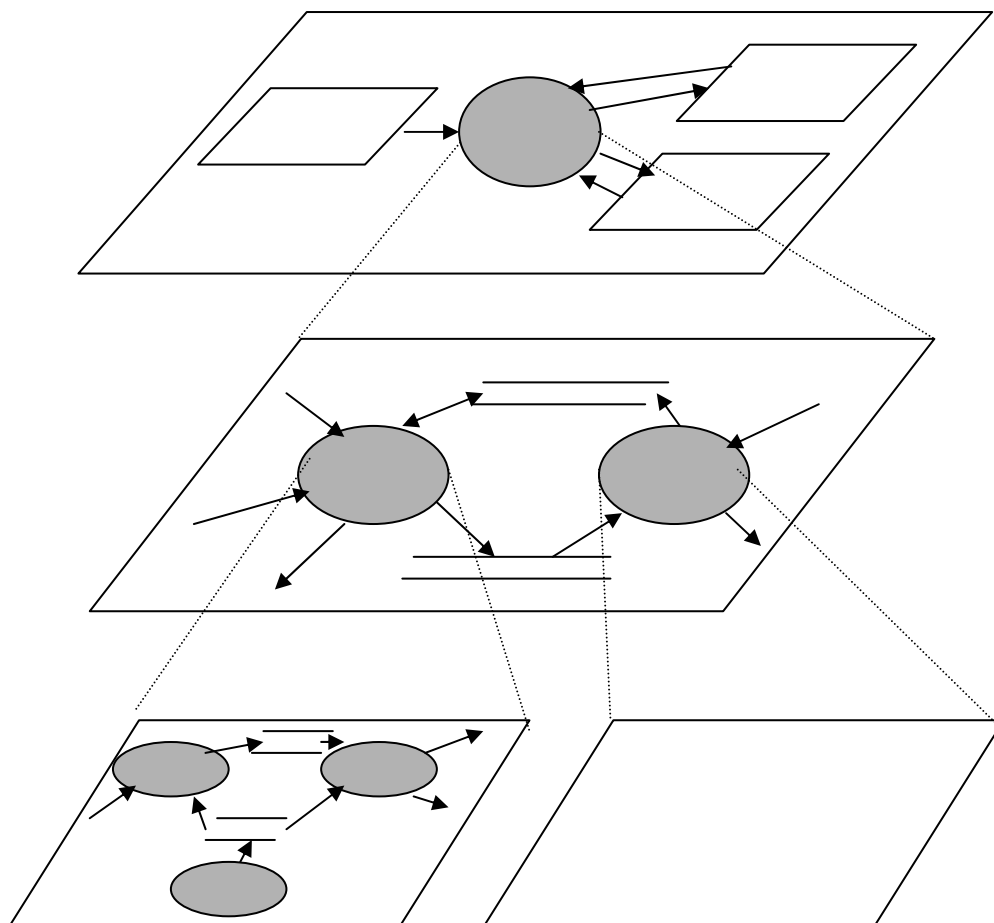
- Deben tener un **nombre** representativo del contenido de la información que fluye a través de él, englobando toda la información del flujo y no sólo parte del mismo. Únicamente los flujos que entren o salgan de almacenes que tengan una estructura simple no tienen nombre, dado que su estructura es la misma que la del almacén.
- Si los datos que viajan por un flujo de datos tienen distintos propósitos o no viajan juntos, entonces estos datos no constituyen uno sino dos o más flujos de datos.
- El contenido de un flujo de datos puede ser de varios tipos:
 - **Elemento:** Es un flujo de datos que contiene un *dato elemental* (campo o atributo), es decir, una pieza insoluble de datos.
 - **Grupo:** Es un flujo de datos discreto que contiene varios elementos de datos.

- **Par de Diálogo:** Sólo puede especificarse este tipo de flujo mediante uno de doble flecha en el que se incluyen dos nombres, uno es el iniciador y el otro indica la respuesta asociada al primero.
- **Múltiple:** Es el flujo formado por un conjunto de flujos de datos. En el DFD se representa como un flujo de datos simple.
- Un flujo de datos se puede desdoblar en varios flujos de datos (cada uno con los mismos datos) o puede aparecer varias veces en un DFD. Asimismo, varios flujos de datos pueden unirse en uno sólo. Lo que no se permite es separar el contenido de un flujo de datos en el mismo diagrama.

Los flujos de datos no indican el control de la ejecución de los procesos ni cuando va a comenzar o terminar de realizarse un proceso.

3.4-1-2.- Descomposición en niveles de un Diagrama de Flujo de Datos

Como el modelo de un sistema grande no puede representarse en una sola página mediante un DFD, suele representarse por "capas" quedando cada capa definida mediante un DFD siguiendo un criterio de aproximación descendente (top - down) en el que cada nivel proporciona una visión mas detallada de una parte definida en el nivel anterior. Las ventajas de este tipo de estudio son:



- Ayuda a construir la especificación de arriba abajo. Al comenzar el modelado del sistema no se conocen todos los detalles, sino sólo los aspectos mas generales. Estos representan los diagramas superiores y, a medida que se va recopilando información ésta se va incluyendo en los niveles inferiores.

- Los niveles pueden ir dirigidos a personas diferentes
- Facilita el trabajo de los analistas que pueden trabajar paralelamente modelando funciones independientes del sistema
- Facilita la documentación del sistema , ya que cada diagrama puede escribirse en cada página.

El nivel mas alto de la jerarquía se comienza mediante un DFD denominado **Diagrama de Contexto** en el que solo hay un proceso que representa al sistema completo. El siguiente nivel representa la descomposición de este proceso en otro DFD (llamado **Diagrama del Sistema**) que representa las funciones principales o subsistemas. Posteriormente se descomponen cada uno de los procesos en nuevos diagramas que representan funciones mas simples, procediendo de esta forma hasta que todas las funciones estén lo "*suficientemente detalladas*" como para que no sea ya necesaria la definición de nuevos DFD.

En resumen, un conjunto de DFD queda definido por:

- ❖ Un **Diagrama de Contexto**, único y en la parte superior.
- ❖ Uno o varios **Niveles Medios** formados por el resto de los diagramas
- ❖ Un conjunto de **Funciones Primitivas** que están presentes tanto en los niveles intermedios como en los últimos niveles de la jerarquía, que se corresponden con procesos que no se explotan en nuevos DFD.

Diagrama de Contexto

El Diagrama de contexto o "*Diagrama de Nivel 0*" tiene por objeto delimitar la frontera entre el sistema con el mundo exterior y definir sus interfaces, esto es, los flujos de datos de entrada y salida con el entorno, o, lo que es lo mismo, el contexto.

Está formado por un único proceso que representa la "*caja negra*" del sistema completo, un conjunto de entidades externas que representan la procedencia y destino de la información del sistema y un conjunto de flujos de datos que representan los caminos por los que fluye dicha información.

Es necesario que en el diagrama de contexto estén representadas todas las entidades externas e, incluso, según algunos autores, es el único diagrama en el que pueden aparecer, si bien puede ser conveniente que se incluyan en los niveles inferiores para ayudar a la comprensión de los mismos.

Niveles Medios

El diagrama sobre el que se descompone el diagrama de contexto se denomina **Diagrama del Sistema** o **Diagrama 0** y en él se representan las funciones principales que debe realizar, así como las relaciones entre ellas (Principales interfaces entre estas funciones).

Es conveniente que las funciones de este diagrama sean conceptualmente diferentes entre sí, lo que facilita la descomposición de cada una para su desarrollo por analistas diferentes.

El proceso, que corresponde a la definición del proceso 0 suele denominársele también como **Diagrama de Nivel 1**.

Procesos Primitivos

Los procesos o funciones primitivas son aquellos procesos de un DFD que ya no se descomponen en más diagramas de nivel inferior. Por cada función primitiva habrá una especificación que la describa.

En general, un proceso no se debe descomponer más cuando:

- ❖ Se puede especificar en menos de una página mediante un lenguaje de especificación o *pseudo código*
- ❖ Los procesos del diagrama tienen pocos flujos de datos de entrada y salida.
- ❖ Al descomponer una función de un nivel determinado, se pierde significado de lo que tiene que hacer dicha función.

La metodología MÉTRICA versión 2 recomienda realizar sólo cuatro niveles de descomposición:

- **Nivel 0:** Diagrama de contexto
- **Nivel 1:** Subsistemas.
- **Nivel 2:** Funciones de cada subsistema.
- **Nivel 3:** Subfunciones asociadas a cada uno de los eventos del sistema.
- **Nivel 4:** Procesos necesarios para el tratamiento de cada subfunción.

Es necesario comprobar la consistencia entre los distintos niveles de DFD, esto es, que la información que entra y sale de un proceso de nivel N sea consistente con la información que entra y sale del DFD en el que se descompone. Para ello se sigue la **Regla de Balanceo** que se enuncia:

- ❖ Todos los flujos de datos que entran en un diagrama hijo deben estar representados en el padre por el mismo flujo de datos entrando en el proceso asociado
- ❖ Las salidas del diagrama hijo deben ser las mismas salidas del proceso padre asociado con una excepción: los *rechazos triviales* (camino de rechazo que no requieren ninguna revisión de la información establecida) no necesitan estar balanceados entre padre e hijo.

Convenciones a la numeración

La convención de la numeración de diagramas y procesos es la siguiente:

- ⊗ Cada diagrama recibe el número y el nombre del proceso que descompone (proceso padre)
- ⊗ El proceso del diagrama de contexto siempre es numerado como cero.
- ⊗ Los procesos del diagrama del sistema se enumeran con un entero, comenzando por 1 y de forma creciente hasta completar el número de procesos del diagrama
- ⊗ En los restantes niveles, los números de los procesos están formados por la concatenación del número de diagrama en el que están, más un punto y un número entero único para identificarlo dentro del diagrama (En particular, la MÉTRICA versión 2 recomienda seguir la notación AXX.XX donde AA son las letras para identificar la aplicación y XX.XX son dígitos que identifican el nivel y el proceso dentro del mismo).

3.4-2.- Diccionario de datos

Se define el **Diccionario de Datos (DD)** como una lista organizada de los datos utilizados por el sistema que gráficamente se encuentran representados por los flujos de datos y almacenes presentes sobre el conjunto de DFD.

El DD se crea a la vez que los DFD durante el análisis del sistema. Las entradas son realizadas cada vez que se identifica un elemento (flujo de datos, almacenes y datos elementales).

Estas entradas deben ser únicas para cada componente del DFD, esto es, habrá una entrada en el DD por cada flujo de datos que aparezca en el conjunto de DFD.

3.4-2-1.- Definición del flujo de datos

La definición de flujos de datos sigue una aproximación *top-down*. Las componentes son definidas, a su vez, mediante componentes mas detalladas, sucesivamente hasta obtener partes indivisibles, es decir, datos elementales (atributos o campos).

Así si un flujo de datos A está compuesto por un flujo B y un flujo C y B está compuesto por los flujos B₁, B₂ y B₃ mientras que el flujo C es siempre C₁ y C₂ puede escribirse:

$$A = B_1 + B_2 + B_3 + C_1 + C_2$$

Pero quizá sea mejor definir los flujos en función de sus componentes subordinados: Estos componentes serán las entradas al DD:

$$\begin{aligned} A &= B + C \\ B &= B_1 + B_2 + B_3 \\ C &= C_1 + C_2 \end{aligned}$$

Un flujo de datos, ya sea un grupo o flujo múltiple, se puede definir teóricamente mediante la inclusión, selección e iteración de sus componentes. Además se incluyen otros símbolos que aportan mas significado a cada entrada del DD:

SÍMBOLO	SIGNIFICADO
=	Composición: Está compuesto de, es equivalente a, ...
+	Inclusión: y
[]	Selección: Selección de una de las opciones encerradas entre corchetes, y separadas por el símbolo " "
{ }	Iteración: Iteraciones del componente encerrado entre llaves
()	Opción: Significa que el componente encerrado es opcional (puede estar presente o ausente)
* Texto *	Comentario: El texto entre asteriscos es un comentario aclarativo de una entrada del DD
@	Identificador: Se utiliza para señalar un campo o conjunto de campos que identifican cada ocurrencia de un almacén

Composición e Inclusión

Se utiliza el símbolo "=". Si un flujo múltiple está compuesto por un flujo B y uno C la definición se representa:

$$A = B + C$$

E indica que "A está compuesto de B y de C, es decir, a cada ocurrencia de A le corresponde una ocurrencia de B y una de C". Por ejemplo:

$$\text{PETICIÓN LIBROS} = \text{CARNET BIBLIOTECA} + \text{FICHA LIBROS}$$

A su vez:

CARNET BIBLIOTECA = NUM.CARNET + APELLIDOS + NOMBRE + TIPO CARNET

Selección

La selección de un componente (sea un elemento o un conjunto de elementos) se representa entre corchetes separando cada opción con el símbolo "|":

TIPO CARNET = [ALUMNO|COLABORADOR|PROFESOR]

Iteración

Representa la repetición de los elementos incluidos entre paréntesis

En el ejemplo seguido, la ficha de libros está compuesta por una estructura repetitiva de libros que considera el ISBN, el título y el autor. La definición queda:

FICHA LIBROS = {LIBROS}

LIBROS = ISBN + TÍTULO + AUTOR

Pueden representarse los límites inferior y superior sobre las ocurrencias de una estructura repetitiva. Así, si el préstamo está restringido a 5 libros puede representarse:

FICHA LIBROS = 1{LIBROS}5

Opción

Un dato opcional indica que puede o no estar presente. Así si el carnet de biblioteca puede opcionalmente recoger el número de teléfono. Se representa:

CARNET BIBLIOTECA = NÚM CARNET + APELLIDOS + NOMBRE + TIPO CARNET + (TELÉFONO)

3.4-2-2.- Definición de almacenes

Los almacenes se definen como entidades repetitivas de datos y/o grupos de datos. Se selecciona uno o más datos para organizar la colección de entradas en un almacén que se denomina **identificador**. Un ejemplo para un almacén de libros disponibles es:

LIBROS DISPONIBLES = @ISBN + TITULO + AUTOR + NUM UNIDADES

El isbn identifica cada ocurrencia del almacén

3.4-2-3.- Alias

A veces, al modelar un sistema, hay datos que se nombran de distinta forma y, en realidad, son el mismo dato. Esto se puede definir en el DD creando un **alias** o sinónimo de una entrada del DD, ya se trate de un flujo de datos o de un elemento de datos. No es aconsejable su utilización pero se utiliza con frecuencia.

Como es muy difícil eliminar completamente todos los alias existentes en un sistema, es preciso documentarlos en el DD aunque aumente la redundancia. Así si la petición de libros es sinónimo de la petición de préstamo, debe incluirse una nueva entrada en el DD que indique:

PETICIÓN LIBRO = CARNET BIBLIOTECA + FICHA LIBROS

PETICIÓN LIBRO = PETICIÓN PRÉSTAMO

3.4-3.- Especificación de procesos

La especificación de proceso, también llamada **mini especificación** es la técnica que define el procedimiento que realiza un proceso primitivo. Debe describir de una manera

mas o menos formal cómo se obtienen los flujos de datos de salida a partir de los flujos de datos de entrada, mas la información local del proceso.

Las posibles alternativas para describir el procedimiento son:

- ✿ **Lenguaje estructurado**
- ✿ **Árboles de Decisión**
- ✿ **Tablas de Decisión**
- ✿ **Diagramas de Acción**

3.4-3-1.- Lenguaje estructurado

Es un lenguaje formado por un subconjunto de palabras de las que se utilizan para formar las construcciones propias de la programación estructurada (*secuencia, repetitiva, alternativa*) y otras incluyen un conjunto de verbos (LEER, ESCRIBIR, BORRAR, ENCONTRAR, CALCULAR, VALIDAR; etc.) que reflejan acciones simples.

Alternativa	SI condición Bloque SI NO Bloque FIN SI
Repetitiva	MIENTRAS Condición Bloque FIN MIENTRAS ----- REPETIR Bloque HASTA Condición
Secuencia	Está formada por un conjunto de sentencias (Bloque) y cada una de ellas puede ser una acción sencilla o una estructura de las anteriores

3.4-3-2.- Árboles de decisión

El Árbol de decisión es "un modelo de una función discreta en la que se determina un valor de una variable y, en función de ese valor, se lleva a cabo una acción".

Se representa en forma de árbol que representa los posibles valores de las variables y las acciones tomadas para cada valor, así como ,el orden en que se realiza la decisión.

Se utiliza cuando el número de condiciones no es muy alto, en otro caso es mejor utilizar una tabla de decisión.

3.4-3-3.- Tablas de decisión

Es un modelo alternativo que muestra la función en forma tabular o matricial. Para ello se define la **parte de condición**, formada por un conjunto de condiciones y entradas de condiciones, y la **parte de acción**, formada por un conjunto de acciones y entradas de acciones.

3.4-3-4.- Diagramas de acción

Es una técnica de especificación que utiliza niveles anidados de corchetes que representan la estructura lógica utilizada para transformar los datos de entrada en los datos de salida.

3.4-3-5. - Precondiciones y postcondiciones

Se centran más en la relación que deben tener las entradas y las salidas del sistema que en su algoritmo. Por un lado indican las condiciones que se tienen que cumplir en el proceso para comenzar (precondiciones) así como las condiciones que han de cumplirse cuando el proceso ha concluido (postcondiciones).

3.4-3-6. - Ejemplo de aplicación

Se analiza la política de descuentos que realiza una empresa sobre los pedidos de los clientes dependiendo del volumen de compras del año anterior. Si se trata de clientes de más de 5 años de antigüedad se le aplica un descuento de un 25 % si el valor de los pedidos anuales es superior a 30.000 €. Si el montante de los pedidos se encuentra entre los 15.000 y los 30.000 € el descuento aplicado será del 15 % y si no se alcanza el valor de 15.000 € se aplicará un 10 %.

Para clientes entre 3 y 5 años de antigüedad se aplicará un 11 % para compras superiores a 25.000 € y el 5% por valor igual o inferior.

Si tienen menos años de antigüedad se aplicará el 9% si el valor de las compras es superior a 25.000 €.

A los clientes clasificados como especiales se les aplicará un descuento del 25% si el volumen de compras supera 30.000 € o del 20% en caso contrario.

Árbol de Decisión

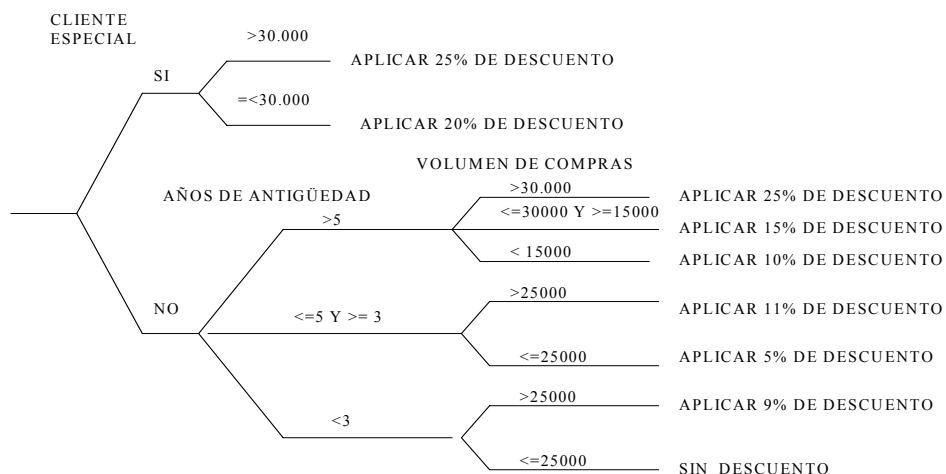


Tabla de Decisión

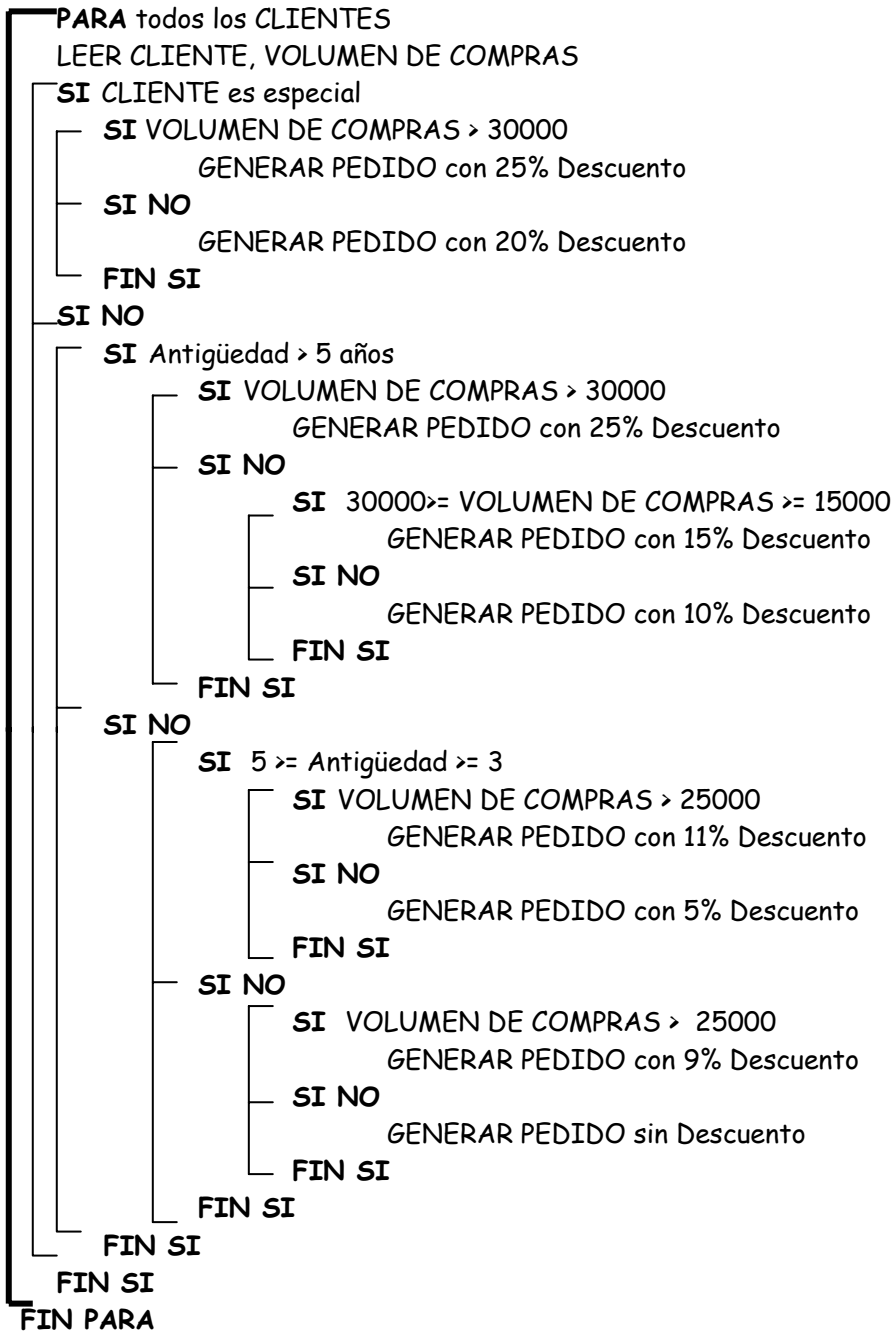
Condiciones

Cliente especial	Si	Si	No	No	No	No	No	No	No
Compras > 30.000	Si	-	Si	-	-	-	-	-	-
Compras <= 30.000	-	Si	-	No	-	-	-	-	-
30.000 >= Compras >= 15.000	-	-	-	Si	-	-	-	-	-
Compras < 15.000	-	-	-	-	Si	-	-	-	-
Compras > 25.000	-	-	-	-	-	Si	-	Si	-
Compras <= 25.000	-	-	-	-	-	-	Si	-	Si
Antigüedad > 5 años	-	-	Si	Si	Si	-	-	-	-
5 años >= Antigüedad >= 3 años	-	-	-	-	-	Si	Si	-	-
Antigüedad < 3 años	-	-	-	-	-	-	-	Si	si

Acciones

Aplicar 25% de descuento	X		X						
Aplicar 20% de descuento		X							
Aplicar 15% de descuento				X					
Aplicar 11% de descuento						X			
Aplicar 10% de descuento					X				
Aplicar 9% de descuento								X	
Aplicar 5% de descuento							X		
Sin descuento									X

Diagrama de Acción



3.4-4.- Diagramas de descomposición funcional

Los diagramas de descomposición funcional (DDF) son técnicas que utilizan determinadas metodologías (por ejemplo ORACLE*METHOD) con el objetivo de representar la jerarquía de los procesos del sistema en diferentes niveles de abstracción.

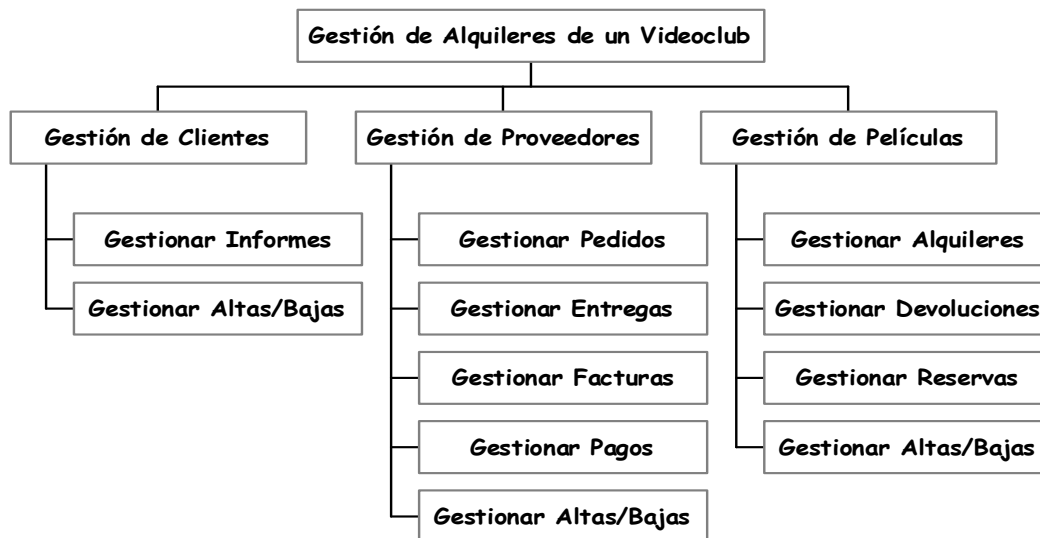
Para ello, se descompone una función de alto nivel (que en este caso es el sistema) en funciones de mas bajo nivel, y así sucesivamente.

Los DDF se utilizan principalmente para representar las funciones, aunque pueden ayudar a representar otros tipos de información (por ejemplo, estructura de una organización, estructuras de documentos, estructuras de menús, etc.).

Pueden considerarse tres tipos de DDF:

- ♦ **Tipo I:** Las funciones del sistema a diferentes niveles de abstracción, pero no los flujos de datos entre ellas. Es el tipo utilizado con mayor frecuencia.
- ♦ **Tipo II:** Las funciones y sus estructuras de los datos de entrada y salida.
- ♦ **Tipo III:** Funciones y los datos de entrada y salida, pero siguiendo determinadas reglas específicas que pueden definirse mediante axiomas matemáticos. Este tipo de DDF puede verificarse automáticamente por medio de herramientas CASE.

Un DDF de tipo I está representado en la figura:



3.4-5.- Comprobaciones sobre una especificación estructurada

Una vez realizada la especificación estructurada, formada por el DFD, el diccionario de datos y las especificaciones de proceso, es preciso revisarla considerando primordialmente cuatro aspectos:

- **Compleción:** Verificando si los modelos de especificación estructurada son completos.
- **Integridad:** Verificando si no existen contradicciones ni inconsistencias entre los distintos modelos.
- **Exactitud:** Verificando si los modelos cumplen los requisitos del usuario.
- **Calidad:** Verificando el estilo, la legibilidad y la facilidad de mantenimiento de los modelos producidos.