INSTITUTO de ENSEÑANZAS a DISTANCIA de ANDALUCÍA

2º de Bachillerato

Tecnologías de la Información y Comunicación

Contenidos

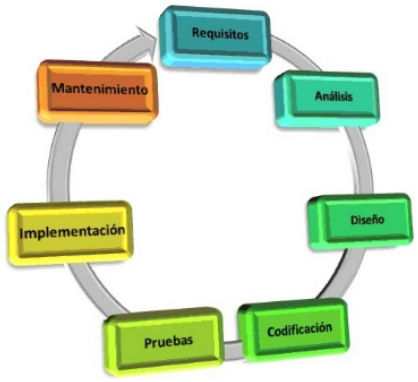
Ciclo de vida:
Ciclo de vida de desarrollo de software

Como vimos en el tema anterior, el proceso de desarrollo del software debe hacerse **por pasos**, cada uno de ellos se identifica con una etapa, dichas etapas siguen un orden establecido y cada una de ellas parte de la etapa anterior.

Cada etapa:

- Pretende alcanzar una serie de **objetivos**.
- Realiza una serie de **acciones**.
- Aplica una serie de **técnicas**.
- Obtiene unos **productos**.

Plantearemos este tema aplicando los conceptos teóricos, que se expondrán en cada etapa, a **un ejemplo** práctico.



Fases o Etapas del Ciclo de Vida del Software

1. ¿Qué voy a conseguir?

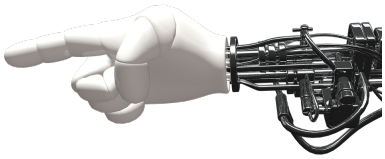


Imagen en pixabay de DirtyOpi bajo licencia Creative Commons

Al finalizar el tema, y **según cada fase del ciclo de vida** de desarrollo del software, los objetivos a conseguir son:

Respecto a la etapa de **especificación de requisitos**:

- Conocerás las técnicas que permiten obtener las necesidades del cliente.
- Identificarás y definirás los requisitos que debe cumplir la aplicación.
- Identificarás los distintos tipos de requisitos.
- Conocerás en qué consiste un estudio de viabilidad.

Relativos al **análisis de procesos**:

- Entenderás en qué consiste la fase de análisis del sistema.
- Distinguirás las dos partes de que consta la aplicación.
- Conocerás los diferentes procesos que debe realizar una aplicación.
- Sabrás construir el Modelo de Procesos con la técnica DFD.
- Entenderás qué es el Diccionario de Datos.

Para el **análisis de datos**:

- Sabrás construir el Modelo de Datos con la técnica Modelo E/R Simplificado.
- Averiguarás qué datos usa una aplicación.
- Organizarás la información que utiliza la aplicación de manera coherente mediante un Modelo de Datos.

En relación con el **diseño de procesos**:

- Entenderás en qué consiste el diseño de procesos.
- Aprenderás las técnicas de diseño estructurado de procesos basadas en los Diagramas de Cuadros de Constantine Simplificados.

Referente al **diseño de datos**:

- Entenderás en qué consiste el diseñó de datos.
- Valorarás la importancia de la estructura de datos Registro.

Respecto a la **implementación** del sistema:

- Valorarás los principios de implementación que subyacen en la programación estructurada.
- Entenderás ejemplos en pseudocódigo del paso a la implementación de los módulos que forman la fase de diseño.

Por último, respecto a las etapas de **pruebas, implantación y mantenimiento** del sistema:

- Conocerás los distintos tipos de pruebas que existen.
- Entenderás cuándo se puede hablar de cierra del proyecto.
- Reconocerás los distintos tipos de modificaciones que se contemplan en las actividades de mantenimiento del sistema.

2. Especificación de requisitos

La primera fase o etapa de cualquier ciclo de vida de una aplicación informática es la denominada especificación de requisitos. El objetivo de la misma es doble:

- Por un lado, se intentan **identificar las necesidades del cliente** para definir el problema a resolver.
- Además, se **realiza un estudio de viabilidad** económico y técnico para decidir si son factibles las posibilidades de realización.

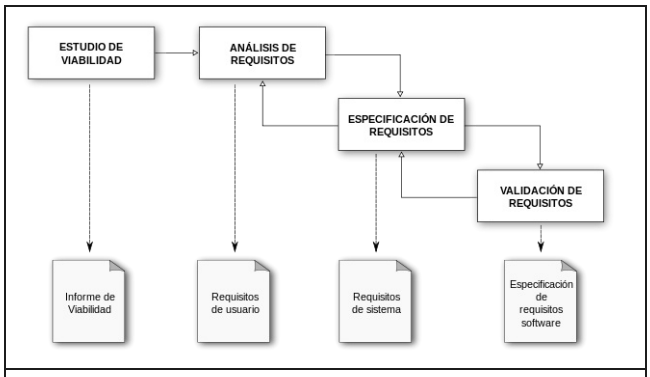


Imagen en Wikimedia de ebnz bajo licencia Creative Commons

A) Definición del problema

Para intentar resolver el problema planteado en nuestra aplicación previamente es necesario definirlo detalladamente. Así tendremos que:

1. **Definir el ámbito y alcance del proyecto**: de acuerdo con el cliente, debe quedar claro qué se debe resolver y qué queda fuera del alcance de dicho proyecto. Además, se deberán identificar los potenciales tipos de usuarios del sistema.
2. **Identificar y definir requisitos**: descomponiendo el problema principal en subproblemas que deben detallarse completamente. Para conseguir esto, se emplearán las técnicas que se especifican a continuación.

B) Técnicas de definición del problema

Las principales técnicas que se usan en la práctica son dos: entrevistas y cuestionarios.

1. Entrevistas

La entrevista es la técnica principal para identificar las necesidades del cliente y, por tanto, poder definir el problema. Consiste en **realizar de forma presencial una serie de preguntas al cliente** con el fin de recoger la máxima información. En ocasiones, el cliente puede necesitar asesoramiento nuestro a modo de guía para ayudarle a "sacar" dicha información. Previamente a "sentarse con el cliente", se debe preparar qué preguntas hacer y qué tipo de preguntas emplear (abiertas, cerradas). Tras finalizar la entrevista, se analizarán las respuestas y **se extraerán las conclusiones** oportunas.

2. Cuestionarios

En ocasiones, **no es posible preguntar físicamente** de forma directa al cliente, en estos casos el cuestionario suele ser la técnica empleada para recabar la información. El hecho de no poder asesorar o no estar de forma presencial con el cliente para aclarar posibles dudas en las cuestiones a preguntar, se hace necesario emplear un **estilo de redacción directo y con un lenguaje claro**, que no de lugar a especulaciones o malentendidos en las preguntas. En ocasiones, los cuestionarios se usan como complementos a las entrevistas.

C) Productos de la especificación de requisitos

El **Catálogo de Requisitos** es el producto generado en esta primera etapa, en él se recogen de una forma organizada dos tipos de requisitos:

- Requisitos **funcionales**: contienen aspectos relativos al comportamiento del sistema.
- Requisitos **no funcionales**: contienen propiedades del sistema organizados en tres tipos:
  - Restricciones: describen los límites del sistema.
  - De funcionamiento: especifican el hardware y software necesarios.
  - Manejo de excepciones: recogen los comportamientos anormales del sistema y sus tratamientos correspondientes.

D) Estudio de viabilidad

En este punto se trata de dar respuesta a la pregunta **¿es viable nuestro proyecto?**, es decir, ¿podremos realizarlo?

Los **pasos** que se seguirán irán en función de la envergadura concreta del proyecto, sin embargo, en líneas generales:

1. Se considerarán las posibles **alternativas** que solucionen el problema.
2. Se **evaluarán** desde los puntos de vista técnico, económico y legal las diferentes alternativas consideradas.
3. Se realizará una toma de decisión para **seleccionar** la alternativa más adecuada.

Para saber más

Documentación de la especificación de requisitos

La especificación de requisitos deberá tener la siguiente estructura formal:

1. Índice
2. Descripción del ámbito y alcance del proyecto.
3. Lista de usuarios participantes.
4. Descripción del sistema actual.
   - 4.1. Modelo físico.
   - 4.2. Lista de problemas y necesidades.
   - 4.3. Diagrama de flujo de datos.
5. Catálogo (priorizado) de requisitos del sistema.
   - 5.1. Funcionales.
   - 5.2. No funcionales.
     - 5.2.1. Restricciones.

5.2.2. De funcionamiento.

5.2.3. Manejo de excepciones.

6. Análisis de alternativas.

6.1. Descripción de la alternativa 1.

6.2. Descripción de la alternativa 2.

...

6.n. Descripción detallada de la alternativa seleccionada.

6.n.1. Modelo lógico de procesos.

6.n.2. Análisis coste-beneficio.

6.n.3. Diferencias con las demás alternativas.

7. Apéndices (si procede).

2.1. Ejemplo

¿Recuerdas el **ejemplo** planteado en el tema anterior? Considerábamos el caso de una **empresa de ropa deportiva** (ERD) que acudía a una empresa de informática (EI) para el desarrollo de un software que le permitiera vender de forma online.

A lo largo de este tema se irán desglosando las distintas etapas para este proyecto. En esta primera etapa detallamos la especificación de requisitos, siguiendo la estructura presentada en el apartado referente a la documentación que se debe aportar.

Para obtener la documentación correspondiente, se propone en primer lugar la realización de una entrevista con el cliente y posteriormente la cumplimentación de un cuestionario complementario.

.

Entrevista con el cliente

EI: La venta fuera de España implica un aumento considerable en los gastos de envío a cargar al cliente y, en parte, mayores gastos de gestión a asumir por la empresa ¿están dispuesto a asumirlos?

ERD: En principio, no podemos afrontarlos, nos centraremos en la venta nacional.

EI: ¿Cuáles son las distintas secciones o categorías de productos?

ERD: Tanto para ropa de hombre como para ropa de mujer distinguiremos entre camisetas y pantalones. De cada uno de estos dividiremos a su vez entre manga corta o manga larga.

EI: ¿Cuáles serán los posibles métodos de envío de un pedido?

ERD: Sólo por correo.

EI: ¿Y las formas de pago?

ERD: Sólo con tarjeta.

EI: ¿Es necesario un pedido mínimo?

ERD: Sí, consideramos que al menos el cliente debe realizar un pedido de 20€.

...

.

Cuestionario (complementario)

Instrucciones:

- Para las preguntas cerradas, marque su respuesta con un aspa.
- Para las preguntas a valorar con un número, el menor indica más pequeño, menor, peor,... y el mayor indica grande, mayor, mejor,...

Preguntas:

1. En los 20€ del pedido mínimo están incluidos los gastos de envío:

   -Sí -No

2. ¿Qué datos son necesarios para registrarse como cliente?

   - Nombre y apellidos
   - DNI
   - Dirección postal
   - Email
   - Otros (especificar): _____

   _____

   _____

...

.

Documentación de la especificación de requisitos

.

1. Índice

...

.

2. Descripción del ámbito y alcance del proyecto

- El objetivo del proyecto es la venta online de productos de de ropa deportiva.
- Por motivos de gestión, relativa a la forma y método de envío de los pedidos, se considerará sólo la posibilidad de vender en territorio nacional.
- Los principales usuarios del sistema serán: la empresa (empleados), los clientes, los técnicos (asesoramiento).
- A groso modo, el funcionamiento del sistema partirá de una página web principal desde la que se podrá acceder a las distintas secciones.
- Las secciones a considerar son:

1. Ropa de hombre
   - 1.1 Camisetas
     - 1.1.1. Manga larga
     - 1.1.2. Manga corta
   - 1.2. Pantalones
     - 1.2.1. Pantalón largo
     - 1.2.2. Pantalón corto
2. Idem Ropa de mujer (...)

- Una vez dentro de una sección se mostrarán los artículos catalogados en la misma.
- Al seleccionar un artículo concreto se mostrará por un lado (parte izquierda) fotografías del artículo, por otro lado (parte derecha), características de dicho artículo y un enlace para añadirlo al pedido.
- Se podrán añadir al carrito o cesta de compra uno o varios productos, respetando las condiciones del pedido mínimo.
- Una vez añadidos todos los productos, el usuario tendrá la posibilidad de formalizar o cancelar el pedido. Para ello será necesario estar registrado previamente o, en caso de ser la primera vez que compra, acceder con sus datos de acceso.

- Los datos de acceso como cliente estarán compuestos por usuario y clave, los cuales se completarán al realizar el proceso de registro.
- Una vez identificado como cliente (datos de acceso), se podrá pagar el pedido o bien cancelarlo.
- El proceso de pago del pedido tendrá un paso intermedio para que el cliente compruebe si es correcto.
- Tras el pago del pedido el cliente recibirá vía email un justificante de compra.

3. Lista de usuarios participantes

- Usuarios públicos: consultan la web sin acceder como clientes.
- Clientes: consultan la web con la posibilidad de comprar artículos (accediendo previamente con sus datos de acceso).
- Técnicos: empleados de la EI para asesorar o resolver posibles incidencias o problemas.

4. Descripción del sistema actual

4.1. Modelo físico

En este momento la empresa se dedica a la venta de ropa deportiva en tienda (de la forma clásica). Dispone de un local comercial con los productos expuestos para la venta y un apartado con un empleado para tramitar la compra por parte del cliente. Sólo es posible la venta de los productos expuestos físicamente en tienda.

4.2. Lista de problemas y necesidades

- En temporada de ofertas, un empleado es insuficiente para atender la alta demanda de clientes.
- Cada vez se van introduciendo más tipos de productos, tallas y colores, de esta forma, el local se va quedando pequeño.
- Por el hecho de vender en tienda, el alcance de mercado es muy limitado, esto es, en la localidad de referencia y algunas muy cercanas.
- Si falta un producto en la tienda, y es el que busca un cliente, se pierde esa venta.
- Trasladanto el "escaparate" a la página web se consiguen mostrar todos los productos que la empresa trabaja, dando la posibilidad de no tener que tenerlos físicamente disponibles.
- Etc.

4.3. Diagrama de flujo de datos

Gráficamente, el camino que sigue la información (en tienda) es el siguiente:

PROCESOS:	**Exposición** del producto (talla, color, precio, etc)	→	**Elección** del/los producto/s	→	**Preparación** del pedido	→	**Pago**	→	**Recogida** del pedido
DATOS:	*marca, modelo, talla, color, precio, etc*		*cantidad*		*suma de productos*		*Importe en €*		*productos comprados*

5. Catálogo (priorizado) de requisitos del sistema

En orden de prioridad los requisitos son los siguientes:

5.1. Funcionales

- Desde la página principal se accederá a las secciones principales (ropa de hombre, ropa de mujer).
- Desde cada sección principal se accederá a cada subsección o categoría de artículos (camisetas, pantalones).
- Desde cada categoría de artículos se accederá a los artículos de dicha categoría (listado).
- Desde el listado de artículos se accederá a un artículo concreto.
- Cada vez que un cliente realice una compra se actualizarán los datos de almacén (existencias) y se contabilizará el importe (ventas).
- Si un artículo no está en stock aparecerá como "no disponible en este momento".
- Etc.

5.2. No funcionales

- El logotipo de la empresa debe estar visible en todas las páginas.
- Para cada producto se especificará su marca, modelo, talla, color y precio.
- Se usará el color azul como base y sus tonalidades según interese.
- Se comprobará la correcta visualización según distintos dispositivos (PC, tablet, móvil, etc) y navegadores (Firefox, Chrome, Internet Explorer, etc).
- Si se produce un error al formalizar un pedido (caída del servidor, etc) se comprobará su estado antes de permitir repetir el pedido.
- Etc.

6. Análisis de alternativas

6.1 Alternativa 1 (no seleccionada)

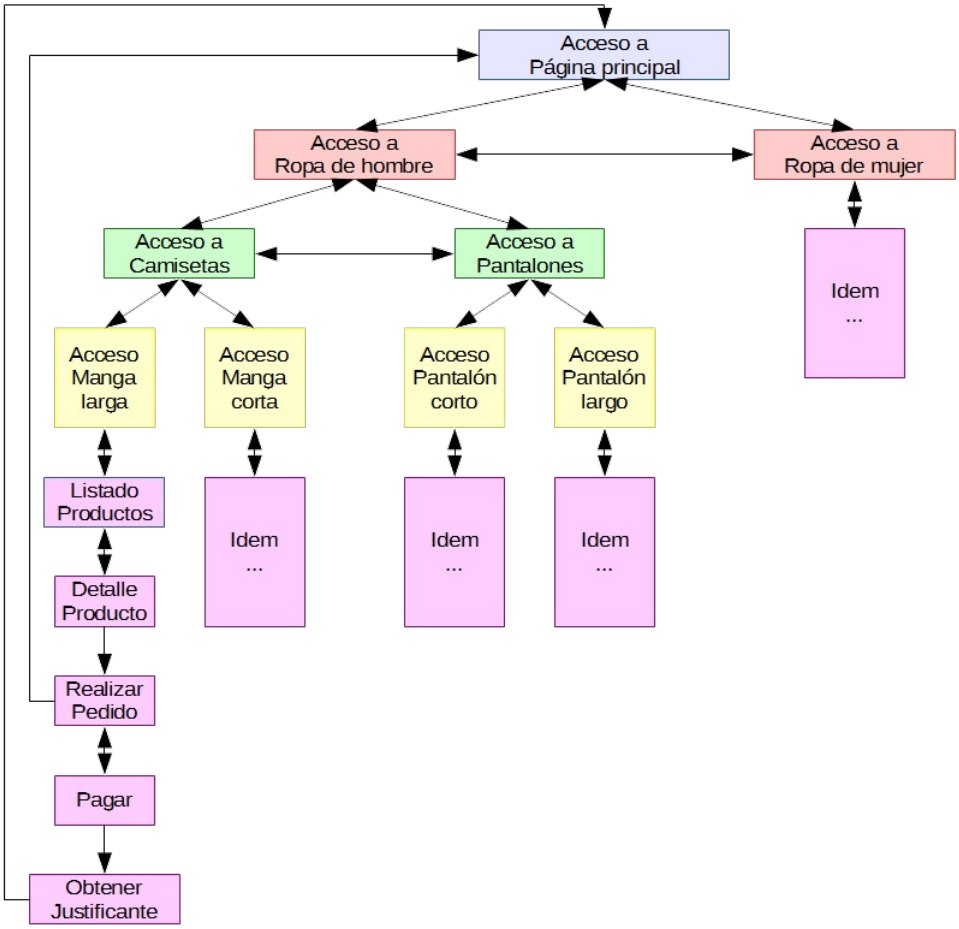
A modo de ejemplo y por simplificar, considerar otro modelo con distinto de acceso a la información, otras formas de pago,...

6.2. Alternativa 2 (seleccionada)

Por simplificar, suponemos que la alternativa seleccionada se ajusta a lo detallado en los apartados anteriores.

6.2.1. Modelo lógico de procesos

Gráficamente, los procesos que se llevarían a cabo y sus relaciones serían:



(Imagen de creación propia)

En el gráfico anterior, las flechas reflejan los posibles caminos a seguir para llegar de un proceso a otro. El sentido de dichas flechas indica si el camino es sólo de ida o de ida y vuelta.

6.2.2. Análisis coste-beneficios

En líneas generales el **coste** del proyecto se puede calcular según el tiempo a emplear y el personal de desarrollo del mismo.

- La página principal, como primer escaparate, necesitará de un mayor diseño web (cartelería, tipografías, fotos, etc), aunque menor dificultad operacional (al contrario que la formalización del pedido).
- El trabajo desarrollado en una de las secciones principales, por ejemplo "Ropa de hombre", facilitará las labores de desarrollo de la otra. Análogamente para las subsecciones y categorías de productos.
- El proceso de formalización del pedido será el más dificultoso, en cuanto a programación informática.
- Etc.

Por otro lado, los beneficios que puede aportar la aplicación de venta online irán en función de:

- La posibilidad de disponer de un servicio 24x7 (24 horas al día los 7 días de la semana).
- Atender un mayor número de pedidos al no depender de limitaciones físicas (personal, falta de stock en tienda, etc).
- Etc.

Resumiendo:

- Costes: H horas empleadas x precio/hora = CCCCCC €
- Ventas: P pedidos diarios x margen/pedido = VVVVVV €
- Beneficios: (Nº de días) x VVVVVV - CCCCCC = BBBBBB €
- En función de cuándo se consiga alcanzar BBBBBB>0 la empresa empezará a obtener beneficios.

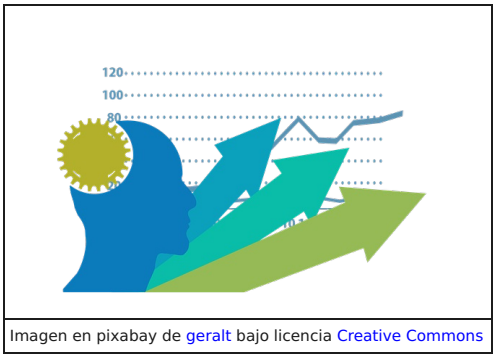
3. Análisis

En la etapa anterior (especificación de requisitos) el objetivo era conseguir que el sistema que se va a desarrollar esté perfectamente especificado, sin posibilidad de error o datos poco definidos. En ella se describe lo que el sistema nuevo debe realizar. Y se ha averiguado, mediante un estudio de viabilidad, si el sistema se puede desarrollar o no.

El siguiente paso del ciclo de vida es el Análisis del sistema. Lo que se pretende es precisamente analizar esas especificaciones para obtener el problema más definido y más cercano a la solución final. **Ahora nos centramos en QUÉ debe hacer el sistema**.

El objetivo del Análisis es entender el problema identificando las necesidades del cliente y las restricciones que se deben cumplir. Para ello se deberá:

- Organizar la información del problema y mostrarlo en un **formato gráfico** más intuitivo y fácil de manejar.
- Depurar todo aquello que no interesa y concentrarse en lo **que debe hacer** el sistema.
- Poner sobre la mesa y resolver posibles **conflictos**.
- **Dividir** el problema principal en subproblemas más fáciles de resolver.



En toda aplicación informática se distinguen principalmente dos partes: los procesos y los datos.

Los **procesos** son la **parte funcional** de la aplicación. Reflejan las funciones o tareas que debe realizar la aplicación. Muestran cómo se transforman los datos.

Los **datos** son la **parte estática** de la aplicación. Se refieren a la información que se necesita almacenar.

En un lenguaje coloquial, **los procesos dicen qué hay que hacer y los datos indican con qué hay que hacerlo**. Ambos usan su propio modelo de representación de la realidad. Así, y como resultado del análisis, se obtendrán:

- El **Modelo de Procesos**: recoge qué funciones, actividades, tareas, acciones, debe realizar la aplicación y cómo manejar los datos.
- El **Modelo de Datos**: describe la información que maneja la aplicación, los datos que debe almacenar y muestra cómo organizarla.

A) El modelo de procesos

El modelo de procesos (o modelo funcional) es la parte del análisis que recoge la **perspectiva funcional** de una aplicación. En él se describen cada uno de los procesos o funciones que debe realizar la aplicación, especificando:

- Qué procesos y subprocesos hay.
- Qué realiza cada proceso.
- Cómo se transforman los datos que entran al sistema en datos de salida.

En esta línea se emplearán las siguientes **técnicas**:

- Diagrama de Flujo de Datos (**DFD**).
- Diccionario de Datos (**DD**).

En este modelo los elementos fundamentales son los procesos. Un proceso, como se ha visto en temas anteriores, es un elemento que realiza cierta transformación sobre un conjunto de datos de entrada, generando unos resultados de salida. Gráficamente:

.

Datos de **entrada**	→	**Proceso**	→	Resultados de **salida**

.

Por último, el modelo de procesos contempla también la existencia de **entidades externas** al sistema que se relacionan con él para realizar operaciones y obtener resultados. Estos usuarios del sistema pueden ser personas físicas u otros sistemas (informáticos o no) e intercambian con el sistema los datos generales de entrada y de salida.

En la práctica, para construir el modelo de procesos debemos **partir de** la especificación de requisitos, principalmente de **los** denominados **requisitos funcionales** e ir aplicando el **principio de descomposición funcional jerarquizada**, comúnmente conocido como "*divide y vencerás*".

.

B) Técnicas de análisis estructurado de procesos

Aunque existen varias técnicas la más importante, y que usaremos en este curso, es la de los diagramas de flujo de datos (**DFD**). Con ella se plasman **gráficamente** los distintos procesos, las relaciones entre ellos, los datos y las entidades externas.

1. Diagrama de flujo de datos

Cuatro son los elementos que forman un DFD:

a) Procesos

Para **identificar funciones** dentro del sistema se usan los procesos, se trata de **operaciones** descritas en el Catálogo de Requisitos. Cualquier proceso implica siempre tratamiento y transformación de unos datos de entrada en otros de salida.

Se representan mediante círculos u óvalos, indicando en su interior el nombre y número identificador del proceso. Un proceso indica siempre una acción, por tanto, **el nombre debe ser un verbo** o una frase que describa su comportamiento. **Su número identificador respetará un sistema de jerarquía** y nos servirá para identificar qué proceso/s es/son suproceso/s de otro jerárquicamente superior.

b) Flujo de datos

Para **representar el camino que siguen los datos** dentro del sistema se usan los flujos de datos, reflejan así la **visión dinámica** de los datos. Así, se indicará, para un proceso, sus datos de entrada y de salida. Los datos que van o vienen directamente desde las entidades externas se denominan **entradas y salidas generales** del sistema. Todos los demás flujos de datos se llaman flujos de datos interiores, los cuáles son a la vez salidas de unos procesos y entradas de otros.

Se representan con flechas desde unos procesos a otros y se distinguen por un nombre que identifica a los datos que representa. Por ello, **su nombre debe ser un sustantivo** o un sintagma nominal escrito en minúsculas y en singular.

c) Almacenes

Los almacenes **representan los datos desde el punto de vista estático**, es decir, atendiendo a la forma en que éstos perduran en el sistema con el paso del tiempo.

Gráficamente **se representan mediante un nombre entre dos líneas horizontales y paralelas**. Dicho nombre, al igual que en los flujos de datos, debe ser **un sustantivo** o un sintagma nominal escrito en mayúsculas y en plural.

Cuando un flujo de datos entra en un almacén procedente de un proceso significa que se está realizando desde ese proceso una operación que modifica el contenido del almacén (**operaciones de escritura**). Por el contrario, cuando un flujo de datos va de un almacén a un proceso significa que el proceso está realizando una operación de consulta sobre el almacén, lo cual no modifica el valor de los datos contenidos en éste (**operación de lectura**).

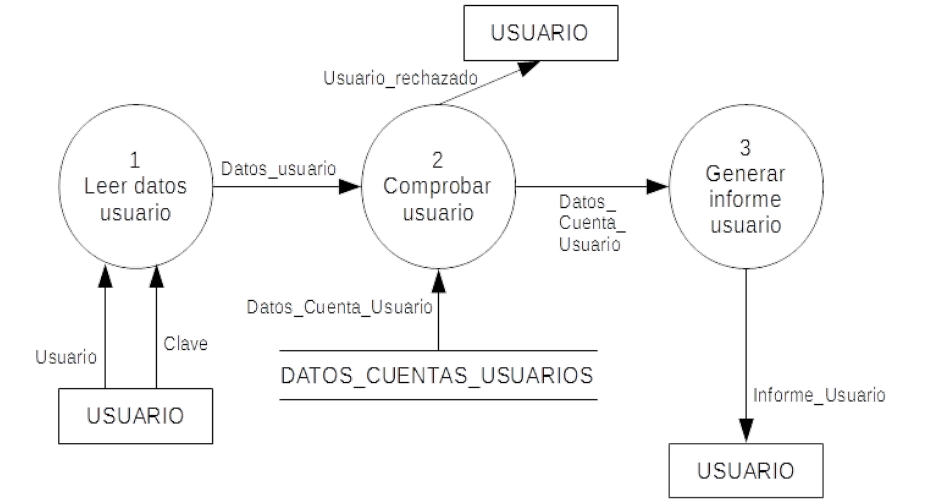
d) Entidades externas

En la fase de especificación de requisitos, debíamos identificar los potenciales **usuarios externos del sistema**, éstos son representados por las entidades externas. Al hablar de usuarios se hace referencia bien a usuarios humanos o bien otros sistemas informáticos (o no), q**ue proporcionan al sistema sus entradas de datos generales y reciben de éste a cambio los resultados o datos de salida generales**. Estas entidades están fuera del sistema y por tanto marcan sus límites.

Las representamos por medio de rectángulos indicando en su interior un **nombre** identificativo. También ha de ser **un sustantivo** o sintagma nominal que suele escribirse en singular y en mayúsculas.

.

Ejemplo:



(Imagen de creación propia)

Algunas **consideraciones o reglas** que debemos tener en cuenta **para crear un DFD** son:

- Se comenzará el DFD con lo que se denomina **diagrama de contexto (DC) o DFD de nivel 0**. En él sólo se representarán las entidades externas y un sólo proceso identificando el problema principal.
- En un diagrama de contexto **nunca debe representarse ningún almacén**.
- El diagrama de contexto se descompondrá en uno o varios **DFD de nivel 1**. En éste se reflejan las **funciones principales** del sistema o el menú principal de la aplicación.
- El resto de DFD surgirán de descomponer los DFD de nivel 1, obteniendo así **DFD de nivel 2, 3, etc**.
- El **nivel** de un DFD representa su **número identificativo**.

2. Diccionario de datos

Para **recoger las definiciones de todos los datos que se manejan en el sistema** se usa el diccionario de datos (DD). Esto incluye a los flujos de datos que entran y salen de los procesos, pero también a los almacenes y a las entidades externas.

Se denominan **entradas** a las definiciones contenidas en el DD, las cuáles **se clasifican en**:

- **simples**: se definen indicando el tipo de dato que representan o enumerando los posibles valores que toman.
- **compuestas**: se definen indicando los elementos que la componen y la forma en que están combinados. Lógicamente, una entrada compuesta puede estar formada por entradas simples, pero también por otras entradas compuestas. Los elementos que forman una entrada compuesta se pueden combinar de la siguiente forma:

TIPO	FÓRMULA	SIGNIFICADO
Contatenación	Entrada = Componente1 + Componente2	Es la regla de composición de elementos más sencilla: una entrada compuesta formada por dos o más componentes.
Disyunción	Entrada = [Componente1 | Componente2]	Establece alternativas exluyentes: la entrada compuesta está formada por uno de entre varios componentes posibles.
Opcionalidad	Entrada = (Componente1)	Supone que un determinado componente puede estar presente en la formación de una entrada compuesta, pero no obligatoriamente.
Repetición	Entrada = { Componente1 }	Permite establecer que una entrada compuesta está formada por dos o más componentes del mismo tipo. El número mínimo y máximo de componentes del mismo tipo que forman parte de una entrada compuesta se puede indicar mediante subíndices y superíndices situados a la derecha de la llave de cierre, pero este añadido no es de uso obligatorio. En concreto, la ausencia del subíndice se interpreta como valor mínimo igual a uno y la ausencia del superíndice significa valor máximo indefinido o desconocido.

Para saber más

Documentación del análisis de procesos

1. Productos de la fase anterior con los que se comienza el análisis.
   - Catálogo de requisitos del sistema.
2. Diagrama de contexto del sistema.
   - Entidades externas.
   - Flujos de datos de entrada generales.
   - Flujos de datos de salida generales.
3. Diagramas de flujo de datos (DFD).
   - Procesos.
   - Flujos de datos internos.
   - Almacenes.
4. Diccionario de datos (DD).

A) El modelo de datos

Imagen en pixabay de Pexels bajo licencia Creative Commons

Entendemos por **dato** toda aquella información que la aplicación debe recordar. Así, el modelo de datos refleja la **parte estática** de la aplicación. Recoge la información que la aplicación maneja respondiendo a la pregunta: ¿qué datos se deben recordar?

Aunque en proyectos informáticos profesionales el modelo de datos se afronta principalmente con técnicas de Bases de Datos Relacionales, para simplificar, en este curso **haremos uso de las estructuras de datos** vistas en temas anteriores para el almacenamiento y gestión de información.

B) Técnicas de análisis y especificación de datos:

Como se ha mencionado en el párrafo anterior, hoy en día las Bases de Datos Relacionales son usadas para el análisis y gestión de la información. Estas usan para su análisis la técnica llamada Modelo Entidad/Relación. Fuera del alcance de este curso, nos basaremos en dicha técnica y emplearemos una parte de ella, concretamente, se considerarán algunos símbolos propios del Modelo E/R sin entrar muchos detalles técnicos. De esta forma **usaremos un Modelo Entidad/Relación simplificado**.

En nuestro caso emplearemos una serie de símbolos y reglas para representar los elementos (información) que forman parte del problema.

Los elementos que forman el modelo E/R simplificado son:

a) Entidades

Una entidad **es un objeto real o abstracto del que se quiere obtener una información**. Las entidades son los conceptos que reflejan los datos que le interesa a la aplicación, las cuáles existen en el mundo real. Son entidades no sólo las cosas tangibles, sino también las intangibles (departamentos, proyectos, profesiones de personas, incidencias temporales, acciones que se repiten en el tiempo, etc).

En la práctica, **las representaremos mediante un rectángulo** en cuyo interior aparece **su nombre**, el cuál suele ser **un sustantivo**.

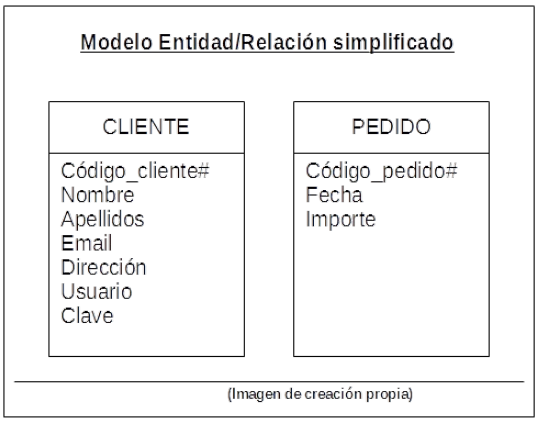
b) Atributos

Todas las entidades tienen asociados unos atributos o **características. Se trata de propiedades** de aquello que representa la entidad. Existen **dos tipos** de atributos:

- **identificadores** de entidad: son atributos que identifican de manera unívoca cada ocurrencia de una entidad.
- **descriptores** de entidad: son atributos que muestran una característica de la entidad.

De esta forma, **siempre debe existir**, al menos, **un atrituto identificador**. A los atributos identificadores se les añade el carácter almohadilla (#) en su nombre.

Ejemplo:



Para saber más

Documentación del análisis de datos

1. Descripción de las entidades.
   - Nombre de la entidad.
   - Breve descripción de lo que contiene.
   - Atributos:
     - Nombre del atributo.
     - Función (identificar/describir).
     - Descripción de los valores que contiene.
     - Dominio de valores que puede tomar.

Para saber más

La **base de datos relacional** (BDR) es un tipo de base de datos (BD) que cumple con el modelo relacional (el modelo más utilizado actualmente para implementar las BD ya planificadas).

Tras ser postuladas sus bases en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. (Texto de Wikipedia)

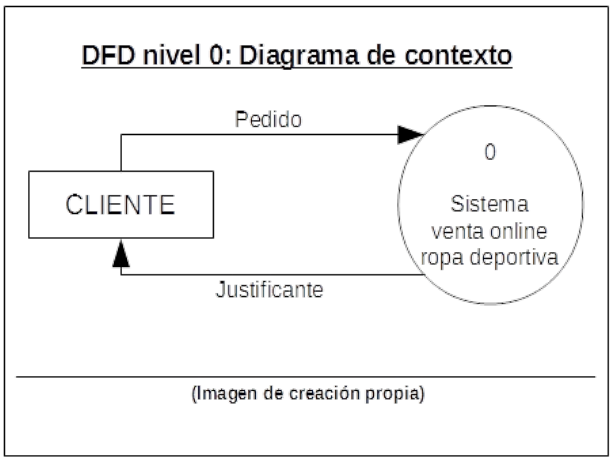
Imagen en Wikimedia de JuralMin bajo licencia Creative Commons

A continuación **se muestra parte del análisis de procesos y parte del análisis de datos** del caso de la empresa de venta de ropa deportiva.

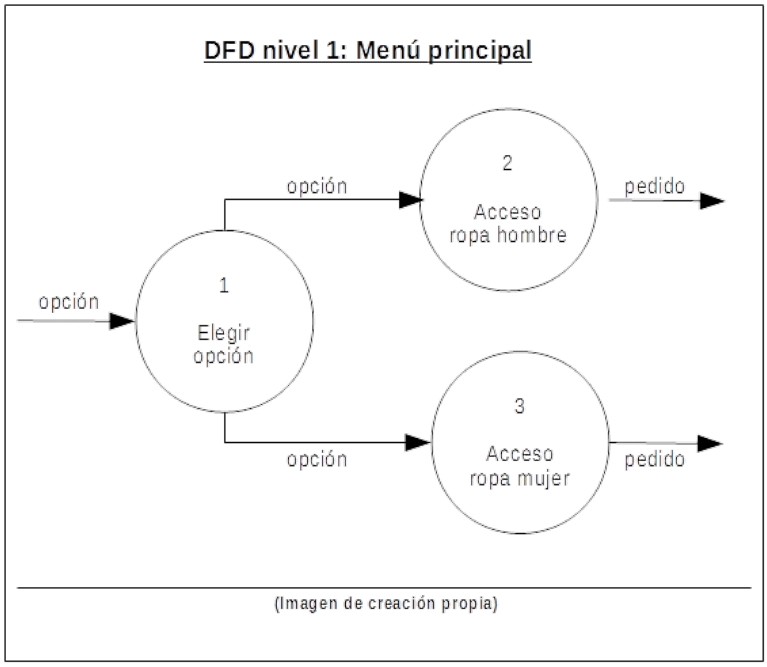
1. Análisis de procesos

Un **resumen** del análisis de procesos podría ser este:

a) Diagramas de flujo de datos (DFD):



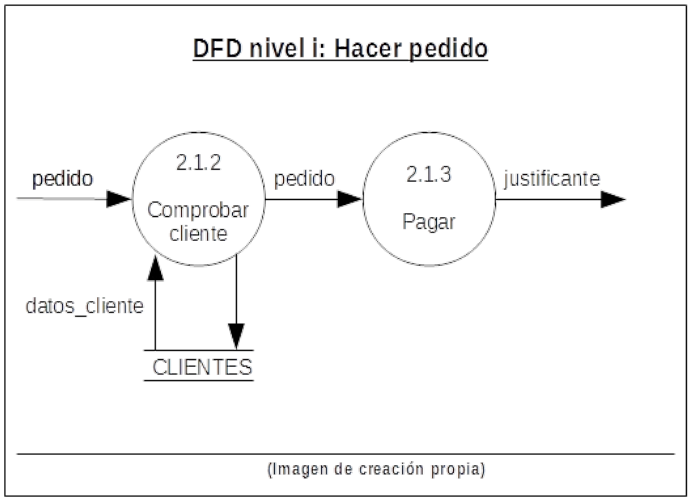
Las primeras dos figuras corresponder a los **DFD de nivel 0 y nivel 1** respectivamente. Como vemos tanto la entrada como la salida (flujo de datos) del diagrama de contexto parten y terminan en el *cliente*, en este sistema esta entidad externa es la que interactúa para hacer peticiones y obtener resultados.



En el DFD de nivel 1 "*opción*" hace referencia a la opción o elección por parte del usuario a la hora de acceder a una u otra sección.

El Proceso "*1 Elegir opción*" representa el cómo se accedería a las alternativas posibles (ropa de hombre/ropa de mujer).

Finalmente vemos que tanto el flujo de datos de salida del proceso "*2 Acceso ropa de hombre*" como del proceso "*3 Acceso ropa de mujer*" se etiqueta como "*Pedido*" representando así que una vez que el cliente accede a una sección y posteriormente a los artículos sobre los que está interesado, el siguiente paso es el de formalizar el pedido.



El siguiente paso sería construir los **DFD de niveles inferiores**, por ejemplo, el correspondiente para **realizar el pedido** podemos verlo más abajo.

Este DFD parte de que, una vez que el usuario intenta formalizar el pedido, con el proceso "*2.1.2 Comprobar cliente*" el sistema previamente debe comprobar si se trata o no de un usuario registrado, esto es un cliente, y de pendiendo de esto tendrá la posibilidad de registrarse (caso de que no sea cliente) o acceder con sus datos de cliente.

Si se trata de un cliente (y el pedido es correcto) con el proceso "*2.1.2 Pagar*" se procede al pago del pedido, obteniendo así el justificante correspondiente.

b) Diccionario de datos (DD):

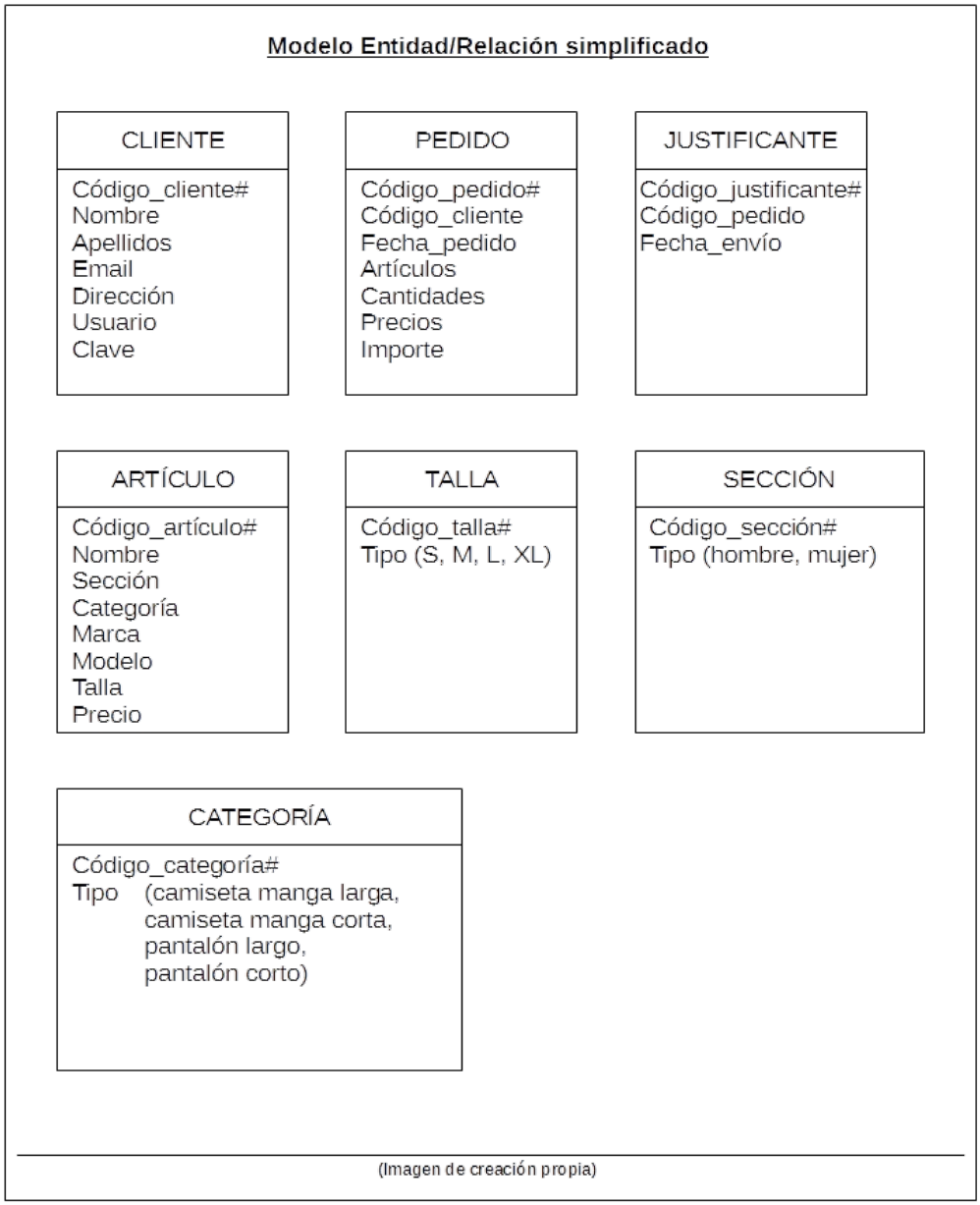
Un extracto del DD sería el siguiente:

Entradas	Aclaraciones
• Pedido = Código_cliente + Código_Pedido	• **Pedido** está compuesto por la concatenación del Código del cliente y el

● Código_cliente = 'C' + {Letra}2 + {Dígito}5 ● Letra = [A|B|C|...|Z] ● Dígito = [0|1|...|9] ● Código_pedido = 'P' + {Dígito}5 ● Justificante = 'J' + Pedido + Fecha_pedido ● ...	Código del pedido. ● El **Código del cliente** está formado por la concatenación del carácter 'C', dos letras y cinco dígitos. ● ...

2. Análisis de datos

Un **extracto** del análisis de datos, según el modelo E/R simplificado sería:



En algunos atributos se indica entre paréntesis los posibles valores que puede tomar.

4. Diseño

En este apartado tratamos la siguiente fase del ciclo de vida del softare: el diseño. Partiendo del análisis, cuyo objetivo era responder a la pregunta ¿qué hacer?, en esta fase nos centramos en ¿cómo hay que hacerlo?

Imagen en pixabay de geralt bajo licencia Creative Commons

Al igual que en el análisis, el diseño se enfoca desde el punto de vista funcional (procesos) y estático (datos) del sistema.

La idea es transformar los modelos construidos durante el análisis en modelos de solución del problema. Así hablaremos de diseño funcional y diseño de datos.

En un lenguaje más formal los **objetivos** del diseño son:

- Pasar de preguntarse ¿QUÉ hay que hacer? a plantearse ¿**CÓMO** hay que hacerlo?
- Obtener una **solución correcta**, esto es, que funcione, que sea mantenible, que se entienda (documentada) y que se pueda probar.
- **Preparar** el camino a la implementación.

La principal técnica usada en el diseño de procesos recibe el nombre de **Diagramas de Cuadros de Constantine**. Su estudio al detalle escapa de los objetivos de este curso, por tanto, emplearemos una modificación de dicha técnica **simplificada**. Nos basaremos en los principios básicos del desarrollo estructurado (descomposición del problema, solución de subproblemas e integración de soluciones parciales en una solución global) para transformar los DFD de la fase de análisis de datos en diagramas modulares.

Análogamente, el estudio completo del diseño lógico de datos requiere del uso del Modelo Relacional. En el análisis de procesos hemos usado un Modelo Entidad/Relación simplificado, de esta forma, en lugar de usar el citado Modelo Relacional **emplearemos las estructuras de datos**, vistas en temas anteriores, **para el diseño de datos**.

A) Técnicas del diseño de procesos

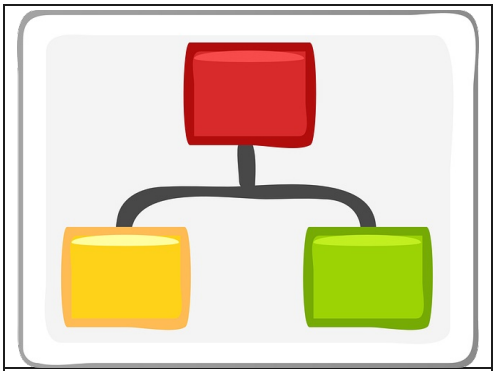


Imagen en pixabay de Clker-Free-Vector-Images bajo licencia Creative Commons

Como se ha mencionado anteriormente, para el diseño de procesos emplearemos una simplificación de los Diagramas de Cuadros de Constantine. A dicha técnica le llamaremos Diagramas de Cuadros de Constantine Simplificados.

Un diagrama de este tipo está formado por dos elementos básicos:

a) Módulos

- Un módulo es un programa o **subprograma** que puede ser llamado independientemente.
- Un componente software que **admite** parámetros de entrada y puede devolver parámetros de salida.
- Un proceso que **convierte** datos de entrada en datos de salida.

Aunque no existen unas directrices exactas, las **características deseables** de un módulo son:

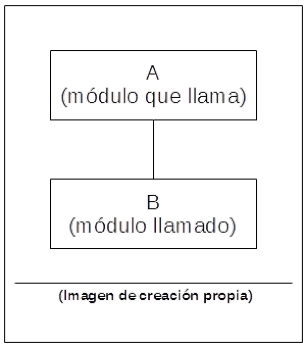
- **Pequeño** tamaño.
- **Interfaz clara** y limitada a través de sus parámetros.
- **Independencia** o mínimo de acoplamiento para poder modificarse sin cambiar otros módulos.
- Máxima **cohesión interna** para realizar sólo una función.

Para representar un módulo usaremos **un rectángulo** en cuyo interior se indicará el nombre de su función.

b) Conexiones

Una conexión entre dos módulos representa la llamada que uno de ellos hace a otro, para que éste ejecute alguna función para él. Se representa por medio de una línea que une al módulo llamador con el módulo llamado.

Ejemplo:

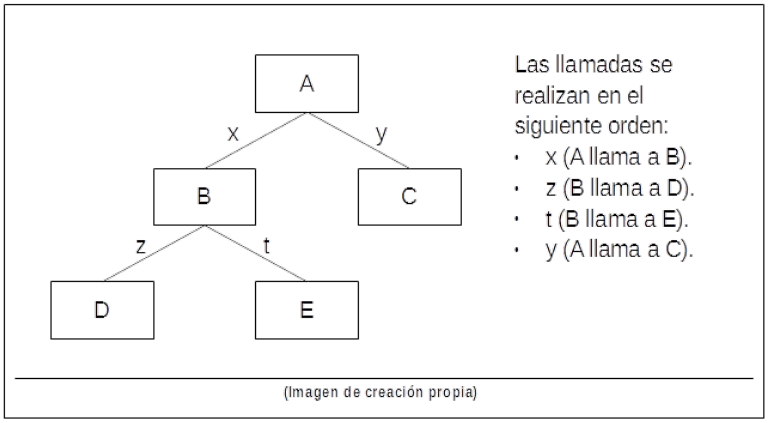


En este ejemplo:

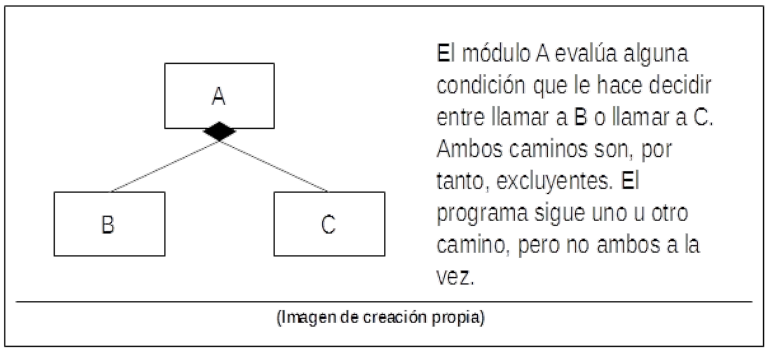
- El módulo A llama al B, cediéndole el control de ejecución.
- El módulo B realiza su función.
- El módulo B devuelve el control de ejecución al A, junto con los valores de retorno correspondientes (si es que los hay).
- El módulo A continúa ejecutando donde se había quedado.

Las distintas variantes que nos encontraremos, al representar este tipo de diagramas, están en función de las **estructuras de control básicas** propias de la programación estructurada:

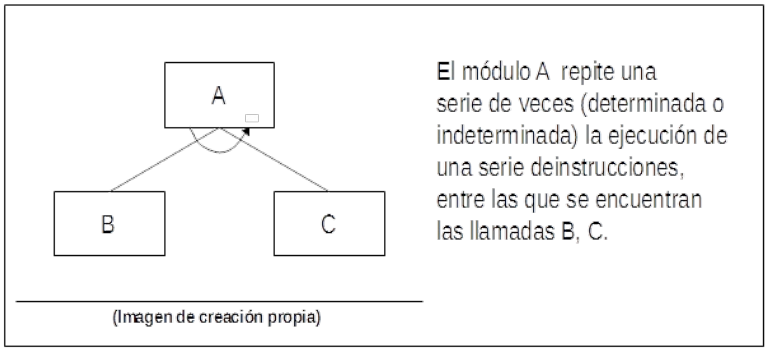
- Secuencia o **iteración**: las llamadas entre los módulos se ejecutan de arriba abajo y de izquierda a derecha.



- Alternativa o **bifurcación**: se representan por medio de un rombo relleno del que salen las conexiones que representan las distintas alternativas.



- Bucle o **repetición**: se representan por medio de una flecha curva que abraza a las conexiones cuyas llamadas han de repetirse un cierto número de veces.



Para saber más

Documentación del diseño de procesos

1. Productos de la fase anterior con los que se va a trabajar.
   - DFD's.
   - DD.
2. Diagramas de Cuadros de Constantine Simplificados.
   - Módulo del programa principal.
   - Módulos de subprogramas.
   - Módulos elementales.
   - Conexiones entre módulos.
     - Estructuras iterativas.
     - Estructuras alternativas.
     - Estructuras repetitivas.

4.2. Diseño de datos

En la fase de análisis de los datos se ha obtenido el Modelo Entidad/Relación Simplificado. Dicho modelo corresponde al ámbito conceptual, al mundo de las ideas. Como toda fase de análisis, se centra en qué datos hay que reflejar y no en cómo. Ahora en la fase de diseño, debe traducirse a una estructura que pueda ser manejada por el ordenador.

La idea es poder almacenar cada entidad con sus correspondientes atributos (fase de análisis) **en estructuras de datos apropiadas** (fase de diseño).

Según el dato a almacenar, se optará por estructuras estáticas o dinámicas de datos. Recordemos el tipo de dato Registro, se trata de un tipo de estructura estática cuyos elementos pueden ser de distinto tipo.

Así, por ejemplo, podríamos almacenar en un Registro la entidad "Pedido" cuyos atributos serían, cada uno de ellos, de un tipo adecuado, esto es:

Atributo	Tipo de dato
Código_pedido#	Cadena
Fecha_pedido	Fecha
Importe	Número decimal

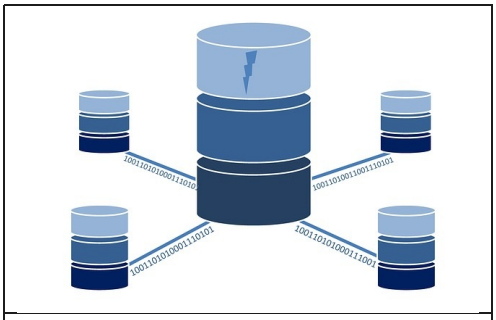
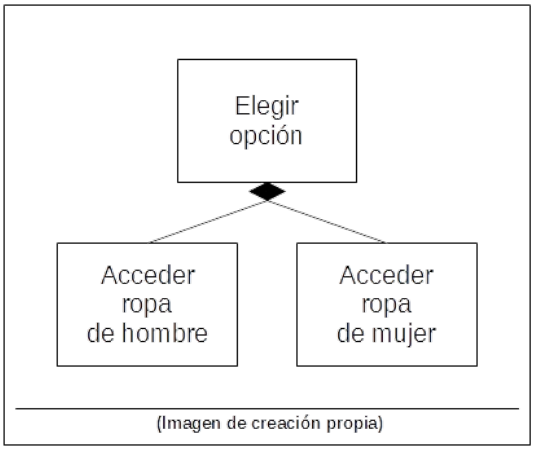


Imagen en pixabay de Tumisu bajo licencia Creative Commons

Partiendo del ejemplo mostrado en la fase de análisis, un extracto del diseño de procesos y de datos sería:

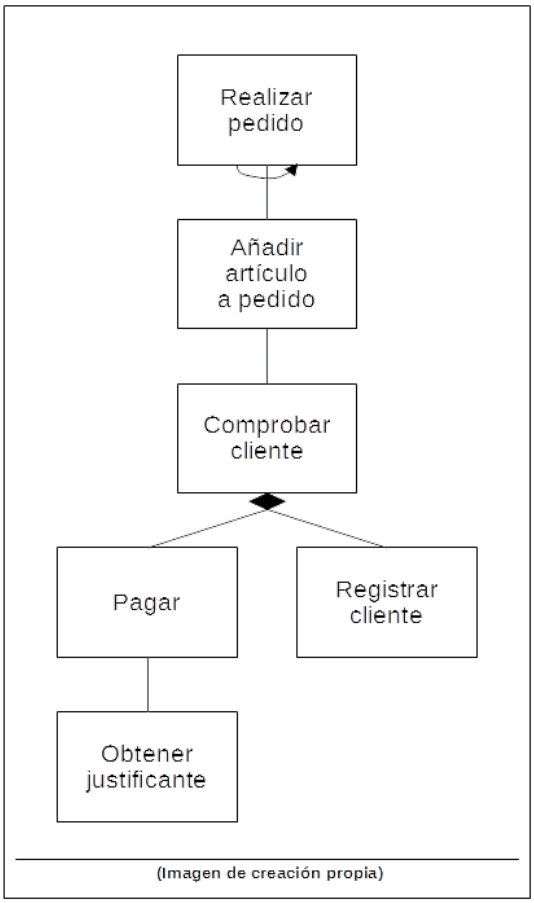
1. Diseño de procesos

Partiendo del DFD de nivel 1 (Menú principal):



(Imagen de creación propia)

El diseño para realizar un pedido quedaría:



(Imagen de creación propia)

2. Diseño de datos

Algunos de los datos que debemos almacenar son los siguientes:

- **Registro** Cliente (**Cadena** Código_cliente, **Cadena** Nombre, **Cadena** Apellidos, **Cadena** Email, **Cadena** Dirección, **Cadena** Usuario, **Cadena** Clave)
- **Registro** Pedido (**Cadena** Código_pedido, **Cadena** Código_cliente, **Fecha** Fecha_pedido, **Registro** Artículos, **Vector Entero** Cantidades, **Vector Decimal** Precios, **Número Decimal** Importe)
- **Vector Decimal** Precios []
- Etc.

Aunque depende del dato en sí, observamos como el **tipo de dato Registro** es el apropiado para contener elementos de distinto tipo, pudiendo ser alguno de éstos un Registro también.

5. Implementación

En las etapas anteriores se ha analizado el problema y diseñada la solución que lo resuelve. Ahora, en esta etapa del ciclo de vida, el objetivo principal es la de construir dicha solución.

Por tanto con la implementación **se pretende**:

- Programar las **funciones** del sistema.
- Crear y rellenar las **estructuras de datos**.

Como hemos observado en apartados anteriores, los diseños realizados son estructurados, por tanto, las técnicas de implementación que **usaremos** serán las de **programación estructurada**.

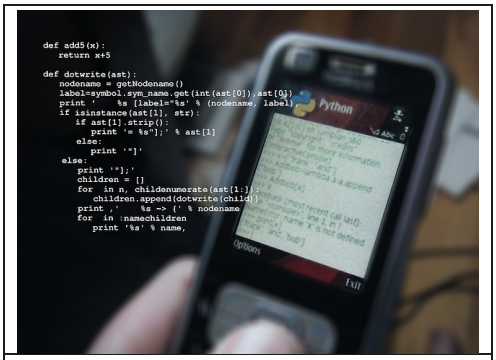


Imagen en pixabay de geralt bajo licencia Creative Commons

A) Implementación funcional del sistema

Consiste en la programación de todas las funciones identificadas y diseñadas.

En este punto, se remite al alumno a los temas anteriores, dónde se han tratado los distintos tipos de instrucciones propios de la programación estructurada.

Así, basándonos en los principios de la programación estructurada, implementaremos los diferentes módulos diseñados con los Diagramas de Cuadros de Constantine Simplificados. Para ello **haremos uso del lenguaje pseudocódigo como lenguaje general** que, según se elija, se codificará a un lenguaje de programación específico (PHP, ASP, Java, etc).

La técnica de resolución de problemas **"divide y vencerás"** es la base para construir funciones estructuradas. Por tanto, a partir de los módulos que obtengamos en el diseño del sistema construiremos un programa o algoritmo principal, para posteriormente dividirlo en subprogramas que faciliten la implementación del sistema. No debemos olvidar la parte estática de la aplicación (datos). De esta forma podemos decir que:

ALGORITMO = INSTRUCCIONES + DATOS

Como ya se ha estudiado, los **tipos de instrucciones** propios de la programación estructurada son:

- **iterativas**.
- **alternativas** (simple o múltiple).
- **repetitivas** (mientras, hasta, para).

Por otro lado, los **tipos de datos** que podemos usar son:

- **constantes** y **variables**.
- datos **predefinidos** y datos **definidos** por el programador.
- datos **simples** y **compuestos**.
- datos **estáticos** y **dinámicos**.

La regla mencionada "divide y vencerás" intenta, como su nombre indica, dividir un problema general o de mayor dificultad en otro/s más sencillo/s y por tanto más manejables y entendibles. Hablamos, por tanto, de **modularidad al considerar estos subprogramas o módulos** y más concretamente de la llamada programación modular.

Así, en la práctica, tendremos:

- un **programa principal** que tendrá el control general del sistema y llamará a diversos subprogramas que resuelvan problemas para él.
- un **conjunto de subprogramas** que trabajarán directamente para el programa principal y que, a su vez, podrán llamar a otros subprogramas menores que también resolverán alguna tarea para ellos. A esto se le llama **profundidad** de los subprogramas. Si el número de subprogramas crece excesivamente y su función es cada vez más especializada, **nace el concepto de módulo, como agrupación de subprogramas con una función o comportamiento afín** y que se almacenan y compilan independientemente.

Es importante también considerar lo que se llama **reglas de visibilidad**:

- Un subprograma ve todo lo definido en su **interior**, sean datos u otros subprogramas. En concreto todo subprograma "es consciente de su propia existencia", lo que le permite llamarse a sí mismo.
- Un subprograma definido dentro de otro ve a todos aquellos subprogramas definidos **al mismo nivel y con posterioridad** a él.

De esta forma, un subprograma no tiene a su alcance ni puede invocar a ningún subprograma al que no vea, porque no sabe de su existencia.

Las dos **formas** que tiene un subprograma **de pasar los datos de entrada a otro subprograma** cuando lo invoca son:

- Paso **por valor**: las modificaciones que el subprograma llamado haga sobre esos datos no permanecerán, pues el subprograma que lo invocó le pasó unas copias de esos datos.
- Paso **por referencia**: se pasan como parámetros los datos propiamente dichos, por lo que los cambios de valor que se les apliquen tendrán efecto.

B) Implementación de los datos

La implementación de los datos implica, además de la **definición formal de cada tipo de dato** a considerar (registros, arrays, cadenas,...), las **operaciones básicas que se pueden hacer** sobre ellos, esto es, creación, lectura, escritura, modificación y borrado.

Por tanto, en la práctica, debemos crear o disponer de las correspondientes funciones o subprogramas que realicen este tipo de operaciones.

Para saber más

Documentación de la implementación del sistema

1. Programación de las funciones del sistema.
   - Cabecera.
     - Nombre.
     - Parámetros y tipos.
       * Por valor.
       * Por referencia.

- Valores de retorno.

- Cuerpo.
  - Especificaciones de código.
  - Condiciones de control
    * Estructuras de bifurcación.
    * Bucles.
- Control de versiones.
  - Autor.
  - Fecha.
  - Número de versión.
  - Revisiones y cambios.

2. Implementación de los datos.

- Creación de registros.
  - Nombre.
  - Parámetros y tipos.
- Eliminación de registros.
- Modificación de registros.
  - Cambios de parámetros.
  - Inserción de datos.
  - Borrado de datos.
  - Modificación de datos.
- Consultas.

5.1. Ejemplo

En este apartado **vamos a implementar (en pseudocódigo) algunos módulos, funcionales y de datos**, diseñados en el ejemplo que venimos desarrollando a lo largo del tema.

1. Implementación funcional

```
Programa Principal
<<DEFINICIÓN DE SUBPROGRAMAS>>
    /* MóduloElegirOpcion*/
    Función MóduloElegirOpcion(Entero opcion){
        Comienzo
            si(opcion==1){
                MóduloRopaHombre();
            }sino{
                MóduloRopaMujer();
            }
        Fin
    }

    /* MóduloRopaHombre*/
    Función MóduloRopaHombre(Entero opcion){
        Comienzo
            según(opcion){
                caso 1:
                    MóduloCamisetasMangaLargaHombre();
                    salir;
                caso 2:
                    MóduloCamisetasMangaCortaHombre();
                    salir;
                caso 3:
                    MóduloPantalonCortoHombre();
                    salir;
                caso 4:
                    MóduloPantalonLargoHombre();
                    salir;
            }
        Fin
    }

    /* Otros Módulos */
        ...
<<FIN>>
```

```
/* MóduloRealizarPedido */
Funcion MóduloRealizarPedido(){
    Variables
        Entero fin_pedido= 0; //Pedido aún no finalizado
        Cadena Código_pedido= MóduloCalcularNumPedido(Código_cliente);
    Comienzo
        repetir{
            MóduloAñadirArtículo(Código_artículo);
        }hasta(fin_pedido==1)
        si(ModuloComprobarCliente(Código_cliente)==1){
            MóduloPagar(Código_pedido);
            MóduloObtenerJustificante(Código_pedido);
        }sino{
            MóduloRegistrarCliente();
            MóduloPagar(Código_pedido);
            MóduloObtenerJustificante(Código_pedido);
        }
    Fin
}
```

2. Implementación de datos

```
/* Creación del Registro (Almacén) Pedidos */
Función CrearPedidos(){
    Variables
        Registro Pedidos (Cadena Código_pedido,
                          Cadena Código_cliente,
                          Fecha Fecha_pedido,
                          Registro Artículos,
                          Vector Entero Cantidades,
                          Vector Decimal Precios,
                          Número Decimal Importe)
    Comienzo
        Global Registro Pedidos; //variable global
    Fin
}
```

```
/* Mostrar un Pedido concreto:
    Buscamos un pedido dentro del Registro Pedidos y lo mostramos
*/
Función MostrarPedidoUno(Registro Pedidos, Entero Pedido){
    Variables
        Entero i; //contador
    Comienzo
        para(i=0;i<NumElementos(Pedidos);i++){
            si(i==Pedido){
                Mostrar(Pedidos[i]);
            }else{
                Mostrar("El pedido buscado no está registrado.")
            }
        }
```

```
                                                    .
/* Mostrar todos los Pedidos */
Función MostrarPedidosTodos(Registro Pedidos){
    Variables
        Entero i; //contador
    Comienzo
        para(i=0;i<NumElementos(Pedidos);i++){
            MostrarPedidoUno(Pedidos[i]);
        }
    Fin
}
```

```
    Fin
}
```

6. Pruebas, implantación y mantenimiento

Por estar estas tres etapas del ciclo de vida del software (pruebas, implantación, mantenimiento) fuertemente relacionadas, se tratarán en un único apartado. Son etapas complementarias pero necesarias **antes del uso real del sistema**.

En este punto del camino, podría pensarse (una vez analizado, diseñado e implementado el sistema) que el trabajo del equipo informática ha terminado. Sin embargo, **el proyecto aún no se ha cerrado**.

En lineas generales, **faltaría probar que todo lo que se hecho hasta ahora se ha hecho bien**. Para ello es necesario probarlo e implantarlo para verlo en funcionamiento y poder detectar así posibles fallos, mejoras, etc.

Imagen en pixabay de animatedheaven bajo licencia Creative Commons

1. Pruebas del sistema

Hablar de pruebas es hablar de **calidad**, y aunque durante todo el proceso de análisis y diseño se intenta que esta se cumpla, pueden aparecer errores que sin la fase de pruebas serían imposibles de detectar y por tanto corregir.

En esta fase, **se somete a cada módulo a** una batería de pruebas, llamados **planes de pruebas**, con el objetivo de comprobar que todo funciona correctamente.

Los diferentes **tipos de pruebas** podemos clasificarlos:

- **Según la forma** en la que se realizan:
  - Pruebas **de caja negra**: comprueban el funcionamiento de un módulo a través de su interfaz, sin entrar en ver su funcionamiento interno. Para ello, se estudian los valores límites, datos erróneos y datos equivalentes.
  - Pruebas **de caja blanca**: se centran en en cómo un módulo resuelve un determinado problema atendiendo a los detalles internos de implementación.
- **Según el momento** de realización:
  - Pruebas **unitarias**: su objetivo es comprobar el funcionamiento de un componente individual, aislado del resto del sistema.
  - Pruebas **de integración**: en este caso se comprueba, después de haber probado componentes individualmente, que funcionan correctamente cuando trabajan juntos (integrados).
  - Pruebas **de subsistema y de sistema**: comprueban que, módulos (subsistemas) probados individualmente, funcionan correctamente cuando trabajan juntos. Los fallos que suele producirse se deben a una errónea comunicación entre subsistemas.
  - Pruebas **de carga**: usamos datos reales de ejecución para simular el funcionamiento del sistema.
  - Pruebas **de aceptación**: son las últimas pruebas que se realizan esta etapa del ciclo de vida. Su objetivo es obtener el visto bueno del cliente sobre la calidad de funcionamiento del sistema desarrollado y probado. En este sentido, se muestra al cliente el sistema completo.

2. Implantación del sistema

Se habla de puesta en producción o implantación del sistema a las **actividades que se realizan para instalar la aplicación en las máquinas del cliente y ponerlas en funcionamiento**.

En esta etapa será necesario considerar la posibilidad (según el caso) de un **período de transición o directamente implantar el sistema**. En ocasiones, la implantación puede suponer el contar con **recursos adicionales** (hardware y/o software).

Como se ha comentado, las últimas pruebas a realizar (de aceptación) son necesarias para, marcar el **cierre del proyecto** y considerar así el sistema implantado.

3. Mantenimiento del sistema

El mantenimiento del sistema supone, implícitamente, que éste **puede estar sujeto a cambios en su estructura, aspecto y funcionamiento** durante todo el tiempo que se utilice. Así, estos cambios son la mejor forma de **evitar que el sistema quede obsoleto** y tenga que ser reemplazado por otro mejor, más moderno, más completo.

Tres son los **tipos de cambios** que gestionan las tareas de mantenimiento:

- **Depuración de errores**: aunque se ha intentado respetar criterios de calidad y se han realizado las pruebas correspondientes, pueden aparecer errores que a priori no habíamos visto. El uso cotidiano del sistema puede hacerlos visibles y poder así depurarlos.
- **Cambios de requisitos**: aunque en principio se fijaron con el cliente unos requisitos puede que, una vez visto el sistema funcionando, se establezcan o cambien algunos.
- **Mejoras y ampliaciones**: su objetivo es considerar la posibilidad de dotar al sistema de nuevas funcionalidades.

Para saber más

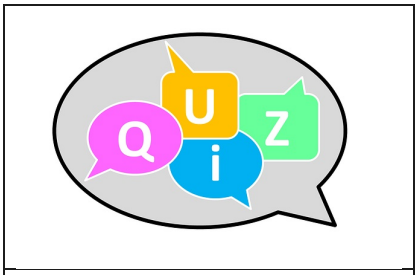
Implantar el sistema implica una **formación de los usuarios** que usarán el sistema. En este sentido, el plan de formación incluye **manuales y cursos**.

Los primeros deben ser precisos y emplear un lenguaje claro, ya que no estará presente ningún técnico al usarlos. Los manuales de usuario son también conocidos como guías de usuario.

La recomendación principal al diseñar un curso de formación es que debe ser práctico, para ayudar al usuario a aprender a trabajar con el sistema desarrollado.

Comprueba lo aprendido

En los apartados anteriores han aparecido muchos conceptos. Intenta recordar.



La primera fase o etapa de cualquier ciclo de vida de una aplicación informática es la denominada especificación de [ ]. El objetivo de la misma es doble:

- Por un lado, se intentan identificar las [ ] del cliente para definir el problema a resolver.
- Además, se realiza un estudio de viabilidad [ ] y técnico para decidir si son factibles las posibilidades de realización.

Las principales técnicas de definición del problema que se usan en la práctica son dos: [ ] y [ ].

El [ ] de Requisitos es el producto generado en esta primera etapa, en él se recogen de una forma organizada dos tipos de requisitos: [ ] y [ ].

El siguiente paso del ciclo de vida es el [ ] del sistema. Ahora nos centramos en [ ] debe hacer el sistema.

En toda aplicación informática se distinguen principalmente dos partes: los [ ] y los [ ].

Los [ ] son la parte [ ] de la aplicación. Reflejan las funciones o tareas que debe realizar la aplicación. Muestran cómo se transforman los datos.

Los [ ] son la parte [ ] de la aplicación. Se refieren a la información que se necesita almacenar.

Así, y como resultado del análisis, se obtendrán:

- El Modelo de [ ] : recoge qué funciones, actividades, tareas, acciones, debe realizar la aplicación y cómo manejar los datos.
- El Modelo de [ ] : describe la información que maneja la aplicación, los datos que debe almacenar y muestra cómo organizarla.

En la práctica, las técnicas usadas en el análisis de procesos son: los diagramas de [ ] de datos ([ ]) y el [ ] de datos ([ ]).

Cuatro son los elementos que forman un [ ] : los [ ] (para identificar funciones), los [ ] de datos (para representar el camino que siguen los datos), los [ ] (representan los datos desde el punto de vista estático) y las [ ] [ ] (para identificar los potenciales usuarios externos del sistema).

Para el análisis y especificación de datos hemos usado el Modelo Entidad/Relación [ ] . Los elementos que forman este modelo son: las [ ] (que se representan mediante un rectángulo) y los [ ] .

En la fase de diseño pasamos de preguntarnos ¿ [ ] hay que hacer? a ¿ [ ] hay que hacerlo?

Para el diseño de procesos, se ha presentado la técnica de los Diagramas de [ ] de Constantine [ ] . Estos diagramas están formados por: [ ] (subprogramas representados por [ ]) y [ ] (representadas por [ ]).

La implementación funcional del sistema consiste en la [ ] de todas las funciones identificadas y diseñadas. Así, basándonos en los principios de la programación [ ] , implementaremos los diferentes módulos diseñados con los Diagramas de [ ] de Constantine [ ] . Para ello haremos uso del lenguaje [ ] como lenguaje general que, según se elija, se codificará a un lenguaje de programación específico (PHP, ASP, Java, etc).

La implementación de los datos implica, además de la [ ] formal de cada tipo de dato a considerar (registros, arrays, cadenas,...), las [ ] básicas que se pueden hacer sobre ellos, esto es, creación, lectura, escritura, modificación y [ ] .

Los tipos de [ ] que se llevan a cabo en la etapa de pruebas del sistema se clasifican: según la [ ] en que se realizan y según el [ ] de realización.

La etapa del ciclo de vida llamada implantación es también conocida como puesta en [ ] .

En la etapa de mantenimiento del sistema se llevan a cabo: la [ ] de errores, los cambios de [ ] y las [ ] y ampliaciones.

Enviar

Aviso legal

El presente texto (en adelante, el "**Aviso Legal**") regula el acceso y el uso de los contenidos desde los que se enlaza. La utilización de estos contenidos atribuye la condición de usuario del mismo (en adelante, el "**Usuario**") e implica la aceptación plena y sin reservas de todas y cada una de las disposiciones incluidas en este Aviso Legal publicado en el momento de acceso al sitio web. Tal y como se explica más adelante, la autoría de estos materiales corresponde a un trabajo de la **Comunidad Autónoma Andaluza, Consejería de Educación (en adelante Consejería de Educación**).

Con el fin de mejorar las prestaciones de los contenidos ofrecidos, la Consejería de Educación se reservan el derecho, en cualquier momento, de forma unilateral y sin previa notificación al usuario, a modificar, ampliar o suspender temporalmente la presentación, configuración, especificaciones técnicas y servicios del sitio web que da soporte a los contenidos educativos objeto del presente Aviso Legal. En consecuencia, se recomienda al Usuario que lea atentamente el presente Aviso Legal en el momento que acceda al referido sitio web, ya que dicho Aviso puede ser modificado en cualquier momento, de conformidad con lo expuesto anteriormente.

1. Régimen de Propiedad Intelectual e Industrial sobre los contenidos del sitio web

1.1. Imagen corporativa

Todas las marcas, logotipos o signos distintivos de cualquier clase, relacionados con la imagen corporativa de la Consejería de Educación que ofrece el contenido, son propiedad de la misma y se distribuyen de forma particular según las especificaciones propias establecidas por la normativa existente al efecto.

1.2. Contenidos de producción propia

En esta obra colectiva (adecuada a lo establecido en el artículo 8 de la Ley de Propiedad Intelectual) los contenidos, tanto textuales como multimedia, la estructura y diseño de los mismos son de autoría propia de la Consejería de Educación que promueve la producción de los mismos.

Descargar PDF