Caso práctico

*Llegados a este punto, **Víctor** reflexiona sobre todo lo que ha aprendido hasta ahora, que no es poco, desde que empezó a formarse como analista en **SI Andalucía**. En este momento comprende completamente lo que **María** trataba de explicarle al principio de su formación, cuando le decía que disponer de un buen **Análisis** y un buen **Diseño** de una aplicación, cuando se va a empezar a programar, es tan importante como disponer de unos buenos planos cuando se comienza a construir una casa. Ahora sabe que antes de ponerse a programar en proyectos de cierta entidad es fundamental realizar un análisis y diseño previos, si se quiere realizar un trabajo con criterios empresariales de calidad y eficiencia. Si no se hace así, un proyecto puede crecer de manera desmedida y llevar muchas más horas de trabajo que las que se pensaban inicialmente.*

__María__ le felicita por el esfuerzo de aprendizaje que le ha traído hasta aquí, y se alegra de poder contar con él para empezar a sacar adelante el análisis y diseño de alguno de los proyectos en los que están inmersos. Cuenta con él, con sus ganas de aprender y con sus recién adquiridos conocimientos.

*Y para demostrárselo le ha asignado el análisis y diseño de una aplicación de pequeño tamaño. Como tienen varios proyectos de aplicaciones pendientes de desarrollar para varios clientes, **María** le dice a **Víctor** que elija el que más atractivo le parezca, y comience a realizar el análisis del mismo.*

*__María__ le comenta también que aunque ha aprendido muchísimo, debe tener presente que un analista nunca termina de formarse, y que siempre va a seguir teniendo bastantes puntos en los que profundizar, o actualizarse. Así, aunque hoy en día, en **SI Andalucía** utilizan análisis estructurado en la mayor parte del análisis y diseño de sistemas, en algunos proyectos utilizan otras metodologías, y llevan un tiempo planteándose, que quizá en un futuro cercano deban utilizar otras metodologías, más cercanas a la orientación a objetos, que existen o que irán apareciendo y quizá incluso abandonar el uso del análisis estructurado.*

*Por ello, dado que **Víctor** va a realizar el análisis de su primer proyecto serio, va a contar con la ayuda de **María**, con la que colaborará trabajando en equipo para realizar las tareas necesarias.*

*__Víctor__ elige la aplicación a desarrollar de la lista que le suministra **María**, y se empiezan a planificar el trabajo. Está seguro de que va a tener que utilizar todo lo que ha aprendido hasta ahora sobre análisis y diseño estructurado, y también intuye que deberá enfrentarse a problemas que no había visto hasta ahora.*

Pero a pesar de eso, encara el reto con optimismo, y se pone a trabajar.

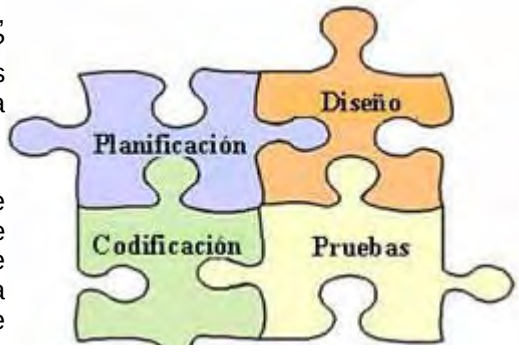
Introducción

Al igual que Víctor, ahora también tú debes **aplicar** todo lo aprendido a la realización de un **proyecto**, es decir, al **análisis y diseño de una aplicación**. Se trata de demostrar que has asimilado adecuadamente todo lo que has aprendido hasta ahora, y de que lo integres en un único trabajo de complejidad media, para que sea abarcable en unas **tres semanas** aproximadamente. Lo cierto es que no es mucho tiempo, incluso para una aplicación de tamaño y complejidad media, así que debes emplearte a fondo.

A la hora de valorar y **evaluar tu trabajo**, será muy importante que queden bien reflejadas todas las funcionalidades y procesos que se especifican en el enunciado del problema, que no se olviden flujos de datos, o entidades que intervengan claramente en el sistema, etc. Por ello, es importante leer varias veces y comprender bien el problema al que te enfrentes. Eso hará que realices unos DFDs correctos. Recuerda que **no hay una solución única** para un problema, sino que puede haber muchas soluciones válidas, siempre que cumplan con las especificaciones y restricciones que se tengan que cumplir.

Consideraciones generales para todos los proyectos

Cuando te matriculaste en este módulo, ¿tenías en mente alguna empresa, o en general, algún sistema de información, que te gustaría **modelar**? Seguramente alguna vez a lo largo de las unidades de este módulo habrás pensado en alguna aplicación práctica de lo que estabas aprendiendo para uso personal, o para algún conocido o amigo.



La experiencia demuestra que los mejores **proyectos** son siempre los que más **motivan** a las personas que los desarrollan. Dicha afirmación se puede aplicar a todos los órdenes de la vida, y en particular al análisis y diseño de sistemas. Quizá tengas en mente **montar tu propia empresa**, y ahora puede ser un buen momento para analizar y diseñar el sistema de

información que constituya la misma.

¿Qué queremos decir con esto?

Pues que a continuación, en los apartados siguientes te vamos a hacer algunas **sugerencias** sobre posibles **proyectos** que puedes realizar, con el objeto de que elijas uno. Pero esos proyectos, al no estar tan cercanos a ti, como quizá uno que a ti se te ocurra, puede que te resulten poco atractivos.

Por el contrario, si tú mismo te propones desarrollar el análisis y diseño de un sistema con el objetivo concreto de poder utilizarlo en la realidad, sí que vas a tener mucho más claro lo que necesitas que ese sistema haga, vas a reparar en detalles que, en otros casos, pasarían inadvertidos para ti, debido a que verías más lejano o extraño a ti cualquier otro ejemplo que te propongamos, y así vas a disponer de unos requerimientos de mucha más calidad que te van a ayudar en la realización del trabajo.

Además, vas a estar mucho más motivado, porque el trabajo que hagas va a tener un doble fruto:

- Te va a permitir aprobar este módulo profesional, pero al mismo tiempo,
- te va a proporcionar una herramienta útil para el propósito que te hayas fijado.

Y si además de cumplir con la tarea de este módulo tu trabajo resulta útil para ti, o para algún conocido tuyo, mucho mejor. Puede que incluso tengas posibilidad de usar ese análisis para el **proyecto final** de algún otro módulo de este ciclo, en el que tengas que programar una aplicación. Si así fuera, te resultaría de gran ayuda para conseguir una aplicación mejor, y para desarrollarla en menos tiempo.

> **Eso sí, antes de empezar a desarrollar la aplicación, debes ponerte en contacto con tu tutor, e indicarle qué proyecto vas a realizar.**

Si es uno **propio** que tú mismo te has puesto, deberás redactar un **documento** indicando en qué consiste, y presentárselo a tu tutor para que te dé el visto bueno, o te indique las mejoras que debes tener en cuenta para que cumpla con los requisitos generales que debe cumplir cualquier proyecto.

Dicho esto, hay que tener en cuenta que, entre la multitud de ideas que se os pueden ocurrir, no todas van a tener la misma **complejidad**, ni la misma **dificultad**, o el mismo **tamaño**. Y ya que tenemos que evaluaros con unos criterios lo más objetivos y equitativos posible, es necesario que fijemos las características básicas que deberá incluir cualquier proyecto que realicéis, tanto si lo proponéis vosotros mismos como si cogéis uno de los enunciados que mostramos como propuestas en esta unidad.

Requisitos mínimos a cumplir por todos los proyectos

¿Pero, qué debes tener en cuenta para conseguir una buena calificación de tu proyecto?

El proyecto consiste básicamente en el análisis y diseño de una aplicación informática por parte de cada uno de los alumnos (de forma individual) que debe incluir los siguientes puntos:

- **Cuestionario de entrevista** que se le haría a una persona que trabaje o sea usuario en el sistema objeto de estudio, de modo que como analistas, obtuviéramos información de utilidad para nuestro diseño del sistema.
- **DFDs del sistema**. De modo que se realice el diagrama de contexto, el diagrama de sistema y se expanda o explosione dicho diagrama hasta llegar a procesos primitivos. Se deberán explicar los diagramas, y en particular cualquier detalle que no se observe a simple vista en la especificación. Además, se deben observar la convenciones en los nombres, tales como que los procesos empiecen con un verbo en infinitivo.
- Diccionario de Datos, en el cual se describan todos los flujos de datos, almacenes, y entidades externas que se hayan observado en los DFDs.
- Miniespecificación **en** pseudocódigo de los procesos primitivos, es decir, de los que no se pueden descomponer más.
- **Diagrama Entidad-Relación** y explicación del mismo: entidades identificadas, por qué se han adoptado tales o cuales cardinalidades, etc.
- **Modelo relacional** obtenido a partir del modelo Entidad-Relación y posterior normalización.
- Diseño de al menos **una** interfaz **gráfica** correspondiente a una supuesta pantalla de la aplicación, para dar de alta algunos datos.
- Diseño de al menos **un informe** que generaría la aplicación una vez construida.
- También se puede incluir cualquier otra información o documentación que se considere oportuna por vuestra parte, para facilitar la comprensión de vuestro análisis y diseño.

Ejemplo guía

Todas esas especificaciones generales del apartado anterior pueden parecerte poco precisas. La idea es deliberadamente dejar el ejercicio bastante **abierto** y permitirte desarrollar tu proyecto de una forma flexible, al mismo tiempo que demuestras la adecuada adquisición de los conocimientos y destrezas que como analista debes poseer al terminar este módulo.

- Es posible que en la realización de tu análisis y diseño necesites establecer algún otro requisito, que a tu juicio sea necesario. En tal caso, explícalo adecuadamente a tu tutor en la documentación que acompañes, para que lo tenga en cuenta en la evaluación.
- Ten presente que cualquier suposición que hagas deberá estar justificada, y suponer algún tipo de mejora sobre lo que inicialmente se había considerado en el enunciado.

Pero para que tengas una **idea más concreta** de lo que se te está pidiendo que hagas, te vamos a mostrar aquí un ejemplo desarrollado. Seguramente te dará algunas ideas, y te aclarará bastante el trabajo, teniendo en cuenta que lo que debes conseguir es el análisis y diseño de una aplicación con una funcionalidad parecida.

Ten en cuenta, que en este **ejemplo** se realizan antes los DFDs que el modelo entidad-relación.

¿Debéis hacerlo así vosotros en vuestro proyecto?

No es obligatorio, en la práctica, hay analistas que lo hacen así, y otros que ven más claro realizar el modelo entidad-relación y más tarde los DFDs.

Análisis y diseño de la tienda de informática: "INFOANDALUS". Especificación del problema

La descripción del problema, en caso de elegir uno de los ejercicios propuestos, os la proporcionamos nosotros. En caso de que el proyecto lo propongáis vosotros, tendréis que redactarla también vosotros. Sería por tanto, el enunciado del ejercicio, que en este caso de ejemplo es el siguiente.

Tienda de informática InfoAndalus

El propietario de la tienda de informática llamada "InfoAndalus" pretende automatizar la gestión de su tienda dedicada a la venta de ordenadores y material ofimático: cds, consumibles para impresora, etc. La aplicación informática que desea que se realice, debe ocuparse de las ventas mediante pedido. Esta tienda sólo vende al por mayor, no a cualquier cliente de la calle, sino a determinados clientes que le suelen comprar grandes cantidades.

Tras una serie de entrevistas con el propietario, para recabar información, Víctor y María se dan cuenta de que, entre otras cosas, la aplicación a realizar va a necesitar un mantenimiento de los clientes que van a comprar ordenadores.

Se han percatado también, mediante la realización de las pertinentes y adecuadas preguntas, de que al propietario de la tienda, le interesa almacenar los datos relevantes de los clientes, en concreto han averiguado que sobre todo hay que guardar:

NIF
Nombre
Dirección
Teléfono
Código postal

Población

Provincia

Han averiguado que los clientes formulan pedidos que serán debidamente registrados para la confección de los correspondientes albaranes y facturas.

El propietario les ha comentado que se ha de tener en cuenta que puede guardarse información de clientes que no tengan ningún pedido registrado todavía en la tienda.

En cuanto a los pedidos que haga un cliente, cuando éste realice uno, la aplicación debe comprobar si hay existencias suficientes para atenderlo; si es así, el pedido se marca como servido, se actualiza el número de existencias de cada artículo y se genera el albarán.

Este albarán es el que le llega a un mozo de almacén, que prepara los artículos para ser enviados. Sí no hay existencias suficientes para atender el pedido en su totalidad, entonces se marca como pedido pendiente y se queda a la espera de recibir los artículos necesarios para poder ser servido.

Los pedidos pueden cancelarse antes de que se hayan servido y no se admiten devoluciones de pedidos.

Un mozo de almacén debe ser el encargado de introducir en la aplicación los artículos que se reciban de los proveedores que suministran a la tienda, para actualizar las existencias de dichos artículos. Estos artículos se sirven a la tienda acompañados con un albarán de entrega.

La dirección de la empresa será quien seleccione los proveedores y artículos con los que trabaje la empresa. Los datos que interesa que maneje la aplicación a construir, respecto a los artículos son:

su nombre,

descripción,

precio de venta al público,

número de existencias

y stock mínimo que debe haber. Puede darse el caso de que un artículo se suministre por varios proveedores.

Como la dirección de la empresa suministrará a la aplicación los datos relativos a los proveedores, debemos tener en cuenta que el sistema deberá contemplar un mantenimiento de proveedores.

Los datos a tener en cuenta para cada proveedor serán:

CIF,

nombre,

dirección,

teléfono,

fax,

código postal,

población

y provincia.

La dirección podrá dar de alta proveedores en la base de datos aunque no suministren ningún artículo. Además, se podrá dar de baja a un proveedor siempre que en la base de datos no haya ningún artículo suministrado por él.

Para la dirección de la empresa, el sistema debe generar un informe de artículos por debajo del stock mínimo, teniendo en cuenta los pedidos pendientes que incluyen estos artículos, para calcular la cantidad a pedir a los proveedores de cada artículo. El informe mostrará por cada articulo:

el stock mínimo,

el número de existencias

y el número de unidades contenidas en pedidos que no se han podido servir aún por falta de existencias.

También se debe generar para la dirección, un informe listando un resumen de las facturas enviadas a los clientes.

Además, el sistema tiene que generar un informe de precios de compra de artículos a proveedores, también para la dirección de la empresa. Dicho informe incluirá los precios de cada artículo según el proveedor que lo suministra.

Víctor pregunta al propietario que cuándo se imprimen las facturas. El propietario responde que cada mes, el sistema imprimirá las facturas correspondientes a las ventas realizadas durante el mes anterior. Los clientes que hayan efectuado varias compras el mismo mes, tendrán varias facturas.

El sistema gestiona los pedidos pendientes de servir. Comprueba si ya se dispone de las cantidades solicitadas de cada artículo, ya que a la tienda entran nuevos artículos con mucha frecuencia.

Para simplificar el presente supuesto práctico, omitimos lo referente a la gestión de pedidos a proveedores que la tienda haga para aumentar sus existencias.

"INFOANDALUS". Cuestionario de entrevista

¿Recuerdas qué se pretendía con las entrevistas, y que había que prepararlas minuciosamente? Seguro que tienes claro que elaborar un buen cuestionario para la entrevista es de vital importancia

Se debe proponer un cuestionario, que sería el que el analista haría a las personas implicadas en el sistema, a la gerencia, a los operarios, para averiguar:

- cómo operan habitualmente,
- sus necesidades,
- qué esperan de la automatización del sistema,
- etc.

Se trata de hacer las preguntas necesarias para obtener una especificación como la dada en el epígrafe anterior.

Hay que tener en cuenta, que quizás no baste con una única entrevista, debido a que con la información que obtengamos en una primera entrevista, nos pongamos a trabajar en el problema, y descubramos que nos faltan detalles sobre algún aspecto sobre el cual no habíamos caído inicialmente. La práctica, la experiencia como analistas, nos hará ir mejorando con el tiempo, de tal modo que vayamos afinando y mejorando nuestros cuestionarios.

Por ejemplo, para el caso que nos ocupa, algunas preguntas interesantes serían las que siguen a continuación.

¿Cuántas personas trabajan en la tienda incluida la dirección?
¿Qué funciones o tareas desempeñan cada una de estas personas en la tienda?
¿Cuál es el proceso habitual en la realización de un pedido por parte de un cliente?
¿Quién prepara los pedidos cuando se reciben?
¿Quién y dónde guarda los datos del pedido a preparar?
¿Se genera ticket o/y factura de compra cuando se sirve un pedido?
¿Es posible que un cliente cancele un pedido?
¿Qué se hace normalmente cuando ocurre esto?
¿Qué datos nos interesa almacenar sobre los clientes?
¿El sistema debe obligar a que se introduzcan todos los datos en el proceso de alta de un nuevo cliente, o se puede permitir la omisión de algún dato?
¿Qué datos nos interesa almacenar sobre los proveedores?
¿Qué criterios se siguen para realizar los pedidos a nuestros proveedores?
¿Nos ofrecen los proveedores algún tipo de descuentos?
¿Qué informes necesita la dirección y en qué soporte?

"INFOANDALUS". Diagramas de flujo de datos

¿Recuerdas la utilidad de los diagramas de flujo de datos?

Tras obtener los datos necesarios de las entrevistas realizadas a todos los implicados en el sistema y analizarlos, podemos pasar a **realizar los diagramas de flujo de datos necesarios para entender de una manera gráfica el sistema**.

Por tanto comenzamos esta fase cuando disponemos de toda la **información** que necesitamos para comenzar a diseñar. De nuevo insistimos aquí, en que puede que hayamos olvidado preguntar algún detalle, y que conforme avancemos en el diseño nos demos cuenta. Esto no supone ningún problema, ya que podríamos consultarlo a quien proceda y ya está. **Sí que puede representar un problema, haber olvidado algo importante, porque nosotros como analistas, no hayamos sido lo suficientemente hábiles para recabar la información importante**. O también podemos tener el problema de que se nos pida una cosa, y estando inmersos en el diseño, se nos cambie la especificación de lo que se quería, por otra cosa.

En este último caso, tenemos un **problema muy grande**, ya que todo el diseño que lleváramos realizado hasta que nos hayan notificado las nuevas especificaciones, es probable que no nos sirva, y tengamos que **rediseñar** todo. Para evitar esas situaciones, o al menos, si no podemos evitarlas, sí que podemos contemplar dicha posibilidad, y que nuestro trabajo no haya sido gratuito, hay que tener en cuenta que **se debe formalizar un contrato con el cliente**. Así, en ese contrato se pacta:

- El trabajo que le vamos a hacer y
- Cuánto le vamos a cobrar.
- Si resulta que el cliente se ha equivocado o se le ocurre que ahora quiere otra cosa, como tenemos un contrato, podremos exigirle un costo adicional, ya que eso no era lo pactado.

Adentrándonos ya en los DFDs, recordemos, que éstos, constituyen una técnica del análisis estructurado, cuyo objetivo fundamental es la descomposición de un problema complejo en otros más sencillos y manejables.

El **diagrama de contexto**, de acuerdo a la especificación que tenemos, es el siguiente:

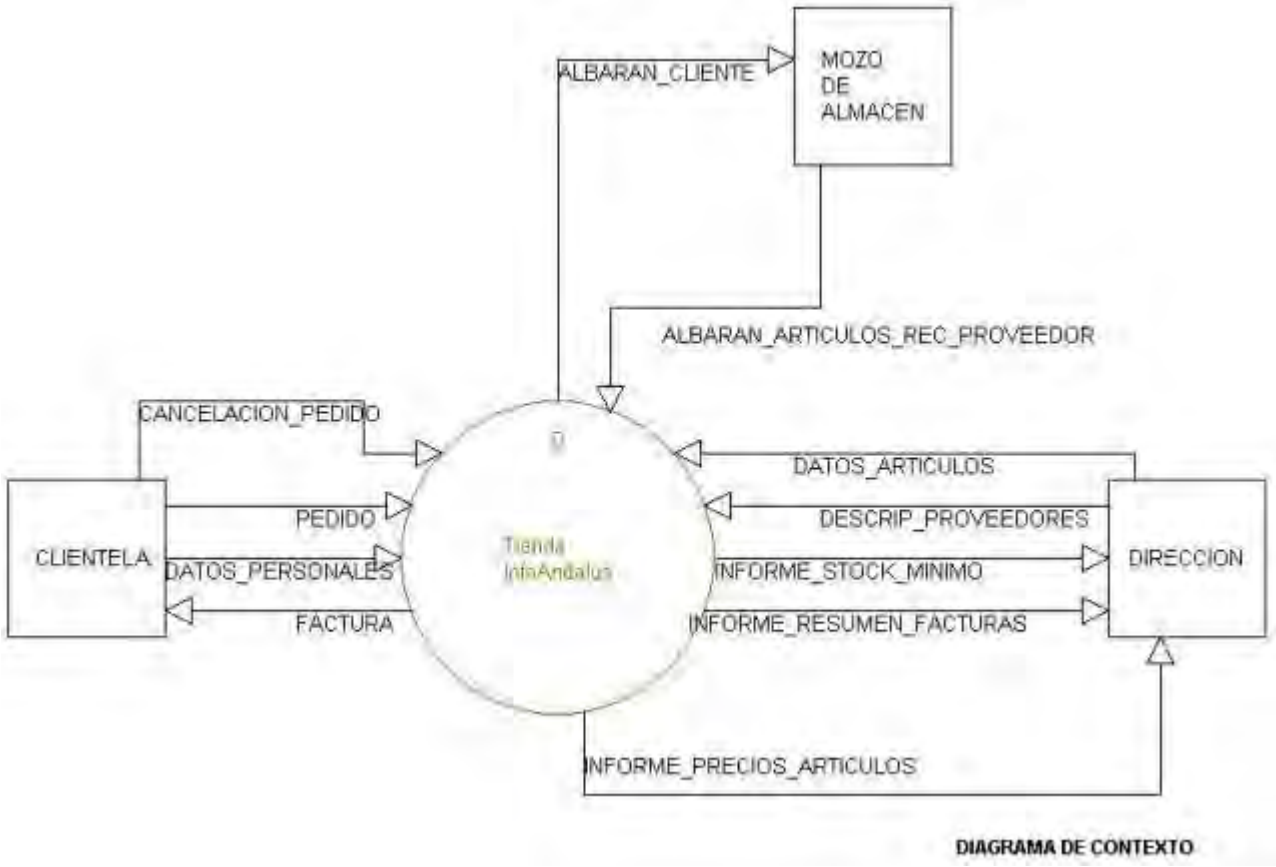


DIAGRAMA DE CONTEXTO

El **sistema** es la tienda en sí, y hemos considerado como entidades externas:

- clientela, entendiendo a esta entidad como cualquier cliente que va a comprar a la tienda,
- mozo de almacén, siendo esta entidad la que representa a cualquiera de los operarios que trabajan en el almacén de la tienda, y
- dirección, que representa a la dirección de la empresa o tienda.

Se puede apreciar como se ha intentado que el nombre de los flujos de datos sea lo más parecido posible a los datos que se pretende que representen.

También recordar en este punto, que en cualquier DFD, **los flujos que se representan son de datos y no de materiales**. Así, por ejemplo, no hay ningún flujo en este diagrama de contexto que represente a un ordenador fluyendo desde el sistema hasta un cliente. Sin embargo, sí que hay un **flujo factura**, que representa los datos que lleva una factura y que le llegarán al cliente por medio de la factura física, en papel, que le entregarán cuando compre un ordenador.

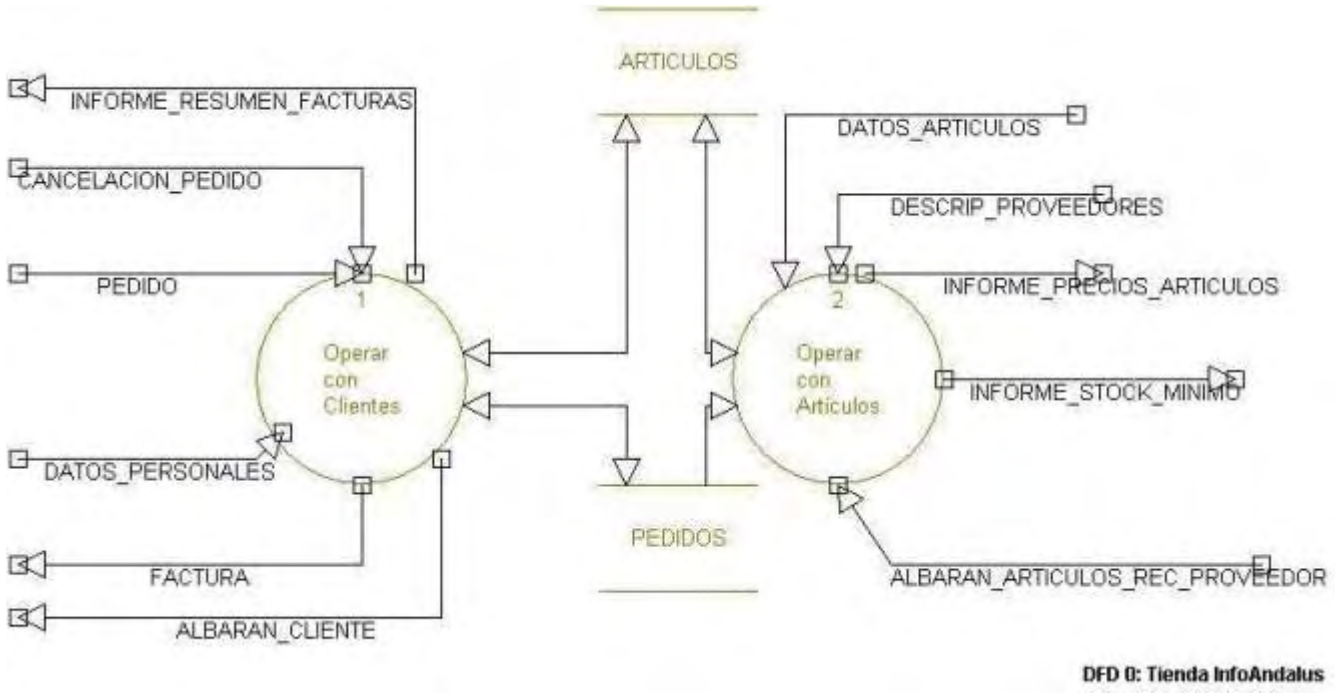
Del diagrama de contexto anterior, y tras pensar lo suficiente en la especificación que tenemos, podemos llegar a la conclusión de que **hay dos procesos fundamentales** que se realizan en el sistema, y que por tanto nos harán explosionar dicho diagrama, en el diagrama de sistema de más abajo. Estos dos **procesos** son por un lado los que tienen que ver con las operaciones relacionadas con lo que concierne a los clientes:

1. ***Operar con Clientes****, y por otro lado, las operaciones que tienen que ver con los artículos:*
2. ***Operar con Artículos***

Comentar aquí, que quizá otro analista considerara algún proceso más y no exactamente los que hemos observado, normalmente, no hay una única solución, sino que diez analistas, probablemente hagan diez análisis diferentes de un mismo sistema.

La cuestión es que dicho análisis sea riguroso y cumpla con los requisitos y especificaciones necesarias sin olvidar ni contradecir ninguno.

Por tanto, en nuestro caso, vemos claramente un proceso que tiene que ver con las operaciones a realizar con los clientes de la tienda, y otro proceso, que tendría que ver con las operaciones a realizar con los artículos de la tienda, como se puede apreciar en el diagrama siguiente.



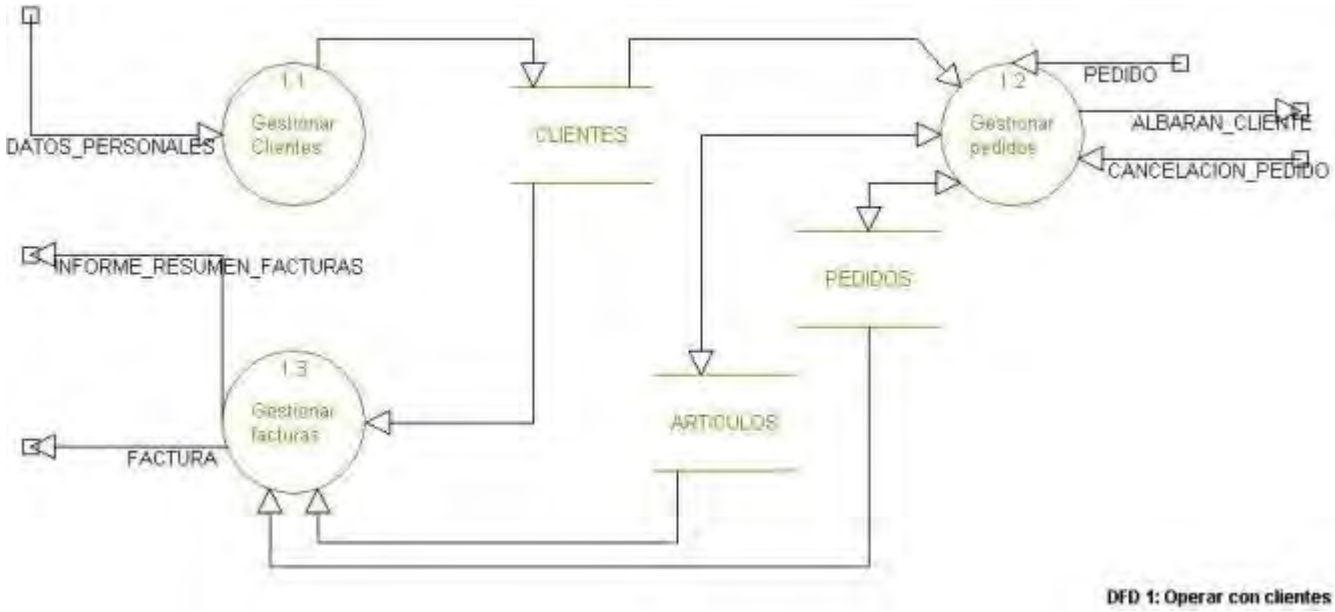
DFD 0: Tienda InfoAndalus

Observamos en este **diagrama de sistema** cómo nos aparecen dos almacenes.

- El almacén ARTICULOS, representa el repositorio en donde el sistema tiene almacenada la descripción, precio y resto de datos necesarios de cada uno de los artículos que comercializa la tienda.
- El almacén PEDIDOS, contendrá los pedidos que van haciendo los clientes a la tienda.

Es importantísimo también observar cómo en este diagrama de sistema **aparecen exactamente los mismos flujos de datos que aparecen en el diagrama anterior, o sea, en el diagrama de contexto, para no infringir la regla de balanceo de niveles.**

Una vez terminado el diagrama de sistema, analizamos la posibilidad de explosionar los procesos que hemos obtenido en este nivel. De esta manera, nos ponemos a pensar en qué procesos podríamos particionar la burbuja que representa Operar con clientes. En nuestro caso, hemos determinado que podemos considerar tres procesos, tal y como se muestra en el siguiente diagrama:



DFD 1: Operar con clientes

En el diagrama apreciamos que, como siempre, se está respetando el balanceo de niveles, y por ello, los flujos de datos que entran y que salen son los mismos que en el diagrama anterior a éste, entraban y salían del proceso numerado como 1, o sea, Operar con clientes.

Observamos además, que en este diagrama aparecen los **dos almacenes** que se veían en el diagrama de sistema y uno más, CLIENTES. Éste último almacén se utiliza como soporte para que el sistema guarde los datos de cada uno de los clientes que pueden comprar en la tienda.

No debemos olvidar en ningún momento, que con cualquier diagrama, lo que se pretende es que de un vistazo se comprenda lo que se desea representar, incluso que se comprenda o se intuya por alguien que no esté relacionado con el mundo de la informática. Por ello, por ejemplo si observamos en el diagrama anterior, el proceso numerado como 1.2, lo que pretendemos representar es que un cliente puede realizar un PEDIDO a la tienda. Los datos de ese pedido se almacenarán en un almacén PEDIDOS.

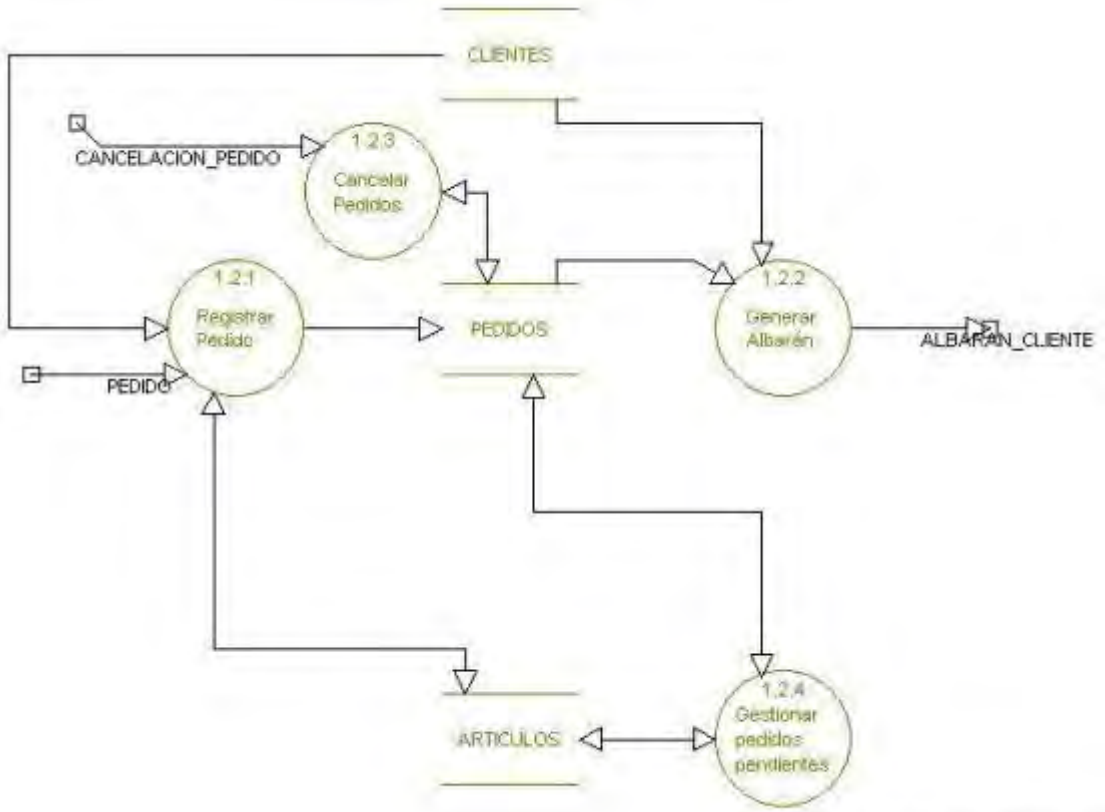
También se ve cómo sale un flujo de datos denominado ALBARAN_CLIENTE, que es el que sale hacia la entidad mozo de almacén, que representa el albarán que se genera cuando un cliente ha hecho un pedido a la tienda, y para ser servido se le pasa a un mozo de almacén para que lo prepare. Se aprecia en este proceso 1.2, que un cliente puede cancelar un pedido.

¿Por qué hay un flujo bidireccional desde ese proceso hacia el almacén de datos ARTICULOS?

Para averiguarlo, debemos pensar en lo que debe pasar en ese proceso, es decir, para saber si podemos satisfacer un pedido, tenemos que comprobar si hay unidades suficientes en el almacén de cada uno de los artículos del pedido. Así pues, habría una **flecha con los datos** desde el almacén hasta el proceso. Pero como estamos satisfaciendo ese pedido, debemos descontar existencias a las actuales, y de ahí la necesidad del flujo en la dirección desde el proceso al almacén, para actualizar las existencias del artículo, por tanto, por ello el flujo es en dos direcciones.

La interpretación de todos los DFDs va en esa línea, se trata por tanto de intentar plasmar gráficamente la especificación del problema que tenemos, intentando ir de lo general, en el diagrama de contexto, a lo particular, detallando cada vez más en los DFDs de los niveles sucesivos. Llegará un momento, en el que se llegará a procesos que ya no se pueden descomponer más, y esos son los que se especificarán en una miniespecificación, normalmente en pseudocódigo.

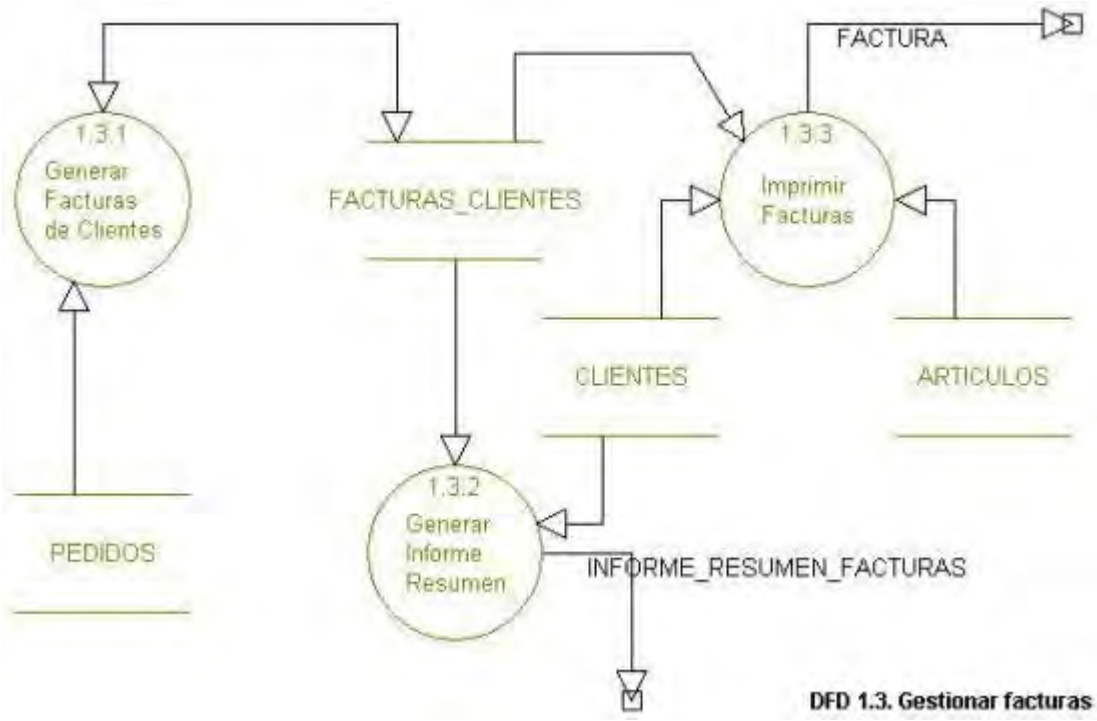
Por otro lado, el proceso 1.1. Gestionar Cliente ya no lo podemos descomponer más y haremos su miniespecificación posteriormente. El proceso 1.2. Gestionar pedidos sí consideremos que se puede descomponer, y sería de la forma que se muestra a continuación:



DFD 1.2. Gestionar Pedidos

Ahí se ve cómo lo que se había englobado con el nombre Gestionar pedidos, se detalla ahora más en esos cuatro procesos que se observan. Estos procesos ya no podemos descomponerlos más, así que también los describiremos posteriormente.

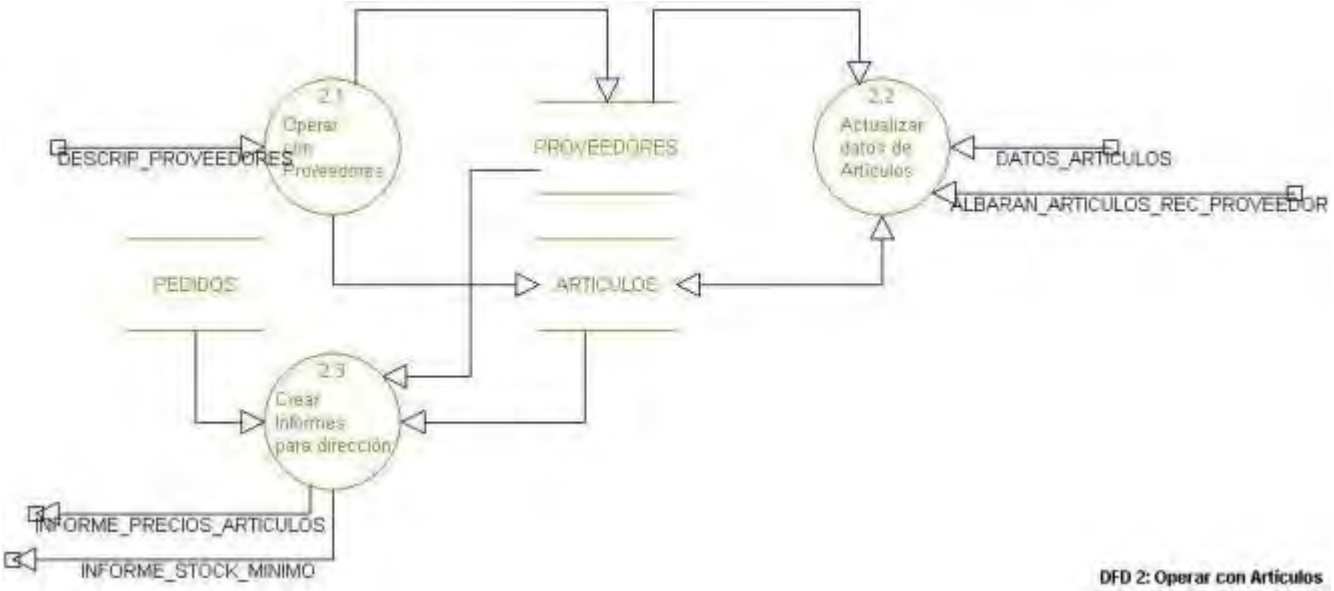
Volvemos ahora atrás y descomponemos el proceso 1.3. Gestionar facturas, que podríamos esquematizar de la siguiente manera:



DFD 1.3. Gestionar facturas

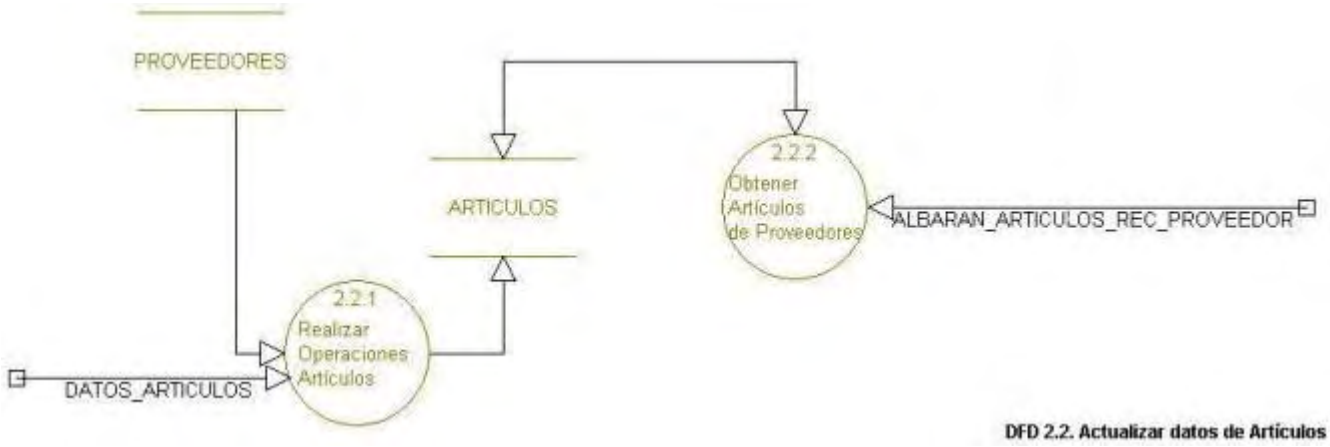
Este diagrama, consideramos que ya no lo podemos descomponer más, y de nuevo, posteriormente haremos la especificación de estos procesos.

En este punto, y dado que ya hemos terminado de descomponer todo lo concerniente al primer proceso, volvemos atrás y tenemos que descomponer el proceso 2. Operar con artículos. Una solución para descomponerlo, podría ser la que presentamos a continuación:



DFD 2: Operar con Artículos

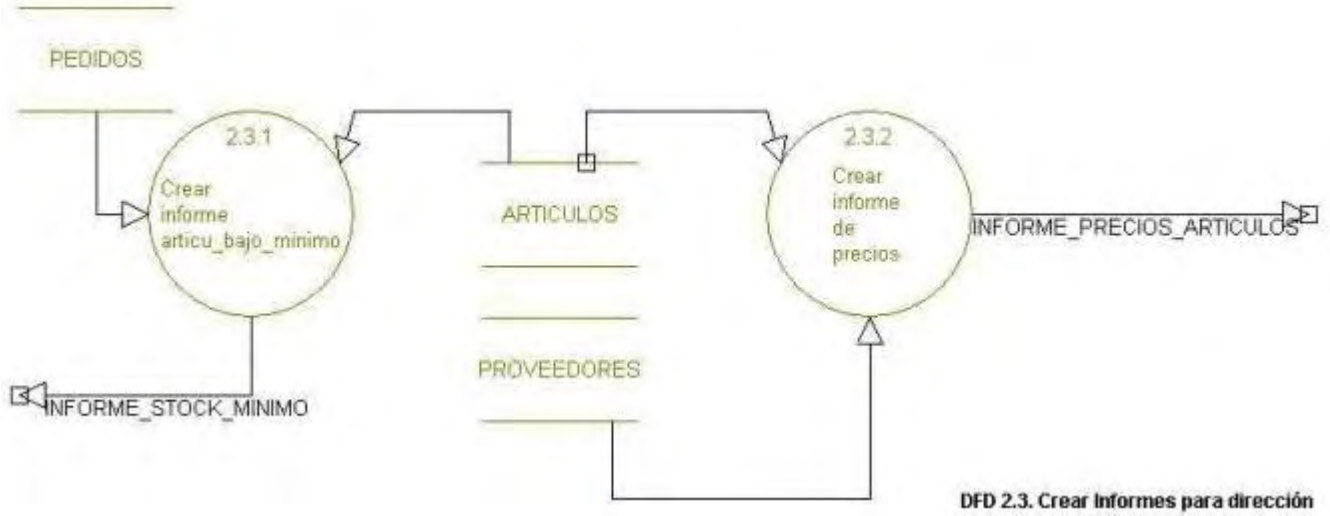
En este mapa, ya no podemos **descomponer** más el proceso 2.1. Operar con Proveedores, así que lo especificaremos con pseudocódigo. En cuanto al proceso 2.2. Actualizar datos de Artículos, se puede detallar un en nivel más, descomponiéndose como se muestra en la siguiente imagen:



DFD 2.2. Actualizar datos de Artículos

Estos dos procesos que hemos obtenido ya no podemos descomponerlos más, y después los especificaremos en pseudocódigo.

El proceso que sí se puede descomponer es el 2.3. Crear informes para dirección, que podemos explosionarlo en el siguiente diagrama que presentamos a continuación:



DFD 2.3. Crear Informes para dirección

Y con esto habríamos **concluido la tarea de descomponer** los DFDs. A continuación realizaremos el diccionario de datos y seguidamente la especificación de los procesos primitivos.

Insistir aquí, por un lado en que, si nos fijamos en la descomposición que hemos llevado a cabo, todos y cada uno de los diagramas que hemos realizado son conexos, no hay almacenes, procesos, o flujos de datos aislados, y por otro lado, en que hemos respetado en todo momento las reglas de balanceo en la descomposición de los sucesivos niveles.

"INFOANDALUS". Diccionario de datos

¿Recuerdas que en unidades anteriores vimos que el **diccionario de datos** sirve para recoger las definiciones de todos los datos que se manejan en el sistema?

Por tanto, se deben contemplar en este punto las definiciones de los flujos de datos, los almacenes y las entidades externas.

Presentamos a continuación dichas definiciones, y tras ellas, la miniespecificación de los procesos que ya no podemos descomponer más en los DFDs anteriores.

DICCIONARIO DE DATOS

ALMACENES

ARTICULOS = @código_artículo + descripción + unidades + PVP + stock_minimo + {CIF + precio}

CLIENTES = @NIF + nombre_cliente + dirección_cliente + teléfono_cliente + CP + localidad +

provincia

PEDIDOS = @número_de_pedido + fecha_pedido + NIF + estado + fecha_servido +

{num_linea + código_artículo + unidades + PVP}

estado = [pendiente | servido]

PROVEEDORES = @CIF + nombre_proveedor + dirección_proveedor + teléfono_proveedor + fax_proveedor + CPProveedor + localidad_proveedor + provincia_proveedor

FACTURAS_CLIENTES = @número_factura + NIF + número_de_pedido + fecha_factura + { núm_linea + código_artículo + cantidad + PVP } + IVA

ENTIDADES EXTERNAS

DIRECCION = Representa al departamento de dirección de la empresa

CLIENTELA = Representa a los clientes que van a comprar a la tienda

MOZO DE ALMACEN = Representa a los operarios que tiene la tienda en el almacén

FLUJOS DE DATOS

ALBARAN_CLIENTE = fecha_venta + datos_cliente + número_de_pedido + fecha_pedido + { código_articulo + descripción + unidades }

ALBARAN_ARTICULOS_REC_PROVEEDOR = código_artículo + número_de_unidades

CANCELACION_PEDIDO = número_del_pedido

DATOS_ARTICULOS = código_artículo + descripción_artículo + existencias + PVP + stock_mínimo + { CIF + precio_compra}

DATOS_PERSONALES = NIF + nombre + dirección + teléfono + código_postal + localidad + provincia

DESCRIP_PROVEEDORES = CIF + nombre_proveedor + dirección_proveedor + teléfono_proveedor + fax_proveedor + CP_proveedor + localidad_proveedor + provincia_proveedor

FACTURA = número_factura + datos_cliente + número_de_pedido + fecha_factura +{ núm_linea + código_artículo + descripción_artículo + cantidad + PVP + total_línea_factura} + total_sin_IVA + IVA + total_factura_con_IVA

INFORME_STOCK_MINIMO = { código_artículo + existencias + stock_mínimo + unidades_pedidas_sin_servir }

INFORME_PRECIOS_ARTICULOS = { código_artículo + descripción + {CIF + nombre_proveedor + precio}}

INFORME_RESUMEN_FACTURAS = fecha_actual + {número_factura + NIF + nombre + total_factura } + total_mensual

PEDIDO = fecha_pedido + NIF + {código_artículo + cantidad }

"INFOANDALUS". Miniespecificaciones

Con las miniespecificaciones, pretendemos especificar en pseudocódigo aquellos procesos que consideramos que ya no se pueden descomponer en otros.

En cuanto a las **convenciones que se han seguido** para redactar las miniespecificaciones, comentar que:

- Se ha tratado de documentar cada miniespecificación, línea a línea. Cada línea que representa un comentario empieza con dos asteriscos.
- Se utiliza LEER algo, para indicar que se lee bien desde teclado o en un almacén.
- Se utiliza ESCRIBIR algo, para indicar que se escribe bien a la pantalla o en un almacén. En ese último caso, sería ESCRIBIR algo, nom_almacén
- Se usa resultado = BUSCAR(Nif, AL) , para indicar que se busca en el almacén de nombre AL, una fila de datos

cuyo dato NIF sea Nif. Si se encuentra, se guardará un valor verdadero, en resultado, y si no se encuentra, se guardará un valor falso en resultado.

- Se usa IMPRIMIR albarán, para mandar a la impresora el albarán.
- Se emplea SUPRIMIR PEDIDOS para borrar el registro actual del almacén PEDIDOS.
- Se utiliza AVANZAR CASA para avanzar en el almacén CASA, al siguiente registro. Así, por ejemplo, si el almacén CASA tuviera el siguiente contenido que se muestra, y se supone que el puntero que señala en qué registro estamos, señala al primer registro, al ejecutarse la instrucción que nos ocupa, dicho puntero se situaría en el segundo registro, o sea, en el de Casa verde clara:

CODIGO	NOMBRE	TAMAÑO
000001	Casa azul	Grande
000002	Casa verde clara	Pequeña

- Se utiliza ACTUALIZAR código_articulo, ARTICULOS, (datos), para indicar que en el almacén ARTICULOS, se actualiza la información del artículo cuyo código es código_articulo con los datos datos.

Se han realizado las siguientes miniespecificaciones, que se encuentran disponibles en los archivos que puedes obtener de los siguientes enlaces:

Archivo miniespecificaciones

- El proceso 1.1. Gestionar Clientes, describe las opciones que permiten dar de alta un cliente, borrar un cliente o modificar un cliente en el almacén Clientes.
- El proceso 1.2.1. Registrar pedido, se puede observar cómo describe el proceso de registrar un pedido, comprobando si hay suficiente mercancía en el almacén como para ser servido o no.

En el proceso 1.2.2. Generar Albarán, puede observarse cómo se solicitaría la introducción al usuario de dos datos: El NIF del cliente y el número de pedido. Entonces se busca en el almacén PEDIDOS el pedido cuyo número se introdujo. Si se encuentra, en encontrado se guardará un valor verdad, con lo que si se cumple la condición que se comprueba con la instrucción condicional SI, se buscará el cliente cuyo NIF es el introducido, y si se encuentra se imprimirá el albarán buscado.

El proceso 1.3.1. Generar Facturas de Clientes lo que hace es lo siguiente:

Ordena los registros del almacén PEDIDOS ascendentemente por fecha, localizar el primer registro del almacén, y mientras haya registros de fecha el mes anterior, escribe en el almacén FACTURAS_CLIENTES los datos del pedido. Además marca ese pedido como ya facturado en el almacén PEDIDOS.

Una miniespecificación que merece un comentario especial es la del **proceso** 1.2.4. Gestionar Pedidos Pendientes**.** Como se puede ver, no se ha empleado pseudocódigo, sino que se ha dejado comentada. Estos comentarios son los que recibiría el equipo de programadores en lugar de un pseudocódigo. Y como en el recurso se indica, serían los programadores los que se encargarían de implementarla de la manera más adecuada, teniendo en cuenta las características del lenguaje de programación que se utilizará.

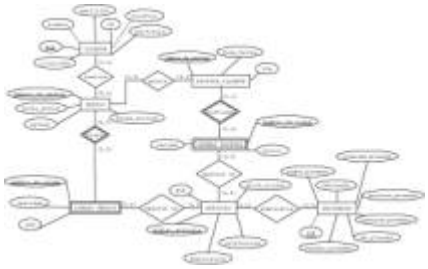
Por tanto en este último caso, se trata de una consideración que hacemos como analistas y diseñadores, de que debe ser el programador el que decida la manera más óptima de realizar la implementación del código que se necesita para realizar el proceso.

El resto de miniespecificaciones, no debe suponer mayor problema a la hora de entenderse, si hemos entendido las que acabamos de comentar.

"INFOANDALUS". Modelo Entidad/Relación

Como ya estudiamos en la unidad dedicada a la modelización conceptual, el **modelo Entidad-Relación** es la técnica de análisis y especificación de datos más ampliamente utilizada. El diagrama Entidad-Relación pretende representar y definir los datos que se almacenan en un sistema de información de una manera conceptual, y por ello sin entrar en detalles de su tratamiento y manipulación.

En el siguiente enlace encontrarás el diagrama Entidad-Relación del ejemplo que nos ocupa, y que seguidamente comentamos.



Si recordamos, las entidades del modelo son conceptos que reflejan los datos que interesan a la aplicación, de los cuales se necesita almacenar información en la base de datos del sistema. Los atributos son las características o propiedades que presenta una entidad.

En base a esos conceptos, podemos observar en nuestro diagrama entidad-relación, cómo hemos identificado las entidades:

- cliente,
- pedido,
- factura,
- articulo,
- proveedor,
- lineas_factura y
- lineas_pedido.

Hacemos hincapié en que las entidades lineas_pedido y lineas_factura son débiles.

¿Por qué razón?

Pues debido a que su existencia depende de otra entidad. Así, lineas_pedido depende de la existencia de la entidad pedido, y lineas_factura depende de la entidad factura. Si se piensa es lógico ya que una factura estará compuesta de una o varias líneas. Una línea de factura no tiene sentido si no existe una factura de la que formar parte. El mismo razonamiento se aplica en el caso de la entidad lineas_pedido.

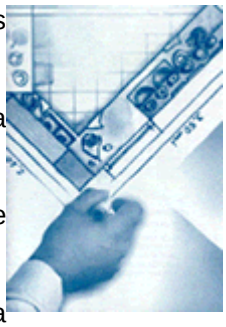
Resaltar también, que las relaciones "suministra" y "aparece en" deben leerse de derecha a izquierda, es decir, un artículo aparece en una línea de pedido, y un proveedor suministra un artículo.

También hay que apreciar en el diagrama, cómo la relación de dependencia tanto entre pedido y líneas pedido, como entre factura y líneas factura es en identificación. Recordamos que una dependencia se dice que es en identificación cuando las ocurrencias de la entidad débil no se pueden identificar simplemente mediante sus propios atributos, sino que se les tiene que añadir el identificador de la ocurrencia de la entidad fuerte de la que dependen. Esto es lógico, ya que pensemos por ejemplo que un artículo, digamos tinta marca "pepito", puede estar almacenado varias veces en el almacén correspondiente a líneas pedido, debido a que varios clientes hayan efectuado pedidos de dicho artículo.

¿Cómo sabríamos a qué pedido corresponden?

La solución es por tanto ayudarse del identificador de la entidad pedido.

"INFOANDALUS". Modelo Relacional: Paso a tablas



¿Recuerdas que en la unidad donde estudiábamos **diseño estructurado de sistemas**, veíamos cómo se realizaba el proceso de traducir el modelo Entidad-Relación al modelo relacional?

Ahora rescatamos esos conocimientos de nuevo, y pasaremos nuestro diagrama Entidad-Relación a tablas.

Tendremos en cuenta que indicaremos en negrita y subrayado la clave primaria, y en cursiva la clave ajena o foránea.

Sabemos que una entidad se transforma en una tabla. Así pues, empezando por ejemplo con la entidad cliente, obtendremos la siguiente tabla:

CLIENTES (**NIF**, nombre_cliente, direccion_cliente, telefono_cliente, CP, localidad, provincia)

En cuanto a la tabla de pedidos, en principio sería:

PEDIDOS (numero_de_pedido, fecha_pedido, estado, fecha_servido)

Pero como la relación entre la entidad cliente y la entidad pedido, denominada realiza, posee cardinalidad 1:N, y como la cardinalidad de la entidad de lado uno es (1,1), entonces se propaga el identificador de la entidad de cardinalidad máxima 1 a la que tiene cardinalidad N. Además la clave propagada es clave ajena en la tabla a la que llega. Por ello, la tabla quedará de la siguiente forma:

PEDIDOS (**numero_de_pedido**, fecha_pedido, estado, fecha_servido, *NIF*)

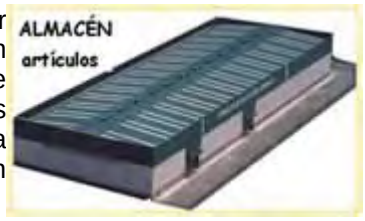
con NIF como clave ajena.

En cuanto a la entidad proveedores, se transforma en la siguiente relación:

PROVEEDORES (**CIF**, nombre_proveedor, dirección_proveedor, teléfono_proveedor, fax_proveedor, CPProveedor, localidad_proveedor, provincia_proveedor)

La entidad articulos, se transforma en la siguiente relación:

ARTICULOS (**codigo_articulo**, descripcion, unidades, PVP, stock_minimo)



La entidad débil lineas_pedido, presenta en principio tres atributos y está afectada por dos relaciones. Debido a la relación "tiene", que es 1:N, y recordamos que es la relación por la que esta entidad presenta dependencia en identificación de la entidad pedidos, se propaga el identificador numero_de_pedido a esta entidad lineas_pedido. Como decimos que es débil en identificación, el identificador de la entidad fuerte debe introducirse en la tabla de la entidad débil y formar parte de la clave de ésta. Así pues, ahora la relación sería:

LINEAS_PEDIDO (***numero_de_pedido*, numero_de_linea**, cantidad, PVP)

formando parte de la clave y siendo además clave ajena el atributo numero_de_pedido. Pero ahora, consideramos la relación correspondiente a que un artículo aparece en cero o muchas líneas de pedido. Esa relación es de nuevo 1:N y hace que se propague el atributo codigo_articulo a esta relación. Por ello, la relación queda:

LINEAS_PEDIDO (***numero_de_pedido*, numero_de_linea**, cantidad, PVP, *codigo_articulo*)

con código también como clave ajena de la tabla.

Ahora pasamos a considerar la entidad de facturas a clientes, y así para su transformación a relación hay que considerar primeramente que en principio presenta tres atributos. Después observamos que hay una relación denominada "genera", que es de tipo 1:1. Como las cardinalidades son (1,1) y (0,1), se pone la clave ajena en la entidad que participa con cardinalidad (0,1) y en caso de haber atributos en la relación, se colocarían en el extremo (0,1). Por tanto, la relación quedaría como:

FACTURA_CLIENTE (**numero_factura**, fecha_factura, IVA, *numero_de_pedido*)

En cuanto a la entidad lineas_factura, se sigue el mismo razonamiento que el mostrado para las líneas de pedido y por tanto la tabla resultante será:

LINEAS_FACTURA (***numero_de_factura*, numero_de_linea**, cantidad, precio, *codigo_articulo*)

Con lo que ya tendríamos todas las tablas necesarias. En estos momentos, hay que repasar las tablas obtenidas y normalizar aquellas que no lo estén. Habitualmente, se suele normalizar hasta tercera forma normal, 3FN.

"INFOANDALUS". Modelo Relacional: Normalizar tablas

¿Recuerdas que en unidades anteriores vimos que la normalización de tablas sirve para evitar redundancias, inconsistencias y otras propiedades no deseadas en las tablas que componen una base de datos?

Pues en este apartado debemos comprobar si las tablas que obtuvimos están normalizadas, y si no lo están debemos normalizarlas para conseguir un buen diseño y así evitar posibles errores en el almacenamiento de datos de nuestro sistema.

Veámoslas de una en una. Primero cogemos la tabla de clientes:

CLIENTES (**NIF**, nombre_cliente, direccion_cliente, telefono_cliente, CP, localidad, provincia)

Recordamos que una relación satisface la primera forma normal si y sólo si todos los dominios subyacentes de la relación contienen valores atómicos. En este caso vemos que así es.

Puesto que un NIF determina el nombre, la dirección y el teléfono de una persona, y dado que un código postal determina una localidad y una provincia, podemos considerar respecto a la tabla Clientes las dependencias funcionales siguientes:

NIF -> nombre_cliente

NIF -> direccion_cliente

NIF -> telefono_cliente

NIF -> CP

CP -> localidad

CP -> provincia

También recordamos que una relación satisface la segunda forma normal si y sólo si satisface la primera forma normal y cada atributo no primo de la relación depende funcionalmente de forma completa de la clave primaria de esa relación. Si la clave es simple, o sea, está formada por un único atributo, y la tabla está en 1FN, entonces también está en 2FN.

Por tanto la tabla Clientes está en 2FN.

Sabemos que una tabla está en 3FN si está en 2FN y cada atributo no primo de la relación no depende funcionalmente de forma transitiva de la clave primaria de esa relación. Es decir, no pueden existir dependencias entre los atributos que no forman parte de la clave primaria de la relación.

Por eso vemos que Clientes **no está en 3FN**, ya que las dos últimas dependencias listadas violan la tercera forma normal.

Tenemos una dependencia transitiva NIF (localidad según la cual el atributo no primo localidad depende funcionalmente de forma transitiva de la clave primaria de la relación, y lo mismo para provincia.

Entonces tenemos que hacer la descomposición por aplicación de la 3FN. En el siguiente cuadro te recordamos cómo se hacía ese proceso.

> *Dada una relación de clave x y dos atributos no primos **y** y **z**, y en esta relación están presentes las siguientes dependencias funcionales: **x(y** , **y(z** y por tanto la dependencia transitiva **x (z**, el proceso de descomposición se realizará de la forma siguiente:*
>
> *De la relación se elimina el atributo que mantiene una dependencia funcional transitiva con la clave de la relación, es decir, el atributo **z**, quedando igual el resto.*
>
> *Se construye una nueva relación con la siguiente estructura:*
>
> *El atributo **z** que mantenía una dependencia funcional transitiva*
>
> *El atributo **y** el con cual el atributo z mantenía una dependencia funcional completa.*
>
> *La clave de la nueva relación será el atributo **y** y en esta relación sólo existirá una dependencia funcional completa de la forma **y(z***
>
> *En la relación original se define y como clave ajena de la relación nueva.*

Aplicado a nuestro caso, quitamos por tanto los atributos localidad y provincia de la tabla clientes, por ello esta tabla queda como sigue, observando que ahora CP es clave ajena:

CLIENTES (**NIF**, nombre_cliente, direccion_cliente, telefono_cliente, *CP*)

y se construye una nueva relación con la siguiente estructura:

CODIGOS_POSTALES (*CP*, localidad, provincia)

En la que se tendrían las siguientes dependencias funcionales:

CP -> localidad

CP -> provincia

Por tanto esas dos tablas están en 3FN.

Pasamos ahora a otra tabla, por ejemplo la correspondiente a los pedidos que, según el apartado anterior es:

PEDIDOS (**número_de_pedido**, fecha_pedido, estado, fecha_servido, *NIF*)

Si consideramos las siguientes dependencias funcionales:

número_de_pedido -> fecha_pedido, estado, fecha_servido, NIF

o sea que un número de pedido, nos va a determinar su fecha de pedido, su estado, la fecha cuando se sirve y el NIF de la persona que lo ha realizado, por los mismos razonamientos seguidos anteriormente, llegamos a la conclusión de que está en 1FN, 2FN y 3FN.

Pasamos a tratar ahora la relación proveedores que es:

PROVEEDORES(**CIF**, nombre_proveedor, direccion_proveedor, telefono_proveedor, fax_proveedor, CPProveedor, localidad_proveedor, provincia_proveedor)

Parece lógico contemplar las siguientes dependencias funcionales:

CIF -> nombre_proveedor, direccion_proveedor, telefono_proveedor, fax_proveedor, CPProveedor

CPProveedor -> localidad_proveedor, provincia_proveedor

y entonces estamos en la misma situación que vimos con la tabla de clientes. Siguiendo el mismo razonamiento, obtenemos la tabla normalizada hasta tercera forma normal siguiente:

PROVEEDORES (**CIF**, nombre_proveedor, direccion_proveedor, telefono_proveedor, fax_proveedor, *CPProveedor*)

donde CPProveedor es clave ajena de la tabla, y hace referencia a la clave primaria de la tabla CODIGOS_POSTALES obtenida anteriormente.

Ahora pasamos a otra tabla, la de artículos, que es:

ARTICULOS (**codigo_articulo**, descripcion, unidades, PVP, stock_minimo)

Podemos considerar que las dependencias funcionales son:

codigo_articulo -> descripcion, unidades, PVP, stock_minimo

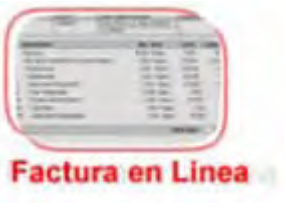
con lo que está en 1FN, 2FN y 3FN.

Ahora cogemos la tabla de líneas de pedido,

LINEAS_PEDIDO (***numero_de_pedido*, numero_de_linea**, cantidad, PVP, *codigo_articulo*)

donde podemos considerar que tanto PVP como cantidad y codigo_articulo, en cada línea de un pedido depende funcionalmente del par numero_de_pedido, numero_de_linea:

numero_de_pedido, numero_de_linea -> codigo_articulo, cantidad, PVP



Estaría por tanto en 1FN, y también en 2FN puesto que cada atributo no primo depende de la clave completa, es decir, la cantidad de un artículo en una línea de pedido depende del par de atributos numero_de_pedido, numero_de_linea. Lo mismo ocurre en el caso del PVP.

Procedemos ahora con la siguiente tabla.

FACTURA_CLIENTE (**numero_factura**, fecha_factura, IVA, *numero_de_pedido*)

Si consideramos las siguientes dependencias funcionales:

numero_factura -> fecha_factura, IVA, numero_de_pedido

concluimos que la relación está en 3FN.

Y por último, la relación que nos queda por comprobar es:

LINEAS_FACTURA (***numero_de_factura*, numero_de_linea**, cantidad, precio, *codigo_articulo*)

numero_de_factura, numero_de_linea -> cantidad, precio, codigo_articulo

que está en 1FN, y también en 2FN puesto que cada atributo no primo depende de la clave completa, es decir, la cantidad de un artículo en una línea de factura depende del par de atributos numero_de_pedido, numero_de_linea. Lo mismo ocurre en el caso del PVP o del código del artículo. Y está en 3FN ya que está en 2FN y no existen dependencias entre los atributos que no forman parte de la clave primaria de la relación, o sea, entre cantidad, precio y codigo_articulo.

Esta última apreciación es muy **importante** por la siguiente razón. Alguien podría decir, y no sin parte de razón, que el código del artículo determina funcionalmente al precio del artículo, y por tanto considerar una dependencia codigo_articulo (precio. Si estuviéramos analizando una tabla de artículos, sin duda la consideraríamos. Pero en este caso, la tabla LINEAS_FACTURA, va a constituir al fin y al cabo un fichero histórico, en el cual se van a ir almacenando muchas líneas de facturas. Pero con el paso del tiempo, un artículo cualquiera, por ejemplo un cd, no va a tener el mismo precio (en 2001 costaba x dinero y actualmente costará x+y dinero). Por eso, en este caso no debemos considerar esa dependencia funcional, puesto que en este caso, el código del artículo no nos va a determinar el precio del artículo.

Ahora, ya hemos terminado con este apartado.

"INFOANDALUS". Interfaz gráfica

En este apartado debemos describir mediante imágenes, los prototipos de las pantallas que consideramos que se les deben presentar a los usuarios de la aplicación, cuando se les solicite información de algún tipo.

De modo que en este caso, se acompaña un **ejemplo del diseño** de pantalla que esperamos que se muestre al usuario, cuando éste necesite dar de alta a un nuevo cliente en la aplicación.

En la documentación que elaboramos como analistas-diseñadores, podemos acompañar a la pantalla con cualquier tipo de información que consideremos necesaria indicar al equipo de programación que implementará la aplicación. Por ejemplo se les podría indicar que el campo del NIF debería tener una longitud de nueve posiciones, o que el campo dirección debería admitir al menos treinta caracteres.

Por tanto, para realizar el alta de un nuevo cliente en el sistema, diseñamos una pantalla como la que podemos observar, y será necesario tener en cuenta, a la hora de programarla en el lenguaje de programación que se considerara, controlar que el NIF no exista ya en la base de datos, para evitar duplicidades e incoherencias.

También sería lógico, que al introducir un código postal, la aplicación mostrara en los campos al efecto, la población y provincia correspondientes.

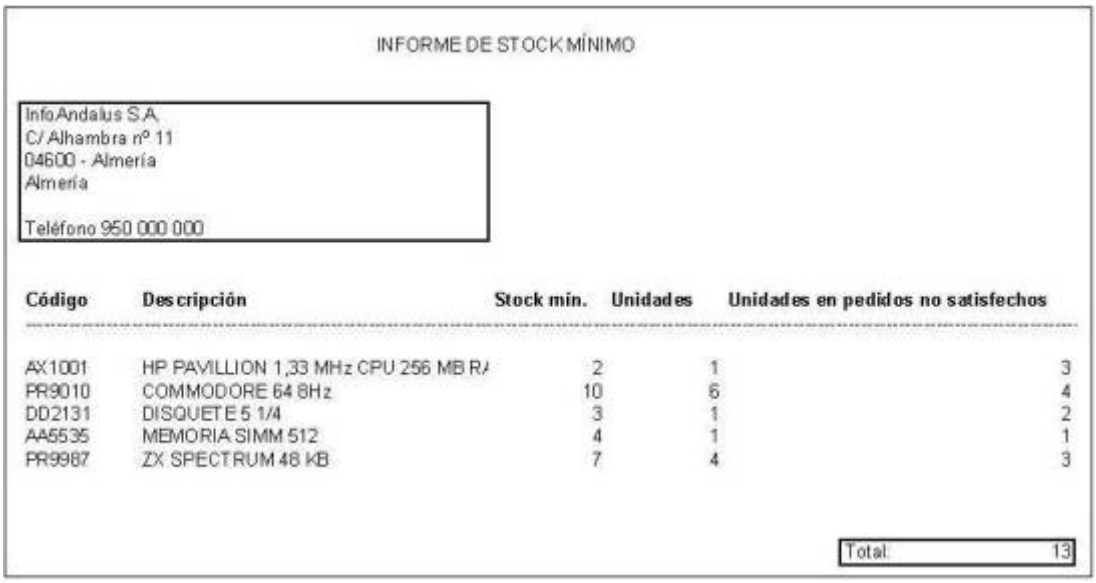
Por último, sería conveniente, que el sistema permitiera dar de alta a un cliente, sin necesidad de introducir su teléfono o dirección, según la práctica habitual de los encargados de dar de alta a clientes.

Ten en cuenta que lo único que tenemos que especificar es el aspecto que va a tener esa ventana, no tenemos que dotar de ningún tipo de funcionalidad a cada uno de los componentes que forman parte de la misma, ya que eso es tarea del programador.

Si trabajáramos en un proyecto real, deberíamos desarrollar los prototipos de todas las pantallas de la aplicación, y documentarlos de la mejor manera posible. Pero en este caso, lo dejamos aquí con este ejemplo que acabamos de ver y pasamos al siguiente apartado.

"INFOANDALUS". Informes

La última parte del proyecto consiste en realizar al menos uno de los informes que emitiría la aplicación. Para el ejemplo que nos ocupa, vamos a presentar el informe de los artículos que se encuentran por debajo del stock mínimo aconsejado:



INFORME DE STOCK MÍNIMO

InfoAndalus S.A.
C/ Alhambra nº 11
04600 - Almería
Almería

Teléfono 950 000 000

Código	Descripción	Stock mín.	Unidades	Unidades en pedidos no satisfechos
AX1001	HP PAVILLION 1,33 MHz CPU 256 MB R/	2	1	3
PR9010	COMMODORE 64 8Hz	10	6	4
DD2131	DISQUETE 5 1/4	3	1	2
AA5535	MEMORIA SIMM 512	4	1	1
PR9987	ZX SPECTRUM 48 kB	7	4	3

Total: 13

Por tanto y en resumen podemos visualizar el proceso de análisis y diseño de un proyecto observando la siguiente animación:

Proceso de análisis y diseño de un proyecto

Sugerencias de posibles proyectos a realizar

En apartados anteriores se comentó la posibilidad de que tú mismo definas el proyecto que desees realizar basándote en los requisitos mínimos que hemos marcado para cumplir en todos los proyectos.

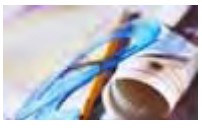
Pero por si alguien no llega a tener claro qué proyecto podría realizar, vamos a **sugerir algunos posibles proyectos**, para que al igual que Víctor en el Caso práctico del comienzo de la unidad, elijas el que más te motive, el que más atractivo te resulte, o el que vaya más en la línea de lo que piensas que te podría resultar útil. Puedes elegir cualquiera de ellos.

En principio están pensados para hacerlos de forma individual o en grupos de dos personas, pero como están muy abiertos, la posibilidad de extenderlos y completarlos puede hacer que el tamaño aumente considerablemente, de forma que exceda la disponibilidad de tiempo real que tenéis en este módulo.

En cualquier caso, en la nota del proyecto se tendrá en cuenta tanto la dificultad y complejidad de la aplicación entregada como el número de personas que han intervenido en su desarrollo. Pero recuerda que previamente deberás contar con el visto bueno del tutor para hacer el trabajo en grupo, y tener en cuenta que el nivel de exigencia a la hora de realizar la evaluación del proyecto será necesariamente mayor, proporcional a la dificultad, el tamaño final y complejidad del proyecto, y en relación con el número de desarrolladores que hayan intervenido en el mismo.

Una vez que hayas dejado volar tu imaginación un rato, valorando el proyecto que vas a realizar, el siguiente paso, es ponerte en contacto con tu tutor e indicarle qué proyecto has elegido, tanto si se trata de uno de los que te sugerimos, como si se trata del tuyo propio. En este último caso, deberás detallar a tu tutor las funcionalidades que va a incluir el mismo, para que le de el visto bueno.

Posibles proyectos



Sistema de gestión de unas salas multicines

Sistema de reserva y venta de billetes

Cine Consu

Cadena de campamentos Campingdulce

Tarjeta “cliente preferente” de la empresa ALPI

Jardinerías Picoypala S.A.

Alquiler de coches "Los autos locos"

Guía de evaluación del proyecto

¿Cómo vamos a **evaluar el proyecto**?

Son diversos los aspectos que podemos valorar en un proyecto, pero seguramente te resultará de utilidad saber los criterios que se van a seguir de forma general en la evaluación.

Lo primero que tenemos que decir es que la valoración se hará de una forma **global**. La calidad del proyecto y de la aplicación se tiene que valorar en su conjunto. No obstante, como este proyecto no tiene una finalidad comercial, sino que su propósito es valorar la consecución de los objetivos de este módulo profesional, los puntos en los que nos vamos a fijar pueden ser ligeramente distintos de los que exigiría el cliente al que se le intentara vender una aplicación.

A continuación vamos a dar las indicaciones que nosotros mismos vamos a seguir a la hora de evaluar y poner nota a tu proyecto. Se trata de indicarte los puntos en los que nos vamos a fijar y la importancia relativa dentro del total que le vamos a conceder a cada uno.

Presentación y documentación

Procura que tu proyecto sea claro, que en la presentación cumplas con todos los requisitos que se te han solicitado, y que esté bien documentado.

Ya hemos indicado anteriormente la documentación que te solicitamos. Ten en cuenta que el tiempo del que dispones es escaso, y hacer una completa documentación es algo que lleva mucho tiempo, pero cuanto más clara y completa esté, pues mejor, por eso se te pide que el proyecto esté lo más documentado posible. Ten en cuenta, que lo has desarrollado tú, y no puedes dar por supuesto que el resto de personas que lo puedan ver lo vayan a comprender si no está bien documentado.

Por ello es importante la **calidad de los comentarios**, que será valorada implícitamente en cada apartado del proyecto: explicar las decisiones tomadas, por qué se ha optado por tales DFDs, por qué tal campo es clave primaria o clave ajena, qué finalidad tiene diseñar de tal o cual manera una interfaz, etc.

Defensa del proyecto

Aquí no es posible que cada uno de vosotros haga una **defensa presencial** de su proyecto ante su tutor. Pero sí deberíais elaborar un breve documento en el que detalléis los aspectos más importantes o los que consideréis mejores o de especial interés de vuestra aplicación, y aquellos detalles que pueden haberos llevado mucho tiempo de análisis y diseño pero cuyo resultado no siempre es llamativo.

También podéis dar una lista de posibles mejoras para vuestra aplicación, que han tenido que quedar aparcadas por falta de tiempo. Con esto pretendemos saber si lo que no has incluido es por desconocimiento o sencillamente por falta de tiempo.

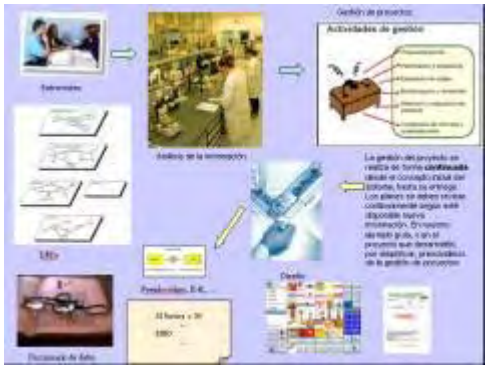
La valoración de este apartado en la nota global del proyecto será de un 5%

Análisis y diseño detallado de la aplicación desarrollada

La parte más importante de la nota, como no podía ser de otra manera, se la llevará el **análisis y diseño de la**

aplicación propiamente dicha, que se valorará como el 95% restante de la nota.

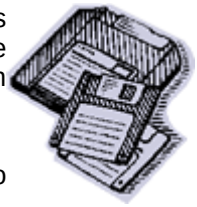
No obstante, hay muchos apartados que se valorarán dentro de la aplicación. Te los enumeramos a continuación, con una indicación de su valoración global dentro de la nota del proyecto. Debes tener en cuenta que la mayoría de esos apartados están íntimamente relacionados entre sí, por lo que se pueden arrastrar desde una parte a otra. Por ejemplo si se realiza un diagrama entidad-relación incorrecto, las tablas que obtendremos al obtener el modelo relacional serán también incorrectas o quizá inadecuadas.



Cuestionario de entrevista.

Se pretende que se componga de preguntas, que en un caso real, nos aportaran información sensible, es decir importante, que nos ayudaría a realizar un buen análisis del problema que tenemos, y por ello, nos facilitaría el diseño de nuestra aplicación. La valoración de este apartado en la nota global del proyecto será de un 5%

Diseño completo con unos DFDs correctos.



Se valorará que el diseño de la aplicación mediante los DFDs resuelva correctamente todos los aspectos del problema. Por ejemplo, si en la especificación del problema se ve claramente que hay que generar informes de cierta cosa, no sería admisible que en ninguno de los procesos que dibujemos en nuestros DFDs, no apareciera por ningún lado los flujos de datos representando dichos informes.

Además, es primordial en la elaboración de los DFDs, respetar en todo momento las reglas de balanceo en la descomposición de los sucesivos niveles.

En definitiva, en este apartado se trata de ver que el diseño es completo y correcto.

La valoración de este apartado en la nota global del proyecto será de un 28%

Diccionario de datos.

Debe ser completo, en el sentido de que deben recogerse todos los flujos de datos, los almacenes y las entidades externas, tal y como hemos visto en el ejemplo guía.

La valoración de este apartado en la nota global del proyecto será de un 5%

Miniespecificación de los procesos primitivos en pseudocódigo.

Aquí se valorará la claridad y sencillez, que estén adecuadamente documentados. Recuerda también, que como norma general, una especificación en pseudocódigo no debe ocupar más de un folio por una cara. De lo contrario, quizá esto nos ponga sobre la pista de que hemos realizado una descomposición incompleta, y por eso quizá, el proceso que estamos intentando describir como primitivo no sea tal, y sea susceptible de poderse descomponer en otros niveles.

La valoración de este apartado en la nota global del proyecto será de un 15%

Modelo Entidad-Relación adecuado.

Se trata aquí de ver que somos capaces de obtener el modelo entidad-relación del problema, que nos servirá para, en el siguiente paso, obtener las tablas con la información que el sistema almacenará en su base de datos. Este apartado es también bastante importante, ya que si obtenemos un mal diseño aquí, las tablas que obtendremos en el siguiente apartado serán probablemente inadecuadas o incorrectas.

La valoración de este apartado en la nota global del proyecto será de un 27%.

Modelo Relacional normalizado hasta 3FN.

En este punto, veremos que partiendo del modelo entidad-relación, se ha realizado correctamente el paso a tablas normalizándolas hasta 3FN.

La valoración de este apartado en la nota global del proyecto será de un 14%.

Diseño de un interfaz gráfico.

Se trata aquí de diseñar prototipos de las diversas pantallas que necesitaría la aplicación, para dar de alta nuevos clientes, modificar tal o cual dato, etc. Al menos se debe realizar una pantalla.

La valoración de este apartado en la nota global del proyecto será de un 3%

Diseño de un informe.

Por último, se debe realizar al menos uno de los informes que generaría la aplicación.

La valoración de este apartado en la nota global del proyecto será de un 3%.

Para saber más

Presentaciones muy completas sobre lo que se ha visto en el módulo, y más aún. Contiene entre otras cosas, ejemplos resueltos de diagramas DFDs.

Apuntes de Análisis y Diseño del Software

http://dis.um.es/~lopezquesada/ISw.htm

Últimas palabras

Aquí termina este módulo profesional.

Ha sido un placer compartir contigo el camino que nos ha traído hasta aquí.

Esperamos que te haya resultado enriquecedor. Sobre todo que lo hayas disfrutado, y encontrado útil, con independencia de la nota que finalmente obtengas, y que siempre

hayas encontrado en nosotros la ayuda necesaria.

¡Mucha suerte con la realización de tu proyecto!