UNIVERSIDAD POLITECNICA DE VALENCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA
ESCOLA TÉCNICA SUPERIOR D'INFORMÀTICA APLICADA

DESARROLLO DE UNA APLICACIÓN DE GESTIÓN DE VENTA PARA DISPOSITIVOS MÓVILES

PROYECTO FIN DE CARRERA

Autor: Fernando Miranda Alonso
Director: Joan Josep Fons Cors

Mayo de 2007

ÍNDICE

ÍNDICE

1. INTRODUCCIÓN

1.1.- CONTEXTO Y MOTIVACIÓN DEL PRODUCTO.

El Proyecto Fin de Carrera de Fernando Miranda está realizado en el contexto del trabajo que desarrolla en la Empresa Alfa Informática Empresarial, S.L., ya que se le encargó hacer un desarrollo nuevo de una Aplicación de Gestión de Ventas, y se aceptó la predisposición de la Empresa de que la Aplicación se pudiera presentar como PFC, con el visto bueno y las orientaciones del Profesor Joan Fons sobre las características técnicas del Trabajo que se solicitaba.

Este PFC hay que verlo como el resultado de Aplicar los Principios de Ingeniería de Software explicados en la E.T.S.I.A., a la consecución de un Producto Comercial, pero cumpliendo con las exigencias de un desarrollo de Software.

Hay que contemplarlo, por tanto, no como un trabajo de Diseño Conceptual, sino como un trabajo de Desarrollo Real para una Empresa Informática, que es muy extenso en cuanto a objetivos a cumplir, (y que se han cumplido), pero que hubiera resultado de una embergadura excesiva si en esta Memoria se reflejaran con todo detalle todos los apartados, cosa que sí hubiera sido necesaria si el proyecto tuviera una amplitud de desarrollo mucho menor.

Soy consciente de que en la Especificación de Requisitos Software, la Descripción de cada Requisito Funcional podría haber quedado redactada con un nivel de detalle mucho mayor para que se reflejara todo lo que se me ha transmitido por parte de la Empresa como requisitos Software (y que efectivamente la Implementación contempla).

También asumo que la totalidad de los Diagramas de que consta un Diseño Conceptual UML no han quedado reflejados en esta memoria, y los que he incorporado no tienen el detalle de un 100 % que explique el trabajo de implementación final, pero he optado por mostrar, a modo de ejemplo, sólo algunos diagramas que analizan una pequeña parte del proyecto, precisamente por la amplitud del mismo.

1.2.- OBJETIVOS.

El **Objetivo** de Alfa Informática, es comercializar la Aplicación de Gestión de Venta, para emplearla en Dispositivos móviles, con las siguientes características:

El entorno de programación de la Empresa desde hace años es el Visual Basic, por tanto ha de estar desarrollada en Visual Basic.NET (Compact Framework).
El Gestor de Base de Datos será el Microsoft SQL Server 2005 Mobile Edition, y la tecnología de acceso a datos será ADO.NET.

Se desarrollará siguiendo una separación entre las 3 Capas de: Presentación, Lógica de Negocio y Acceso a Datos, para poder adaptarla más fácilmente a otros dispositivos móviles con características diferentes.

Hay necesidad de implementación en PDAs y en otros dispositivos específicos diferentes, con distintos Sistemas Operativos y con distintos tamaños y resoluciones de Pantalla, con lo que se tendrá que preparar la Capa de Presentación específica para ellos.
Los dispositivos que tengan teclado físico y teclas de función han de poder aprovechar al máximo estas características para trabajar con dichas teclas.

La impresión ha de estar preparada: para que se adapte a diferentes impresoras (anchura, forma de conexión, papel continuo o formulario, ...), para que un mismo documento tenga diferentes diseños de impresión a elegir por cada Empresa, y para que se pueda repetir y/o previsualizar en pantalla en cualquier momento cualquier documento de impresión .

El Interface será con ficheros de texto y se tendrá que poder interconectar con diferentes Aplicaciones de Gestión Comercial. El envío y recepción de dichos ficheros de Interface ha de prepararse para que pueda hacerse de diferentes formas: por Internet o conectando la PDA con un PC.

Tendrá que estar totalmente parametrizada para que responda a las necesidades de funcionamiento de diferentes Empresas con distintas políticas de ventas.
También las pantallas que presenten los datos en forma de tablas (listviews), han de prepararse para que el usuario final pueda adaptar la anchura de las columnas de las tablas al aspecto que desee.

Ha de diseñarse para que el funcionamiento sea rápido y con las mínimas interacciones por parte del Agente Comercial, para reducir su dedicación en tareas administrativas, y ha de prepararse para que sólo se presente por pantalla la parte de la Aplicación que cada Agente (según su perfil) necesite para su trabajo.

2. ESPECIFICACION DE REQUISITOS SOFTWARE

2.1. INTRODUCCIÓN.

2.1.1 PROPÓSITO.

El propósito de esta ERS es formalizar la especificación de funcionalidades de la Aplicación, junto a la empresa ALFA.
Va dirigida, por tanto, a las personas de la empresa ALFA que tienen que decidir y revisar las especificaciones funcionales, antes de seguir adelante.

2.1.2 ÁMBITO.

El producto a desarrollar va a ser la Aplicación de Ventas para Dispositivos Móviles de la empresa ALFA INFORMATICA.
Está destinada a ser la Aplicación que usen muchos tipos de empresas diferentes del sector de la Comercialización/Distribución.
El sistema a desarrollar debe proporcionar un conjunto de aplicaciones para facilitar el trabajo de los Agentes Comerciales y los Vendedores / Repartidores que se desplazan con sus vehículos al domicilio de los Clientes, permitiendo disminuir los tiempos administrativos de la visita en casa del Cliente, agilizar el tratamiento de dicho Pedido al poderlo enviar inmediatamente a la Oficina de Administración de su Empresa, es decir, directamente al Ordenador Central.

2.1.3 DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS.

AlfaPDA: La Aplicación de Venta en Dispositivos móviles (marca registrada de la empresa Alfa Informática).
Vendedor, Agente Comercial o Usuario: El personal de la Empresa que está utilizando la PDA para realizar su trabajo, con su vehículo.
Visita: Es el encuentro del Usuario con uno de sus Clientes, para realizar: Un Pedido, una Venta o un Cobro.
Recorrido: La Ruta que tiene asignada cada Usuario cada día de la semana, y que constará de los Clientes que tiene que visitar. Un Cliente puede estar en recorridos de distintos días en la misma semana.
Proveedor: Es el suministrador de un Artículo o de un Servicio. El Usuario puede también realizar compras a Proveedores, y esos artículos pasan al stock de su vehículo.
En esta Aplicación también se utilizará el concepto de Proveedor para controlar los gastos que el usuario tenga que realizar en su Ruta (en gasolineras, restaurantes,...).

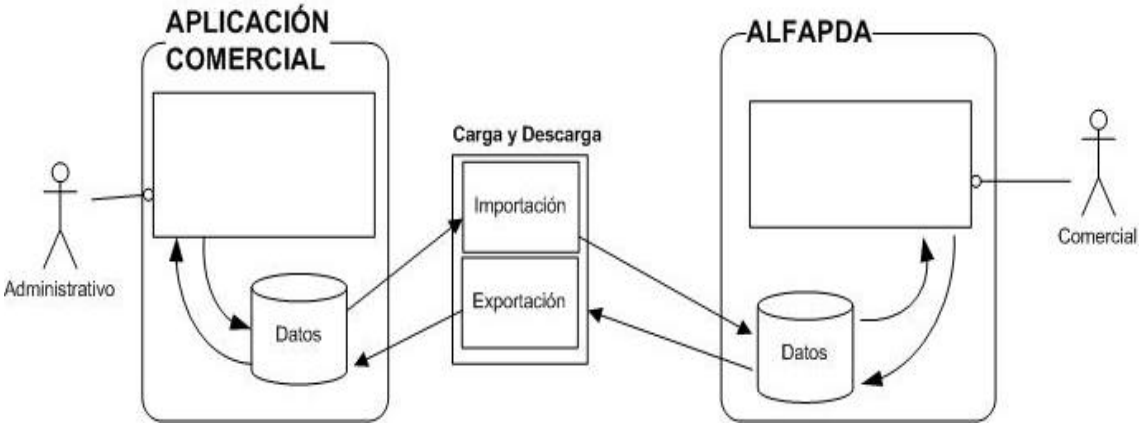
2.1.4 REFERENCIAS.

Este Documento se ha realizado según los Conceptos explicados en la Asignatura de Ingeniería del Software.
Se han seguido las Especificación de Requisitos Software IEEE-STD-830-1998.

2.2. DESCRIPCIÓN GENERAL.

2.2.1 PERSPECTIVA DEL PRODUCTO.

Es un producto diseñado para funcionar en cualquier dispositivo móvil, y es totalmente independiente de cualquier Aplicación de Gestión Comercial Central, en el sentido de que puede adaptarse a diversas Aplicaciones Comerciales que existan en el mercado, pero que necesita Cargar y Descargar los Datos mediante ficheros de Interface a la Aplicación Central, ya que realiza sólo el trabajo del Agente Móvil.



2.2.2 FUNCIONES DEL PRODUCTO.

El sistema a desarrollar debe proporcionar un conjunto de aplicaciones que a continuación describiremos:

1.- Parámetros de funcionamiento general de la Aplicación en cada PDA.
2.- Módulo de Carga y Descarga de Ficheros. Diariamente desde la Aplicación Central de Gestión Comercial de la Empresa, se le generará a cada vendedor un fichero Interface sólo con la información que necesita para tener actualizada la Base de Datos de su PDA: la lista de artículos, clientes, stock con la carga de su camión (almacén), ...
De igual forma se haría la descarga de las Ventas, Cobros, ... desde la PDA a la Aplicación Central.
Dicho intercambio de ficheros se realizará: bien conectándose a la red de la Empresa o bien desde el propio PDA conectándose por internet a un Servidor de FTP.
Se podrá repetir la recepción o envío de los ficheros si surgen problemas de transmisión.
3.- Rutas diarias asignadas al Vendedor.
4.- Módulo Gestión de Preventa: Pedidos.
5.- Módulo Gestión de Ventas: Albaranes y Facturas.
6.- Módulo Gestión de Compras a Proveedores.
7.- Módulo Gestión de Almacén: traspaso mercancía entre “almacenes / vehículos”.
8.- Módulo de Cobros a Clientes.

9.- Módulo de Control de Gastos del Vendedor.
10.- Módulo de Consultas, Resúmenes e Informes, que se podrán sacar eligiendo visualización o impresión (mediante Bluetooth a la miniimpresora que llevarán conectada al encendedor del vehículo). Los documentos tipo albaranes, facturas... se realizarán directamente a la impresora. En cualquier momento se podrá repetir cualquier documento ya impreso.

2.2.3 CARACTERÍSTICAS DEL USUARIO.

El producto va dirigido a los profesionales de la Venta que se desplazan en vehículo a casa del Cliente (Agentes Comerciales / Vendedores / Repartidores), sin necesidad de conocimientos informáticos, por lo que no se requiere ninguna formación específica previa, tan sólo los conocimientos propios de la actividad que realizan.

2.2.4 RESTRICCIONES GENERALES

El Interface con La Aplicación de Gestión Comercial de la Empresa ha de ser con ficheros de texto tanto para la Carga de Datos como para la Descarga de la Gestión Comercial realizada durante el día.
No se quiere una Aplicación tipo Web realizada para trabajar directamente a través de Internet con el Servidor Web de la Empresa y con la Base de Datos de la Empresa.
Se quiere desarrollar una Aplicación que sirva para cualquier punto geográfico donde se encuentre el Agente de Ventas con su vehículo y evitar la dependencia, restricciones y cortes de servicio que puedan derivarse de los problemas que tengan las distintas Empresas que dan servicio de Internet a los dispositivos móviles.
Por tanto se quiere trabajar con una Aplicación Local en la PDA con la Base de Datos Local que será la que necesite el Vendedor en ese día.
Pero para que la conexión con el Ordenador de la Empresa pueda ser inmediata se habilitará el envío y recepción de Información a un servidor de FTP de la Empresa, pudiendo realizar estos envíos todas las veces que el Vendedor desee.
La aplicación debe desarrollarse en Visual Basic.Net, para funcionar en dispositivos móviles, con entornos Windows Mobile, Windows CE y Pocket PC.
El interface gráfico debe ser compatible con cualquier tamaño y resolución de pantalla.

2.2.5 SUPUESTOS Y DEPENDENCIAS.

Para funcionar existe dependencia con la Aplicación de Gestión Comercial que tenga la Empresa, ya que se alimenta de Datos que se le deben suministrar desde ella, pero no está condicionada por una Aplicación concreta.
Las Tablas a importar y exportar, los datos de cada tabla con la definición y longitud del tipo de campo y el orden de esos datos en los ficheros de interface estarán controlados por un fichero de Control del Interface, de manera que no se tengan que modificar programas para adaptar el contenido de los ficheros de interface a cada Empresa.

2.3. REQUISITOS ESPECÍFICOS.

En esta sección de la ERS vamos a relacionar todos los requisitos software con un nivel de detalle suficiente para poder permitir a partir del mismo diseñar un sistema que cumpla con todos esos requisitos, y que se puedan realizar pruebas que comprueben que dichos requisitos verdaderamente se satisfacen.

2.3.1. REQUISITOS DE INTERFACE EXTERNOS.

2.3.1.1. Interfaces de usuario.

Al ser la pantalla del dispositivo de dimensiones reducidas, se ha de aprovechar al máximo el contenido en pantalla, evitando todo aquello que no sea necesario para la operación concreta que esté realizando el usuario.
Se dispondrá de un teclado emergente numérico y otro alfanumérico, que se presentarán automáticamente sólo cuando se necesiten, según el dato a rellenar. También pueden visualizarse o esconderse cuando el usuario los seleccione.

Hay que tener en cuenta que los dispositivos móviles que se utilicen pueden ser:
- con o sin teclado físico, por lo que a aquellos que tengan teclado no se les presentará automáticamente el teclado virtual emergente.
- con o sin teclas de función, y diferente número de teclas de función: hay que prever que el usuario pueda cambiar la tecla asignada a cada función.
- con o sin lector de código de barras, por lo que aquellas que tengan lector de código de barras deben poder hacer un uso automático de dicha prestación.

Los diseños de pantalla deben estar preparados para funcionar en cualquier tamaño y resolución de pantalla, incluso en diferente posicionamiento de pantallas (vertical, apaisada).
La Aplicación debe estar preparada para permitir diferentes diseños de pantalla al cliente que lo solicite, sin que por eso tenga que modificarse el código del programa, y debe prepararse para que el coste de tiempo de dicha adaptación al cliente sea mínima.

Al tener los listview que se visualizan en la Aplicación una serie de columnas y al tener la suma de sus longitudes más longitud que la anchura de la pantalla de la PDA, todos los listview tienen que estar preparados para que el propio usuario pueda redimensionar la anchura de cada campo del listview con el puntero, para darle a cada campo el tamaño óptimo e incluso para anular algún campo que no sea de relevancia para él. Ha de tener la opción de que esos tamaños se conviertan en los definitivos para él en su fichero de configuración, teniendo siempre la opción de cambiar momentáneamente la anchura de alguna columna en alguna ocasión.

Así mismo, todo documento impreso ha de prepararse para que el mismo código de programación pueda tener diferentes formas en el diseño en papel.
Lo más típico es la impresión de un documento de venta (albaran, factura) y de pedido.

Al ser la impresora conectada por BlueTooth con la PDA del tipo miniimpresora, la anchura del papel es unas 60 posiciones, por lo que cada empresa tiende a seleccionar el tamaño adecuado de cada campo a imprimir. Por tanto hay que posibilitar que cada empresa pueda tener a nivel de "diseño de impresión" el que considere adecuado.

2.3.1.2. Interfaces hardware.

La aplicación está diseñada para ser soportada por dispositivos móviles: PDAs y dispositivos industriales.

2.3.1.3. Interfaces software.

La aplicación debe poder ejecutarse en PDAs y dispositivos similares con sistemas operativos Windows Mobile 2005, Windows CE o Pocket PC 2003.
Ha de estar diseñada para que se pueda cambiar fácilmente la programación para elegir cualquier Sistema de Gestión de Base de Datos y cualquier tecnología de Acceso a Datos.
Ahora está ya preparada para trabajar con el sistema de gestión de bases de datos Microsoft SQL Server 2005 Mobile Edition, y tecnología de acceso a datos ADO.NET.

2.3.1.4. Interfaces de comunicaciones.

La comunicación con el Ordenador de la Empresa puede ser:

- Si la PDA tiene conexión a Internet: mediante envío directo por FTP desde la PDA al servidor de FTP que utilice la empresa. Esta es la forma normal en la que funcionará la aplicación en muchas empresas.
- Mediante email desde la PDA: Otra forma alternativa (menos cómoda) de envío y recepción de los ficheros de descarga y carga de la PDA es mediante email, desde el gestor de correo de la PDA, seleccionando el fichero que se quiere adjuntar o grabando el fichero recibido en la carpeta adecuada.

- En el caso de Gestión de Pedidos, para las Empresas que funcionen como Agentes Comerciales distribuyendo artículos que pertenecen a diferentes proveedores, hay opción además de obtener un listado con los artículos a pedir a cada Proveedor, en formato .pdf y enviarlos directamente a la dirección email que tenga puesta cada Proveedor. La AlfaPDA dejará directamente en el Outlook los distintos correos generados.

En estos casos anteriores no hace falta que la PDA tenga conexión a internet, porque se puede sincronizar el Bluetooth de la PDA a un teléfono móvil con bluetooth. Esto es desde fuera de la aplicación AlfaPDA, es decir en los Parámetros de la PDA se selecciona que no es envío por internet.

Cuando el vendedor tenga que desplazarse al final de su jornada a la empresa, entonces lo normal no es el envío por FTP, sino la conexión al PC una vez ha llegado a las Oficina de su Empresa. En este caso puede ser:
- Con la sincronización Microsoft ActiveSync: Requiere la conexión por cable al P.C.
- Mediante wifi utilizando la conexión Internet del PC al que se conecta, y enviando por FTP, de esta forma todos los vendedores pueden estar enviando cuando están ya dentro de la Empresa en cobertura wifi .

2.3.2. REQUISITOS FUNCIONALES.

Vamos a definir las acciones fundamentales que debe realizar el software, es decir, de qué forma se procesan y aceptan un conjunto de entradas para producir una determinada salida.

2.3.2.1. Carga de Datos a la PDA.

Introducción

Se Cargan en la Base de Datos de la PDA los datos destinados a un Comercial concreto provenientes de la Aplicación Comercial de la Empresa.

Entradas:

Un fichero plano de texto con el contenido de todas las tablas a importar.

Proceso:

Leer cada una de las tablas que existen en el fichero, sustituyendo la tabla existente en la Base de Datos por el contenido de la tabla en este fichero de texto. Hay un fichero de control con la definición del formato de campos que tiene el fichero de entrada. Este fichero de Control es el que va a permitir que el formato de Entrada y de Salida de Datos pueda ser diferente en cada Empresa según los datos que necesite y según la Aplicación de Gestión Comercial Central que tenga la Empresa.

Salidas:

La actualización de la Base de datos en la PDA.

2.3.2.2 Inicialización de la aplicación.

Introducción

Se leen los parámetros de configuración que existen en el fichero para cada PDA y se mantienen en memoria durante toda la ejecución de la Aplicación.

Entradas:

Un fichero plano de texto con el contenido de todos los Parámetros que definen la Configuración del Terminal.

Proceso:

Se realizan comprobaciones iniciales de Licencia, entorno PDA de Fecha, alfabeto, punto de millar, coma decimal, memoria disponible, apertura de la B.D., comprobación de número de usuario, mensajes al vendedor. Visualizar/ Ocultar los menús que deben verse y que no deben verse.

Salidas:

Si se superan todas las comprobaciones se va al Menú General.
Si la PDA no tiene un vendedor asignado, va a pedir vendedor y a realizar una Carga de Datos.

2.3.2.3. Ir al Rutero.

Introducción

El rutero es la lista en un orden concreto de los Clientes que el Comercial tiene que visitar en cada día. Ese orden ya está preparado para que sea la ruta que ha de seguir el Comercial en base a la ubicación geográfica del cliente.

Proceso:

Ordena los clientes según el día y con opción de sacar sólo los pendientes de visitar, los ya visitados o todos. También se pueden seleccionar sólo los que tengan Pedidos Pendientes de Servir.

Salidas:

Una Tabla con la relación de Clientes a visitar.

2.3.2.4. Selección de cliente.

Introducción

Se elige el Cliente al que se va a visitar.

Proceso:

Se recuperan los datos de dicho cliente. Algunos se visualizarán y otros se tendrán disponibles para poder realizar operaciones con ellos durante la visita (pedido / venta / cobro).

Entradas:

La selección del Cliente bien en el Rutero o bien en la pantalla de Búsqueda de Cliente ayudado de las posibilidades generales de las búsquedas (selección de campo de búsqueda: código, nombre, población, ...)

Salidas:

Datos del Cliente: Código y Nombre.

2.3.2.5. Realización de una Venta / Pedido.

Introducción

Se introducen las líneas de detalle de la venta (o pedido) con los artículos elegidos).

Proceso:

Se recuperan los datos de dicho cliente. Algunos se visualizarán y otros se tendrán disponibles para poder realizar operaciones con ellos durante la visita (pedido/venta).

Se van introduciendo los artículos y se recupera el precio según los datos del cliente leído, teniendo en cuenta precios especiales de cliente, de cadena, y tarifas de cliente y generales.

Se recuperan también los descuentos y promociones que tenga ese artículo para ese cliente o bien a nivel general.

Entradas:

Se van introduciendo las líneas de la venta:

Artículo: bien introduciendo el código o con el lector del código de barras, o yendo a la búsqueda de artículo seleccionando la búsqueda por cualquiera de los campos posibles, grupo, subgrupo, código, descripción.

Se introduce la Cantidad, y la unidad de medida, el precio si se quiere cambiar, así como los posibles descuentos, y si es el caso el numero de piezas y el lote.

Si se introduce una cantidad negativa pide la selección del motivo de devolución, y si se introduce precio cero pide la selección del motivo de precio cero.

Se comprueba stock del artículo y se comprueba también las promociones regalo que pudiera tener ese artículo, así como si es un artículo “envase” o un artículo sujeto a “cer” (coste de eliminación de residuos) o artículo con “punto verde”.

Salidas:

Se visualizan en la pantalla los datos actualizados en el documento de Venta/Pedido.

2.3.2.6. Obtención del Precio que se aplica a un Cliente.

Introducción

Se obtiene el precio de venta de un artículo, que se aplica a un cliente.

Proceso:

Hay un orden asociado a una política de precios y por consiguiente una búsqueda en distintas tablas para la obtención del PRECIO.

Se recuperan también los descuentos y promociones que tenga ese artículo para ese cliente en un orden de prioridad de búsqueda establecido.

Entradas:

Código de Artículo o código de barras.

Salidas:

Precio del artículo por unidad de venta.

2.3.2.7. Modificar Línea de Venta/Pedido.

Introducción

Se elige la línea a modificar y se selecciona pulsando con el puntero.

Proceso:

Se recuperan los datos de dicha línea desde la tabla de la pantalla, y se habilita la modificación de cualquier campo de la línea.

Entradas:

Se puede cambiar cualquier campo de la línea de venta/pedido.

Salidas:

Registro de línea de venta/pedido modificado.

2.3.2.8. Borrar Línea de Venta/Pedido.

Introducción

Se elige la línea a borrar y se selecciona pulsando con el puntero.

Proceso:

Se recuperan los datos de dicha línea desde la tabla de la pantalla, y se elige la opción de menú “borrar línea de detalle”, y se da conformidad.

Entradas:

La línea a borrar que figura en la pantalla.

Salidas:

Registro de línea de venta/pedido modificado.

2.3.2.9. Borrar Documento de Venta/Pedido.

Introducción

Se recupera el documento a borrar y se visualiza en pantalla.

Proceso:

Se recuperan los datos de dicho documento, y se elige la opción de menú “borrar documento”, y se da conformidad.

Entradas:

El documento a borrar que figura en la pantalla.

Salidas:

Borrar en la Base de Datos el documento. En el caso de una Venta el documento no se borra físicamente de la B.D., sino que se marca como anulado.

2.3.2.10. Realización de una Carga de Mercancía.

Introducción

Se introducen los artículos que constituyen la carga de mercancía del vehículo.
La pantalla de captura de Carga de Mercancía debe tener las opciones de recuperar documento de carga, modificar línea de detalle, borrar línea de detalle, borrar documento de carga completo e imprimir.

Proceso:

Se selecciona el Almacén de Origen y el de Destino y a continuación se introduce la relación de artículos cargados.

Entradas:

Se van introduciendo los artículos que se cargan en el vehículo:
Artículo: bien introduciendo el código o con el lector del código de barras, o yendo a la búsqueda de artículo seleccionando la búsqueda por cualquiera de los campos posibles, grupo, subgrupo, código, descripción.
Se introduce la Cantidad, y la unidad de medida, el precio, y si es el caso el numero de piezas y el lote.

Salidas:

Se visualizan en la pantalla los datos actualizados en el documento de Carga de Mercancía.

2.3.2.11. Realización de un Recuento de Mercancía.

Introducción

Es una operatoria parecida a la Carga de Mercancía. Se diferencia en que se selecciona un único Almacén, que es sobre el que se hace el recuento.

2.3.2.12. Realización de un Cobro.

Introducción

Se visualiza el detalle de Deuda de un Cliente y se realiza un cobro sobre una de las líneas de detalle.

Proceso:

Se recuperan los datos de dicho cliente y todo el detalle de deuda de ese cliente y sobre una de las líneas de detalle se captura un cobro total o parcial indicando la forma de cobro (contado, cheque, ...) y actualizando la deuda del cliente.

Entradas:

Se selecciona una línea de detalle.

Salidas:

Se visualizan en la pantalla los datos actualizados en el documento de Pendiente de Cobro.

2.3.2.13. Descarga de Datos de la PDA.

Introducción

Se Descargan los Datos de la PDA con la gestión del Comercial a un fichero de texto.

Entradas:

Las Tablas de la B.D. que han recogido la actividad del Comercial.

Proceso:

Leer cada una de las tablas de salida que existen en la B.D. y va grabando los registros en un fichero de texto según el fichero de Control que define el formato a generar para dicho interface.

Salidas:

Un fichero plano de texto con el contenido de todas las tablas a exportar.

2.3.3. REQUISITOS DE EFICIENCIA.

Requisitos de Rendimiento: Se ha de desarrollar intentando que el tiempo de respuesta de la PDA sea lo más rápido posible, ya que uno de los objetivos es que el vendedor reduzca su tiempo de gestión administrativa en la venta/pedido, (aparte de los otros objetivos de disponibilidad de la información que necesita, ...) .

Hay que tener en cuenta que estos dispositivos no tienen ni la memoria de proceso ni la velocidad de procesador de un PC.

2.3.4. RESTRICCIONES DE DISEÑO.

Requisitos en el diseño de la Base de Datos:

A cada PDA se le ha de enviar sólo la información que necesite para realizar su trabajo del día.

Cuando realice la Descarga diaria y a continuación vuelva a hacer una Carga, se van a sustituir todas las tablas que tiene en la PDA.

Hay opción de que la Carga borre / inserte / modifique datos a una tabla sin borrarla por completo, pero lo normal es que se le envíen completos todos los datos que necesita en el día y que por tanto se borren por completo los datos del día anterior.

2.3.5. ATRIBUTOS.

Se debe cuidar que el mantenimiento posterior de la aplicación se desarrolle teniendo en cuenta el parque de aplicaciones ya instaladas, para que no genere ningún problema la actualización de las versiones antiguas a la nueva versión.

Para ello hay que prever una actualización automática de los nuevos parámetros que se incorporen, poniendo la opción por defecto más adecuada a nivel general.

Hay que prever la posibilidad por parte de las empresas de que establezcan claves para acceder a determinados programas o para realizar determinadas operaciones.

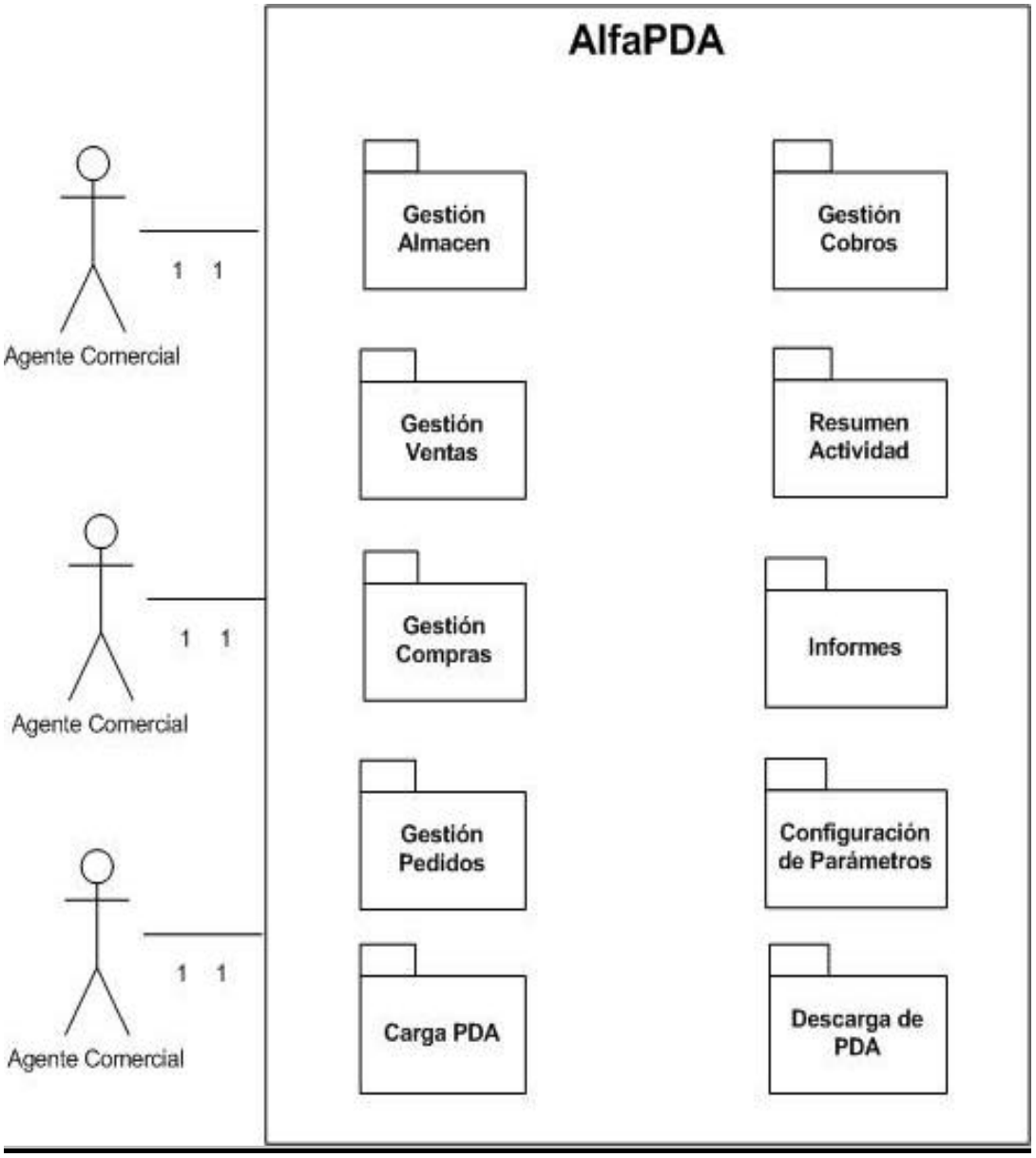
También se ha de garantizar la grabación correcta en la base de datos sin posibilidad de deterioro de información.

Ante la posibilidad de que una PDA se averíe, ha de ser rápido y sencillo por parte de la persona encargada en cada empresa, realizar una instalación completa de la Aplicación en otra PDA y que conserve las características a nivel de parámetros asignadas a ese número de vendedor, y a continuación realizar una Carga de la PDA con los datos que necesita en ese día para realizar su trabajo.

3.- ESPECIFICACIÓN CONCEPTUAL DEL SISTEMA

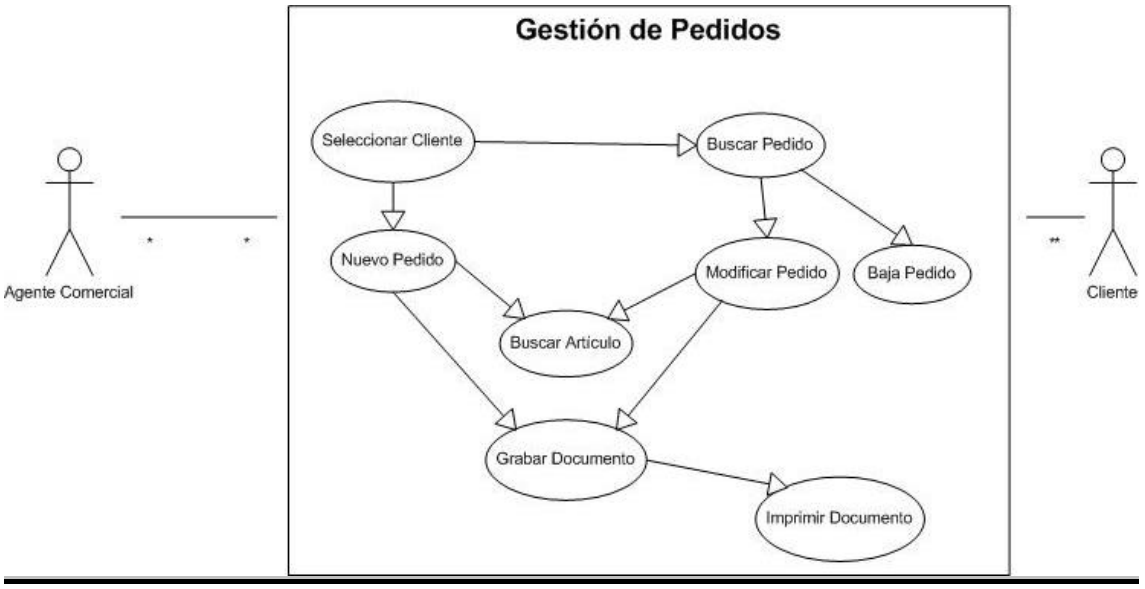
3.1. CASOS DE USO.

La Aplicación AlfaPDA tiene los siguientes componentes:



Voy a exponer de forma concisa lo que sería solamente un caso de uso: el de Nuevo Pedido.

Diagrama de contexto:



Plantilla de Caso de Uso:

Caso de uso	Nuevo Pedido
Actor principal	Agente Comercial
Propósito	Dar de alta un nuevo pedido
Resumen	El comercial visita a un cliente. El cliente le solicita artículos. El comercial crea un nuevo pedido con los artículos solicitados
Precondiciones	
Poscondiciones	El pedido se almacena en el sistema
Incluye	Buscar Artículo. Grabar Documento
Extiende	
Hereda de	

Intenciones de usuario	**Obligaciones del sistema**
1.- El comercial solicita nuevo pedido	El sistema carga todos los datos iniciales para comenzar el pedido
Mientras se estén añadiendo líneas:	
2.- El comercial selecciona un artículo	El sistema carga ese artículo con las condiciones de precio y descuentos del cliente y las características de ese artículo en cuanto a cer, envase, regalo, ...y le va pidiendo el resto de datos del artículo
3.- El comercial introduce la cantidad	Si la cantidad es negativa, el sistema le solicita el motivo de la devolución.

	Si el artículo está en alguna promoción regalo, el sistema comprueba si le corresponde regalo y le informa.
4.- El comercial puede introducir, si el sistema se lo permite, el precio, lote, piezas, descueto, promoción	Si el precio es cero el sistema le solicita el motivo.
5.- El comercial pulsa fin de línea	El sistema hace las comprobaciones de que la introducción de la línea es correcta, añade la línea en el pedido y en el listview de la pantalla con los artículos solicitados. El sistema hace también automáticamente la gestión de incluir las posibles línea de regalo y cer.
6.- El comercial puede seleccionar una línea ya intrtoducida para poder modificarla	El sistema recuperará los datos de dicha línea y los visualizará en la zona inferior de la pantalla con todos los datos rellenos para que el comercial pueda modificar lo que desee.
7.- El comercial pulsa otros datos del pedido	El sistema le muestra la pantalla para introducir el resto de datos a nivel de pedido: observaciones, fecha entrega, ruta repartidor.
8.- El comercial va introduciendo dichos datos	El sistema hace las comprobaciones pertinentes
9.- El comercial pulsa fín Pedido	El sistema graba el Pedido y si es el caso lo imprime

Extensiones síncronas
#1 Si en 2 el comercial introduce un artículo inexistente, el sistema le muestra un mensaje de error
#1 Si en 8 el comercial introduce una fecha de servicio, esta debe ser posterior a la fecha de pedido

3.2. DIAGRAMA DE CLASES.

En esta sección voy a ir relacionando las clases de que consta la Aplicación, las características de cada una, y las relaciones entre ellas.
Y para terminar expondré lo que sería el diagrama de clases que corresponde a la realización de una Venta/Pedido.

En la siguiente sección de Diseño Arquitectónico del Sistema iré mencionando cada uno de los Requisitos Software y exponiendo la solución elegida, con sus características y su funcionamiento, agrupando las clases en una estructura de capas.

En la sección de Implementación pondré algún ejemplo para que se pueda ver cómo está desarrollada la programación.

DESCRIPCIÓN DE LAS CLASES

CLASE DE ACCESO A DATOS

CLASE Tabla :

Está todo el sistema de acceso a datos.
Gestiona una Tabla que es el resultado de una tabla de la B.D. o de una Vista de varias tablas de la B.D.

El atributo cBD que es privado de la clase y compartido por todos los objetos instanciados de Tabla, define la conexión a una base de datos.
Métodos de clase AbrirBD y CerrarBD.

Métodos genéricos de los objetos instanciados:
Hay definidos una serie de métodos que permiten realizar todas las operaciones que se puedan necesitar de una B.D., y también hay otros métodos que facilitan información de alguna característica de la operación en curso.
Estos métodos son los que se utilizan desde el resto de la Aplicación, y por tanto no están condicionados por una tecnología concreta, si hubiera que cambiar la tecnología empleada se crearía otra clase Tabla1 que contendría esos métodos con los mismos parámetros requeridos, pero la programación de cada uno de esos métodos se adaptaría a la nueva tecnología.
En la sección de implementación pondré unos ejemplos.

```
' Leer       LeerDH      LeerAgrupado  LeerSQL
 ' Coger      Dejar       Add           Borrar            BorrarTodo
 ' LeerS      LeerA       Actualizar    InicializarRegistro
 ' nrr        contador    status        NumeroCampos      NombresCampos
```

CLASES BÁSICAS DE LA LÓGICA DE NEGOCIO

CLASE Cliente

Es una de las clases básicas de la lógica de negocio.
Un objeto cliente de esta clase, suministra toda la información que se necesita y tiene relación con dicho cliente y que está almacenada en diversas tablas de la B.D.
Al crear un objeto se recupera toda esa información y además tiene métodos que realizan la gestión, actualización, grabación en la Base de Datos.
Altas: de nuevos clientes, de direcciones de envío. En la PDA se permite dar de Alta de nuevos Clientes.

CLASE Artículo

Suministra toda la Información relacionada con un artículo. No se permiten dar altas de artículos en la PDA.

CLASE CodigoBarras

Ayuda en la identificación de un artículo cuando lo que nos han introducido en el campo artículo es un código de barras. La Aplicación permite el uso del lector de código de barras cuando la PDA lo posee.
El código de barras puede ser de un artículo, de un lote, de un artículo en unidad de medida de almacenaje, de un código de artículo con numeración del proveedor, ...

CLASE ArtCli

Es la clase que tiene la información de un artículo concreto para un cliente concreto.
Está relacionada con el objeto cliente y con el objeto artículo, y tiene una serie de métodos que definen unas condiciones de venta.
Es una clase que contiene la lógica de la política comercial de la Empresa.
Hay una serie de tablas en la B.D. que permiten definir distintas políticas comerciales a elegir por cada Empresa y los métodos definidos permiten informar de las condiciones económicas de la venta de un artículo concreto para el cliente, en función de tarifas, precios especiales, descuentos, promociones, pertenencia a cadenas de clientes, ...

CLASES Visita, VisitaCabecera, VisitaDetalle

Es la clase que gestiona una visita a un cliente. Es decir es la clase que contiene una Venta o un Pedido. Al crear un objeto visita se define si va a ser una venta o un pedido. Las diferencias entre los dos son muy pocas y están bien identificadas, y la mayor parte de los métodos son únicos, por ejemplo a la hora de hacer el cálculo del importe de un documento de venta o de un pedido.

La clase Visita tiene un objeto Cliente, un objeto de la clase VisitaCabecera y cero o muchos objetos de la clase VisitaDetalle. (Se admite Visita con sólo cabecera para recoger el motivo de visita nula). VisitaDetalle tiene un objeto artículo.

Creación del objeto visita:
New para dar de alta nueva visita

New para leer (recuperar) una Visita ya existente

Métodos para la valoración de una línea de detalle.
Métodos para la valoración de CalculosTotalCabecera.
Grabar cabecera en memoria
Grabar_DetalleVisita_enMemoria (alta, modificación y eliminación)
Gestion de la Grabación de Linea Regalo
Gestion de la Grabación de Linea CER (coste de eliminación de residuos)
FinaldeVisita, grabación / modificación del objeto visita en la Base de Datos.
EliminarVisitadeBD
Métodos de recuperación de la información para la impresión:
I_DatosCabecera I_DatosDetalle

CLASE AlbProveedor

Es una clase con un funcionamiento parecido a la clase Visita, pero lo que aquí se gestiona es un documento de compra (Albarán) a un Proveedor. También tiene una cabecera de documento y una o varias líneas de detalle con métodos que permiten realizar todas las operaciones posibles.

CLASE CargaMercancia

Es una clase que gestiona el documento de traspaso de Mercancía entre Almacenes. Cada camión de reparto es un almacén y el funcionamiento normal es la carga de mercancía diaria desde el Almacén General.
El documento de Carga tiene una gestión parecida a los dos documentos anteriores de Visita y AlbProveedor: hay una cabecera de documento y unas líneas de detalle de documento y también hay métodos que permiten realizar todas las operaciones posibles.

CLASE Recuento

Es una clase que gestiona el documento de realizar un Recuento (Inventario) de artículos en un Almacén, en este caso recuento de artículos del camión.
El recuento puede ser total: los artículos que no figuran en el recuento se actualiza su stock con valor cero. Parcial: los artículos que no figuran en el recuento se dejan con el stock que tengan.

CLASE GestionStock

Es la clase que gestiona el stock del almacén-camión y que da servicio a todas las clases anteriores

CLASE Cobros

Los cobros pendientes de un cliente se traspasan a la PDA para que el Agente pueda realizar Gestión de Cobros.
Esta Clase permite la realización de Cobros a un Cliente. Gestiona por cada cliente los distintos “documentos de cobro” pendientes de cobrar. De cada uno de ellos se pueden ir haciendo cobros parciales y cada cobro parcial puede ser en efectivo, cheque o pagaré. También gestiona la introducción de un importe que el cliente entrega y lo distribuye entre los distintos documentos pendientes.
Se puede también anular un cobro concreto ya realizado.

En la B.D. hay una tabla TECO (de cabecera de cobros) donde cada cliente tiene varios elementos en donde consta entre otras cosas la deuda original y el total ya cobrado entregado a ese documento, y en otra tabla TSCE de detalle se relacionan cada uno de los cobros que se han realizado a cada documento pendiente, y el tipo de cobro.

CLASES BÁSICAS DE LÓGICA DE NEGOCIO: GESTIÓN DE REQUISITOS

CLASE Rutero

En el formulario FrmRutero se presentan los clientes que posee una ruta concreta (un Agente de Ventas con su vehículo) y están agrupados por cada día de la semana (un mismo cliente puede aparecer en varios días) , en el orden en que se deben visitar (según criterio geográfico de la empresa) y poniéndolos en distintas listas según hayan sido visitados ya o no, y distinguiendo los que tienen pedidos pendientes de servir de aquellos que no tienen.
Esta Clase se encarga de suministrar toda esta información a la clase Frm00Rutero que gestiona la presentación en los formularios Frm01Rutero, Frm02Rutero, ...(En la siguiente sección explicaré las distintas presentaciones en pantalla de un mismo formulario).

A este Rutero se va desde distintos botones del Menú General, para realizar: Ventas / Pedidos / Cobros.
El rutero de cobros tiene un filtro que hace que sólo aparezcan los clientes con cobros pendientes.

CLASES VenPed y VenPedMasDatos

Dan servicio a las clases Frm00VenPed y Frm00VenPedMasDatos que gestionan a los formularios de Ventas y Pedidos Frm01VenPed, Frm01VenPedMasDatos, ...

CLASE EntregaporcuentaProveedor

Hay a veces unos acuerdos comerciales con algunos grandes clientes en los que es el Proveedor/Fabricante de una Marca Comercial el que establece las condiciones económicas de Venta a dichos Clientes. En estos casos la Empresa que distribuye esos productos al cliente hace sólo de transportista para entregar mercancía, y la entrega al Cliente por cuenta del Proveedor.

Este es un tipo de Gestión en el que se necesita mantener unas tablas en la B.D. donde figuren:
Los Proveedores involucrados en esa política.
Los Clientes involucrados y con qué Proveedores.
Los artículos que cada proveedor puede vender.
Los artículos que cada cliente puede comprar a cada proveedor.
Los códigos de artículos con la codificación del proveedor y los códigos de artículos con la codificación del cliente.
Las condiciones de venta de precio, descuentos, promociones, tarifas que tiene cada Proveedor a cada Cliente.

En resumen: cuando se selecciona un cliente que está identificado de forma que algunas marcas de productos sólo se los puede vender el Proveedor, el funcionamiento interno de la gestión de la política de precios y de los artículos que se le pueden vender es completamente diferente de aquellos clientes "normales" de la empresa, aunque la operativa del formulario de ventas sea completamente igual en todos los clientes, sólo que internamente se ejecutan otros procesos que permiten seleccionar o no un artículo y se calculan unos precios diferentes.

Toda esta gestión de venta especial está desarrollada en esta clase y suministra la información de precios y condiciones que necesita la clase Frm00VenPed que es la clase que gestiona la pantalla de Ventas, formulario Frm01VenPed.

CLASE Envases

Realiza la Gestión y Control de los envases en depósito/fianza en un Cliente.
Esta clase da soporte a la clase Cliente para suministrarle la información que necesita y a la clase Visita para gestionar la fianza/depósito de envases en un documento de venta.

CLASE Regalo

Realiza la Gestión de la Política de Regalos y Promociones especiales en función de distintos criterios y posibilidades a elegir por cada Empresa.
Esta clase da soporte a las clases Frm00VenPed y Visita para suministrarles la información que necesitan.

CLASE Cer

Realiza la Gestión del "Coste por Eliminación de Residuos" que algunas Empresas están obligadas a llevar. Da soporte a la clase Visita.

CLASE PedPenSer

Realiza la incorporación a la Venta de los Pedidos que tiene un Cliente pendiente de Servir.
Hay un formulario Frm01PedPenSer que es el que soporta la operatiria con el Vendedor y que hace uso de esta clase.

CLASE HisVenPed

HisVenPed Realiza la incorporación a la Venta/ Pedido desde un Histórico de Ventas/Pedidos. Desde el formulario de Venta / Pedido y un Menú de traspasos con los posibles traspasos de artículos desde Histórico de Ventas y Pedidos, y también desde Últimas Ventas / Pedidos
Hay un formulario Frm01HistVenPed que es el que soporta la operatiria con el Vendedor y que hace uso de esta clase.

CLASE UltimVenPed

Realiza la incorporación a la Venta/ Pedido desde las Últimas Ventas/Pedidos.
Hay un formulario Frm01UltimasVentas que es el que soporta la operativa con el Vendedor y que hace uso de esta clase.

CLASE ArtPrefer

Realiza la incorporación a la Venta/ Pedido de los artículos que la Empresa ha marcado como preferenciales a la hora de vender.

RESTO DE CLASES DE LA APLICACIÓN

Voy a relacionar unos grupos de clases que no explicaré aquí porque las voy a exponer más detalladamente en el diseño arquitectónico explicando su funcionamiento y la forma como están construidas.

Juego de clases del nivel de Presentación: Formularios, con varias versiones de cada formulario.

Juego de clases donde está el código de programación de cada formulario. Es un código común para las distintas versiones de cada formulario.

Juego de clases que realizan el Interface de la Aplicación de la PDA con la Aplicación Comercial de la Empresa. Es la Carga y Descarga de Datos.

Juego de clases que realizan la impresión de documentos e informes, la visualización, la generación del informe en fichero de texto o en formato pdf y el envío por email de algún listado.

Juego de clases que realizan la búsqueda de Cliente, Artículo, documentos de venta, ...

Juego de clases que dan servicio a toda la Aplicación en general: PDAini, Contador, Licencia.

Juego de clases que dan servicio a los Formularios. Son clases que contribuyen a la personalización de la Aplicación:
Frm, AdaptarListview, TecladoVirtual, TeclasFuncion, Pegasus.

POR ÚLTIMO TRES CLASES MUY IMPORTANTES

Voy a explicar ahora tres clases que he incorporado en la Aplicación y que trabajan de manera “silenciosa”, pero que garantizan que la Aplicación funcione adecuadamente en la PDA.
En caso de una anomalía darán el correspondiente mensaje al vendedor.
Son las Clases: InicioPrevio, EstadoMemoria, Errores.

CLASE InicioPrevio :

Antes de visualizar el Menú General de la Aplicación se realizan unos controles:
Comprobamos que Hay sufuciente memoria para ejecutar la Aplicación.
Comprobar que la fecha de la PDA está con configuración en español.

Comprobar que la coma decimal y el punto de millar están como la utilizamos nosotros.
Comprobamos que la Aplicación tiene Licencia.
Leemos el fichero de configuración .INI.
Abrimos la conexión a la Base de Datos y leemos algunas tablas básicas cuya información ha de estar disponible para todos los programas de forma permanente.

CLASE EstadoMemoria :
Las PDAs no se caracterizan por tener demasiada memoria, habiendo bastante diferencia de memoria libre entre unas marcas y modelos de PDAs y otras y además como se pueden utilizar para muchas aplicaciones: teléfono incorporado, navegador GPS, aplicaciones diversas de gestión personal, juegos, ..., e incluso algunas PDAs tienen la carácterística de que se puede balancear la parte de memoria que se asigna a ejecución de programas y la parte de memoria que se asigna a almacenamiento de ficheros.
Por tanto, nos podemos encontrar con que la memoria que queda libre para ejecutar la AlfaPDA es escasa o que es tan justa que no va a permitir unos albaranes de venta de varias decenas de líneas de detalle (las pruebas de memoria permiten hacer un albaran de venta de más de cien líneas de artículos diferentes, cosa que es más que suficiente para una Autoventa con vehículos de reparto).

Para prevenir posibles problemas de saturación de memoria se hace un control inicial de memoria disponible y además en los documentos con captura de líneas de artículo (que se gestionan mediante un objeto “documento” que maneja en memoria el documento completo y que cuando se ha terminado el documento, graba todo de una vez a la base de datos) se hace periódicamente cada cierto número de líneas una comprobación de memoria disponible y en el caso de que la memoria libre sea insuficiente para seguir introduciendo líneas se solicita del vendedor que termine el actual documento y si es el caso vuelva a abrir otro documento para continuar con el cliente actual. Insisto en que la AlfaPDA admite más de 100 líneas de artículos diferentes por documento, cosa que es suficiente para cualquier reparto de mercancía.

La Aplicación AlfaPDA continuamente va destruyendo los objetos en cuanto no se necesitan y liberando la memoria utilizada por ellos.

CLASE Errores :
Por último, una característica importantísima de la AlfaPDA para dar un buen servicio de atención al cliente cuando tenga problemas, es esta clase, que Gestiona la grabación de un histórico de posibles errores o mensajes críticos.
Esta clase se utiliza para grabar en un fichero de texto los mensajes que le hayan podido salir al Vendedor y que sean consecuencia de alguna anomalía susceptible de comunicar a su Empresa. Es un histórico de errores en donde figura la fecha, hora, el método donde se ha producido el mensaje y la explicación del mensaje.
Lo normal es que no se tenga que grabar nunca ningún mensaje en este fichero, porque implicaría una incidencia de funcionamiento que hay que comunicar.

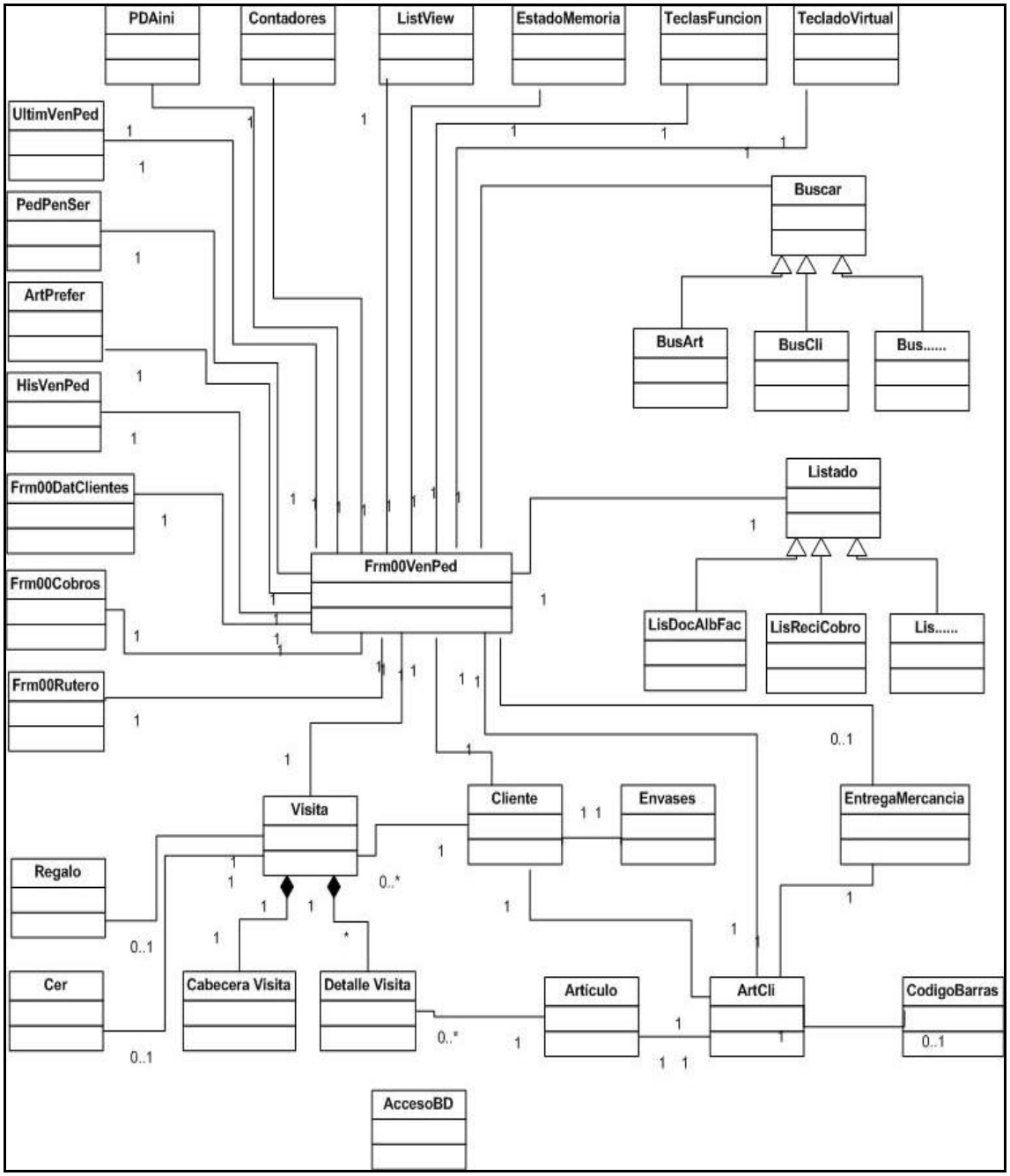
La nomenclatura de estos ficheros de Errores es:
AlfaPDAErrortttmmdd.txt
siendo: ttt = numero terminal mm= mes dd = dia
En el caso de que hubiera que grabar los mensajes de error, se crearía el fichero de ese día con el primer mensaje de error a grabar.
Cuando el vendedor sale de la Aplicación, si hay mensajes grabados, verá un mensaje recordatorio de que comunique a la Empresa que ha habido errores y se le informará del nombre del fichero donde están grabados dichos errores.
Por ejemplo: si un vendedor ha instalado la Aplicación y quiere empar a trabajar y no ha realizado ninguna Carga de la PDA, la base de datos estará vacía, ni siquiera tendrá el número de terminal que tiene asignada la PDA, la Aplicación le pedirá que introduzca ese número y a continuación le pedirá que haga una Carga de la PDA. Si hace una Carga pero selecciona un fichero de Interface que no tiene una estructura correcta, la Aplicación le avisará con mensajes de ese problema. Estos mensajes se grabarán en este Log de errores para que quede constancia de que la Carga no ha sido correcta.

Estos ficheros de error se conservan unos cuantos días y la Aplicación los borra automáticamente al pasar esos días.

Esta Gestión de errores es básica para dar el servicio de mantenimiento adecuado a las Empresas por parte de Alfa Informática, ya que si hay alguna incidencia en alguna PDA nos traemos esos ficheros de error y vemos lo que le ha pasado a la PDA, pues de lo contrario a veces sólo tendríamos la información de que “esto no funciona y salen errores”.

DIAGRAMA DE CLASES DE LA REALIZACION DE UNA VENTA / PEDIDO

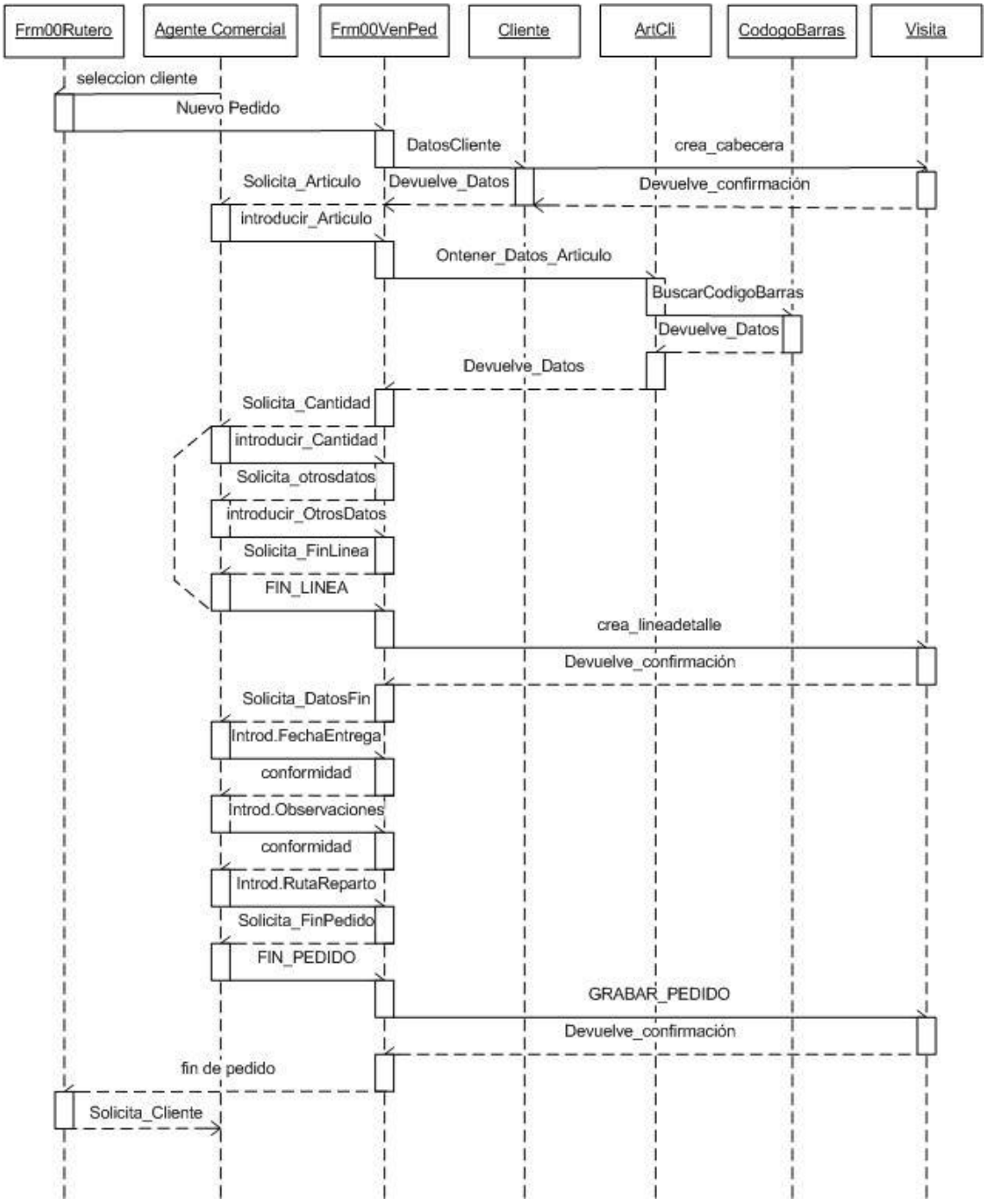
A continuación voy a exponer en un único cuadro lo que sería el diagrama de clases que corresponde a la realización de una Venta/Pedido. No relaciono en el diagrama ni los atributos (que son todos privados) ni los métodos (que los hay privados, protegidos y públicos) porque se desbordaría el gráfico.



3.3. DIAGRAMA DE SECUENCIA (TRAZA DE EVENTOS).

Dentro de los Diagramas de Comportamiento, voy a elegir un Diagrama de Interacción, el Diagrama de Secuencia, para reflejar lo que sería el Nuevo Pedido descrito en el caso de uso.

Nuevo_Pedido



4.- ASPECTOS DE DISEÑO DEL SISTEMA

4.1. ESTILO ARQUITECTÓNICO.

En la Especificación de Requisitos Software se han ido relacionando una serie de objetivos que se tenían que cumplir, ahora voy a exponer la solución elegida para cada uno de los requisitos que ya se han mencionado anteriormente.

Formularios

En Requisitos de Interfaces de Usuario se especificaba:

Requisito:
Los diseños de pantalla deben estar preparados para funcionar en cualquier tamaño y resolución de pantalla, incluso en diferente posicionamiento de pantallas (vertical, apaisada).
La Aplicación debe estar preparada para permitir diferentes diseños de pantalla al cliente que lo solicite, sin que por eso tenga que modificarse el código del programa, y debe prepararse para que el coste de tiempo de dicha adaptación al cliente sea mínima.

Solución:
La Aplicación consta de un total de 39 Formularios. Se han diseñado ya 4 Presentaciones diferentes. Cada Presentación hace referencia a los diseños de los 39 Formularios.
Hay 4 clases que realizan la Gestión de su Presentación respectiva. Además, como en las PDAs la memoria es escasa, para que el ejecutable ocupe lo mínimo, cuando se hace una complicación para la Presentacion 1, las 3 clases restantes (y sus juegos de formularios) están excluidas del proyecto, y análogo para las otras.

Tenemos pues:
Clases Formu1 Formu2 Formu3 Formu4 gestionan Presentaciones 1-2-3-4.

Formularios Frm01AlbProveedor Frm01 CargaMer Frm01Cobros .....
Son la versión 01 de los formularios

Clases Frm00AlbProveedor Frm00CargaMer Frm00Cobros .....
Son la clase donde está la programación de los Formularios.
Frm00AlbProveedor es la clase con la programación que gestiona los formularios: Frm01AlbProveedor Frm02AlbProveedor Frm03AlbProveedor Frm04AlbProveedor

Objetos FrmAlbProveedor FrmCargaMer FrmCobros .....
Son los objetos instanciados de las clases Formularios.
FrmAlbProveedor es el objeto de: Frm01AlbProveedor Frm02AlbProveedor Frm03AlbProveedor Frm04AlbProveedor

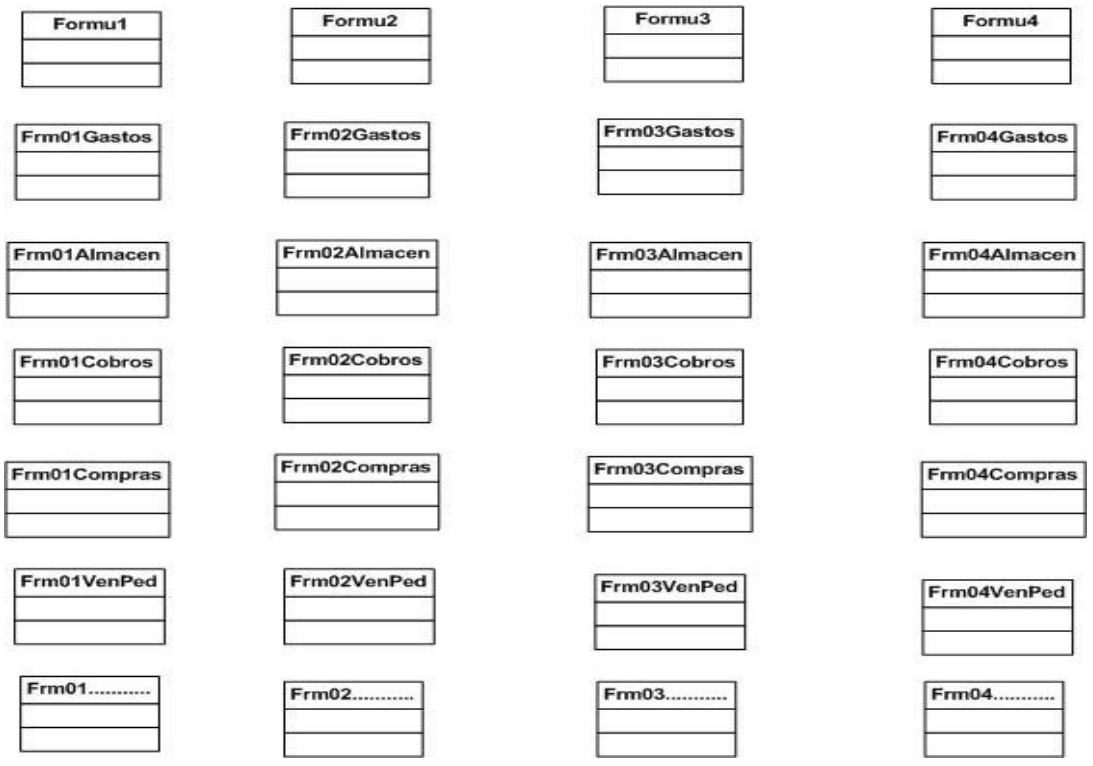
Objetos Frm0AlbProveedor Frm0CargaMer Frm0Cobros .....
Son los objetos instanciados de las clases Frm00AlbProveedor ...

En el código de los formularios sólo está la referencia a la ejecución del método correspondiente en el objeto que gestiona el código. Y este código además es el mismo en los distintos diseños de formularios
Ejemplo: En Frm01AlbProveedor.vb están los siguientes métodos

```
Private Sub FrmAlbProveedor_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    Frm0AlbProveedor.FrmAlbProveedor_Load()
End Sub
Private Sub TxtCantidad_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtCantidad.GotFocus
    Frm0AlbProveedor.TxtCantidad_GotFocus(sender)
End Sub
Private Sub TxtCantidad_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtCantidad.KeyPress
     Frm0AlbProveedor.TxtCantidad_KeyPress(sender, e)
End Sub
Private Sub TxtCantidad_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtCantidad.LostFocus
     Frm0AlbProveedor.TxtCantidad_LostFocus(sender)
End Sub
Private Sub BtnBusArt_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BtnBusArt.Click
     Frm0AlbProveedor.BtnBusArt_Click()
End Sub
```

.....

Quedarían los 4 juegos de Presentaciones con la siguiente distribución de Clases:



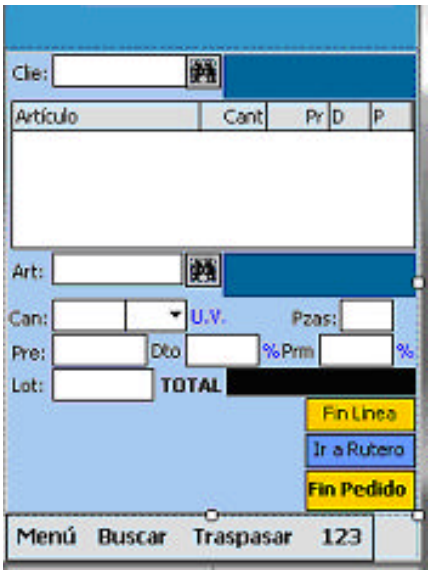
A continuación, a modo de ejemplo, vemos para el formulario de Ventas/Pedidos los 4 juegos de Presentaciones .
La diferencia es:

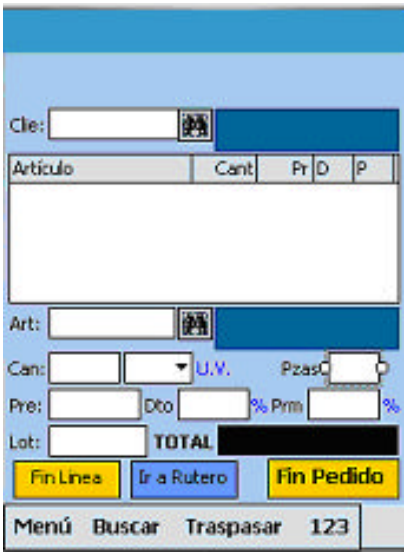
El primero trabaja con Windows Mobile 2005 que no tiene teclado físico y que se le reserva la zona inferior izquierda de la pantalla para el teclado numérico virtual.

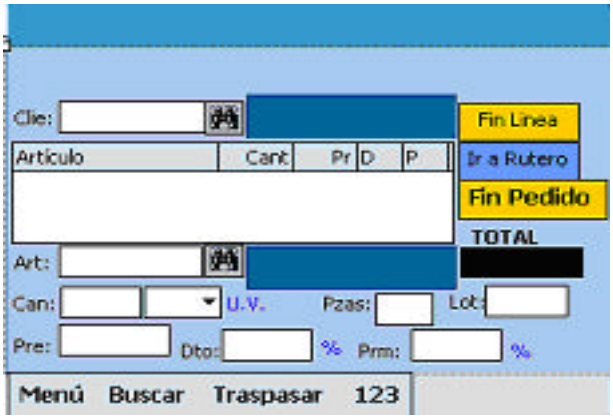
El segundo trabaja con Windows CE 5 con teclado físico y que por tanto no necesita reserva de espacio para teclado virtual, pero se puede solicitar pulsando <123> y aparecería momentáneamente encima del listview, pero la zona de pantalla a diseñar es más pequeña que en Windows mobile 2005

El tercero es un diseño de pantalla completamente diferente para una forma de introducción de Pedidos con unas necesidades particulares.

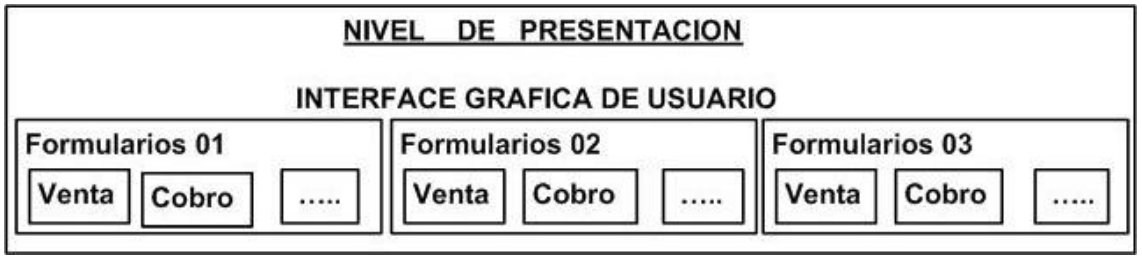
El cuarto trabaja con un dispositivo industrial que tiene una pantalla de tipo apaisada con Windows CE 5.







Obtendríamos muy esquemáticamente el nivel de Presentación



En INICIO de la Aplicación hay definidos los siguientes objetos que valen para cualquier juego de formularios de Presentación:

```
Frm0AlbProveedor As Frm00AlbProveedor
Frm0Almacen As Frm00Almacen
Frm0Buscar As Frm00Buscar
Frm0BuscarOpciones As Frm00BuscarOpciones
Frm0CargaMercancia As Frm00CargaMercancia
Frm0CargaPDA As Frm00CargaPDA
Frm0Cobros As Frm00Cobros
Frm0Compras As Frm00Compras
Frm0DatClientes As Frm00DatClientes
Frm0DatClientesMasDatos As Frm00DatClientesMasDatos
Frm0DescargaPDA As Frm00DescargaPDA
Frm0DesResuActi As Frm00DesResuActi
Frm0Envios As Frm00Envios
Frm0FormaPago As Frm00FormaPago
Frm0Gastos As Frm00Gastos
Frm0HisVenPed As Frm00HisVenPed
Frm0ImpFacPen As Frm00ImpFacPen
Frm0IncidenciasCli As Frm00IncidenciasCli
Frm0Informes As Frm00Informes
Frm0IniStock As Frm00IniStock
Frm0Listar As Frm00Listar
Frm0ManteClientes As Frm00ManteClientes
Frm0ManteClientesMasDatos As Frm00ManteClientesMasDatos
Frm0ManteDirEnvio As Frm00ManteDirEnvio
Frm0MenuPrincipal As Frm00MenuPrincipal
Frm0Parametros As Frm00Parametros
Frm0PedirDatos As Frm00PedirDatos
Frm0PedPenServ As Frm00PedPenServ
Frm0PenCobros As Frm00PenCobros
Frm0Reenvio As Frm00Reenvio
Frm0ReimDocum As Frm00ReimDocum
Frm0Rutero As Frm00Rutero
Frm0SelDiaSem As Frm00SelDiaSem
Frm0SelDoc As Frm00SelDoc
Frm0SelStk As Frm00SelStk
Frm0UltimasVentas As Frm00UltimasVentas
Frm0VenPed As Frm00VenPed
Frm0VenPedMasDatos As Frm00VenPedMasDatos
Frm0VerDocum As Frm00VerDocum
```

Formu1 tiene entre otras cosas definidos los objetos:

```
FrmAlbProveedor As Frm01AlbProveedor
FrmAlmacen As Frm01Almacen
FrmBuscar As Frm01Buscar
FrmBuscarOpciones As Frm01BuscarOpciones
FrmCargaMercancia As Frm01CargaMercancia
FrmCargaPDA As Frm01CargaPDA
FrmCobros As Frm01Cobros
FrmCompras As Frm01Compras
FrmDatClientes As Frm01DatClientes
FrmDatClientesMasDatos As Frm01DatClientesMasDatos
FrmDescargaPDA As Frm01DescargaPDA
FrmDesResuActi As Frm01DesResuActi
FrmEnvios As Frm01Envios
FrmFormaPago As Frm01FormaPago
FrmGastos As Frm01Gastos
FrmHisVenPed As Frm01HisVenPed
FrmImpFacPen As Frm01ImpFacPen
FrmIncidenciasCli As Frm01IncidenciasCli
FrmInformes As Frm01Informes
FrmIniStock As Frm01IniStock
FrmListar As Frm01Listar
FrmManteClientes As Frm01ManteClientes
FrmManteClientesMasDatos As Frm01ManteClientesMasDatos
FrmManteDirEnvio As Frm01ManteDirEnvio
FrmMenuPrincipal As Frm01MenuPrincipal
FrmParametros As Frm01Parametros
FrmPedirDatos As Frm01PedirDatos
FrmPedPenServ As Frm01PedPenServ
FrmPenCobros As Frm01PenCobros
FrmReenvio As Frm01Reenvio
FrmReimDocum As Frm01ReimDocum
FrmRutero As Frm01Rutero
FrmSelDiaSem As Frm01SelDiaSem
FrmSelDoc As Frm01SelDoc
FrmSelStk As Frm01SelStk
FrmUltimasVentas As Frm01UltimasVentas
FrmVenPed As Frm01VenPed
FrmVenPedMasDatos As Frm01VenPedMasDatos
FrmVerDocum As Frm01VerDocum
```

Y también tiene el método ejecutar con la forma de gestionar los formularios, tanto si no hay que pasar parámetros como si hay que pasar parámetros, que por supuesto hay que pasarlos al objeto que tiene la programación del formulario Frm0Buscar, no al objeto que tiene el diseño de formulario FrmBuscar.

```
Public Shared Sub ejecutar(ByVal NomPrograma As Short, ByVal
ParamArray parametro() As Object)
Select Case NomPrograma
            Case Programa.FrmAlbProveedor
                        Frm0AlbProveedor = New Frm00AlbProveedor
                        FrmAlbProveedor = New Frm01AlbProveedor
                        FrmAlbProveedor.ShowDialog()
                        Frm0AlbProveedor = Nothing
                        FrmAlbProveedor = Nothing
            Case Programa.FrmAlmacen
```

```
                        Frm0Almacen = New Frm00Almacen
                        FrmAlmacen = New Frm01Almacen
                        FrmAlmacen.ShowDialog()
                        Frm0Almacen = Nothing
                        FrmAlmacen = Nothing
            Case Programa.FrmBuscar
                        Frm0Buscar = New Frm00Buscar
                        FrmBuscar = New Frm01Buscar()
                        Frm0Buscar.Nueva(parametro(0), parametro(1))
                        FrmBuscar.ShowDialog()
                        Frm0Buscar = Nothing
                        FrmBuscar = Nothing
.........................................
```

Requisito:
Se dispondrá de un teclado emergente numérico y otro alfanumérico, que se presentarán automáticamente sólo cuando se necesiten, según el dato a rellenar. También pueden visualizarse o esconderse cuando el usuario los seleccione.
Hay que tener en cuenta que los dispositivos móviles que se utilicen pueden ser:
- con o sin teclado físico, por lo que a aquellos que tengan teclado no se les presentará automáticamente el teclado virtual emergente.
- con o sin teclas de función, y diferente número de teclas de función: hay que prever que el usuario pueda cambiar la tecla asignada a cada función.
- con o sin lector de código de barras, por lo que aquellas que tengan lector de código de barras deben poder hacer un uso automático de dicha prestación.

Solución:
Para conseguir la versatilidad en la Presentación y la Personalización del Usuario he creado unas clases basándome en un criterio de Agrupación Funcional, y que dichas clases tengan dentro de ellas todo lo necesario para dar el servicio completo a todos los formularios de la Aplicación, en cualquiera de sus versiones de Presentación.
Voy a relacionar dichas Clases.

La clase PDAini gestiona la disponibilidad del acceso a todos los parámetros desde cualquier punto de la Aplicación.
Entre los parámetros están:
Activar_Teclado_Numerico="S"
Activar_Teclado_Alfanumerico="S"
Activar_Teclado_Numerico_Automatico="N"
Activar_Teclado_Alfanumerico_Automatico="N"

Hay una clase TecladoVirtual en donde está definida tanto el objeto visual de teclado como los métodos que gestionan todo el tratamiento, y estos métodos son únicos y comunes para todos los formularios de todas las versiones de Presentación.

Están también los parámetros:
Utilizar_Teclas_Funcion="S"
F1_Buscar="F1"
F2_Seleccion_Tabla="F2"

F3_Tecla3="F3"
F4_Carga_Aplicacion="F4"
F5_Menu1="F5"
F6_Menu2="F6"
F7_Tecla7="F7"
F8_Fin_Linea="F8"
F9_Salir="F9"
F10_Fin_Venta="F10"

De esta forma cada empresa elige si utiliza o no teclas de función y qué tecla es la que quiere asociar a cada función que por defecto ya tienen una asignación.

Hay una clase TeclasFunción en donde está definida todos los métodos de todos los formularios de todas las versiones de Presentación.

Los parámetros:
Gestion_Codigo_Barras="S"
Prioridad_Codigo_Barras_Busqueda_Articulo="S"

Junto con la clase CodigoBarras dan ese servicio cuando está definido.

La clase Pegasus tiene la gestión del teclado e impresora en un dispositivo industrial específico que tiene todo incorporado en un bloque compacto y que además tiene tratamiento especial porque teclado e impresora comparten el mismo puerto.

La clase Frm tiene métodos que gestionan necesidades que tienen todos los formularios, por ejemplo:
ValidarNum SoloNum NumeroDeDecimales ControlarDecimales
SubirLineaList BajarLineaList .....

Requisito:
Se ha de procurar que cada Empresa sólo vea en Pantalla lo que realmente necesita en función de lo que haya definido en los parámetros.

Solución:

Para esto se definen los parámetros:
[Accesos]
Menu_Parametros="S"
Parametros_Privados="S"
Menu_Ventas="S"
Menu_Pedidos="S"
Menu_Cobros="S"
Menu_Compras="S"
Menu_Gastos="S"
Movimientos_Mercancia="S"
Menu_Ini_Stocks="S"
Listados="S"

Resumen_Ventas="S"
Resumen_Pedidos="S"
Reimpresion_Documentos="S"
Listado_Stock="S"
Listado_Pendiente_Cobro="S"
Listado_Ventas="S"
Recorrido_Diario="S"
Listado_Clientes_Sin_Visitar="S"
Ver_Documentos_Tipo_Texto="S"

Y según lo que se defina aparece o no aparece en pantalla.
Ejemplo, las pantallas siguientes muestran diferentes presentaciones según lo que cada Empresa quiere utilizar.

Requisito:
Al tener los listview que se visualizan en la Aplicación una serie de columnas y al tener la suma de sus longitudes más longitud que la anchura de la pantalla de la PDA, todos los listview tienen que estar preparados para que el propio usuario pueda redimensionar la anchura de cada campo del listview con el puntero, para darle a cada campo el tamaño óptimo e incluso para anular algún campo que no sea de relevancia para él. Ha de tener la opción de que esos tamaños se conviertan en los definitivos para él en su fichero de configuración, teniendo siempre la opción de cambiar momentáneamente la anchura de alguna columna en alguna ocasión. También ha de poder cambiar el tamaño de letra y el estilo de presentación.

Solución:
Para esto se definen los parámetros:
[Frm1]
Tamaño_Fuente_Lista="8"
Fuente_Negrita="N"
Estilo_Subrayado="N"

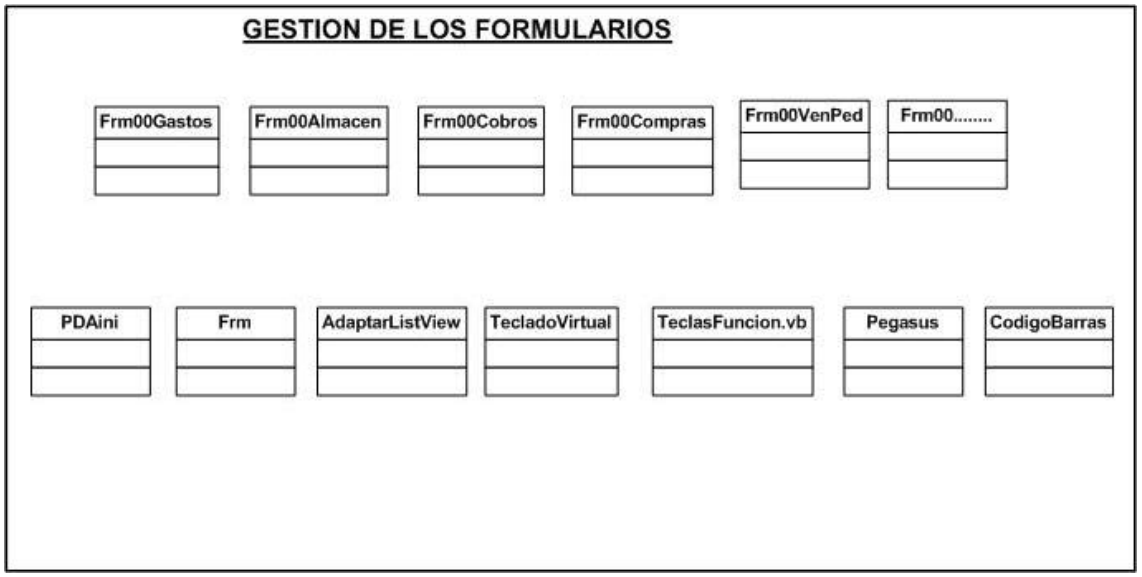
Alternancia_Registros="S"
Tamaño_Fuente_Listado="8"

[Frm2]
Actualizacion_Resize_Automatica="S"
Parámetros automáticos de la Gestión de todas las columnas de todos los listview de la Aplicación

Que junto con la clase AdaptarListView dan ese servicio.

Todos estos objetivos permiten que cada Empresa funcione de la manera que mejor se adapte a ella.

Obtendríamos por tanto dentro del NIVEL DE APLICACIÓN un primer NIVEL DE GESTION DE FORMULARIOS, y dentro de este nivel diferenciaríamos también dos niveles, las clases del nivel inferior darían servicio a todas las clases del nivel superior.



CARGA /DESCARGA DE PDA

En Supuestos y Dependencias se especificaba:

Requisito:
Para funcionar existe dependencia con la Aplicación de Gestión Comercial que tenga la Empresa, ya que se alimenta de Datos que se le deben suministrar desde ella, pero no está condicionada por una Aplicación concreta.
Las Tablas a importar y exportar, los datos de cada tabla con la definición y longitud del tipo de campo y el orden de esos datos en los ficheros de interface estarán controlados por un fichero de Control del Interface, de manera que no

se tengan que modificar programas para adaptar el contenido de los ficheros de interface a cada Empresa.

Solución:
La Carga y la Descarga de Datos a / desde la PDA la he diseñado para que se pueda adaptar a cualquier Aplicación Comercial que la Empresa tenga instalada en sus PCs .
Para eso he creado un fichero AlfaPDATerDat.txt de Definición del Interface de dichas Carga y Descarga.
Ese fichero de texto se puede cambiar y elegir distintas posibilidades:
Si se quiere un solo fichero de interface con todas las tablas o un fichero por cada tabla.
El nombre de las tablas que le llegan desde la Aplicación Central.
El número de campos de cada Tabla, el orden en que le llegan, el tamaño de cada campo, número de enteros y decimales, saltarse campos que no se necesitan en la PDA, incluir campos que no le llegan y que la PDA necesita, y asignarles un valor de inclusión por defecto.
También se puede hacer el Interface sustituyendo la Tabla completa o sólo realizando variaciones de Datos, y en este caso se pueden dar bajas de registro que tenga la PDA, modificaciones de registros y Altas de registros que no estén en la PDA.

Este fichero AlfaPDATerDat.txt está autoexplicado, incluyendo comentarios de cómo especificar los cambios para que se adapten a otras Aplicaciones Copmerciales.

En resumen: El programa de Interface de la PDA está preparado para que no se tenga que modificar y que sirva para adaptarse a cualquier fichero de interface que nos obliguen desde la Aplicación Comercial General que tenga instalada la Empresa. Sólo hay que modificar el contenido del fichero de texto que define la estructura de ese interface

Los siguientes parámetros completan las personalización :
UnSolo_Carico_Entrada="S"
UnSolo_Carico_Salida="S"
NumDias_A_Mantener_Ficheros_De_Descarga="5"
Descarga_Carga_Automatica="S"
Path_Datos="\DATOS\"
Path_Carga="\CARGA\"
Path_Descarga="\DESCARGA\"
Path_Ayudas="\AYUDAS\"

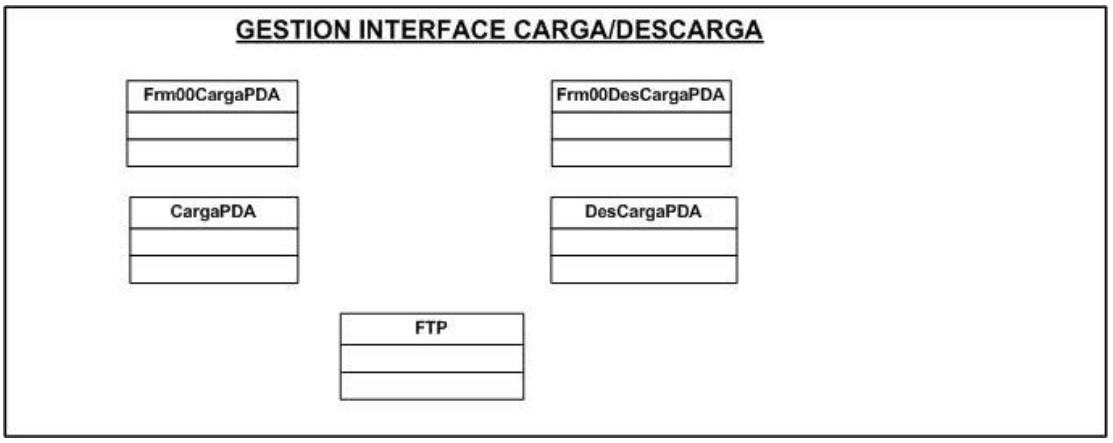
Interface de Comunicaciones:

Requisito:
En este punto se especificaban una gran variedad de formas en que la PDA podía físicamente intercambiar los Datos con el Ordenador Central.

Solución:
Para esto se definen los parámetros:
[Conexiones]
Activar_Transferencia_Internet="N"
Usuario="elquesea"
contraseña="laquesea"
servidor="elquesea.com"
Conectar_Automaticamente="N"
Conexion="nombre"
Retardo="100"

Las siguientes clases resuelven todas las posibilidades de conexión, además del fichero de texto AlfaPDATerDat.txt de Control ya explicado.
Frm00CargaPDA
Frm00DesCargaPDA
CargaPDA
DesCargaPDA
FTP

Obtenemos las siguientes clases:



IMPRESIÓN

Requisito:
La impresión ha de estar preparada: para que se adapte a diferentes impresoras (anchura, forma de conexión, papel continuo o formulario, ...), para que un mismo documento tenga diferentes diseños de impresión a elegir por cada Empresa, y para que se pueda repetir y/o visualizar en pantalla en cualquier momento cualquier documento de impresión .

Solución:
Para esto se definen los parámetros:
[Impresion1]
Pedidos_Cab="00"
Pedidos_Det="00"
Pedidos_Pie="00"
Albaranes_Cab="00"

Albaranes_Det="00"
Albaranes_Pie="00"
Facturas_Cab="00"
Facturas_Det="00"
Facturas_Pie="00"
Recibo_Cobro_Cab="00"
Recibo_Cobro_Det="00"
Recibo_Cobro_Pie="00"
……..
[Impresion2]
Lineas_Por_Pagina="048"
Lineas_Saltar_FinListado="005"
Imprimir_Documentos="S"
Impresion_Cobros_Automatica="S"
Impresion_DocumVenta_Automatica="S"
Salida_Listados="PAPEL"
Tipo_Impresora="STAR"
Imprimir_Datos_Empresa="S"
Papel_Continuo="N"
Copias_Documento_Venta="1"
Copias_Documento_Pedido="1"
Copias_Documento_CargaMercancia="1"
Copias_Documento_Recuento="1"
Copias_Documento_AlbProveedor="1"
Mensaje_Repetir_Listado="S"

Y he diseñado la siguiente estructura de Clases:

LOS LISTADOS CONSTAN:

1.- De una CLASE LisSelecInforme que lo que hace es la gestión de la seleción de un Informe concreto

2.- CLASE Listado: es la clase Padre para todos los listados, en donde están definidos de todos los atributos que son comunes a todos los listados y una serie de métodos, unos son comunes para todos los tipos de listados y otros son comunes para los listados tipo documentos.

Hay dos formas de Listar: Listar Documentos y Listar Informes.
Para todos los casos, los métodos Seleccion_Cabecera, Seleccion_Detalle y Seleccion_pie son los métodos de selección del diseño de impresión que hay a elegido el cliente para ese documento o informe.

Para listar documentos hay que llamar al método ListarDocumentos definido en la clase Listado. El método ListarDocumentos es la Rutina Común de los Documentos de Impresión (Cabecera y Detalle).

Listar Informes llama al método Listar que está definido en cada Informe que se implementa. Para cada Informe Nuevo a implementar hay que crear una clase para ese informe que heredará de la clase Listado.

Estos dos métodos ListarDocumento y Listar generan unos objetos con el contenido del listado.
Dichos objetos son:
Para la Rutina ListarPorPantalla para pasarlo al formulario Form_Listar:
Las variables siguientes tiene el listado completo con todas las páginas y el número de páginas del listado.
Protected Listado() As String ' cada elemento es una página completa
Protected NumeroPagina As Integer ' indica el número de páginas del listado
Para la Rutina ListarPorPapel:
Lo que hay en Listado() también está contenido en:
Protected Linea() As String ' El array Linea contiene las líneas de Cabecera y Pie
Protected LineaDetalle() As String ' El array LineaDetalle contiene todas las líneas de Detalle

Desde dicha rutina LISTAR (o ListarDocumento) se llama (según se haya pedido) a:
- ListarPorPapel
- ListarPorFichero
- ListarPorPantalla (desde esta pantalla también se puede llamar a ListarPorPapel, o ListarporFichero si es eso lo que está en el AlfaPDAini.ini)

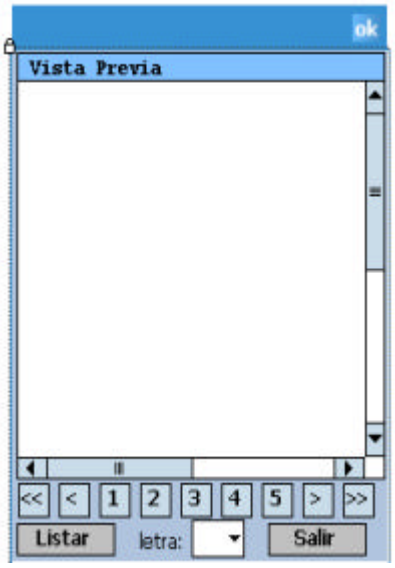
ListarPorPapel es la única rutina de impresión en papel una vez se han generado los objetos (Arrays) anteriores.
Es la Rutina de Impresión única que se utiliza en toda la aplicación tanto para imprimir documentos como para imprimir informes, y es una rutina que ya no hay que modificarla y que se adapta a cualquier diseño de cualquier informe.
Se utiliza tanto para imprimir directamente en papel o para imprimir partiendo de una previsualización en pantalla.
Para esto sólo necesita partir de los arrays ya generados:
Linea() y LineaDetalle().

3.- MUCHAS CLASES, (LisDocAlbFac, LisDocReci, ...) una para cada listado que se implementa. Todas estas Clases heredan de la Clase Listado.
Cada una de estas clases contiene:
En el caso de que sea un "Documento" sólo tiene el “diseño” del listado porque la rutina de impresión es ListarDocumentos que es común a todos los documentos. Por tanto para preparar la impresión de nuevos documentos o de nuevos diseños de “tipos de documentos” ya existentes sólo hay que hacer nuevo el diseño, porque la programación existente sirve para todos.
En el caso de que sea un "Informe" la clase contiene el método listar y el diseño de la impresión.

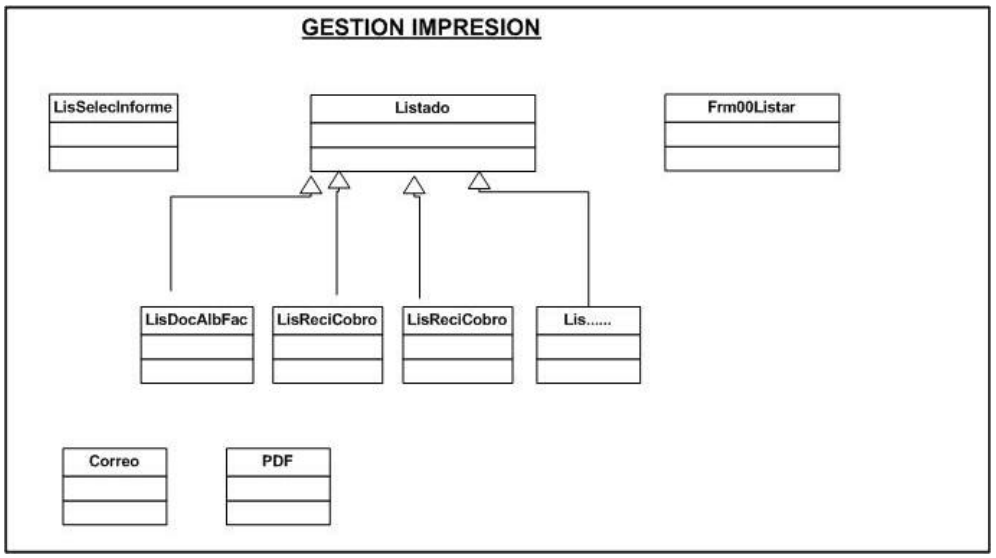
4.- CLASE “VISTA PREVIA” Además cualquier Documento o Informe se puede opcionalmente visualizar antes de imprimir y cualquier documento ya impreso se puede volver a visualizar / imprimir. Es decir, he preparado lo que se llama “Vista Previa” de la impresión.
He diseñado un formulario Frm01Listar con su gestión de código en la clase Frm00Listar, que es único para todos los listados y ya no hay que modificarlo y

sirve para “visualizar en pantalla” cualquier listado o documento que ya exista o cualquiera que se pueda definir en el futuro, y consta de una zona de visualización con las barras deslizantes vertical y horizontal y con una barra inferior de botones que contiene 5 páginas de acceso directo en el entorno de la página actual en la que estemos y los botones de: comienzo listado << fín listado >> página anterior < y página siguiente >.
La salida de los listados puede ser por impresora, por pantalla o grabar en fichero.

5.- CLASES Correo y PDF. Además si se graba en fichero se puede grabar en formato texto o en formato pdf y opcionalmente se puede enviar un email con la impresión en pdf, por ejemplo para realizar los pedidos a proveedores.



Obtenemos las siguientes clases:



BUSQUEDAS

En Requisitos Funcionales se especificaba:

Requisito:
La selección del Cliente se hace bien en el Rutero o bien en la pantalla de Búsqueda de Cliente ayudado de las posibilidades generales de las búsquedas (selección de campo de búsqueda: código, nombre, población, ...)
Selección de Artículo: bien introduciendo el código o con el lector del código de barras, o yendo a la búsqueda de artículo seleccionando la búsqueda por cualquiera de los campos posibles, grupo, subgrupo, código, descripción.

Solución:
Para esto se definen los parámetros:
Actualizacion_Parametros_Automatica="S"
Articulo_Interactivo="S"
Articulo_Grupo="01"
Articulo_Subgrupo="01"
Articulo_Campo="01"
Articulo_Condicion="01"
Articulo_Cabeceras="01"
Cliente_Interactivo="S"
Cliente_Campo="01"
Cliente_Condicion="01"
Cliente_Cabeceras="01"
Proveedor_Interactivo="S"
Proveedor_Campo="01"
Proveedor_Condicion="01"
Proveedor_Cabeceras="01"

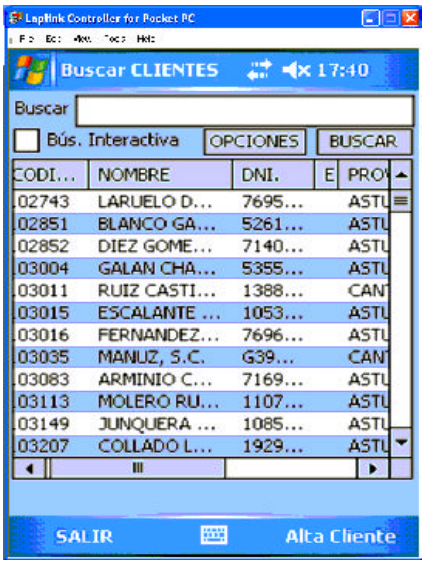
Y he diseñado la siguiente estructura de Clases:

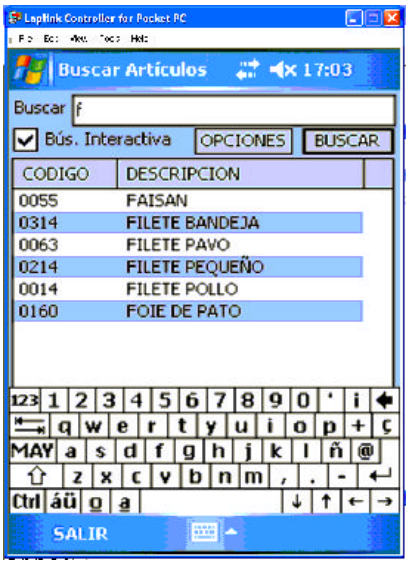
LAS BUSQUEDAS CONSTAN:

1.- CLASE Buscar: es la clase Padre para todas las búsquedas, en donde está definido lo que es común a todas.

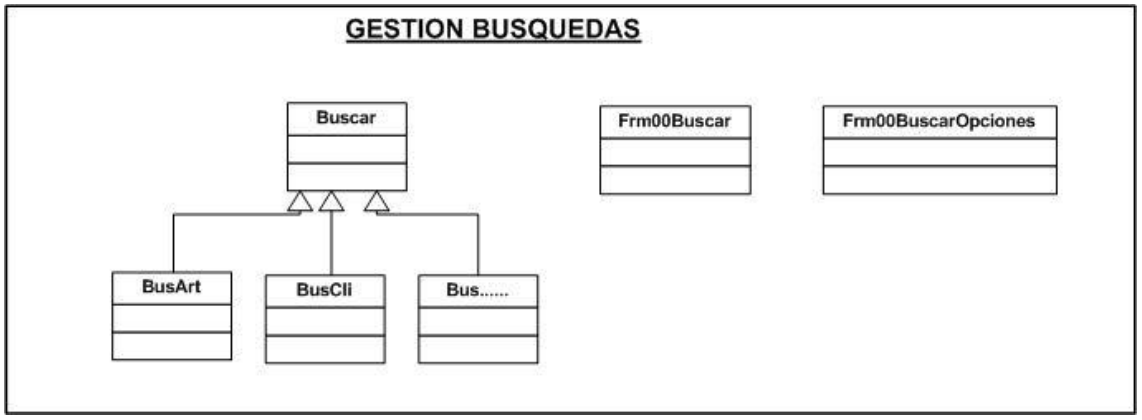
3.- MUCHAS CLASES, una para cada búsqueda que se implementa. Todas estas Clases heredan de la Clase Buscar.
BuscarArticulos
BuscarArticulosPromocion
BuscarClientes
BuscarVentas
BuscarPedidos
BuscarFPago
BuscarAlbaranProveedor
BuscarProveedores
Se definen los campos de la tabla por los que se puede buscar , las distintas condiciones de búsqueda posibles: contiene, comienza por, termina en, igual, y las columnas de la tabla que se quieren ver (encabezado de la lista).

4.- CLASE “VISUALIZACION” común para todas las Búsquedas, para la gestión en pantalla de la búsqueda.
He diseñado los formularios Frm01Buscar y Frm01BuscarOpciones con su gestión de código en las clases Frm00Buscar y Frm00BuscarOpciones, que son únicos para todas las búsquedas y sirven para “visualizar en pantalla” cualquier búsqueda de cualquier tabla de la que tengamos realizada su clase de búsqueda.





Obtenemos las siguientes clases:



Especificación de Requisitos Funcionales

Requisito:
En la especificación de Requisitos Funcionales, en cada Acción que debía reralizar el Software, aparecía una "Funcionalidad" que debía tener la Aplicación.

Solución:
La solución que he elegido es diseñar Clases que resuelvan objetivos según la "funcionalidad" que se tenía que dotar a la Aplicación.

Por ejemplo, he creado una Clase CodigoBarras que se utiliza desde los formularios de Ventas, Pedidos, Carga Mercancía, Recuentos y Pedidos a Proveedor.

Las Clases que he creado que responden a dichos Objetivos Funcionales son:
Rutero
VenPed
VenPedMasDatos
EntregaporcuentaProveedor
HisVenPed
PedPenSer
UltimVenPed
Regalo
Cer
ArtPrefer
CodigoBarras
Envases
GestionStock

Y también he creado unas Clases Básicas para la Aplicación con un gran contenido de Propiedades y Métodos que resuelven toda la problemática que tenga que ver con ellas.
Dichas Clases son:
Cliente
Articulo
ArtCli
Visita
VisitaCabecera
VisitaDetalle
AlbProveedor
CargaMercancia
Recuento
Cobros

A modo de ejemplo:
La Clase "Visita" corresponde tanto a un documento de Venta como a un documento de Pedido.
Consta entre otras cosas de:

Un objeto Cliente de la Clase Cliente
Un objeto CabeceraVisita de la Clase VisitaCabecera
Cero, uno o muchos objetos DetalleVisita de la Clase VisitaDetalle

La Clase "VisitaDetalle" a su vez tiene un objeto Artículo de la Clase Artículo y la información relacionada con una línea de detalle de un documento de venta, junto con los métodos de grabación, recuperación.

La clase Cliente contiene toda la información y gestión que tenga que ver con un cliente.

Para la GESTION DE VENTAS / PEDIDOS intervienen clases a diferentes niveles:
Aparte de las Clases/Formularios a nivel de presentación Frm01VenPed y Frm01VenPedMasDatos y de las clases presentación Frm00VenPed y Frm00VenPedMasDatos que contienen la programación de dichos formularios, intervienen las siguientes clases:

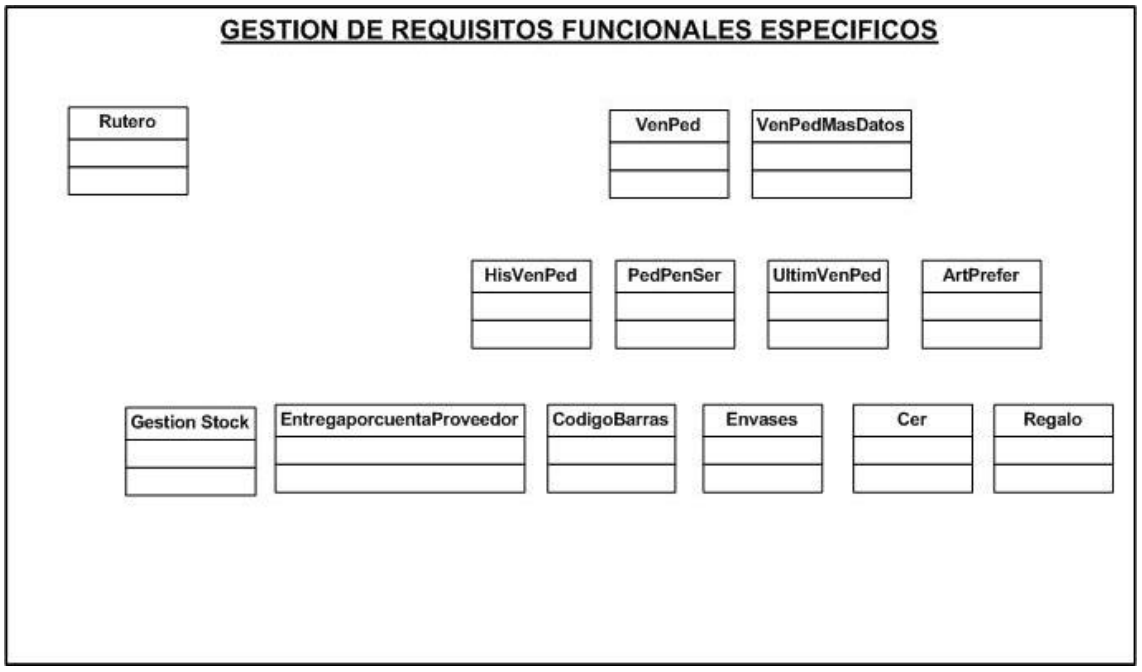
- VenPed Realiza Tareas en la Pantalla de Venta/Pedido
- VenPedMasDatos Realiza Tareas en la Pantalla Otros Datos de Venta/Pedido

Las Clases siguientes son para facilitar la captura de líneas de detalle en una Venta/Pedido, mediante incorporación masiva seleccionando desde otro formulario de captura de datos:
- HisVenPed Realiza la incorporación a la Venta/ Pedido desde un Histórico de Ventas/Pedidos.
- PedPenSer Realiza la incorporación a la Venta de los Pedidos que tiene un Cliente pendiente de Servir
- UltimVenPed Realiza la incorporación a la Venta/ Pedido desde las Últimas Ventas/Pedidos.
- ArtPrefer Realiza la incorporación a la Venta/ Pedido de los artículos que la Empresa ha marcado como preferenciales a la hora de vender.

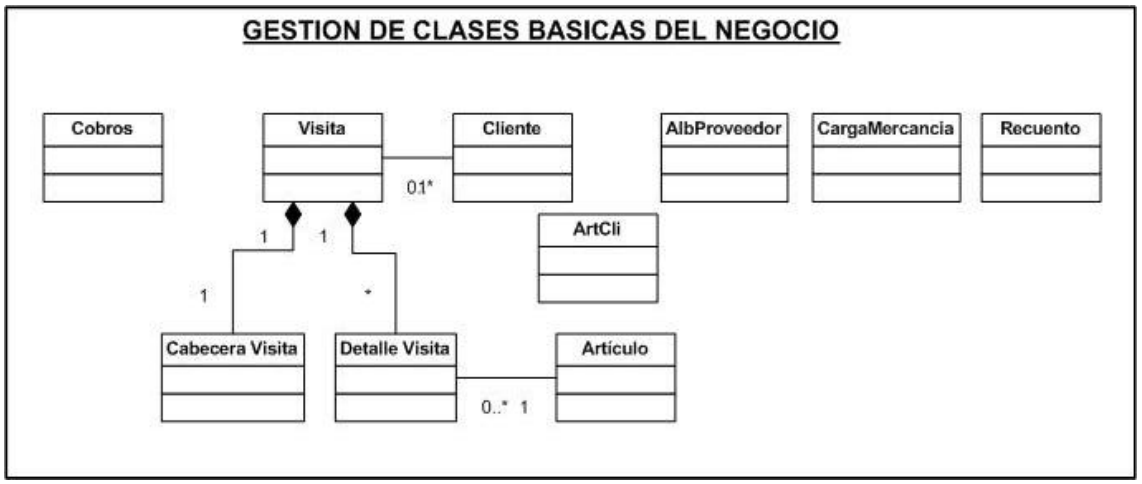
Además intervienen un conjunto de Clases que realizan tareas específicas:
- EntregaporcuentaProveedor Realiza Tareas para ese Tipo de Venta Especial en el que la Venta consiste básicamente en hacer la Entrega de Mercancía ya que las condiciones de Venta se negocian directamente entre El "Cliente o Cadena" con el Proveedor / Fabricante de la Mercancía.
- Regalo Realiza la Gestión de la Política de Regalos y Promociones especiales en función de distintos criterios y posibilidades a elegir por cada Empresa.
- Cer Realiza la Gestión del "Coste por Eliminación de Residuos" que algunas Empresas están obligadas a llevar.
- Envases Realiza la Gestión y Control de los envases en depósito/fianza en el Cliente.
- CodigoBarras Realiza la captura de un código de barras.

Y también intervienen las Clases Básicas del Negocio:
Cliente, Articulo, ArtCli, Visita, VisitaCabecera, VisitaDetalle.

Obtenemos las siguientes clases:

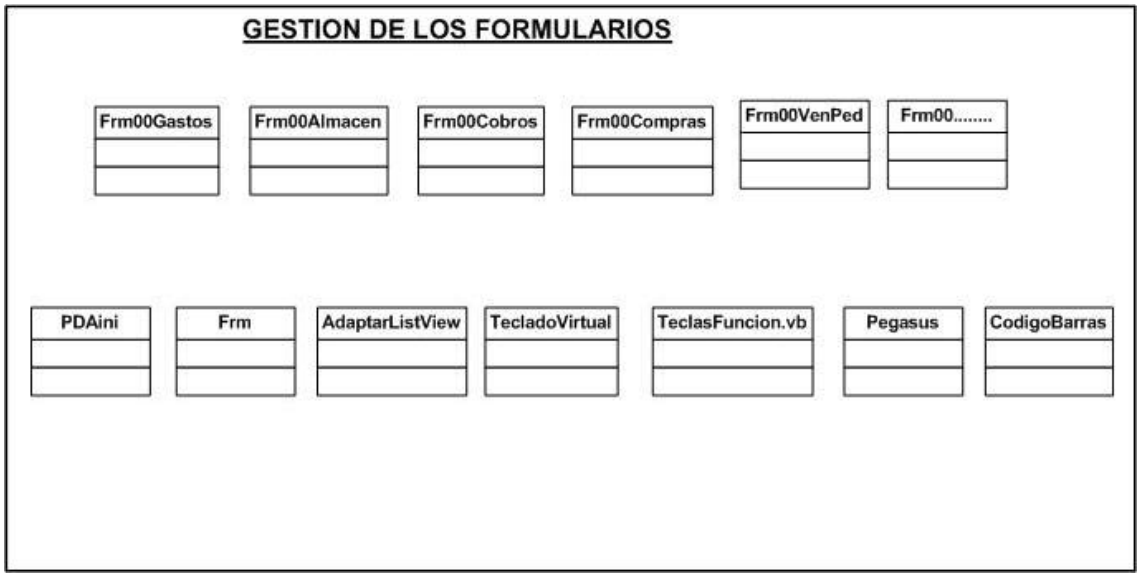


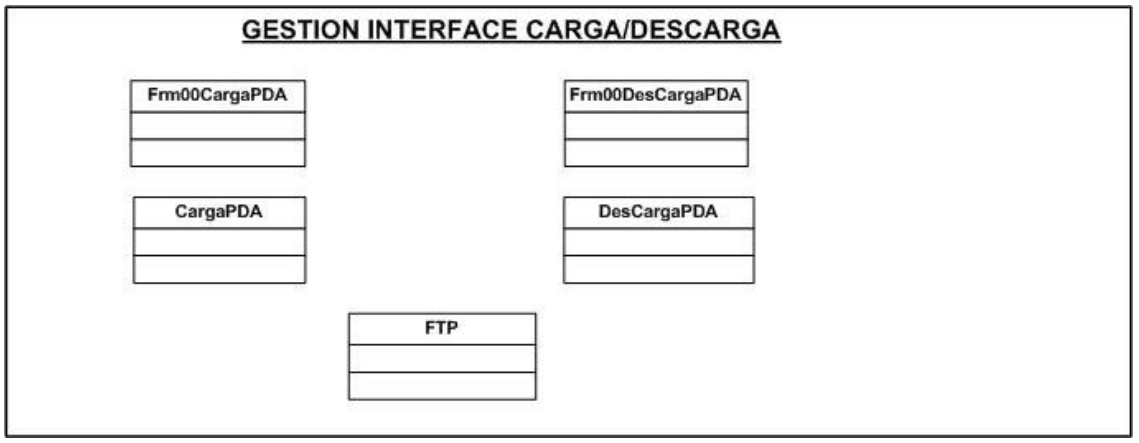
Que se completan con las Clases Básicas del Negocio

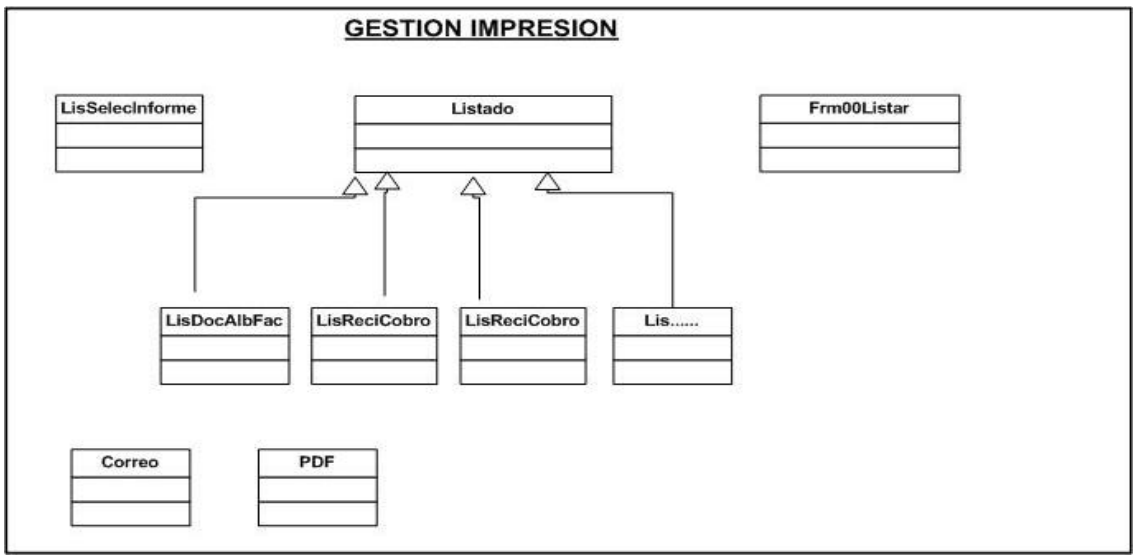


REUNIENDO TODAS LAS CLASES ANTERIORES QUEDARÍA

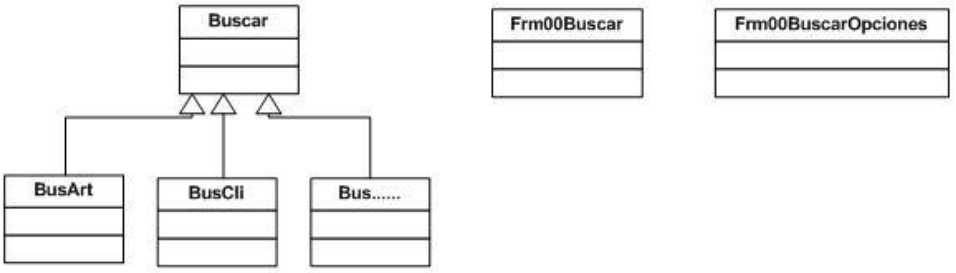
NIVEL DE DOMINIO O APLICACIÓN



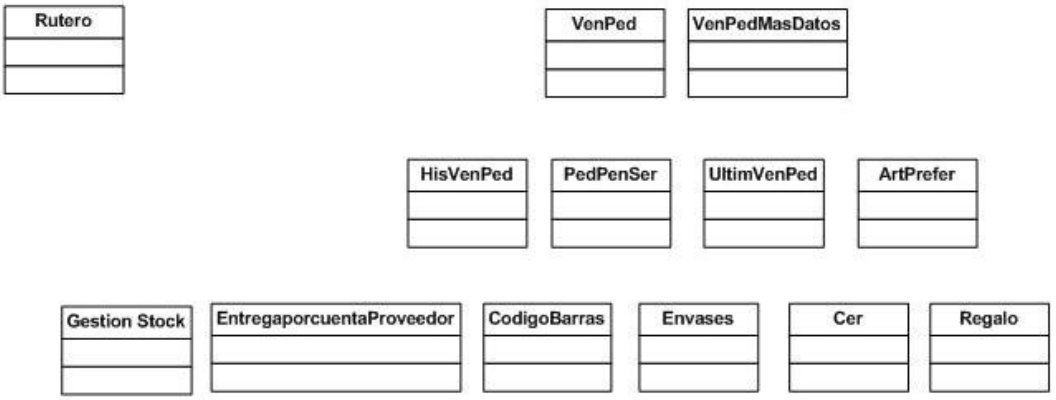




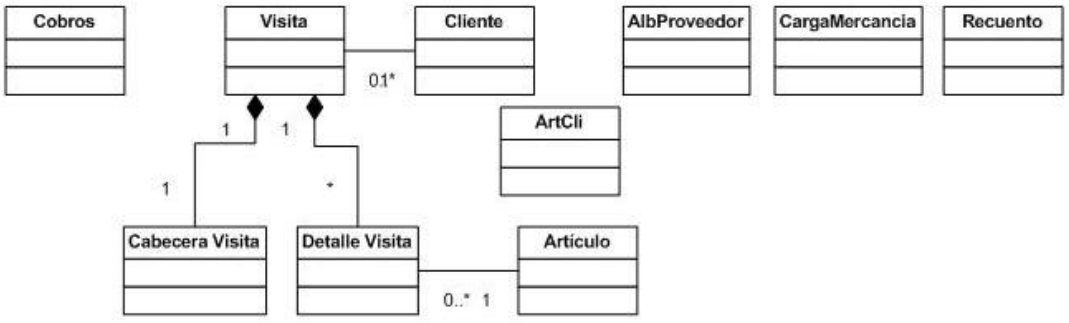
GESTION BUSQUEDAS



GESTION DE REQUISITOS FUNCIONALES ESPECIFICOS



GESTION DE CLASES BASICAS DEL NEGOCIO



ACCESO A DATOS

En Requisitos de Interface Software se especificaba:

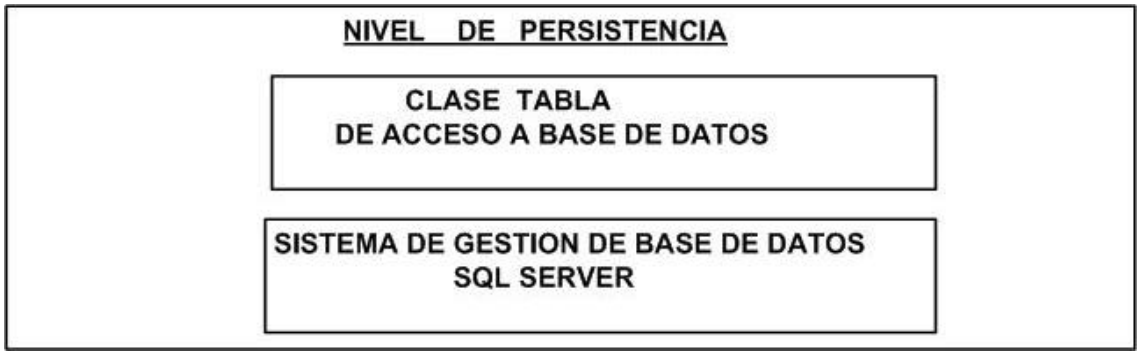
Requisito:
Ha de estar diseñada para que se pueda cambiar fácilmente la programación para elegir cualquier Sistema de Gestión de Base de Datos y cualquier tecnología de Acceso a Datos.

Solución:
He diseñado una CLASE "TABLA" donde está todo el acceso a datos y con tan sólo reescribir el contenido de los métodos genéricos de dicha clase se puede adaptar a otra tecnología de Acceso a Datos.

Dicha clase contiene métodos genéricos como
Leer LeerSiguiente LeerAnterior Coger Dejar Add
Borrar BorrarTodo Actualizar InicializarRegistro, ...

Por ejemplo el método Leer tiene sobrecarga, hay escritos varios métodos Leer que se diferencian en los parámetros que aceptan para dar flexibilidad a las necesidades de lectura.
Desde el resto de programas de la Aplicación, todo el Acceso a Datos está escrito creando un objeto "Tabla" y ejecutando métodos de dicho objeto.
El llamar a esta clase "Tabla" no quiere decir que lo que gestiona es una Tabla de la Base de Datos, sino que lo que gestiona es una Tabla que es el resultado de una Tabla de la B.D. o de una Vista de varias tablas de la B.D.

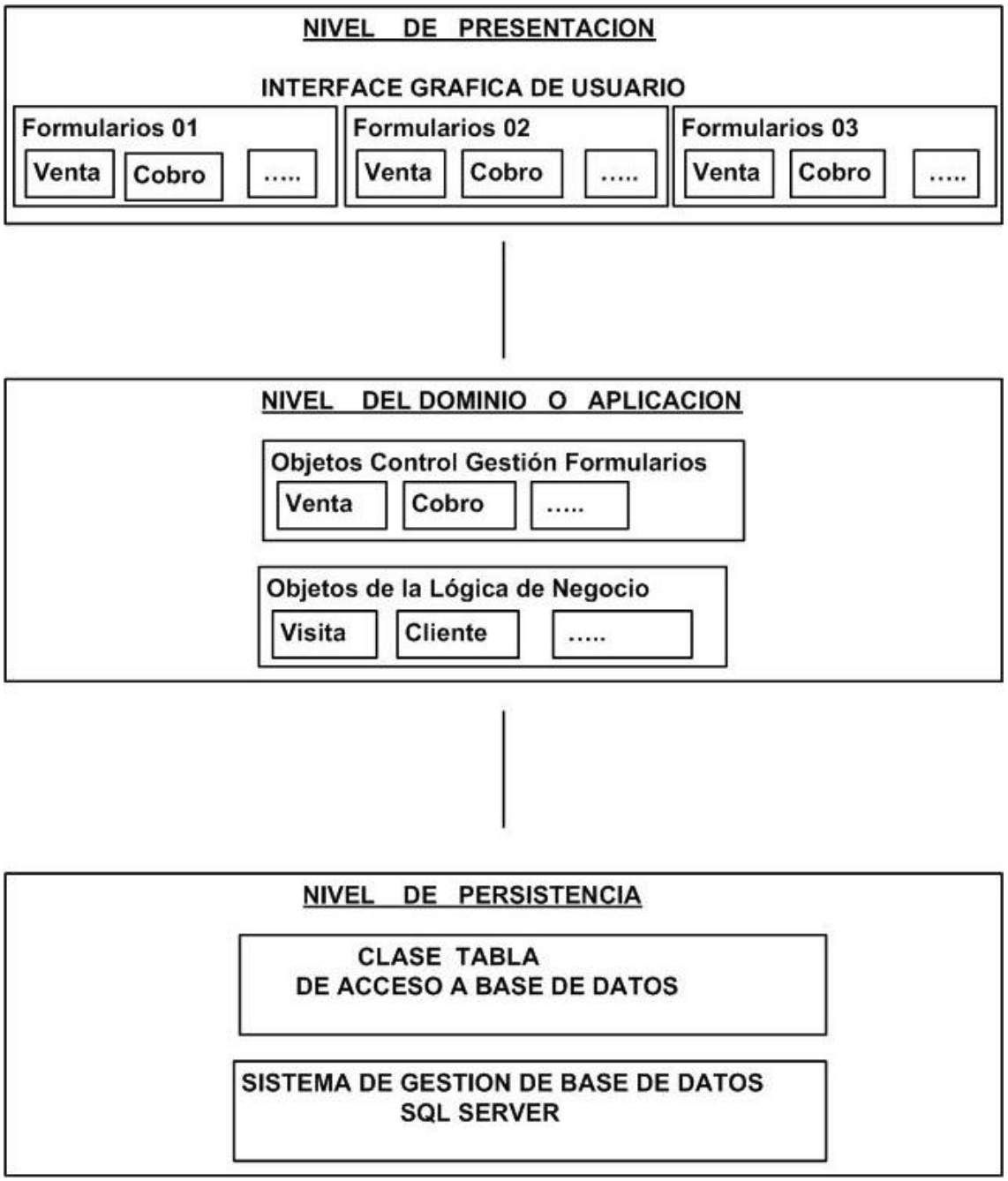
Por tanto la única clase necesaria es la CLASE TABLA.



RESUMEN GENERAL DEL DISEÑO ARQUITECTÓNICO

El Resumen de todo lo expuesto anteriormente sería un diseño totalmente separado en tres niveles, con una separación ya realizadas en el nivel de presentación para dar servicio ya a 4 presentaciones diferentes.

En el nivel de Aplicación se podría hacer también una división para poner por una parte las clases que gestionan cada formulario, por otra las clases que dan servicio a la personalización de todos los formularios y en oytro nivel las clases que gestionan los objetos básicos del negocio: Cliente, Artículo, Visita, AlbaránProveedor, ...



A modo de RESUMEN:

NIVEL DE PRESENTACIÓN:

Aquí está el diseño de los formularios.
Como en las PDAs y Dispositivos industriales móviles hay diferentes tamaños de pantalla, diferentes resoluciones ..., los diseños no tienen código de programación, son sólo diseño de la Interface Gráfica, y por tanto hay diferentes juegos de diseño de formularios: actualmente ya hay 01, 02, 03 y 04.
También se emplea esto para poder prepararle a algún Cliente concreto los formularios con un diseño específico para él, si es que fuera necesario.

NIVEL DE APLICACIÓN:

Aquí está separado en dos niveles: Uno es la programación de la Gestión de cada Formulario y el otro es la Programación de los Objetos Básicos que constituyen la Lógica de Negocio y que se necesitan desde varios formularios.
Por ejemplo toda la programación que hay detrás del formulario de Ventas se realiza desde la Gestión de ese Formulario, pero esta Gestión necesita de ciertos métodos de Objetos que pertenecen a la Lógica de Negocio como son el Objeto Cliente, Artículo, Visita, ...

NIVEL DE PERSISTENCIA:

Comprende el Objeto de Acceso a Datos, donde está programada una tecnología concreta de Acceso a Datos y el Sistema de Gestión de Bases de Datos elegido.
En este proyecto se ha elegido el SQL Server de Microsoft como Sistema de Gestión de Bases de Datos, y la tecnología ADO.Net para acceder a los datos.

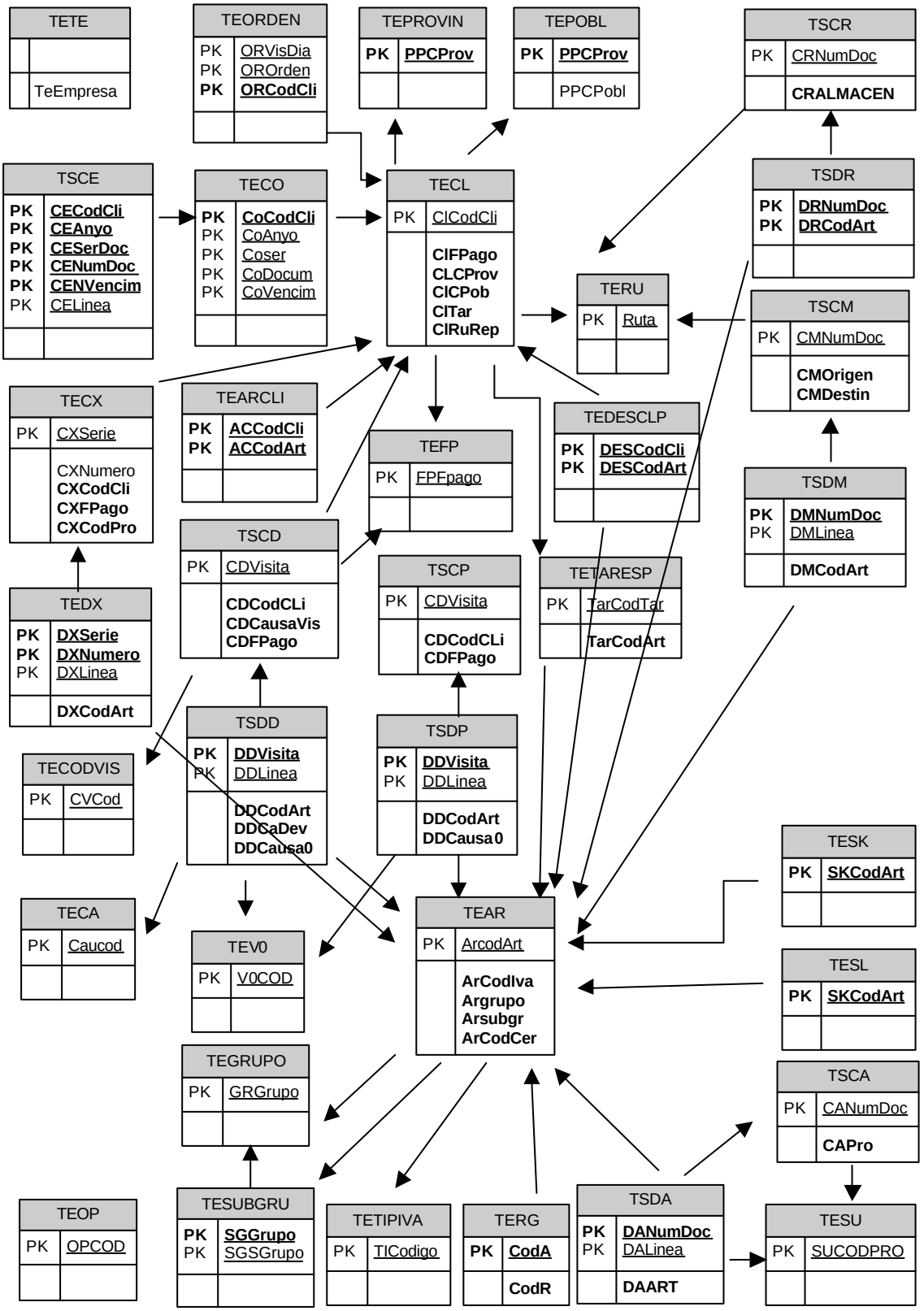
Si se cambiara la tecnología de Acceso a Datos, sólo habría que preparar la versión 2 del Objeto de Acceso a Datos, ya que todo el Acceso a Datos en todos los programas se hace a través de los métodos que están en este Objeto.

Esta separación de Niveles va a hacer posible que la Aplicación AlfaPDA pueda adaptarse fácilmente a muchas Empresas diferentes y que cada Empresa pueda elegir los dispositivos móviles que más se ajusten a sus necesidades.

4.2. MODELO RELACIONAL DE LA BASE DE DATOS.

4.2.1.- MODELO RELACIONAL

Sólo relaciono los atributos que intervienen en la clave principal y en las claves externas para no desbordar el gráfico. Los demás atributos se relacionan a continuación en el punto 4.2.2. Detalle del Esquema.



4.2.2.- ESQUEMAS RELACIONALES

TETE (Registro de Empresa)

Columnas	Tipo de datos
TeEmpresa	NVARCHAR(20)
TeDomic	NVARCHAR(30)
TePob	NVARCHAR(25)
TeCPostal	NUMERIC(5)
TeProv	NVARCHAR(20)
TeCif	NVARCHAR(15)
TeRegMer	NVARCHAR(85)
TeTelef	NVARCHAR(15)
TeTipFac	NVARCHAR(2)
TeNumFac	NUMERIC(9)
TeTipAlb	NVARCHAR(2)
TeNumAlb	NUMERIC(9)
TeDocMov	NUMERIC(9)
TeClave1	NUMERIC(4)
TeClave2	NUMERIC(4)
TeCLave3	NUMERIC(4)
TeClave4	NUMERIC(4)
Teclave5	NUMERIC(4)
TeDocAP	NUMERIC(9)

TECL (CLIENTES)

Columnas	Tipo de datos
ClCodCli	NUMERIC (15)
ClNom	NVARCHAR(40)
ClDirec	NVARCHAR (40)
ClCpostal	NUMERIC(5)
CLCProv	NUMERIC(2)
ClCPob	NUMERIC(3)
ClCif	NVARCHAR (15)
ClEstab	NVARCHAR (40)
ClFPago	NUMERIC(2)
ClTiva	NUMERIC(1)
ClTotVent	NUMERIC(9)
ClLimCre	NUMERIC(9)
ClConObl	NVARCHAR(1)
ArPorCer	NUMERIC(6,4)
ClTar	NUMERIC(3)
ClCad	NUMERIC(4)
ClDtoFr	NUMERIC(5,2)
ClDtoPP	NUMERIC(5,2)
ClPtoVerde	NVARCHAR (1)
ClRuRep	NUMERIC(3)
ClEnvDep	NVARCHAR (1)

ClTelCli	NVARCHAR (15)
ClTelCli2	NVARCHAR (15)
ClPur	NVARCHAR (1)

Clave		**Cardinalidad**
Principal	ClCodCli	
externa	ClFPago → TEFP	varios - a - uno
externa	CLCProv → TEPROVIN	varios - a - uno
externa	CLCProv, ClCPob → TEPOBL	varios - a - uno
externa	ClTar → TETITAR	varios - a - uno
externa	ClRuRep → TERU	varios - a - uno

TEORDEN (Orden de Clientes en el Rutero)

Columnas	**Tipo de datos**
ORCodCli	NUMERIC(15)
ORVisDia	NVARCHAR(1)
OROrden	NUMERIC(3)

Clave		**Cardinalidad**
Principal	ORVisDia, OROrden, ORCodCli	
externa	ORCodCli → TECL	varios - a - uno

TEPROVIN (Tabla de Provincias)

Columnas	**Tipo de datos**
PPCProv	NUMERIC(3)
PPDProv	NVARCHAR (25)

Clave	
Principal	PPCProv

TEPOBL (Tabla de Poblaciones)

Columnas	**Tipo de datos**
PPCPais	NUMERIC(3)
PPCProv	NUMERIC(3)
PPCPobl	NUMERIC(3)
PPDPobl	NVARCHAR (25)

Clave		**Cardinalidad**
Principal	PPCProv, PPCPobl	
externa	PPCProv → TEPROVIN	varios - a - uno

TEFP (Formas de Pago)

Columnas	Tipo de datos
FPFpago	NUMERIC(2)
FPDesc	NVARCHAR(30)
FPTipPag	NVARCHAR(1)

Clave	
Principal	FPFpago

TERU (Rutas Vendedores)

Columnas	Tipo de datos
Ruta	NUMERIC(3)
Nombre	NVARCHAR(30)

Clave	
Principal	Ruta

TECO (Cobros Pendientes)

Columnas	Tipo de datos
CoCodCli	NUMERIC(15)
Coser	NVARCHAR(2)
CoDocum	NUMERIC(8)
CoAnyo	NUMERIC(4)
ColPend	NUMERIC(9,2)
CoVencim	NUMERIC(2)
CoFVencim	NVARCHAR(10)
CoRel	NVARCHAR(9)
COCobro	NUMERIC(9,2)
CoExpedi	NVARCHAR(10)

Clave		Cardinalidad
Principal	CoCodCli, CoAnyo, Coser, CoDocum, CoVencim	
externa	CoCodCli → TECL	varios - a - uno

TSCE (Cobros Efectuados)

Columnas	Tipo de datos
CECodCli	NUMERIC(15)
CESerDoc	NVARCHAR(2)
CENumDoc	NUMERIC(8)
CEAnyo	NUMERIC(4)
CENVencim	NUMERIC(2)

CEVenc	NVARCHAR(10)
CELinea	NVARCHAR(2)
CEICobr	NUMERIC(9,2)
CEFechat	NVARCHAR(10)
CEBco	NUMERIC(4)
CEOfi	NUMERIC(4)
CEDC	NUMERIC(2)
CECta	NUMERIC(10)
CENatu	NVARCHAR(1)
CEFechad	NVARCHAR(10)
CEHora	NVARCHAR(8)
CENumeroT	NVARCHAR(20)

Clave		**Cardinalidad**
Principal	CECodCli, CEAnyo, CESerDoc, CENumDoc, CENVencim, CELinea	
externa	CECodCli → TECL	varios - a - uno
externa	CECodCli, CEAnyo, CESerDoc, CENumDoc, CENVencim → TECO	varios - a - uno

TETIPIVA (Tabla de IVAs)

Columnas	**Tipo de datos**
TICodigo	NUMERIC(2)
TIDescri	NVARCHAR(30)
TIIVA	NUMERIC(4,2)
TIRec	NUMERIC(4,2)

Clave	
Principal	TICodigo

TEAR (ARTICULO)

Columnas	**Tipo de datos**
ArcodArt	NVARCHAR(15)
ArDesc	NVARCHAR(20)
ArUniven	NVARCHAR (2)
ArUniAlm	NVARCHAR (2)
ArFactor	NUMERIC(6,3)
ArPrecio	NUMERIC(9,3)
ArPreMax	NUMERIC(9,3)
Arpremin	NUMERIC(9,3)
ArCodIva	NUMERIC(2)
Argrupo	NVARCHAR (2)
Arsubgr	NVARCHAR (2)
ArCodBar	NVARCHAR (25)
ArEnvase	NVARCHAR (1)
ArCer	NVARCHAR (1)

ArCodCer	NVARCHAR(15)
ArPorCer	NUMERIC(6,4)
ArPur	NVARCHAR (1)
Arcanpur	NUMERIC(7,5)
Arptoverde	NVARCHAR (1)
ArPrePv	NUMERIC(8,5)
ArLote	NVARCHAR (1)
ArPesVar	NVARCHAR (1)
ArPesAprox	NUMERIC(8,3)

Clave		**Cardinalidad**
Principal	ArcodArt	
externa	ArCodIva → TETIPIVA	varios - a - uno
externa	Argrupo → TEGRUPO	varios - a - uno
externa	Argrupo,Arsubgr → TESUBGRU	varios - a - uno
externa	ArCodCer → TEAR	varios - a - uno

TEGRUPO (Maestro de Grupos)

Columnas	**Tipo de datos**
GRGrupo	NUMERIC(2)
GRDesGru	NVARCHAR(10)

Clave		**Cardinalidad**
Principal	GRGrupo	

TESUBGRU (Maestro de SubGrupos)

Columnas	**Tipo de datos**
SGGrupo	NUMERIC(2)
SGSGrupo	NUMERIC(2)
SGDesSGru	NVARCHAR(10)

Clave		**Cardinalidad**
Principal	SGGrupo, SGSGrupo	
externa	SGGrupo → TEGRUPO	varios - a - uno

TEARCLI (Codigos de articulo según la codificacion del cliente)

Columnas	**Tipo de datos**
ACCodCli	NUMERIC(15)
ACCodArt	NVARCHAR(15)
ACCodArCli	NVARCHAR (15)

Clave		**Cardinalidad**
Principal	ACCodCli, ACCodArt	
externa	ACCodCli → TECL	varios - a - uno
externa	ACCodArt → TEAR	varios - a - uno

TECA (Causa de devolucion)

Columnas	Tipo de datos
Caucod	NUMERIC(5)
Caudes	NVARCHAR(20)

Clave	
Principal	Caucod

TEV0 (Causas Venta Precio Cero)

Columnas	Tipo de datos
V0COD	NUMERIC(2)
V0DES	NVARCHAR(50)

Clave	
Principal	V0COD

TECODVIS (Motivos de No Compra)

Columnas	Tipo de datos
CVCod	NUMERIC(5)
CVDescri	NVARCHAR(20)

Clave	
Principal	CVCod

TECX (Cabecera Pedidos Pendientes de Servir)

Columnas	Tipo de datos
CXSerie	NVARCHAR(2)
CXNumero	NUMERIC(6)
CXCodCli	NUMERIC(15)
CXFPago	NUMERIC(2)
CXCodPro	NUMERIC(15)
CXPediP	NVARCHAR(20)
CXFecha	NVARCHAR(10)
CXObserva	NVARCHAR(50)

Clave		Cardinalidad
Principal	CXSerie, CXNumero	
externa	CXCodCli → TECL	varios - a - uno
externa	CXFPago → TEFP	varios - a - uno
externa	CXCodPro → TESU	varios - a - uno

TEDX (Detalle Pedidos Pendientes de Servir)

Columnas	Tipo de datos
DXSerie	NVARCHAR(2)
DXNumero	NUMERIC(6)
DXLinea	NUMERIC(6)
DXCodArt	NVARCHAR(15)
DXPrecio	NUMERIC(9,3)
DXCant	NUMERIC(8,2)
DXProm	NUMERIC(6,3)
DXDesc	NUMERIC(6,3)
DXCoLote	NVARCHAR(15)

Clave		Cardinalidad
Principal	DXSerie, DXNumero, DXLinea	
externa	DXSerie , DXNumero → TECX	varios - a - uno
externa	DXCodArt → TEAR	varios - a - uno

TEDESCLP (Descuentos de Clientes)

Columnas	Tipo de datos
DESCodCli	NUMERIC(15)
DESCodArt	NVARCHAR(15)
DESCliDes	NUMERIC(6,3)

Clave		Cardinalidad
Principal	DESCodCli, DESCodArt	
externa	DESCodCli → TECL	varios - a - uno
externa	DESCodArt → TEAR	varios - a - uno

TEOP (Observaciones Pedidos)

Columnas	Tipo de datos
OPCOD	NUMERIC(2)
OPDES	NVARCHAR(50)

Clave	
Principal	OPCOD

TERG (Promociones Regalos)

Columnas	Tipo de datos
CodA	NVARCHAR(15)
CanC	NUMERIC(9,3)
CodR	NVARCHAR(15)

CanR	NUMERIC(9,3)

Clave		Cardinalidad
Principal	CodA	
externa	CodA → TEAR	varios - a - uno
externa	CodR → TEAR	varios - a - uno

TESK (Stock del vehículo)

Columnas	Tipo de datos
SKCodArt	NVARCHAR(15)
SKCantST	NUMERIC(9,2)
SKCantCar	NUMERIC(9,2)
SKCantVen	NUMERIC(9,2)

Clave		Cardinalidad
Principal	SKCodArt	
externa	SKCodArt → TEAR	varios - a - uno

TESL (Stock del Lote)

Columnas	Tipo de datos
SLCodArt	NVARCHAR(15)
SLCodLote	NVARCHAR(15)
SLCantST	NUMERIC(9,2)
SLFeCad	NVARCHAR(10)

Clave		Cardinalidad
Principal	SKCodArt	
externa	SKCodArt → TEAR	varios - a - uno

TESU (Proveedores)

Columnas	Tipo de datos
SUCODPRO	NUMERIC(15)
SUNOMPRO	NVARCHAR(30)
SUDIRECC	NVARCHAR(30)
SUCProv	NUMERIC(2)
SUCPob	NUMERIC(3)
SUCODPOS	NUMERIC(5)
SUDNICIF	NVARCHAR(15)
SUTELEFO	NVARCHAR(15)
SUOBSERV	NVARCHAR(70)

Clave		Cardinalidad
Principal	SUCODPRO	
externa	SUCProv → TEPROVIN	varios - a - uno

externa	SUCProv, SUCPob → TEPOBL	varios - a - uno

TETARESP (Tarifas)

Columnas	**Tipo de datos**
TarCodTar	NUMERIC(3)
TarCodArt	NVARCHAR(15)
TarPrecio	NUMERIC(9,3)

Clave		**Cardinalidad**
Principal	TarCodTar	
externa	TarCodArt → TEAR	varios - a - uno

TSCA (Cabecera Albaranes Proveedor)

Columnas	**Tipo de datos**
CANumDoc	NUMERIC(6)
CAPro	NVARCHAR(15)
CASerie	NVARCHAR(2)
CARef	NVARCHAR(15)
CAFecha	NVARCHAR(10)
CAHora	NVARCHAR(8)
CAImporte	NUMERIC(9,2)

Clave		**Cardinalidad**
Principal	CANumDoc	
externa	CAPro → TESU	varios - a - uno

TSDA (Detalle Albaranes a Proveedor)

Columnas	**Tipo de datos**
DANumDoc	NUMERIC(6)
DALinea	NUMERIC(4)
DAART	NVARCHAR(15)
DACant	NUMERIC(8,3)
DAPrecio	NUMERIC(9,3)
DALote	NVARCHAR(15)

Clave		**Cardinalidad**
Principal	DANumDoc, DALinea	
externa	DANumDoc → TECA	varios - a - uno
externa	DAART → TEAR	varios - a - uno

TSCD (Cabecera de Ventas)

Columnas	**Tipo de datos**
CDVisita	NUMERIC(5)
CDLetFra	NVARCHAR(2)

CDNumFra	NUMERIC(6)
CDCodCLi	NUMERIC(15)
CDPorDtoSF	NUMERIC(4,2)
CDPorDtoPP	NUMERIC(4,2)
CDACuen	NUMERIC(9,2)
CDImpBI	NUMERIC(9,2)
CDImpIVA	NUMERIC(9,2)
CDImpRE	NUMERIC(9,2)
CDFPago	NUMERIC(2)
CDFecha	NVARCHAR(10)
CDHora	NVARCHAR(8)
CDCausaVis	NUMERIC(5)

Clave		**Cardinalidad**
Principal	CDVisita	
externa	CDCodCLi → TECL	varios - a - uno
externa	CDFPago → TEFP	varios - a - uno
externa	CDCausaVis → TECODVIS	varios - a - uno

TSDD (Detalle de Ventas)

Columnas	**Tipo de datos**
DDVisita	NUMERIC(5)
DDLinea	NUMERIC(4)
DDCodArt	NVARCHAR(15)
DDPrecio	NUMERIC(9,3)
DDCant	NUMERIC(9,2)
DDPromo	NUMERIC(6,3)
DDDesc	NUMERIC(6,3)
DDCaDev	NUMERIC(2)
DDCausa0	NUMERIC(2)
DDPorIVA	NUMERIC(4,2)
DDPorRE	NUMERIC(4,2)
DDImpPUR	NUMERIC(9,3)
DDImpPV	NUMERIC(9,3)
DDLote	NVARCHAR(15)

Clave		**Cardinalidad**
Principal	DDVisita, DDLinea	
externa	DDVisita → TSCD	varios - a - uno
externa	CDFPago → TEFP	varios - a - uno
externa	DDCodArt → TEARCL	varios - a - uno
externa	DDCaDev → TECA	varios - a - uno
externa	DDCausa0→ TEV0	varios - a - uno

TSCP (Cabecera de Pedidos)

Columnas	Tipo de datos
CDVisita	NUMERIC(5)
CDCodCLi	NUMERIC(15)
CDImpBI	NUMERIC(9,2)
CDFPago	NUMERIC(2)
CDFecha	NVARCHAR(10)
CDHora	NVARCHAR(8)
CDFecPre	NVARCHAR(10)
CDObserva	NVARCHAR(50)

Clave		Cardinalidad
Principal	CDVisita	
externa	CDCodCLi → TECL	varios - a - uno

TSDP (Detalle de Pedidos)

Columnas	Tipo de datos
DDVisita	NUMERIC(5)
DDLinea	NUMERIC(4)
DDCodArt	NVARCHAR(15)
DDPrecio	NUMERIC(9,3)
DDCant	NUMERIC(9,2)
DDPromo	NUMERIC(6,3)
DDDesc	NUMERIC(6,3)
DDCausa0	NUMERIC(2)
DDPorIVA	NUMERIC(4,2)
DDPorRE	NUMERIC(4,2)

Clave		Cardinalidad
Principal	DDVisita, DDLinea	
externa	DDVisita → TSCP	varios - a - uno
externa	DDCodArt → TEARCL	varios - a - uno
externa	DDCausa0→ TEV0	varios - a - uno

TSCM (Cabecera de Carga de Mercancía)

Columnas	Tipo de datos
CMNumDoc	NUMERIC(6)
CMOrigen	NUMERIC(3)
CMDestin	NUMERIC(32)
CMFecha	NVARCHAR(10)
CMHora	NVARCHAR(8)

Clave		Cardinalidad
Principal	CMNumDoc	
externa	CMOrigen → TERU	varios - a - uno

externa	CMDestin → TERU	varios - a - uno

TSDM (Detalle de Carga de Mercancía)

Columnas	**Tipo de datos**
DMNumDoc	NUMERIC(6)
DMLinea	NUMERIC(4)
DMCodArt	NVARCHAR(15)
DMCant	NUMERIC(9,2)
DMLote	NVARCHAR(15)

Clave		**Cardinalidad**
Principal	DMNumDoc , DMLinea	
externa	DMNumDoc → TSCM	varios - a - uno
externa	DMCodArt → TEARCL	varios - a - uno

TSCR (Cabecera de Recuento)

Columnas	**Tipo de datos**
CRNumDoc	NUMERIC(6)
CRALMACEN	NUMERIC(3)
CRFecha	NVARCHAR(10)
CRHora	NVARCHAR(8)

Clave		**Cardinalidad**
Principal	CRNumDoc	
externa	CRALMACEN → TERU	varios - a - uno

TSDR (Detalle de Recuento)

Columnas	**Tipo de datos**
DRNumDoc	NUMERIC(6)
DRCodArt	NVARCHAR(15)
DRCant	NUMERIC(9,2)

Clave		**Cardinalidad**
Principal	DRNumDoc , DRCodArt	
externa	CRNumDoc → TSCR	varios - a - uno
externa	DRCodArt → TEARCL	varios - a - uno

5.- IMPLEMENTACIÓN DEL PROTOTIPO/SISTEMA

Se ha utilizado el lenguaje Visual Basic.Net empleando el entorno de desarrollo Microsoft Visual Studio 2005.
El Gestor de Base de Datos es el Microsoft SQL Server 2005 Mobile Edition, y la tecnología de acceso a datos es ADO.NET.

Aspectos a Destacar son:
Chequeo de Entorno de: Fecha, Alfabeto elegido, punto de millar y coma decimal. Chequeo de memoria libre disponible. Control de Licencia. Identificación de la versión y Fecha de la Aplicación instalada en cada cliente. Grabación de un histórico de mensajes de advertencia que le puedan haber salido al usuario.
Configuración de parámetros (Bloque Personalización): Selección del diseño de los Impresos, Selección del diseño de los Formularios, modificación por el usuario del ancho de las columnas de los ListView, modificación por el usuario de los criterios por defecto en las búsquedas. Acceso a ejecución de Programas (los programas no permitidos no aparecen en pantalla). Visualización en pantalla sólo de la informacion/botones/menús que el vendedor necesita para esa operación, ya que es importante no presentar lo que no necesite.

Al ser una Aplicación que va a comercializar una empresa de Desarrollo de Software, no puede figurar el código fuente de los programas en esta Documentación, pero sí voy a reflejar pequeños aspectos parciales del código de algunas clases, y a veces sólo la definición de los métodos, para que se vea cómo está implementado, sin nuevas explicaciones acerca de la finalidad de las clases.

```
'------------------------
Public Class InicioPrevio
'------------------------
Public Shared Sub Preliminares()
 Dim resultado As String
' Comprobamos que Hay sufuciente memoria para ejecutar la Aplicación
' Comprobamos que la Aplicación tiene Licencia
' Leemos el fichero de configuración .INI
' Abrimos la conexión a la Base de Datos
' Leemos Tablas TETE y TEDI
' COMPROBAR QUE LA FECHA DE LA PDA ESTA EN ESPAÑOL
  ComprobarFecha()
' COMPROBAR QUE ESTA LA COMA DECIMAL Y EL PUNTO DE MILLAR
  ComprobarComaDecimal()
' Comprobar licencia de uso o pedir clave si es primera vez
    While Licencia2.ExisteClave = "0"
            resultado = Licencia2.PedirClave()
            If resultado = "CERRAR_APLICACION" Then
                CerrarAplicacion()
                Exit Sub
            End If
    End While
' LEER pdaINI, ABRIR BASE DE DATOS, LEER TABLA: TETE, TEDI,
  resultado = PDAini.LeerPDAini_AbrirBD_LeerTETE_TEDI_TEV0_TETIPIVA()
```

```
' puede ser: "BASE_DATOS_VACIA"   "CERRAR_APLICACION"  OR  ""
 If resultado = "CERRAR_APLICACION" Then
        CerrarAplicacion()
 End If
' COMPROBAR QUE HAY MEMORIA DISPONIBLE SUFICIENTE
   ComprobarMemoria()
' Girar la Pantalla por si no está en la psición apaisada adecuada
   If PDAini.PForm_FrmMenuPrincipal = "03" Then
            Pegasus.InicioEnPegasus()
    End If
End Sub


'------------------------
Public Class Tabla
'------------------------
  Private Shared cBD As Data.SqlServerCe.SqlCeConnection  ' conectar
con la base de datos

Public Shared Function AbrirBD() As Boolean
Dim fichero As String
fichero = "Data Source=" & PDAini.PBus_Path_Datos & "Autoventa.sdf"
AbrirBD = False
Try
   cBD = New Data.SqlServerCe.SqlCeConnection(fichero)
   cBD.Open()
   AbrirBD = True
Catch ex As Exception
   MensajeError = "ABRIR BASE DATOS:  " & ex.Message
   MessageBox.Show(MensajeError)
   SalidaPorCatchBD(MensajeError)
 End Try
 End Function

' definiciones para las instancias de la clase Tabla
' ================================================
    Private NomTabla As String  ' Contiene el nombre de la Tabla
    Private sql As String       ' para definir la sql a ejecutar
    Private da As Data.SqlServerCe.SqlCeDataAdapter    ' para asignar
una vista (instrucción) SQL sobre una conexion (Base de Datos)
    Private cb As Data.SqlServerCe.SqlCeCommandBuilder ' para
inserter, actualizar y borrar
    Private ds As Data.DataSet  'es el dataset (en memoria) que
corresponde al dataadapter obtenido
    Private dsnombre As String = "dsnombre" ' el nombre que le voy a
asignar al dataset va a ser fijo dsnombre
    Private dro As Data.DataRow    ' es una fila del dataset. Lo
utilizamos sólo para añadir registro al dataset
    Private nr As Integer                                ' numero
registro en curso del Dataset (primer registro es el cero)
    Private dr As Data.SqlServerCe.SqlCeDataReader     ' para lectura
secuencial rapida
    Private consulta As Data.SqlServerCe.SqlCeCommand  ' para utilizar
en un datareader o DataSet.

Public Function Leer(ByVal ParamArray parametro() As Object) As
Integer
        ' Se le puede llamar con  0, 2, 3, 4, 5, 6, 7 ó 9 parámetros
```

```
  Public Function Leersql(ByVal Wsql As String, ByVal ParamArray
parametro() As String) As Integer
        ' Leer (en dataset), registros de la tabla NomTabla,
ejecutando la sql del parámetro,
        ' y devuelve el número de registros recuperados


Public Function Coger(ByVal Wcampo As String) As Object
        ' Leer campo Wcampo del registro en curso de (dataset) tabla
NomTabla y devuelve un String o un Numerico


  Public Function Dejar(ByVal Wcampo As String, ByVal Wvalor As
Object) As Boolean
        ' Dejar campo Wcampo en registro en curso de (dataset) tabla
NomTabla


  Public Function Add() As Boolean
        ' Añadir registro en(dataset) tabla NomTabla, el nr se
actualiza con el valor que le corresponde a este registro nuevo


  Public Function Borrar() As Boolean
        ' Borrar registro en curso, del (dataset) y de tabla NomTabla.


Public Function Borrar(ByVal Wcampo As String, ByVal Wvalor As Object)
As Boolean
        ' Borrar registros que en el campo Wcampo  tengan el valor
Wvalor, del (dataset)y de tabla NomTabla


  Public Function BorrarTodo() As Boolean
        ' Borrar la Tabla completa de tabla NomTabla


  Public Function LeerS() As Boolean
        ' Leer el registro siguiente (del dataset) de la tabla
NomTabla



'------------------------
Public Class Formu
'------------------------
 Public Shared Sub ejecutar(ByVal NomPrograma As Short,
                            ByVal ParamArray parametro() As Object)
     Select Case NomPrograma
            Case Programa.FrmVenPed
                        Frm0VenPed = New Frm00VenPed
                        FrmVenPed = New Frm01VenPed
                        Frm0VenPed.P_Origen = parametro(0)
                        Frm0VenPed.P_WCodCli = parametro(1)
                        FrmVenPed.ShowDialog()
                        Frm0VenPed = Nothing
                        FrmVenPed = Nothing

            Case Programa.FrmVenPedMasDatos
                        Frm0VenPedMasDatos = New Frm00VenPedMasDatos
                        FrmVenPedMasDatos = New Frm01VenPedMasDatos
```

```
                        FrmVenPedMasDatos.ShowDialog()
                        Frm0VenPedMasDatos = Nothing
                        FrmVenPedMasDatos = Nothing

            Case Programa.FrmBuscar
                        Frm0Buscar = New Frm00Buscar
                        FrmBuscar = New Frm01Buscar()
                        Frm0Buscar.Nueva(parametro(0), parametro(1))
                        FrmBuscar.ShowDialog()
                        Frm0Buscar = Nothing
                        FrmBuscar = Nothing

          Case Programa.FrmCargaMercancia
                        Frm0CargaMercancia = New Frm00CargaMercancia
                        FrmCargaMercancia = New Frm01CargaMercancia
                        FrmCargaMercancia.ShowDialog()
                        Frm0CargaMercancia = Nothing
                        FrmCargaMercancia = Nothing



'------------------------------
Public Class Frm01AlbProveedor
'------------------------------
Public WTeclado As TecladoVirtual

Private Sub Frm01AlbProveedor_KeyDown(ByVal sender As Object, ByVal e
As System.Windows.Forms.KeyEventArgs) Handles MyBase.KeyDown
      Frm0AlbProveedor.FrmAlbProveedor_KeyDown(sender, e)
End Sub

 Private Sub FrmAlbProveedor_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
     Frm0AlbProveedor.FrmAlbProveedor_Load()
 End Sub

Private Sub TxtCantidad_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtCantidad.GotFocus
      Frm0AlbProveedor.TxtCantidad_GotFocus(sender)
End Sub

    Private Sub TxtObProv_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs)
        Frm0AlbProveedor.TxtObProv_LostFocus(sender)
    End Sub

    Private Sub TxtObProv_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs)
        Frm0AlbProveedor.TxtObProv_TextChanged(sender)
    End Sub

    Private Sub TxtCantidad_KeyPress(ByVal sender As Object, ByVal e
As System.Windows.Forms.KeyPressEventArgs) Handles
TxtCantidad.KeyPress
        Frm0AlbProveedor.TxtCantidad_KeyPress(sender, e)
    End Sub

    Private Sub TxtCantidad_LostFocus(ByVal sender As Object, ByVal e
As System.EventArgs) Handles TxtCantidad.LostFocus
        Frm0AlbProveedor.TxtCantidad_LostFocus(sender)
    End Sub
```

```
    Private Sub ListVAlbProveedor_KeyPress(ByVal sender As Object,
ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles
ListVAlbProveedor.KeyPress
        Frm0AlbProveedor.ListVAlbProveedor_KeyPress(sender, e)
    End Sub

    Private Sub ListVAlbProveedor_ItemActivate(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ListVAlbProveedor.ItemActivate
        Frm0AlbProveedor.ListVAlbProveedor_ItemActivate()
    End Sub

    Private Sub CmdUM_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles CmdUM.KeyPress
        Frm0AlbProveedor.CmdUM_KeyPress(sender, e)
    End Sub

    Private Sub PicBusProv_MouseUp(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.MouseEventArgs) Handles
PicBusProv.MouseUp
        Frm0AlbProveedor.PicBusProv_MouseUp()
    End Sub

    Private Sub PicBusProv_MouseDown(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.MouseEventArgs) Handles
PicBusProv.MouseDown
        Frm0AlbProveedor.PicBusProv_MouseDown()
    End Sub

    Private Sub PicBusProv_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles PicBusProv.Click
        Frm0AlbProveedor.PicBusProv_Click()
    End Sub

    Private Sub BtnBusProv_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles BtnBusProv.Click
        Frm0AlbProveedor.BtnBusProv_Click()
    End Sub

    Private Sub PicBusArt_MouseUp(ByVal sender As System.Object, ByVal
e As System.Windows.Forms.MouseEventArgs) Handles PicBusArt.MouseUp
        Frm0AlbProveedor.PicBusArt_MouseUp()
    End Sub

    Private Sub PicBusArt_MouseDown(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.MouseEventArgs) Handles
PicBusArt.MouseDown
        Frm0AlbProveedor.PicBusArt_MouseDown()
    End Sub

    Private Sub PicBusArt_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles PicBusArt.Click
        Frm0AlbProveedor.PicBusArt_Click()
    End Sub

    Private Sub BtnBusArt_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnBusArt.Click
        Frm0AlbProveedor.BtnBusArt_Click()
    End Sub
```

```
Private Sub TxtArticulo_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtArticulo.KeyPress
        Frm0AlbProveedor.TxtArticulo_KeyPress(sender, e)
End Sub

    Private Sub TxtArticulo_LostFocus(ByVal sender As Object, ByVal e
As System.EventArgs) Handles TxtArticulo.LostFocus
        Frm0AlbProveedor.TxtArticulo_LostFocus(sender)
    End Sub

    Private Sub TxtArticulo_GotFocus(ByVal sender As Object, ByVal e
As System.EventArgs) Handles TxtArticulo.GotFocus
        Frm0AlbProveedor.TxtArticulo_GotFocus(sender)
    End Sub

    Private Sub TxtRef_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtRef.GotFocus
        Frm0AlbProveedor.TxtRef_GotFocus(sender)
    End Sub

    Private Sub TxtRef_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtRef.KeyPress
        Frm0AlbProveedor.TxtRef_KeyPress(sender, e)
    End Sub

    Private Sub TxtProveedor_GotFocus(ByVal sender As Object, ByVal e
As System.EventArgs) Handles TxtProveedor.GotFocus
        Frm0AlbProveedor.TxtProveedor_GotFocus(sender)
    End Sub

    Private Sub TxtProveedor_KeyPress(ByVal sender As Object, ByVal e
As System.Windows.Forms.KeyPressEventArgs) Handles
TxtProveedor.KeyPress
        Frm0AlbProveedor.TxtProveedor_KeyPress(sender, e)
    End Sub

    Private Sub TxtProveedor_LostFocus(ByVal sender As Object, ByVal e
As System.EventArgs) Handles TxtProveedor.LostFocus
        Frm0AlbProveedor.TxtProveedor_LostFocus(sender)
    End Sub

    Private Sub TxtRef_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtRef.LostFocus
        Frm0AlbProveedor.TxtRef_LostFocus(sender)
    End Sub

    Private Sub TxtSerie_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtSerie.GotFocus
        Frm0AlbProveedor.TxtSerie_GotFocus(sender)
    End Sub

    Private Sub TxtSerie_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtSerie.KeyPress
        Frm0AlbProveedor.TxtSerie_KeyPress(sender, e)
    End Sub

    Private Sub TxtSerie_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtSerie.LostFocus
        Frm0AlbProveedor.TxtSerie_LostFocus(sender)
    End Sub
```

```
    Private Sub TxtPrecio_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtPrecio.LostFocus
        Frm0AlbProveedor.TxtPrecio_LostFocus(sender)
    End Sub

    Private Sub TxtPrecio_GotFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TxtPrecio.GotFocus
        Frm0AlbProveedor.TxtPrecio_GotFocus(sender)
    End Sub

    Private Sub TxtPrecio_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtPrecio.KeyPress
        Frm0AlbProveedor.TxtPrecio_KeyPress(sender, e)
    End Sub
    Private Sub TxtLote_GotFocus(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TxtLote.GotFocus
        Frm0AlbProveedor.TxtLote_Gotfocus(sender)
    End Sub



'------------------------------
Public Class Frm00AlbProveedor
'------------------------------
 Dim WIndice As Integer = -1 'controla el indice de la línea con la
que se esta trabajando
 Dim WNuevoAlbaran As AlbProveedor

Public Sub FrmAlbProveedor_KeyDown(ByVal sender As Object, ByVal e As
KeyEventArgs)
        If PDAini.PFun_Utilizar_Teclas_Funcion = "S" Then
            TeclasFuncion.TeclasFuncion_FrmAlbProveedor(sender, e)
        End If
End Sub


Public Sub FrmAlbProveedor_Load()
Cursor.Current = Cursors.WaitCursor
FrmAlbProveedor.TxtArticulo.MaxLength = PDAini.PFrm1_Long_Cod_Art
Inicio()
FrmAlbProveedor.WTeclado = New TecladoVirtual(FrmAlbProveedor)
Cursor.Current = Cursors.Default
End Sub


Public Sub TxtCantidad_GotFocus(ByRef sender As Object)
   Frm.ST(sender)
End Sub

Public Sub TxtObProv_LostFocus(ByVal sender As Object)
   Frm.STS(sender)
End Sub



Public Sub TxtCantidad_KeyPress(ByRef sender As Object, ByRef e As
System.Windows.Forms.KeyPressEventArgs)

If e.KeyChar = ChrW(Keys.Return) Then 'si pulsa enter
 'Validamos que sea un múmero
  If Not Frm.ValidarNum(FrmAlbProveedor.TxtCantidad.Text, 0) Then
  MsgBox("La cantidad no es válida", MsgBoxStyle.Critical, "ATENCION")
  e.Handled = True
```

```
  Exit Sub
  End If
DuranteLinea(sender)
e.Handled = True
Else
 'Se cambia el signo de la cantidad si se pulsa la tecla +/-
       If Not Frm.CambioSigno(sender, Asc(e.KeyChar)) Then
          e.Handled = True
          Exit Sub
       End If
       'Si la pulsación no es válida la cancelamos
       If Frm.ControlarDecimales(Asc(e.KeyChar), 3, sender) = 0 Then
                e.Handled = True
                Exit Sub
       End If
End If
End Sub

 Public Sub TxtCantidad_LostFocus(ByVal sender As Object)
        'Cambiamos los colores
        Frm.STS(sender)
 End Sub
```

```
'------------------------------
Public Class Frm00VenPed
'------------------------------
' ESTE FORMULARIO TIENE LA PROGRAMACIÓN AGRUPADA EN los BLOQUES:
 ' BLOQUE DE  PROPIEDADES
 ' BLOQUE DE  EJECUCION DE LOS DISTINTOS PROGRAMAS DEL  MENU
 ' BLOQUE DE  EJECUCION DE LAS DISTINTAS BUSQUEDAS
 ' BLOQUE DE  EJECUCION DE LOS DISTINTOS PROGRAMAS DE  TRASPASO
 ' BLOQUE DE  EJECUCION DE LOS CAMPOS CABECERA (CLIENTE Y BAJAR LINEA)
 ' BLOQUE DE  EJECUCION DE LOS CAMPOS EN LA LINEA DETALLE
 ' BLOQUE DE  EJECUCION DE LOS BOTONES
 ' BLOQUE DE  FUNCIONES Y PROCEDIMIENTOS DEL FORMULARIO VISITAS

Private WNuevaVisita As Visita   objeto  una nueva venta
Private OArtPrefer As ArtPrefer  objeto Gestion Artículos Preferidos
Private Clien As Cliente
Private ArtiClien As ArtCli
Private RegDet As EstructuraDetalleVisita  ' registro detalle visita
Private WIndice As Integer = -1 'Controla el indice del list, si es -1
nos preparamos para crear una línea nueva
 Private InputPanel1 As Microsoft.WindowsCE.Forms.InputPanel = New
Microsoft.WindowsCE.Forms.InputPanel 'Teclado emergente de windows

' CAMPOS A NIVEL DE CABECERA DE DOCUMENTO
' CAMPOS A NIVEL DE CADA LINEA DE DETALLE

Private Sub FinLinea()
 'Finaliza la línea, controlando que se hayan metido bien los datos
 'y prepara la pantalla para una nueva línea
        With FrmVenPed
```

```
            Dim WLinea As Short
            TxtArticuloEsCodigo = " "  ' reseteamos este indicador
para la próxima introducción de Artículo

            If .TxtArticulo.Text <> "" Then
                ' CHEQUEAR CAMPOS DETALLE
                ' ----------------------
                ChequearCamposDetalle()
                'ALMACENAMOS EN MEMORIA Y CALCULAMOS EL TOTAL VISITA
                ' ------------------------------------------------
                WLinea = Almacenar_LineaDetalleVisita_enMemoria()
                ' SUBIMOS LA LINEA,  limpiamos la pantalla y la
dejamos preparada para introducir otra línea
                ' SubirLineaList( ListIndex, List, CodArt, CodArt,
NomArt, Cant, Piezas, Precio, UniVen,  Descuento, Promocion, Cod.Lote,
NLinDet)
                Frm.SubirLineaList(WIndice, .ListVVenPed,
.TxtArticulo, .TxtArticulo, .LblDesArt, .TxtCantidad, .TxtPiezas,
.TxtPrecio, .CmbUV, .TxtDto, .TxtPromocion, .TxtLote, .TxTDDLinea)
         ' Personalizar tamaño letra, alternancia de filas, negrita...
                AdaptarListView.PersonalizarListView(.ListVVenPed)
                Frm.LimpiarPantalla(.TxtArticulo, .LblDesArt,
.TxtCantidad, .CmbUV, .TxtPiezas, .TxtPrecio, .TxtDto, .TxtPromocion,
.TxtLote)
                Frm.OcultarControles(.LbPiezas, .TxtPiezas, .LblDto,
.TxtDto, .LblTipDto, .LblPromocion, .TxtPromocion, .LblTipProm,
.LblLote, .TxtLote, .PicBusLote, .BtnBusLote)

                Frm.DeshabilitarControles(.TxtCantidad, .CmbUV,
.TxtPiezas, .TxtPrecio, .TxtDto, .TxtPromocion, .TxtLote,
.BtnFinLinea, .PicBusLote, .BtnBusLote)
                Frm.HabilitarControles(.TxtArticulo, .PicBusArt,
.BtnBusArt)

                .TxtArticulo.Focus()
                WIndice = -1 'Valor -1 quiere decir que no se esta
modificanco ninguna línea, por lo tanto está listo para una nueva
línea de detalle
            End If
        End With
End Sub

Public Function Almacenar_LineaDetalleVisita_enMemoria() As Short
' GRABA LA LINEA DE DETALLE EN EL ARRAY DE MEMORIA,
' CALCULA Y DEVUELVE EL IMPORTE TOTAL DEL DOCUMENTO.
' Guardamos la linea detalle en memoria de la clase Visita
' y  nos devuelve el número de linea detalle asignado
WLinea = WNuevaVisita.Grabar_DetalleVisita_enMemoria(WLinea, RegDet)
 'Calculamos el total de visita y visualizarlo
 CalcularTotalCabecera()
' Devolvemos el número de línea grabada para que se grabe en el indice
9 del listvvenped por si luego queremos modificar esta línea bajándola
 Almacenar_LineaDetalleVisita_enMemoria = WLinea
' Marcamos la Visita como Modificada
HayModificacion_deVisita = True
End Function

'------------------------------
Public Class Frm
'------------------------------
' Procedimiento cambio color cuando el campo esta activo para escribir
```

```
 Public Shared Sub ST(ByRef ControlFrm As TextBox)

' Procedimiento cambio color cuando el campo deja de estar activo
Public Shared Sub STS(ByRef ControlFrm As TextBox

'Validar que sea un número el dato de la última pulsación
Public Shared Function ValidarNum(ByVal Texto As String, ByVal iAScii
As Integer) As Boolean

'devuelve el número de decimales que tiene <<cCant>>
Public Shared Function NumeroDeDecimales(ByVal cCant As String) As
Integer

' elimina los ceros a la derecha en los numeros decimales
 Public Shared Function EliminarCerosDerechaNumDecimal(ByVal Num As
String) As Decimal
Public Shared Function ControlarDecimales(ByVal iAscii As Integer,
ByVal iNumDecimales As Integer, ByVal Objeto As TextBox) As Integer

'   Se cambia el signo de la cantidad Public Shared Function
CambioSigno(ByRef ControlFrm As Control, ByVal iAscii As Integer) As
Boolean

'   AÑADE LINEA A UN LISTWIEV
Public Shared Sub SubirLineaList(ByVal index As Integer, ByRef ListV
As ListView, ByRef NuevoFoco As Control, ByVal ParamArray parametro()
As Control)
'Este procedimiento añade una líanea a un Listview en función de la
parametrización
'Definición de los Parametros
 'Index  --> El indice en el que se encuentra el ListView
 'ListV   --> Objeto ListView de la pantalla
 'NuevoFoco   --> Objeto el cual recibe el foco después
 'parametro() --> Resto de campos que aparecen en la línea del
ListView

'  LEE UNA LINEA DE UN LISTWIEV
Public Shared Sub BajarLineaList(ByVal index As Integer, ByVal ListV
As ListView, ByRef NuevoFoco As Control, ByVal ParamArray parametro()
As Control)
 'Este procedimiento lee una línea de un Listview y la pone en
pantalla en función de la parametrización
 'Definición de los Parametros
 'Index       --> El indice en el que se encuentra el ListView
 'ListV       --> Objeto ListView de la pantalla
 'NuevoFoco   --> Objeto el cual recibe el foco después
 'parametro() --> Campos que aparecen en la línea del ListView y que
se corresponden con los de la pantalla


  Public Shared Function BorrarLineaList(ByVal index As Integer, ByRef
ListV As ListView) As ListView


   Public Shared Sub LimpiarPantalla(ByVal ParamArray parametro() As
Control)



'------------------------------
Public Class AdaptarListView
'------------------------------
Public Shared Sub PersonalizarListView(ByRef ListV As ListView)
```

```
'   tamaño de letra, negrita, subrayado y alternancia de registros

Public Shared Sub LeerParametrosAnchoColumnas()

Public Shared Sub DibujarAnchoColumnas(ByRef ListV As ListView, ByVal
lista As ListaListView, ByVal NumColumnas As Short)

Public Shared Sub GrabarParametrosAnchoColumnas(ByRef ListV As
ListView, ByVal lista As ListaListView, ByVal NumColumnas As Short)


'------------------------------
Public Class TecladoVirtual
'------------------------------
Private EjeX As Integer
Private EjeY As Integer

Public Sub New(ByRef Frm As Form)

Private Sub Tecl_Num_Posicion(ByRef Frm As Form)
' proporciona las coordenadas adecuadas del teclado numerico,
' para que éste se visualize en el lugar adecuado del formulario

'    VISIBILIDAD INICIAL AL HACER EL LOAD DEL FORMULARIO
Private Sub Visibilidad_Inicial(ByRef frm As Form)

' VISIBILIDAD AL HACER CLICK EN 123 O POR ACTIVACION AUTOMATICA AL
COGER FOCO UN CAMPO
Public Sub Btn_Tecl_Num_Click(ByRef frm As Form, Optional ByVal SetOn
As String = " ")
'Esta rutina a parte de visualizar y ocultar el teclado cada vez que
se pulse el boton



'------------------------------
Public Class TeclasFuncion
'------------------------------

Public Shared Sub Asignar_Valores_Teclas_Funcion()
' Asigna los valores ASCII que le corresponden a las teclas de función
definidas por el usuario
' Cada tecla F1 F2 ... el usuario ha podido cambiarla y que sean: la
F1 --> F5, la F2 --> F1, ......


Public Shared Sub TeclasFuncion_FrmAlmacen(ByVal sender As Object,
ByVal e As KeyEventArgs)
 Select Case e.KeyValue
            Case PDAini.PFun_F1
                'Aqui hay que ejecutar las acciones deseadas
            Case PDAini.PFun_F2

            Case PDAini.PFun_F3



'------------------------------
Public Class CargaPDA
'------------------------------
' LISTA DE METODOS PUBLICOS EXISTENTES:
' Sub CargaDatos_alPDA()
```

```
' LISTA DE METODOS PRIVADOS EXISTENTES:
    ' Sub CargarFicheroTerDat()
    '        Carga el Fichero TerDat.txt
    ' Sub ObtenerDefinicionTabla()
    '        obtiene toda la definicion de campos de esa Tabla desde
los valores que hemos construido leyendo el TerDat.txt
    ' Sub ObtenerDefinicionCampos(ByVal WVar As String, ByVal WCampo
As Byte)
    '        Trocea la variable WVar (cada linea leida del fichero)
tomando como separacion la #
    ' Sub BorrarTabla_y_Preparar(ByVal WQueFic As String)
    '        Borramos la tabla completa y leemos para tener preparado
un dataset para añadir registros
    ' Sub CreaRegistro()
    '        Crea registro en la tabla (dataset) correspondiente,
según la linea leida en el fichero de entrada Carico
    ' Sub Actualizar_Tabla()
    '        Actualizamos todos los registros de la tabla que estamos
cargando

    ' ESTRUCTURA de llamadas entre métodos:
    ' -------------------------------------
    '  CargaDatos_alPDA() --->
    '    CargarFicheroTerDat()   --->  TrocearLineaTerDat()
    '      Do
    '      case 2    Actualizar_TablaPDA()
    '                BuscarSiglasTabladeCarico()
    '                ObtenerDefinicionTabla()  --->
                                          TrocearLineaTerDat()
    '      case 3   CreaRegistro()


Public Sub CargaDatos_alPDA(ByVal NombresFicheros() As String)
  ' Rutina Principal del módulo a la que se llama desde el formulario.


Private Sub CreaRegistro(ByVal WQueFic As String, ByVal NReg As
Integer)
        ' Crea un registro de salida en la tabla que corresponda,
partiendo del registro leido del Carico.d
        ' Es una única rutina común para todas las tablas que tengamos
que crear, sin necesidad de hacer una rutina para cada tabla


' BUCLE para ir obteniendo los distintos campos del registro leido (
wbu = numero de campo)
 For wbu = 1 To WNumMaxCamposporFicheroenTerdat
 ' SELECCION POR TIPO PROCESO
   Select Case UCase(WVProcesaCampoCaricoFicheroenCurso(wbu))
         Case "S", "E"    ' Sí se procesa
              Select Case UCase(Trim(WVTipoDatoCampoFichenCurso(wbu)))
                     Case "N"    ' Numérico
                          CreaCampoNumerico(WQueFic, wbu, Desde, "3")
                     Case "X"     ' Alfanumérico
                         CreaCampoAlfabetico(WQueFic, wbu, Desde, "3")
                        Case "B"     ' Booleano
                            CreaCampoBooleano(wbu, Desde)
                        Case "F"     ' Fecha
                            CreaCampoFecha(wbu, Desde)
                        Case "H"     ' Hora
                            CreaCampoHora(wbu, Desde)
```

```
                End Select

        Case "N"    ' NO se procesa ' saltamos el campo
                Desde = Desde + WVLongCampoCaricoFicheroenCurso(wbu)
        Case "I"    ' No existe en Carico, Pero se incluye en la PDA
                .......



'------------------------------
Public Class Listado
'------------------------------
  ' ------------------------
    ' Rutina ListarPorPantalla para pasarlo al formulario Form_Listar
    ' ------------------------
    ' Las variables siguientes tiene el listado completo con todas las
páginas y el número de páginas del listado
    Protected Listado() As String     ' cada elemento de Listado es
una página completa
    Protected NumeroPagina As Integer    ' indica el número de páginas
del listado
    ' --------------------
    ' Rutina ListarPorPapel
    ' --------------------
    ' Lo que hay en Listado() también está contenido en:
    Protected Linea() As String            ' El array Linea contiene
todas las líneas de Cabecera y Pie
    Protected LineaDetalle() As String       ' El array LineaDetalle
contiene todas las líneas de Detalle
    Protected NumeroLineaDetalle As Integer  ' indica el número total
de lineas de detalle obtenidas en el Listado


Protected Sub Seleccion_Cabecera(ByVal WTablaCab As String, _
                      ByVal WCab As Integer, _
                      ByVal ParamArray WParametroCab() As Object)

Protected Function Seleccion_Detalle(ByVal WTablaDet As String, _
            ByVal WDet As String, _
            ByVal ParamArray WParametroDet() As Object) As Boolean

Protected Sub Seleccion_Pie(ByVal WTablaCab As String, ByVal WPie As
Integer)


Public Sub ListarDocumentos(ByVal WTipoDocumento As String, _
                         ByVal WValorClaveCab1 As Object, _
                         Optional ByVal WSalida As String = " ", _
                         Optional ByVal WCDCodCLi As Long = 0)


Select Case WTipoDocumento

            Case "FACTURA"
                WPuerto = PDAini.PCon_Puerto_Com
                WClase = "Visita"
                WTablaCab = "TSCD"
                WCampoClaveCab1 = "CDVisita"
                WTablaDet = "TSDD"
                WCampoClaveDet = "DDVisita"
                WCampoLineaDet = "DDLinea"
                WCab = Val(PDAini.PIm1_Facturas_Cab)
```

```
                WDet = Val(PDAini.PIm1_Facturas_Det)
                WPie = Val(PDAini.PIm1_Facturas_Pie)
                WPapelContinuo = False
                If PDAini.PIm2_Papel_Continuo = "S" Then
WPapelContinuo = True
                WSaltoFin = Val(PDAini.PIm2_Lineas_Saltar_FinListado)
                WLinPag = Val(PDAini.PIm2_Lineas_Por_Pagina)
                WEspera =
Val(PDAini.PIm2_Retardo_Impresion_Milisegundos)
                WTitulo = "Visita"
                Visita = New Visita(Operacion.LEER, "VENTA",
WValorClaveCab1)
                WCopias = Val(PDAini.PIm2_Copias_Documento_Venta)

            Case "ALBARAN"
                 ..................
            Case ..............
End Select

ListarDocumentos2(WSalida, _
                          WPuerto, _
                          WClase, _
                          WTablaCab, _
                          WCampoClaveCab1, _
                          WValorClaveCab1, _
                          WTablaDet, _
                          WCampoClaveDet, _
                          WCampoLineaDet, _
                          WCab, _
                          WDet, _
                          WPie, _
                          WPapelContinuo, _
                          WSaltoFin, _
                          WLinPag, _
                          WEspera, _
                          WTitulo, _
                          WCopias, _
                          WCDCodCLi)


Protected Sub ListarDocumentos2(ByVal WSalida As String, _
                            ByVal WPuerto As String, _
                            ByVal WClase As String, _
                            ByVal WTablaCab As String, _
                            ByVal WCampoClaveCab1 As String, _
                            ByVal WValorClaveCab1 As Object, _
                            ByVal WTablaDet As String, _
                            ByVal WCampoClaveDet As String, _
                            ByVal WCampoLineaDet As String, _
                            ByVal WCab As Integer, _
                            ByVal WDet As Integer, _
                            ByVal WPie As Integer, _
                            ByVal WPapelContinuo As Boolean, _
                            ByVal WSaltoFin As Integer, _
                            ByVal WLinPag As Integer, _
                            ByVal WEspera As Integer, _
                            ByVal WTitulo As String, _
                            ByVal WCopias As String, _
                            Optional ByVal WCDCodCLi As Long = 0)
```

```
Private Sub ListarPorPapel_imprimir()
' ------------------------------------------------------------
' IMPRIMIR CABECERA  (UNA SOLA VEZ EN LA PRIMERA PAGINA)
        imprimir_cabecera_enpapel()
BUCLE IMPRIMIR DETALLE  (AQUI ESTA INCLUIDO EL SALTO OVERFLOW)
 ' en este bucle también está el control de salto de página con
impresión de nueva cabecera
        imprimir_detalle_enpapel()
        imprimir_desde_UltimaLineaDetalle_hasta_InicioPie_enpapel()
' IMPRIMIR PIE DE LISTADO (UNA SOLA VEZ EN ULTIMA PAGINA DEL LISTADO)
' Son líneas con datos, (por ejemplo totales, ....)
        imprimir_pie_enpapel()
'IMPRIMIR LINEAS FIN LISTADO (UNA SOLA VEZ EN ULTIMA PAGINA LISTADO)
' Son las líneas en blanco que hay que saltar por fín listado hasta
página nueva
        imprimir_finlistado_enpapel()
' ------------------------------------------------------
```

```
'------------------------------
Public Class LisDocAlbFac
'------------------------------
    Inherits Listado
```

Esta sería el modelo de “diseño” (aunque son líneas de programación) que habría que hacer a cada Enpresa que quiera un modelo de documento de impresión diferente en cabecera, detalle o pie. Sólo habría que crear una Cab01 Cab02, cambiando el contenido de estas rutina, porque la lógica de la impresión no cambia y las 55 variables, Datos(55), que se pueden imprimir en una cabecera de Factura ya están calculadas para todos y están disponibles llamando a Visita.I_DatosCabecera().
Para detalle y Pie (Det01 Det02 o un Pie01 Pie02 ..), sería análogo.
Voy a poner el ejemplo de la cabecera, ya que tiene muchas variables posibles para imprimir.

```
Protected Overrides Sub Cab00()
 Dim Datos(55) As String
 Datos = Visita.I_DatosCabecera()
 INICab = 1
 Linea(1) = Left(Trim(Datos(1)), 50)
 Linea(2) = Left(Trim(Datos(2)), 50)
 Linea(3) = Left(Trim(Datos(3)) & " " & Trim(Datos(4)) & " (" & _
                Trim(Datos(5)) & ")", 60)
 Linea(4) = Trim(Datos(6))
 Linea(5) = Left(PDAini.PTet_Literal1_Facturas & "
", 25) & Left(Trim(Datos(7)) & " " & Trim(Datos(8)), 33)
 Linea(6) = Left(PDAini.PTet_Literal2_Facturas & "
", 25) & Left(Trim(Datos(9)), 33)
 Linea(7) = Left(WDescAlbFac & " " & Trim(Datos(15)) & "/" &
Trim(Datos(16)) & "                        ", 25) &
Left(Trim(Datos(10)), 33)
Linea(8) = Left(FormatDateTime(Datos(17), vbShortDate) & " " &
FormatDateTime(Now, vbShortTime) & "                          ", 25) &
Left(Trim(Datos(11)) & Trim(Datos(12)), 33)
```

```
Linea(9) = Left(Trim(Datos(25)) & "                                ", 25)
& Left("CIF: " & Trim(Datos(14)), 33)

 CabeceraDetalle()
FINCab = 13
End Sub


'------------------------------
Public Class Visita
'------------------------------
Private CabeceraVisita As VisitaCabecera  ' objeto Cabecera de Visita
Private DetalleVisita() As VisitaDetalle ' objetos detalle visita
Private Clien As Cliente      ' objeto Cliente
Private TipoDocumento As String ' identifica el Tipo de Documento
"PEDIDO" / "VENTA"

Public Sub New(ByVal WTipoDocum As String, ByRef WClien As Cliente)
' CREAR OBJETO PARA DAR DE ALTA UNA VISITA (VENTA O PEDIDO)
' esta rutina es para cuando se crea una visita nueva, en vacío.( Para
darla de alta)
' Lo único que necesitamos pasar de momento es si va a ser una Venta o
un Pedido, y el Objeto Cliente
' Asigna el siguiente nuevo número correlativo de visita que
corresponda y Crea los atributos que definen a una Visita.

Public Sub New(ByVal WOperacionLeer As Operacion, ByVal WTipoDocumento
As String, ByVal WVisita As Integer)
' CREAR OBJETO PARA LEER  UNA VISITA (VENTA O PEDIDO) YA EXISTENTE EN
LA BASE DE DATOS
' esta rutina es para cuando se accede a una visita ya existente (
recuperar para leerla en listados o para dar modificacion o baja)
' Crea un objeto de la clase Visita asignámdole el número de visita
que nos dan

Public Function CalcularTotalCabecera() As Object
' SE CALCULA EL DOCUMENTO QUE HAY EN MEMORIA
' y devuelve un array bidimensional
' Wvalor(i, 0) = 0            ' valores (0/1) 0= nula, 1= válida
'Wvalor(i, 1) = 0             ' codigo iva solo a nivel detalle (i<>0)
'Wvalor(i, 2) = 0             ' Importe Suma lineas
'Wvalor(i, 3) = 0             ' Importe Punto Verde
'Wvalor(i, 4) = 0             ' Importe PUR
'Wvalor(i, 5) = 0             ' Importe Bruto
'Wvalor(i, 6) = 0             ' % Dto. PP
'Wvalor(i, 7) = 0             ' Importe Dto. PP
'Wvalor(i, 8) = 0             ' % Dto. S/Fra.
'Wvalor(i, 9) = 0             ' Importe Dto. S/Fra.
'Wvalor(i, 10) = 0            ' Importe Base imponible Iva
'Wvalor(i, 11) = 0            ' % de iva               solo a nivel
detalle (i<>0)
'Wvalor(i, 12) = 0            ' Importe Iva
'Wvalor(i, 13) = 0            ' % Recargo              solo a nivel
detalle (i<>0)
'Wvalor(i, 14) = 0            ' Importe Recargo
'Wvalor(i, 15) = 0            ' Importe Neto  (Imp.Suma Lineas -
Imp.Dto.P.P. - Imp.Dto.S/Fra + Imp.Iva + Imp.RE)
'Wvalor(i, 16) = 0    ' % IRPF     solo a nivel factura (i=0)
'Wvalor(i, 17) = 0   ' Importe IRPF solo a nivel factura (i=0)
'Wvalor(i, 18) = 0   ' Importe Retención IRPF solo nivel factura (i=0)
'Wvalor(i, 19) = 0   ' Importe Total     solo a nivel factura (i=0)
```

```
        CC_InicializarVariablesCabecera()
        ' COMIENZO BLOQUE 1 PRINCIPAL
        ' --------------------------
        If ExisteVisita = True Then
            CC_ObtenerDatosCliente()
            'Recalcular BI,IVA,RE
            If MaxLineaDetalle > 0 Then  ' Existen Lineas de detalle
                ' COMIENZO BUCLE 2 lectura TDeta.
                For lineax = 1 To MaxLineaDetalle
                   If ExisteDetalleVisita(lineax) Then
                        ' Calculos a nivel de cada linea de detalle
                     lineasleidas = lineasleidas + 1
                     CC_DatosArticulo(DetalleVisita(lineax).DDCodArt)
                     CC_Calcularsubindice_yporcentajesIVARE()
                     CC_CalcularNetoLineayBasesIvaRE()
                     If TipoDocumento = "VENTA" Then
                            CC_CalcularImpPVerde()
                            CC_CalcularImpPUR()
                     End If
                   End If
                Next
                ' FIN BUCLE 2 lectura TDeta
            End If
        End If

        ' FIN BLOQUE 1 PRINCIPAL
        ' ----------------------
        If lineasleidas <> 0 Then   ' Hay líneas de detalle
            ' Calculos a nivel de total Factura
            CC_BloqueHayLineasDetalle()
        End If
        CC_AsignarValoresRetorno()
        CalcularTotalCabecera = Wvalor
    End Function

 METODOS GrabarCabecera
 ----------------------
' Contiene las siguientes Funciones:
' Se ejecuta cuando se hace New():
'    caso de nueva visita: N_PrepararCabeceraVisita_enMemoria()
'    caso de visita ya existente :  N_LeerDocumentodedisco
' ActualizarCabeceraVenta_enMemoria()
' ActualizarCabeceraPedido_enMemoria()
' FinDocumento()

METODOS GrabarLineaDetalle
--------------------------
Public Function Grabar_DetalleVisita_enMemoria(ByVal WLinea As Short,
ByVal RegDet As EstructuraDetalleVisita) As Short
' GRABA LA LINEA DE DETALLE EN EL ARRAY DE MEMORIA,
If WLinea > 0 Then
            ' Caso de Dar una Modificación
            Modificar_DetalleVisita_enMemoria(WLinea, RegDet)
            Grabar_DetalleVisita_enMemoria = WLinea
        Else
            ' Caso de Dar una Alta
            Grabar_DetalleVisita_enMemoria =
AlmacenarDatos_DetalleVisita_enMemoria(WLinea, RegDet)
        End If
 End Function
```

```
'------------------------------
Public Class VisitaDetalle
'------------------------------
Private TDeta As Tabla            ' Objeto TablaDetalle
Private WArti As Articulo         ' Objeto Artículo
Private TipoDocumento As String   ' identifica el Tipo de Documento
"PEDIDO" / "VENTA"
Private Sub CogerDatosdeDisco()
Public Sub DejarDatosenDisco()
```

6.- INSTALACIÓN Y CONFIGURACIÓN DEL SISTEMA

6.1. INSTALACIÓN

La instalación nueva en una Empresa implica previamente la Configuración de todos los parámetros conjuntamente con la Persona Responsable de la Empresa y la selección de los diseños de los documentos de impresión que se ajusten a lo que necesitan.
Además está previsto que si no se ajusta ninguno de los que ya hay, se le implemente su versión dado que el mismo documento de impresión comparte código para todas las empresas y difiere en lo que es el diseño de impresión, de forma que la propia empresa puede cambiar en parámetros el número de diseño de cabecera/detalle/pie de cada documento y va obteniendo la impresión de los distintos diseños de impresión de ese documento.

Para la Instalación he preparado unos SETUP (ficheros ejecutables) que instalan tanto la Aplicación como también el software básico que se necesita para que funcione la Aplicación.

Abarcan tanto la instalación nueva en todas las PDAs de una Empresa como la instalación de una nueva versión teniendo en cuenta que lo normal será que una vez instalada en una PDA, se produzcan modificaciones y mejoras que a la Empresa le interese actualizar.
Hay que tener en cuenta que esta Aplicación la he preparado para que funcione con distintos sistemas operativos y con distintos tamaños y resoluciones de pantalla.

La instalación puede ser en la memoria de la PDA o en la Tarjeta de Memoria extraible.
Lo normal es que se elija Tarjeta de Memoria extraible, para que la Aplicación, todos los Datos, y la copia de los ejecutables para la instalación del software básico estén en la Tarjeta y por tanto a salvo de cualquier problema que le pueda ocurrir a la PDA.
Se puede ejecutar desde PC conectado a la PDA por ActiveSync o desde la propia PDA, y en este caso, desde las PDAs que tengan conexión a internet se puede acceder al servidor de FTP donde está el setup de la Aplicación y ejecutarlo para la instalación inicial de la Aplicación o una nueva versión.
En todos los casos, lo que se hace es acceder a un servidor FTP y elegir el setup adecuado y lo puede instalar el usuario.

A una instalación ya realizada, una actualización de la Aplicación le respeta el fichero de configuración de parámetros que ya haya definido, pero le incluye los parámetros que se hayan definido nuevos con la opción más adecuada a su Empresa, y además siempre puede cambiar cualquier parámetro antiguo o nuevo.

6.2. CONFIGURACIÓN DE PARAMETROS. PERSONALIZACIÓN.

La primera tarea a realizar conjuntamente con la empresa a la que se instala la Aplicación es la introducción del valor adecuado en cada parámetro de la Aplicación.
A continuación se relacionan dichos parámetros.

[Conexiones]
Puerto_Com="COM8"
Activar_Transferencia_Internet="N"
Usuario="elquesea"
contraseña="laquesea"
servidor="elquesea.com"
Conectar_Automaticamente="N"
Conexion="nombre"
Retardo="100"

[Impresion1]
Pedidos_Cab="00"
Pedidos_Det="00"
Pedidos_Pie="00"
Albaranes_Cab="00"
Albaranes_Det="00"
Albaranes_Pie="00"
Facturas_Cab="00"
Facturas_Det="00"
Facturas_Pie="00"
Recibo_Cobro_Cab="00"
Recibo_Cobro_Det="00"
Recibo_Cobro_Pie="00"
Albaranes_Proveedor_Cab="00"
Albaranes_Proveedor_Det="00"
Albaranes_Proveedor_Pie="00"
Cargas_Mercancia_Cab="00"
Cargas_Mercancia_Det="00"
Cargas_Mercancia_Pie="00"
Recuentos_Cab="00"
Recuentos_Det="00"
Recuentos_Pie="00"
Recibo_Contado_Cab="00"
Recibo_Contado_Det="00"
Recibo_Contado_Pie="00"
Entrega_Mercancia_Cab="00"
Entrega_Mercancia_Det="00"
Entrega_Mercancia_Pie="00"
Obsequio_Cab="00"
Obsequio_Det="00"
Obsequio_Pie="00"

[Impresion2]
Lineas_Por_Pagina="048"
Lineas_Saltar_FinListado="005"
Max_Segundos_En_Espera_Impresora="015"
Retardo_Impresion_Milisegundos="400"
Imprimir_Documentos="S"
Impresion_Cobros_Automatica="S"
Impresion_DocumVenta_Automatica="S"
Preguntar_Menu_Pendientes="N"
Salida_Listados="PAPEL"
Tipo_Impresora="STAR"
Imprimir_CodArt_Cliente="N"
Imprimir_Codigo_Articulo="S"
Imprimir_Datos_Empresa="S"
Papel_Continuo="N"
Copias_Documento_Venta="1"
Copias_Documento_Pedido="1"
Copias_Documento_CargaMercancia="1"
Copias_Documento_Recuento="1"
Copias_Documento_AlbProveedor="1"
Mensaje_Repetir_Listado="S"

[Impresion3]
Lis_Act_FactAlb_Cab="00"
Lis_Act_FactAlb_Det="00"
Lis_Act_FactAlb_Pie="00"
Lis_Act_Cobros_Cab="00"
Lis_Act_Cobros_Det="00"
Lis_Act_Cobros_Pie="00"
Lis_Act_Entrega_Cuenta_Cab="00"
Lis_Act_Entrega_Cuenta_Det="00"
Lis_Act_Entrega_Cuenta_Pie="00"
Lis_Act_AlbPro_Cab="00"
Lis_Act_AlbPro_Det="00"
Lis_Act_AlbPro_Pie="00"
Lis_Act_Gastos_Cab="00"
Lis_Act_Gastos_Det="00"
Lis_Act_Gastos_Pie="00"
Lis_Act_Resumen_Cab="00"
Lis_Act_Resumen_Det="00"
Lis_Act_Resumen_Pie="00"
Listado_Stock_Cab="00"
Listado_Stock_Det="00"
Listado_Stock_Pie="00"
Pendiente_Cobro_Cab="00"
Pendiente_Cobro_Det="00"
Pendiente_Cobro_Pie="00"
Recorrido_Diario_Cab="00"
Recorrido_Diario_Det="00"
Recorrido_Diario_Pie="00"

Ventas_Articulo_Cab="00"
Ventas_Articulo_Det="00"
Ventas_Articulo_Pie="00"
Clientes_Sin_Visitar_Cab="00"
Clientes_Sin_Visitar_Det="00"
Clientes_Sin_Visitar_Pie="00"
Resumen_Pedidos_Cab="00"
Resumen_Pedidos_Det="00"
Resumen_Pedidos_Pie="00"
Gastos_Cab="00"
Gastos_Det="00"
Gastos_Pie="00"

[Ventas1]
Liberar_Maximo_Minimo="N"
Liberar_Precio_Especial="N"
Permitir_Precio_Cero="S"
Liberar_Descuentos_Promociones="N"
Activar_Descuentos_Promociones="N"
Antes_Descuento_Que_Promocion="S"
Promocion_en_Porcentaje="S"
Descuento_en_Porcentaje="S"
Se_Utiliza_PUR="N"
Se_Utiliza_PuntoVerde="N"
Se_Utiliza_PuntoVerde_Regalo="N"

[Ventas2]
Utiliza_Exclusividad_Cliente_Articulo="N"
Gestion_Articulos_Preferidos_Cantidad_1="N"
Avisar_Si_NoHayStock="AVISA Y DEJA"
Gestion_Lotes="N"
Bloqueo_Recuperar_Venta="N"
Gestion_Codigo_Barras="N"
Prioridad_Codigo_Barras_Busqueda_Articulo="N"
Permitir_Sobrepasar_Max_Env_Deposito="S"
Ver_Mensajes_Informacion_Regalo="S"
Preguntar_sise_Regala="S"
Codigo_Proveedor_Empresa="7"

[Ventas3]
Entrega_A_Cuenta="S"
Permitir_Cobro_Albaranes="S"
Cobro_En_Contado="S"
Activar_Ventas_Nulas="N"
Ventas_PonerenCant_1="S"
Ventas_Saltar_Campo_Cant_1="N"
CarMer_PonerenCant_1="S"
Saltar_Campo_Precio="S"
Liberar_Limite_Credito="S"
Liberar_Limite_NumFrasCredito="S"

[Gastos]
Proveedor_Gasolina="1"
Proveedor_Restaurante="2"
Proveedor_Varios="3"
Articulo_Gasolina="1"
Articulo_Restaurante="2"
Articulo_Varios="3"

[Busquedas1]
Marcar_Carga_como_Articulo_Preferido="N"
Rutero_Con_Detalle="S"
Rutero_Con_Nombre_De_Cliente="N"
UnSolo_Carico_Entrada="S"
UnSolo_Carico_Salida="S"
NumDias_A_Mantener_Ficheros_De_Descarga="5"
Descarga_Carga_Automatica="S"
Path_Datos="\DATOS\"
Path_Carga="\CARGA\"
Path_Descarga="\DESCARGA\"
Path_Ayudas="\AYUDAS\"

[Busquedas2]
Actualizacion_Parametros_Automatica="S"
Articulo_Interactivo="S"
Articulo_Grupo="01"
Articulo_Subgrupo="01"
Articulo_Campo="01"
Articulo_Condicion="01"
Articulo_Cabeceras="01"
Cliente_Interactivo="S"
Cliente_Campo="01"
Cliente_Condicion="01"
Cliente_Cabeceras="01"
Proveedor_Interactivo="S"
Proveedor_Campo="01"
Proveedor_Condicion="01"
Proveedor_Cabeceras="01"
Muestrario_Interactivo="S"
Muestrario_Campo="01"
Muestrario_Condicion="01"
Muestrario_Cabeceras="01"

[Accesos]
Menu_Parametros="S"
Parametros_Privados="S"
Menu_Ventas="S"
Menu_Pedidos="S"
Menu_Cobros="S"
Menu_Compras="S"
Menu_Gastos="S"

Movimientos_Mercancia="S"
Menu_Ini_Stocks="S"
Listados="S"
Resumen_Ventas="S"
Resumen_Pedidos="S"
Reimpresion_Documentos="S"
Listado_Stock="S"
Listado_Pendiente_Cobro="S"
Listado_Ventas="S"
Recorrido_Diario="S"
Listado_Clientes_Sin_Visitar="S"
Ver_Documentos_Tipo_Texto="S"

[Formu]
FrmAlbProveedor="01"
FrmAlmacen="01"
FrmBuscar="01"
FrmBuscarOpciones="01"
FrmCargaMercancia="01"
FrmCargaPDA="01"
FrmCobros="01"
FrmCompras="01"
FrmDatClientes="01"
FrmDatClientesMasDatos="01"
FrmDescargaPDA="01"
FrmDesResuActi="01"
FrmEnvios="01"
FrmFormaPago="01"
FrmGastos="01"
FrmHisVenPed="01"
FrmImpFacPen="01"
FrmIncidenciasCli="01"
FrmInformes="01"
FrmIniStock="01"
FrmListar="01"
FrmManteClientes="01"
FrmManteClientesMasDatos="01"
FrmManteDirEnvio="01"
FrmMenuPrincipal="01"
FrmParametros="01"
FrmPedirDatos="01"
FrmPedPenServ="01"
FrmPenCobros="01"
FrmReenvio="01"
FrmReimDocum="01"
FrmRutero="01"
FrmSelDiaSem="01"
FrmSelDoc="01"
FrmSelStk="01"
FrmUltimasVentas="01"

FrmVenPed="01"
FrmVenPedMasDatos="01"
FrmVerDocum="01"

[Frm1]
Tamaño_Fuente_Lista="8"
Fuente_Negrita="N"
Estilo_Subrayado="N"
Alternancia_Registros="S"
Tamaño_Fuente_Listado="8"
Ver_Mensaje_Novedades="S"
Longitud_Codigo_Cliente="15"
Longitud_Codigo_Articulo="15"

[Frm2]
Actualizacion_Resize_Automatica="S"
Parámetros automáticos de la Gestión de todas las columnas de todos los listview de la Aplicación

[Privado]
Permitir_Anular_Cobro="S"

[Entorno]
Comprobar_Memoria="S"
Permitir_Recuento_Almacen_General="S"
Permitir_Cambiar_Forma_Pago_Cliente="S"
Permitir_Cambiar_PaC_CaP_MaC="S"
Descarga_Carga_Automatica="S"
Gestion_Pedidos="ESTANDAR"
DespuesdeFinVenta_IraRutero="S"

[Entorno2]
Activar_Teclado_Numerico="S"
Activar_Teclado_Alfanumerico="S"
Activar_Teclado_Numerico_Automatico="N"
Activar_Teclado_Alfanumerico_Automatico="N"
Resolucion_Pantalla="320 X 240"

[TeclasFuncion]
Utilizar_Teclas_Funcion="S"
F1_Buscar="F1"
F2_Seleccion_Tabla="F2"
F3_Tecla3="F3"
F4_Carga_Aplicacion="F4"
F5_Menu1="F5"
F6_Menu2="F6"
F7_Tecla7="F7"
F8_Fin_Linea="F8"
F9_Salir="F9"
F10_Fin_Venta="F10"

7.-CONCLUSIONES

Este proyecto ha consistido en la creación de la Aplicación ALfaPDA, con unos condicionantes de requerimientos de tal versatilidad que va a constituir una línea de negocio y por consiguiente el objetivo es que sea un referente para Empresas que necesiten alguna de las siguientes tareas en dispositivos móviles:

Gestión de Pedidos Gestión de Almacén Gestión de Compras
Gestión de Ventas Gestión de Cobros

Como ejemplo, fuera ya de la cobertura del presente proyecto ya he desarrollado la Gestión de Pedidos para una Empresa de Comercialización de Textil dentro de AlfaPDA como una opción dentro de la Gestión de Pedidos, que comparte las Clases y los Métodos definidos en la Gestión estándar, empleando las potencialidades del Diseño Orientado a Objetos.

En estos primeros 5 meses de funcionamiento, la Aplicación AlfaPDA ya se ha instalado en unas 25 empresas, no todas pertenecientes a la Comunidad Valenciana.

Los Dispositivos móviles en los que está ya instalada la AlfaPDA son muy dispares:

Sin teclado físico y con teclado físico.
Sin Teclas de Función y con teclas de Función.
Sin Lector de Código de Barras y con Lector.
Con pantalla de resolución "320 X 240" y otras con resolución de "640 X 480"
Con pantalla de tipo vertical y otras de tipo apaisado.
Con/Sin Tarjeta de conexión a Internet.
Con/Sin wiifi para conectarse a la red de la empresa y a través de esa red a Internet.
Con bluetooth para conectarse también a un teléfono móvil y a través de la conexión de internet del teléfono móvil conectarse a Internet.
Con bluetooth para conectarse a la impresora.
Con dispositivos móviles industriales que tienen integrada la PDA y la impresora en el mismo dispositivo y que por tanto la impresión es a través de conexión física.

El objetivo de versatilidad de la Aplicación y personalización para cada Empresa han conseguido que cada una de ellas sólo vea en pantalla lo que realmente necesita.

La experiencia de ALFA INFORMATICA http://www.alfa-informatica.com en el sector de las Empresas de Distribución y en el desarrollo de Aplicaciones para dispositivos móviles, han posibilitado unas Funcionalidades de la Aplicación AlfaPDA muy amplias, que se han visto complementadas con un diseño adecuado desde el punto de vista de la Ingeniería del Software.
Todo ello ha dado como resultado un **producto "AlfaPDA" de calidad y altamente competitivo** para ser ofertado a todas las Empresas de Distribución que necesiten una Aplicación de Pedidos/Ventas para su flota de vehículos.

8.-ANEXO. MANUAL DE USUARIO DE LA APLICACIÓN

INICIO APLICACION

Para arrancar la Aplicación debemos puntear en el programa AlfaPDA del Inicio de Windows o pulsar <F4> en aquellas máquinas que tengan teclas de función.

Obtendremos el Menú General de la Aplicación, y al personalizarla sólo aparecerán los botones correspondientes a los bloques que la Empresa va a utilizar.

Pulsando AlfaPDA se visualiza la versión de la Aplicación.

Opcionalmente si la Empresa puede dejarle también un fichero al Vendedor para informarle de alguna novedad que considere importante para el trabajo de ese día para el vendedor.

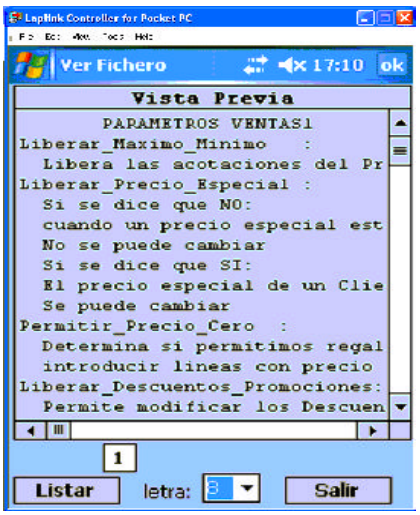
Hay una serie de claves de acceso que si quiere puede habilitar la Empresa para obligar a introducir la clave en algunas operaciones, con el fín de permitir con el mismo menú de funcionamiento, diferentes niveles de posibilidad de realización de operaciones en función de cada Agente concreto.

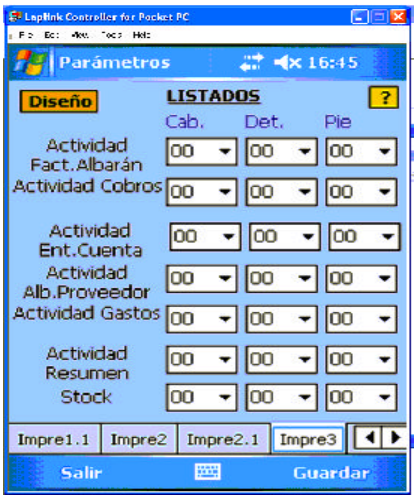
BOTÓN PARAMETROS

La primera tarea a realizar con el Cliente es la configuración de los parámetros de la Aplicación.

La explicación de cada parámetro en cada pantalla se obtiene pulsando ? en esquina superior derecha.

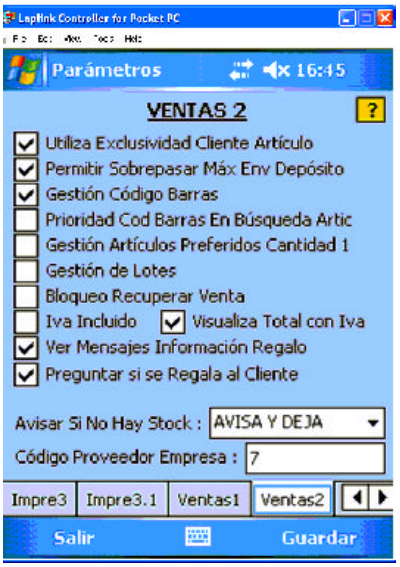
Todo este bloque es la personalización de la Aplicación para el Cliente.

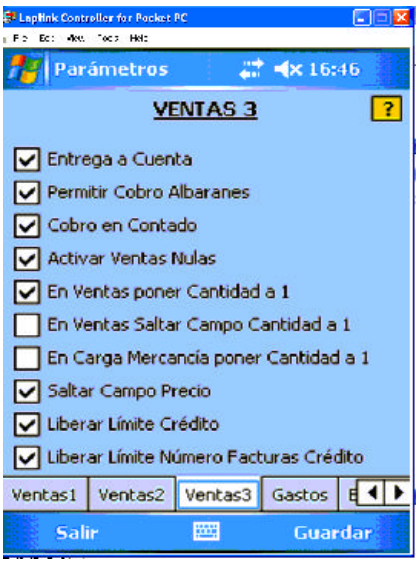


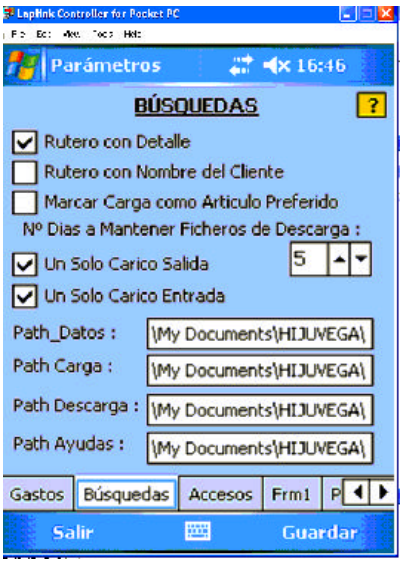


Hay diferentes pestañas que van agrupando los parámetros en bloques de tratamiento.

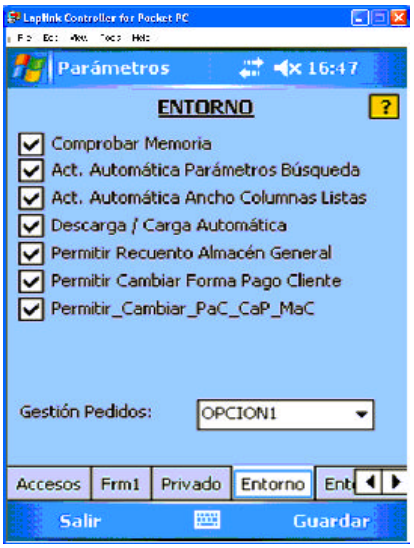
Todo este bloque de parámetros no es para el uso normal del vendedor, sino para configurar el funcionamiento de la Aplicación en las distintas PDAs que tenga la Empresa en función del perfil del Agente y el destino de cada PDA: Ventas, Pedidos, ...







Se puede personalizar para el usuario tanto las opciones (botones) que le saldrán en la pantalla, como el aspecto visual de los distintos listview que tienen los formularios, o la forma de funcionar de los teclados virtuales numérico y alfanumérico.



La Aplicación tiene la posibilidad de utilizar hasta 10 teclas de función diferentes en aquellas PDAs que tengan dichas teclas. Estas Teclas de Función también se pueden elegir por el usuario.

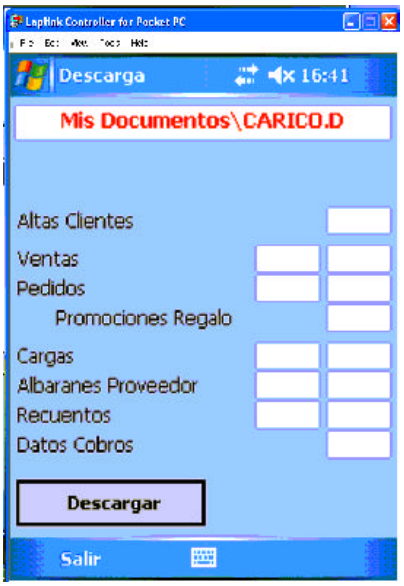
BOTÓN DE CARGA

Se presenta una pantalla con el fichero que se va a importar y se le da la conformidad.
Según haya sido la opción de acceder a ese fichero, se conectará al servidor de FTP o se leerá en modo local, habiéndose cargado previamente.

Las tareas de Comienzo y Final del Día son:
Carga de PDA y
Descarga de PDA
Lo normal es que se haga una Carga al principio del día y que se haga una Descarga al final del día, pero se podrían hacer varias descargas durante el día si se quiere tener inmediatez por ejemplo en la comunicación de los Pedidos a la Empresa.

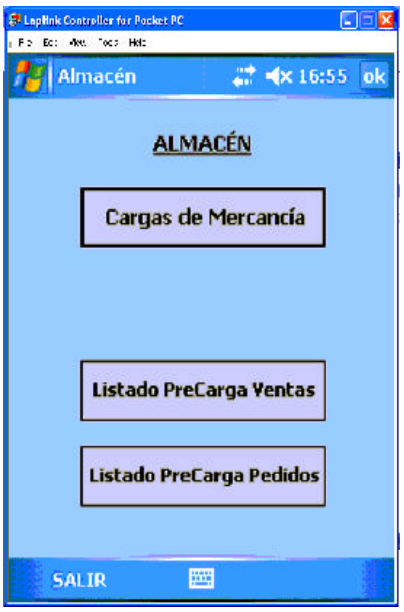
BOTON DE DESCARGA

Se pide conformidad a la generación de la descarga del trabajo realizado durante el día.
Según haya sido la opción de generar ese fichero, se conectará al servidor de FTP para enviarlo o se grabará en modo local, para su posterior envío.



GESTION DE ALMACEN

Consta de los botones Carga Mercancía y dos listados para que la Carga incluya por lo menos el contenido de esos listados.
Listado Precarga Venta: se recogen las ventas realizadas (para reponer mercancía).
Listado Precarga Pedidos: se recogen los pedidos realizados (para cargar mercancía por parte del vendedor), en un funcionamiento diferenciado entre Agente de Pedido y Agente de Venta.



CARGA MERCANCÍA

Contempla la Carga de Mercancía al vehículo (que es el Almacén del Vendedor) para que se recoja el traslado de Mercancía desde el Almacén General o cualquier otro Almacén al vehículo del vendedor. Servirá también para gestionar el stock de los artículos en el vehículo.

RECUENTO

Cuando el Almacén Origen y Destino es el mismo se trata de un Recuento de Mercancía del Almacén (Inventario).
En toda la Aplicación funciona la Introducción del Artículo mediante la Captura del Código de Barras, en las PDAs que tengan lector.
Pulsando en el icono <buscar> se va a una pantalla de Ayuda para selección de Artículo.

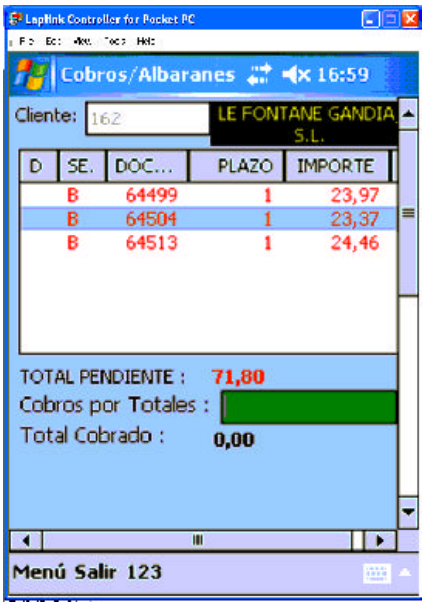
La Aplicación está preparada para que se pueda utilizar como Carga de Mercancía y Recuento para el Almacén Central de la Empresa (no sólo vehículos) pues lo lógico cuando se está cargando mercancía en una nave, es ir con un dispositivo móvil (PDA), capturar el movimiento de Almacén y a continuación se transmite al ordenador de la Empresa.

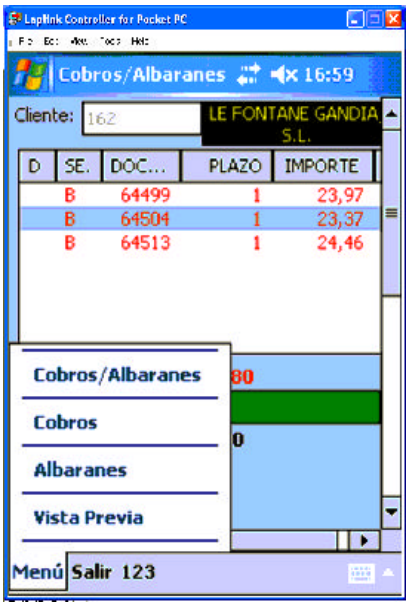
En la Pantalla hay disponible un Menú para realizar:
Recuperar Documento para modificación, Reimpresión del Documento, Borrado de una línea ya introducida, Borrado del Documento, Cargar las Ventas del Día (para rapidez en la Captura de la mercancía, cuando lo que se va a hacer es reponer mercancía al vehículo.

COBROS

Al pulsar el botón de Cobros se visualiza un Rutero con los Clientes con Cobros Pendientes en ese día de la semana, y aparecen ya en el Orden de la Ruta física que debe seguir el vehículo.

Se selecciona un cliente y se visualizan todos los cobros pendientes. Hay opciónes de cobrar total o parcialmente algún cobro pendiente y de cobrar por totales asignando una cantidad y que la vaya asignando a los cobros pendientes.

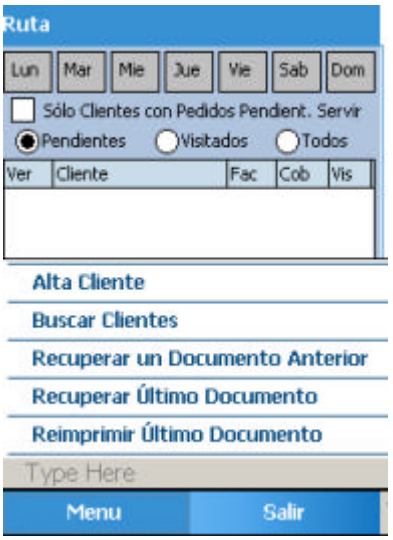




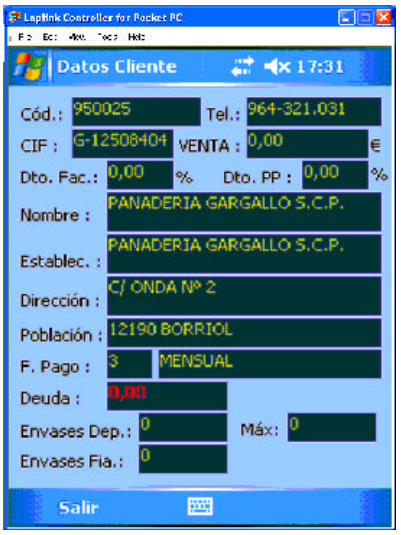
VENTAS Y PEDIDOS

Al pulsar el botón de Ventas se visualiza un Rutero con los Clientes que tienen asignados visitar ese día de la semana, y aparecen ya en el Orden de la Ruta física que se supone debe seguir el vehículo..

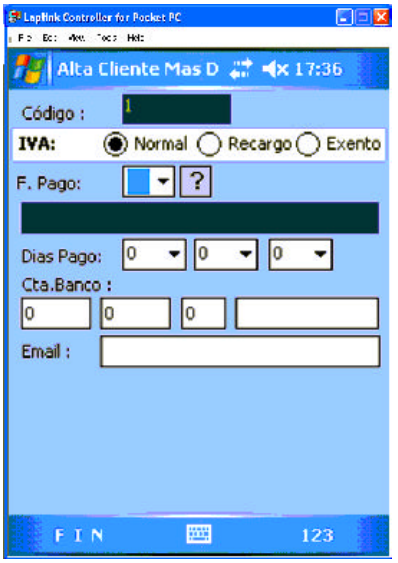
Desde el Rutero hay opción de Seleccionar sólo los Clientes con Pedidos Pendientes de Servir, de ver sólo los Clientes de esa ruta pendientes de visitar o los ya visitados o Todos.
También se puede ver los Datos del Cliente, Realizar un Alta de Cliente, Ir a la Búsqueda de Clientes, Recuperar un Documento (Venta o Pedido) anterior, Recuperar el último Documento Realizado, o Reimprimirlo



DATOS DEL CLIENTE

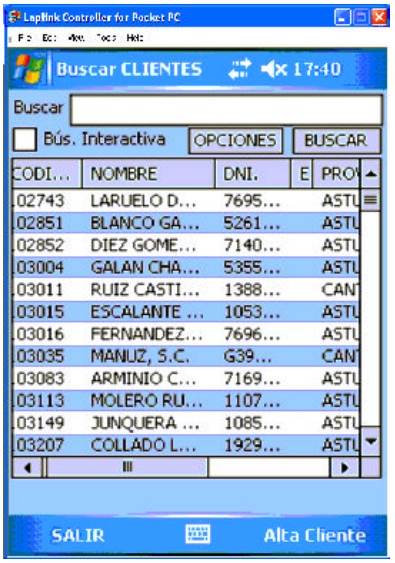


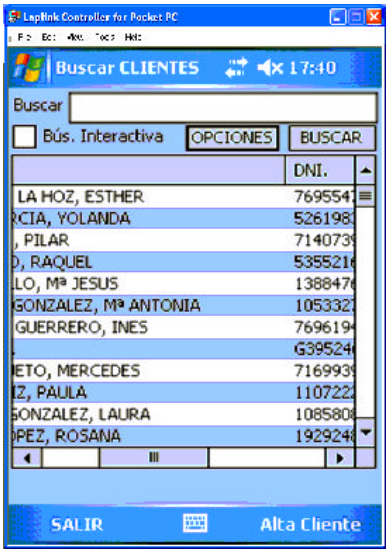
ALTAS DE CLIENTES



BUSCAR CLIENTES

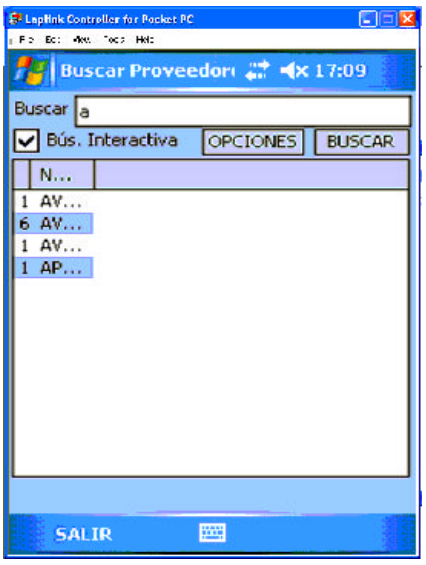
Hay opción de ir a la pantalla de Ayuda en la Selección de Clientes para buscar directamente el Cliente que deseamos, con el apoyo de la pantalla de Opciones de búsqueda donde se puede selecionar el campo por el que se busca (código nombre, población) y el criterio de comparación (que comience por, que contenga, ...)

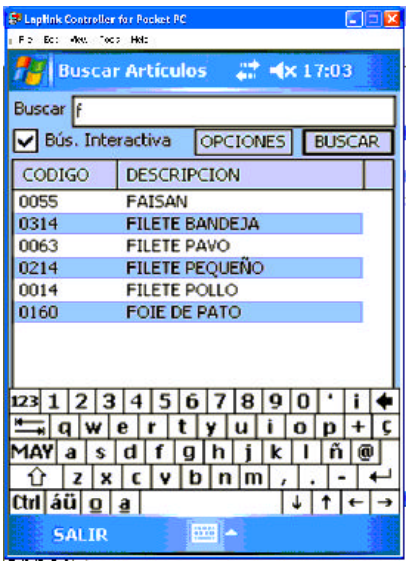




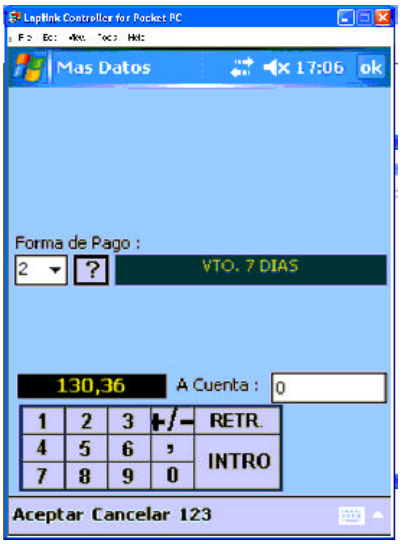
PANTALLA DE VENTA

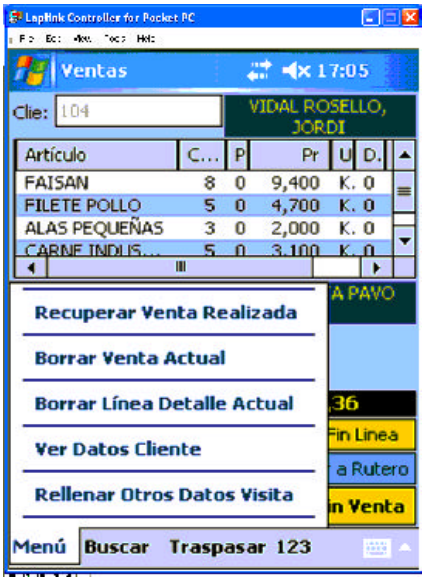
Es también la misma que para realizar un Pedido.





La pantalla de introducir otros Datos sería distinta si se está haciendo un Pedido o si se está haciendo una Venta.

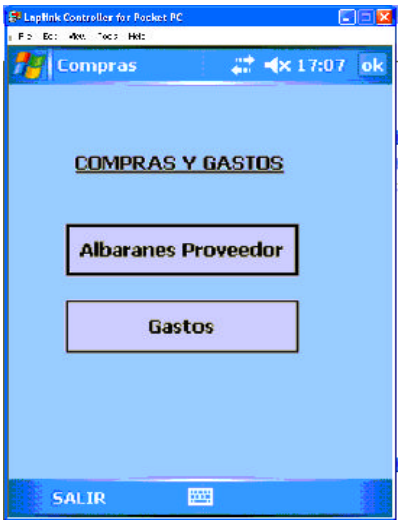


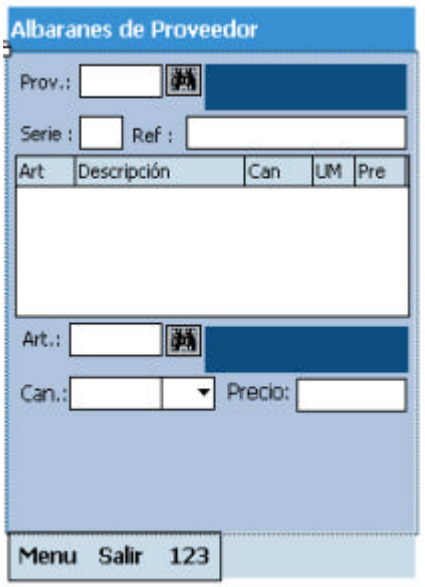


Las Opciones de Traspasar permiten ir a unas pantallas donde se visualizan:
Pedidos Pendientes de Servir a ese Cliente.
Artículos marcados por la Empresa como preferidos a la hora de hacer Ventas.
Histórico de Ventas del Cliente, para seleccionar desde su historial.
Ultimas Ventas del Cliente.

COMPRAS Y GASTOS

Se presenta una pantalla para seleccionar:
Compras de Albaranes a Proveedor.
Gastos del Vendedor.
Los funcionamientos de estos Formularios están claros viendo las pantallas siguientes.





RESUMEN

El botón Resumen se pulsa al finalizar la jornada de trabajo, para obtener los Resúmenes de Actividad del Agente:

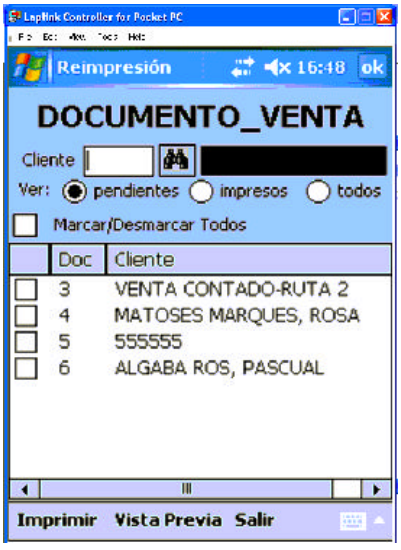
-de Ventas
-de Cobros
-Entregas a Cuenta
-de Compras a Proveedores
-de Gastos
-de Actividad Agrupado

IMPRESION

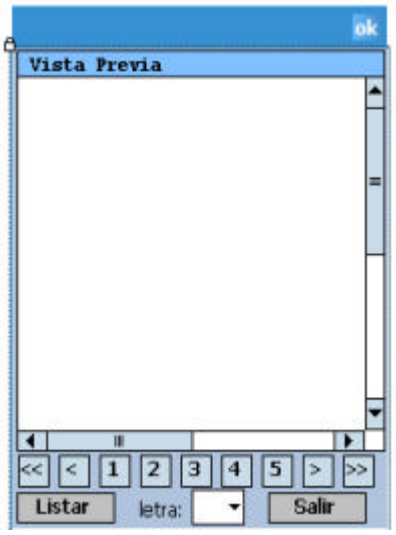
El Bloque de Impresión contempla las posibilidades de Imprimir por papel, generar en fichero o consultar por pantalla cualquier Documento o Listado que tenga necesidad de realizar el Vendedor.

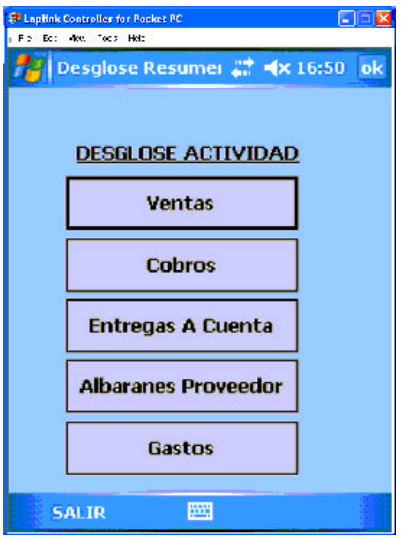
El contenido aparece en la pantalla adjunta.

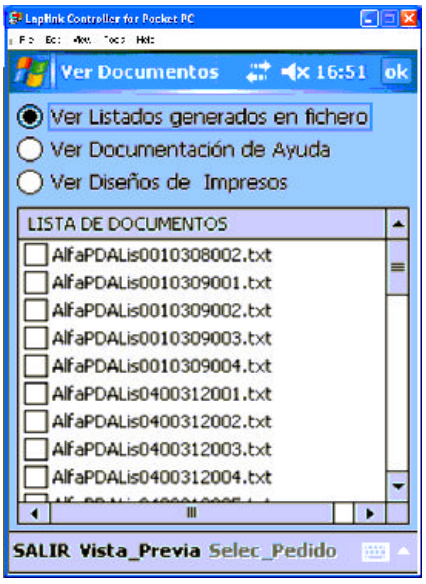
La Reimpresión de Documentos permite volver a imprimir cualquier documento que se haya ya impreso, presentando una pantalla con el contenido de dichos documentos.
En la vista previa de cualquier listado aparece una barra con unos botones de navegación.

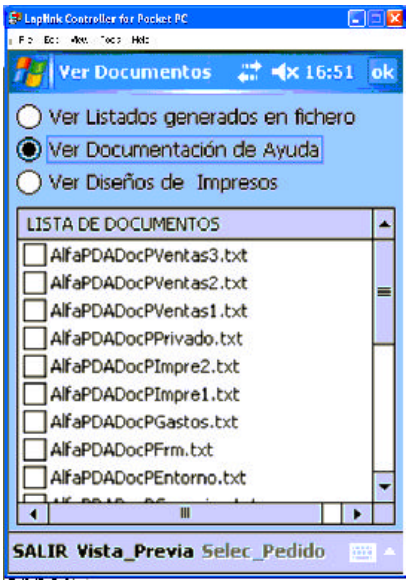


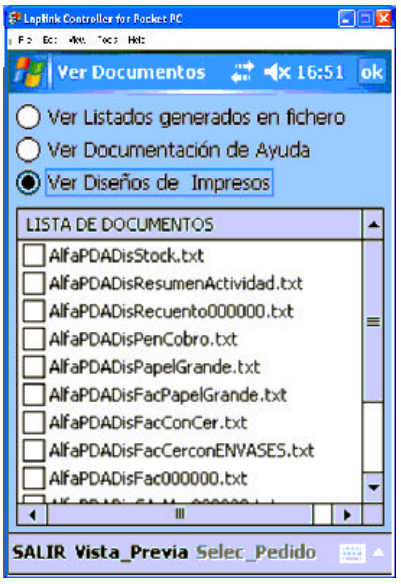

Inicio 16:49 ok
Vista Previa
B-03071800
100 VE
t
Factura: 2/99
07/03/07 16:49
46600A
CONTADO
CIF:
Ruta: 002 AGUSTIN E.G.
Articulo
Caj
JAMONCITO POLLO 10,
CER Desp-Elavorad 10,
CUARTOS POLLO 30,
CER Desp-Elavorad 30,
HIGADO POLLO 40,
1
Listar
letra:
Salir


ok
Vista Previa
<< < 1 2 3 4 5 > >>
Listar
letra:
Salir


Desglose Resumen 16:50 ok
DESGLOSE ACTIVIDAD
Ventas
Cobros
Entregas A Cuenta
Albaranes Proveedor
Gastos
SALIR


Ver Documentos 16:51 ok
Ver Listados generados en fichero
Ver Documentación de Ayuda
Ver Diseños de Impresos
LISTA DE DOCUMENTOS
AlfaPDALis0010308002.txt
AlfaPDALis0010309001.txt
AlfaPDALis0010309002.txt
AlfaPDALis0010309003.txt
AlfaPDALis0010309004.txt
AlfaPDALis0400312001.txt
AlfaPDALis0400312002.txt
AlfaPDALis0400312003.txt
AlfaPDALis0400312004.txt
SALIR Vista_Previa Selec_Pedido


Ver Documentos 16:51 ok
Ver Listados generados en fichero
Ver Documentación de Ayuda
Ver Diseños de Impresos
LISTA DE DOCUMENTOS
AlfaPDADocPVentas3.txt
AlfaPDADocPVentas2.txt
AlfaPDADocPVentas1.txt
AlfaPDADocPPrivado.txt
AlfaPDADocPImpre2.txt
AlfaPDADocPImpre1.txt
AlfaPDADocPGastos.txt
AlfaPDADocPFrm.txt
AlfaPDADocPEntorno.txt
SALIR Vista_Previa Selec_Pedido


Ver Documentos 16:51 ok
Ver Listados generados en fichero
Ver Documentación de Ayuda
Ver Diseños de Impresos
LISTA DE DOCUMENTOS
AlfaPDADisStock.txt
AlfaPDADisResumenActividad.txt
AlfaPDADisRecuento000000.txt
AlfaPDADisPenCobro.txt
AlfaPDADisPapelGrande.txt
AlfaPDADisFacPapelGrande.txt
AlfaPDADisFacConCer.txt
AlfaPDADisFacCerconENVASES.txt
AlfaPDADisFac000000.txt
SALIR Vista_Previa Selec_Pedido

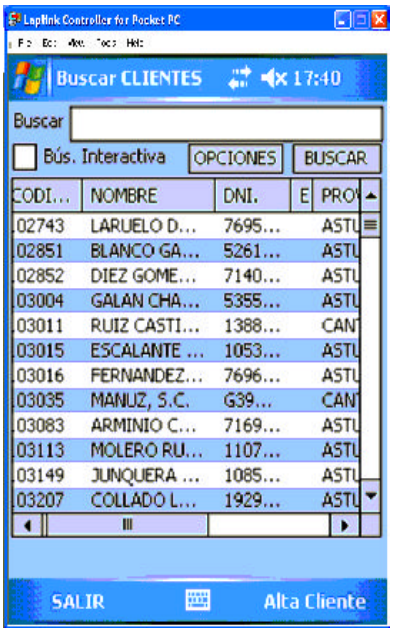

Recorrido Diario 16:51 ok
Recorrido Diario
Lun
Mar
Mie
Jue
Vie
Sab
Dom
Cancelar


Recorrido Diario 16:55 ok
Vista Previa
Fecha: 10/04/2007
Hora: 16:54
RECORRIDO DIARIO
Ruta: 002 AGUSTIN E.G.
Cliente
Telefono
t (100)
(101) 650/916600
(102) 96-2872626
(103) 96-2042895
CARNICERIA LAVAL (104) 96-2861312
(105) 96/2309026
(106) 666-057050
1 2 3 4
Listar
letra:
Salir


Listado de Stock 16:52 ok
Desde
Hasta
Grupo
Principio
Fin
SubGrupo
Principio
Fin
Articulo
Principio
Fin
Articulos Sin Stock
Listar
Vista Previa
Salir

OPCION PEDIDOS TEXTIL

Como ejemplo de lo que sería la implementación de una versión de la Aplicación para otro tipo de Empresas, ya está en funcionamiento una versión de PEDIDOS para una empresa de tipo Textil.
Pondré a modo de ejemplo alguna pantalla de las que son diferentes de la versión Standar en lo que es la Captura de un Pedido de Artículo Textil.



Al pulsar el botón Pedidos va a seleccionar un cliente, y después a seleccionar el muestrario del que se le va a realizar el Pedido.
A continuación se presenta la pantalla de captura en la que ya no sería necesario el teclado Numérico Virtual, pues se introducen los números por los símbolos de incremento y decremento numérico.
Se selecciona una Familia, después un modelo de esa familia, y a continuación se selecciona el color y la talla, todo mediante combobox. Se puede realizar selección múltiples de todos los objetos y/o todas las tallas para agilizar la captura.