Preventa

Eduardo Ortiz Pérez del Molino
ETIG

Consultor: Jordi Ceballos Villach

9/01/2006

Resumen del proyecto

En el sector de la distribución se denomina “Preventa” a la labor de visita de clientes y recogida de pedidos que realizan los comerciales. Una vez recogidos los pedidos se transmiten a la central donde se cargan los camiones que posteriormente entregarán la mercancía. Una actividad similar a esta es la “Autoventa ” donde es el mismo camión de reparto el que va cliente por cliente tomando los pedidos y entregando la mercancía al mismo tiempo.

Se trata de una aplicación de preventa para terminales con Sistema Operativo Windows CE 4.2 y plataforma Pocket PC 2003. La aplicación permitirá al terminal conectarse con un ordenador central (PC) para descargar la información de los clientes, artículos, rutas, etc. Una vez haya recibido la información, el comercial podrá ir visitando los diferentes clientes siguiendo una ruta e introducir los pedidos. En cualquier momento del día el comercial podrá enviar los pedidos a la central mediante una conexión remota o bien mediante una comunicación directa.

Este proyecto no incluye la aplicación de gestión del PC ya que el objetivo es que sea lo más genérica posible y pueda enlazar con cualquier aplicación de gestión de PC. Tanto para el envío de información del PC al terminal como del terminal al PC se utilizarán ficheros de texto.

Área del TFC: Tecnología .NET

Palabras clave: Gestión; Windows CE; Pocket PC; .Net; C#; Smart Device; Compact Framework.

Indice

Indice de contenidos

Indice de imágenes

1- Introducción

1.1- Justificación del TFC

A principios de la década de los 80 aparecieron los primeros terminales de mano. Casi desde el principio se emplearon con aplicaciones de captura de datos incluidas las aplicaciones de preventa (toma de pedidos). Con el tiempo los terminales fueron evolucionando aumentando en capacidad de proceso, memoria, capacidad gráfica y periféricos.

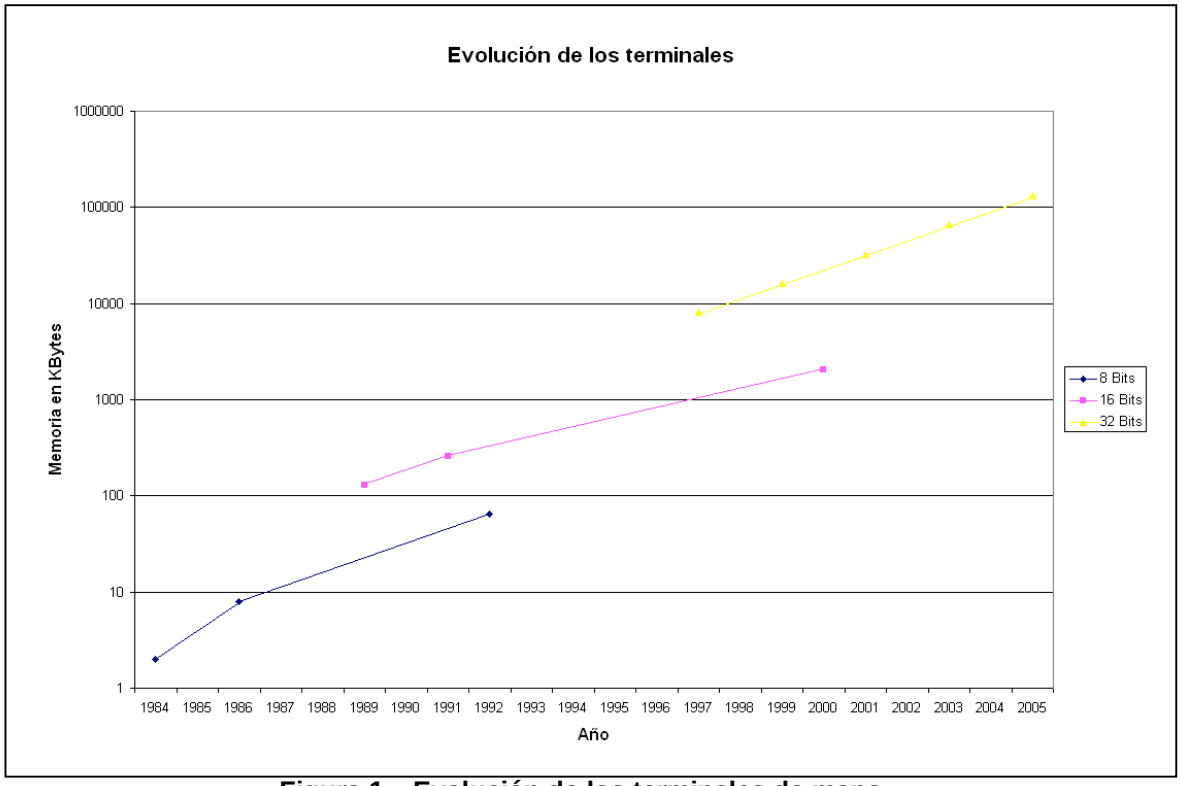

Figura 1 – Evolución de los terminales de mano

Los primeros terminales de 8 bits no tenían un sistema operativo propiamente dicho y se programaban en BASIC y en ensamblador. Con la aparición de los terminales de 16 bits se emplearon SO a medida (EPOC 16) o se adaptaron SO (MS-DOS embebido). Los terminales de 16 bits perviven todavía pero están en franco retroceso en favor de los terminales de 32 bits. Los terminales de 32 bits aparecieron a finales de la década de los 90 con SO hechos a medida (EPOC 32, Palm y Windows CE). En la actualidad los terminales corporativos en su mayoría disponen de 32M de RAM, pantalla gráfica y Windows CE en sus dos plataformas: Windows CE y Pocket PC.

Las herramientas de desarrollo también han ido evolucionando desde las primeras en BASIC, las siguientes en eVB y eVC++ y las actuales en .Net (VB.Net y C#). Cabe destacar que la aparición del VS 2003 ha supuesto una

auténtica revolución ya que multiplica por 5 el rendimiento frente al eVB mientras que el C# aunque no es tan eficiente como el C++, es más sencillo de programar. La pieza clave que hace esto posible es el Compact Framework, integrado por un JIT, un recolector de basura y las librerías de classes.

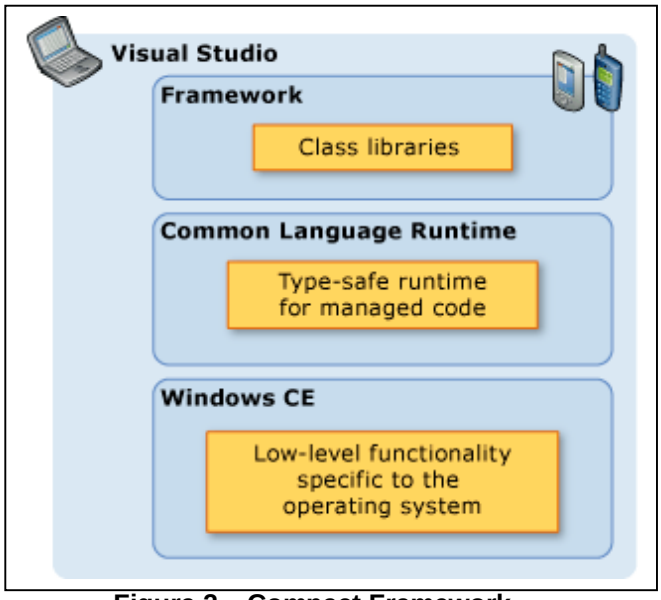

Figura 2 – Compact Framework

Entre los aspectos de los que nos podemos aprovechar son las mayores posibilidades del hardware de los terminales actuales: CPU más potentes, más memoria y más pantalla (1/4 VGA) todavía lejos de las de un PC pero muy superiores a la de los terminales de hace unos pocos años. Otro aspecto muy destacado es la multitud de posibilidades de conexión: serie, USB, Bluetooth, GSM, GPRS, UMTS, IrDA. Pero el aspecto más destacable es el cambio en las herramientas y la plataforma de desarrollo.

Hasta la aparición del VS 2003 apenas había posibilidad de desarrollar una aplicación con programación orientada a objetos y de utilizar un servidor SQL. En la actualidad empieza a haber unas pocas aplicaciones de preventa desarrolladas con el VS y en muchos casos tras un proceso de reconversión de los programadores de lenguajes más tradicionales a VB.Net y C#.

Puesto que es necesario una reconversión de las aplicaciones anteriores de preventa, convendría hacerlo con el menor esfuerzo (coste) posible. Si en vez de partir de 0 se dispusiera de una aplicación de preventa de código abierto las diferentes empresas de software podrían ahorrar costes y reducir los tiempos de desarrollo. En el mejor de los casos las empresas podrían aprovechar todo el proyecto y en el peor de los casos podrían aprovechar alguna classe, comparar el código o sacar alguna idea.

Para terminar, una aplicación de gestión nos permite estudiar la productividad utilizando técnicas clásicas. Podemos analizar cuanto cuesta desarrollar una

aplicación de este tipo y de esta forma estimar el coste de futuras aplicaciones. Esto nos puede proporcionar unos datos muy valiosos ya que al tratarse de una herramienta nueva aplicada a una plataforma distinta al PC apenas hay datos.

1.2- Objetivos

Una aplicación de preventa permite a las empresas reducir los errores en la recogida de los pedidos, reducir costes de mano de obra al no tener que volverlos a introducir en la oficina y sobre todo acortar los tiempos en la recepción de los pedidos.

El objetivo es disponer de una aplicación de preventa completamente operativa. Una vez terminada los programadores dispondrán no solo de un código que ellos pueden modificar sino de un desarrollo completo con su análisis y su diseño. A partir de ahí podrán modificar o ampliar cualquier aspecto de la aplicación. No es por tanto tan importante que la aplicación satisfaga todas las posibilidades, y si que pueda crecer o modificarse a voluntad.

Esta aplicación al ser de código abierto permitirá a los desarrolladores contar con un código que podrán adaptar a sus necesidades ampliando los aspectos que consideren oportunos. Esto reducirá de forma considerable los tiempos de aprendizaje de la herramienta de desarrollo y del lenguaje, así como los tiempos de desarrollo ya que contaran con una aplicación completa.

Al ser una aplicación de código abierto hay que tener en cuenta algunos aspectos de la programación:

- El código ha de estar bien documentado.
- El código ha de ser muy modular y fácil de modificar y ampliar.
- En la medida de lo posible se ha de fomentar la reutilización de las clases más genéricas.
- Se utilizarán únicamente las classes del VS (Compact Framework) y classes de código abierto.
- La aplicación ha de ser lo más genérica posible.
- Si bien es importante la fiabilidad de la aplicación, se adoptará un criterio más relajado en favor de la sencillez y claridad del código.

Por último se ha de calcular el coste de la aplicación en horas de trabajo con el fin de que los programadores, analistas y jefes de proyecto puedan estimar el coste de sus propios proyectos. Aunque los resultados aquí obtenidos no serán en ningún caso concluyentes pueden servir para hacer una primera aproximación de costes.

1.3- Enfoque y método seguido

El desarrollo está enfocado en obtener una aplicación de preventa totalmente operativa en el sentido que la toma de pedidos sea real, las comunicaciones sean reales y la impresión también. Sin embargo se trata más bien de una aplicación para que los programadores puedan trabajar sobre ella, experimentar con el código, etc. Por ese motivo los títulos de las pantallas son el nombre de la classe (ScrInicial, ScrPrincipal, etc) y la aplicación no está en modo kiosco y así permitir a los usuarios al acceso al resto del sistema.

En el desarrollo se ha utilizado el método clásico de: Análisis, Diseño, Implementación y Depuración. También se han seguido los pasos marcados por el tutor del proyecto por lo que a demás se ha efectuado una planificación y se ha desarrollado un prototipo.

- En primer lugar se ha efectuado una planificación de todo el proyecto.
- En segundo lugar se ha hecho el análisis. La pieza fundamental del análisis ha sido los casos de uso que han servido de punto de partida para elaborar los diagramas de colaboración en la etapa de diseño. A demás de los casos de uso se elaboraron una primera versión de un prototipo con las pantallas de la aplicación.
- En tercer lugar se ha hecho el diseño. Para hacer el diseño se han desarrollo los diagramas de colaboración a partir de los casos de uso de la etapa de análisis. A demás se ha diseñado la BD de la aplicación partiendo de un esquema ER. Por último se ha terminado el prototipo.
- En cuarto lugar se ha implementado la aplicación. Aquí se ha seguido el modelo vista controlador, estructurando todo la aplicación entorno a las classes de control y siguiendo los diagramas de colaboración.
- En quinto lugar se ha depurado el código.
- En último lugar se ha escrito esta memoria y se ha hecho la presentación.

1.4- Planificación del proyecto

El desarrollo del proyecto sigue el calendario marcado por el director del trabajo y cumple el ciclo clásico de desarrollo: Planificación, Análisis, Diseño e Implementación. A continuación indicamos el calendario de entregas y una pequeña descripción de cada una:

Fecha de entrega	Trabajo
26/09/2005	Plan de trabajo
17/10/2005	Análisis y prototipo
07/11/2005	Diseño
09/12/2005	Implementación
09/01/2006	Memoria y presentación Powerpoint

- **Plan de trabajo:** Este mismo documento donde se indica la descripción del proyecto, los objetivos, los requisitos y la planificación.

- **Análisis y prototipo:** El análisis de la aplicación y un primer prototipo con las pantallas de la aplicación.

- **Diseño:** El diseño completo de la aplicación y el prototipo definitivo de la interface gráfica de la aplicación.

- **Implementación:** La aplicación completa y terminada.

- **Memoria y presentación Powerpoint:** La memoria de todo el proyecto y una presentación virtual del proyecto en Powerpoint.

	Nombre de tarea	Duración	Comienzo	Fin
1	**Plan de Trabajo**	**10 días**	**sáb 17/09/05**	**lun 26/09/05**
2	**Análisis y prototipo**	**21 días**	**mar 27/09/05**	**lun 17/10/05**
3	Análisis	13 días	mar 27/09/05	dom 09/10/05
4	Diseño de pantallas	8 días	lun 10/10/05	lun 17/10/05
5	**Diseño e Interface gráfica**	**21 días**	**mar 18/10/05**	**lun 07/11/05**
6	Diseño BD	2 días	mar 18/10/05	mié 19/10/05
7	Diseño entidades	2 días	jue 20/10/05	vie 21/10/05
8	Diseño Gestión Usuarios	1 día	sáb 22/10/05	sáb 22/10/05
9	Diseño Comunicación	1 día	dom 23/10/05	dom 23/10/05
10	Diseño Selección Clientes	2 días	lun 24/10/05	mar 25/10/05
11	Diseño Gestión Pedidos	3 días	mié 26/10/05	vie 28/10/05
12	Diseño Configurarión	2 días	sáb 29/10/05	dom 30/10/05
13	Documentación	8 días	lun 31/10/05	lun 07/11/05
14	**Implementación**	**32 días**	**mar 08/11/05**	**vie 09/12/05**
15	Implementación Gestión Usuarios	3 días	mar 08/11/05	jue 10/11/05
16	Implementación Comunicación	6 días	vie 11/11/05	mié 16/11/05
17	Implementación Selección Clientes	2 días	jue 17/11/05	vie 18/11/05
18	Implementación Gestión Pedidos	10 días	sáb 19/11/05	lun 28/11/05
19	Implementación Configurarión	4 días	mar 29/11/05	vie 02/12/05
20	Revisión y documentación	7 días	sáb 03/12/05	vie 09/12/05
21	**Memoria y PPT**	**31 días**	**sáb 10/12/05**	**lun 09/01/06**
22	Memoria	20 días	sáb 10/12/05	jue 29/12/05
23	Presentación PowerPoint	11 días	vie 30/12/05	lun 09/01/06

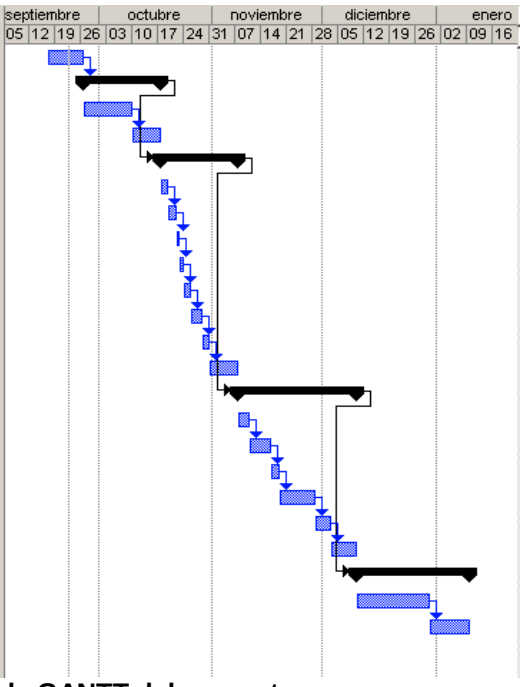

Figura 3 - Diagrama de GANTT del proyecto

1.5- Productos obtenidos

Durante el desarrollo del proyecto se han obtenido los siguientes productos:

- **Plan de Trabajo**: Descripción breve del proyecto y planificación temporal.
- **Análisis**: Contiene una descripción de la aplicación y sus requerimientos, la especificación de las funcionalidades mediante casos de uso, una primera versión de las pantallas de la aplicación.
- **Diseño**: Contiene una descripción de como se va a implementar la aplicación, el desarrollo de los casos de uso mediante diagramas de colaboración y el diseño de la BD.
- **Prototipo**: Una primera versión de la aplicación que muestra las pantallas pero sin datos reales.
- **Implementación**: La aplicación para Pocket PC.
- **Manual de instalación**: La instalación de la aplicación y su configuración.
- **Memoria del proyecto**: Este documento.
- **Presentación del proyecto**: Presentación en PowerPoint de los puntos más destacados del proyecto.

1.6- Descripción de los capítulos siguientes

En los siguientes capítulos veremos:

- **Arquitectura de la aplicación**: Explicamos la arquitectura de la aplicación tanto externa como interna, las classes de la que está compuesta, un pequeño repaso a las librerías utilizadas y la BD.

- **Descripción de la aplicación**: Describimos la aplicación desde un punto de vista funcional, repasando los requerimientos; y a continuación revisamos la aplicación desde el punto de vista del usuario.

- **Valoración económica**: Hacemos un repaso al coste de la aplicación en horas de desarrollo y un estudio de sus puntos de función. Analizamos con un pequeño ejemplo como se podría estimar el coste de una pequeña ampliación.

- **Conclusiones**: Se repasa todo el desarrollo desde un punto de vista crítico, estudiando que ha ido bien y que se puede mejorar del proceso de desarrollo. También se describen posibles mejoras y ampliaciones.

2- Arquitectura de la aplicación

En una empresa de distribución lo habitual es que exista un programa de gestión en la oficina central con diferentes ordenadores conectados en red donde los comerciales ejecutan la aplicación de gestión. El problema surge cuando los comerciales deben visitar a los clientes y tomar los pedidos. Aunque se puede utilizar portátiles conectados mediante GSM/GPRS/UMTS la cobertura de las líneas no siempre es posible. En consecuencia es necesario utilizar una aplicación que pueda funcionar de forma autónoma y de forma regular se conecte con la oficina para intercambiar los datos.

La aplicación de preventa está diseñada para funcionar de forma totalmente autónoma e independientemente del programa de gestión. Al ser una aplicación genérica que puede intercambiar información con diferentes sistemas de gestión se ha optado por utilizar formatos estándar de ficheros de intercambio (ficheros de texto y/o ficheros XML) y protocolos estándar (FTP).

En consecuencia se ha diseñado una aplicación Windows para plataformas Pocket PC que utiliza una BD SQL Server CE 2.0. La aplicación se conecta a un servidor FTP del que recibe los ficheros y al que envía los pedidos. En cierto modo se puede considerar a la aplicación de Preventa un cliente de la aplicación de gestión del PC.

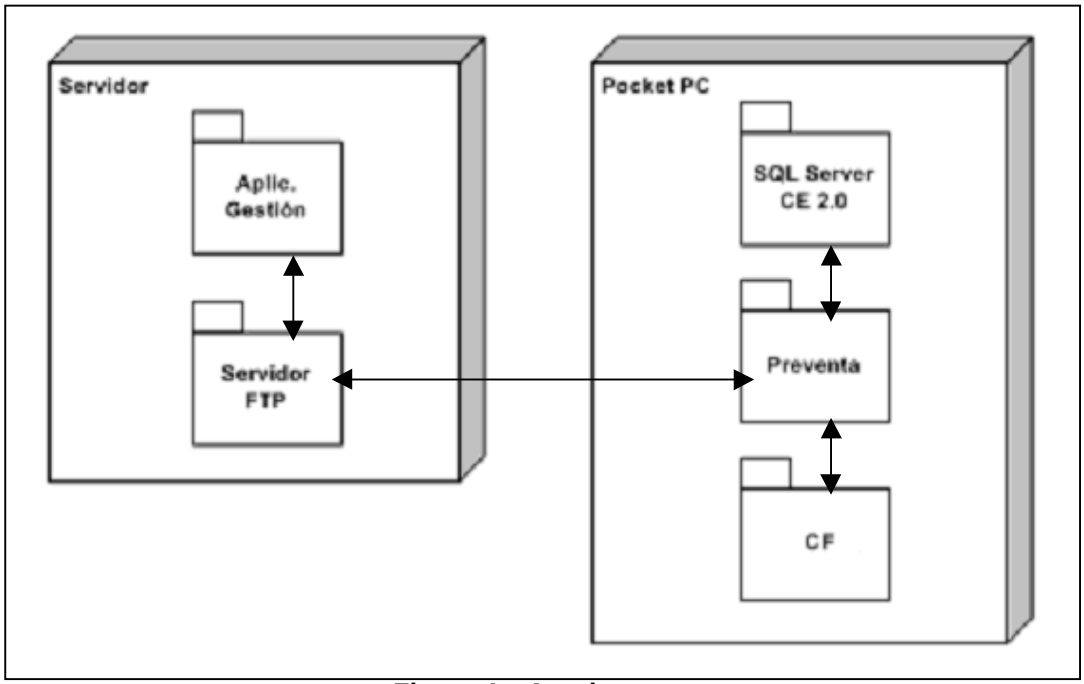

Figura 4 – Arquitectura externa

2.1 Arquitectura interna

Internamente la aplicación está formada por diferentes subsistemas. Dentro de cada subsistema se implementa uno o varios casos de uso. Cada subsistema está controlado por una classe de control que se encarga de la comunicación con los otros subsistemas y classes de control, así como de la creación de classes frontera. La única excepción es el subsistema de gestión de pedidos que debido a su complejidad se han utilizado 3 classes de control. A demás se ha utilizado una classe de control adicional para la gestión de la BD.

Con el fin de hacer la aplicación fácilmente transportable a otras plataformas se ha intentado reducir el código de las classes frontera al mínimo en favor de las classes de control. Al mismo tiempo se ha intentado hacer las entidades lo más independiente del resto de la aplicación con el fin de facilitar futuras ampliaciones y especializaciones de la aplicación. Todo ello hace que la aplicación siga el patrón Modelo-Vista-Controlador y por ende facilita todo el diseño.

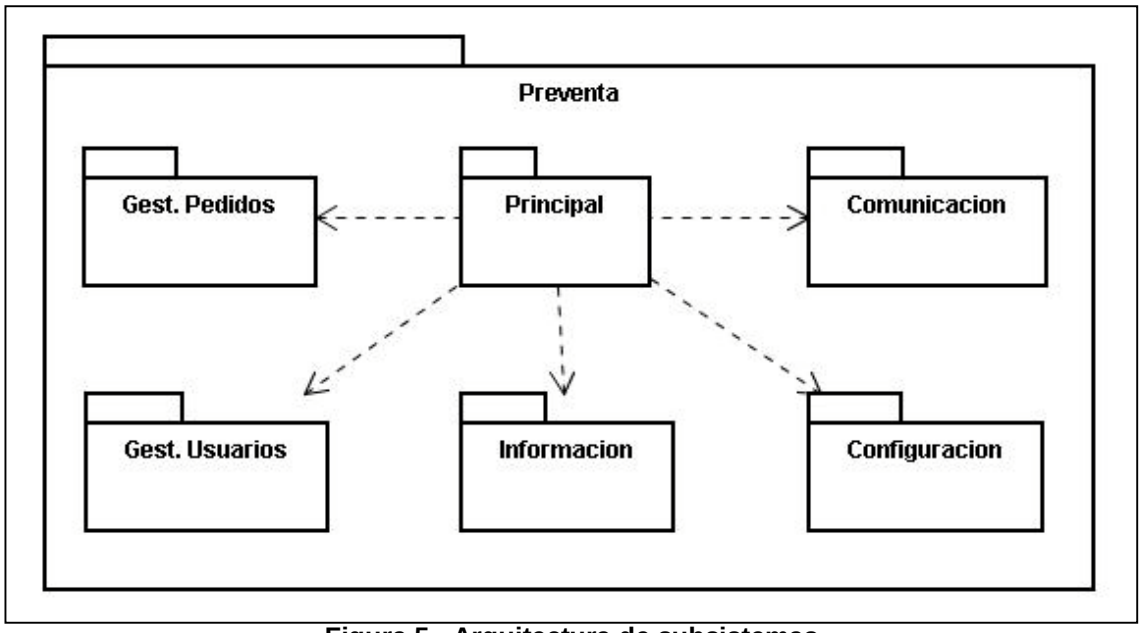

Figura 5 - Arquitectura de subsistemas

- **Subsistema principal:** Es el encargado de iniciar y terminar la aplicación así como de gestionar la pantalla principal desde la cual los usuarios accederán a las diferentes funcionalidades. Hay dos perfiles de usuario en función de los cuales la aplicación mostrará o no diferentes las diferentes opciones.

- **Subsistema de gestión de pedidos:** Es el encargado de la creación y edición de los pedidos. Comprende junto con el subsistema de comunicación las funcionalidades más importantes de la aplicación.

- **Subsistema de gestión de usuarios:** Permite el cambio de usuario y a los administradores permite crear, eliminar y editar a los usuarios.

- **Subsistema de comunicación:** Es el encargado de recibir los clientes, artículos, rutas, etc.; y de transmitir los pedidos. Tanto la información que se recibe como la que se envía, se hace a través de ficheros de texto o XML por lo que es necesario un proceso de importación y exportación a la BD interna.

- **Subsistema de configuración:** Permite ajustar diferentes parámetros de la aplicación. Al tratarse de una aplicación genérica que se puede aplicar en diferentes empresas y con diferentes configuraciones de hardware, es necesario hacerla lo más flexible posible. Los principales ajustes serán los relacionados con la interfaz gráfica, la comunicación y la impresión.

- **Subsistema de información:** Proporciona información al usuario acerca de los pedidos así como del estado del terminal (memoria, baterías, etc.).

2.2 Classes de la aplicación

Al tratarse de POO y estar desarrollada la aplicación en C#, son las classes las piezas fundamentales que nos permiten construirla. En este apartado hacemos un repaso agrupándolas por su tipo.

2.2.1 Classes de control

Las classes de control constituyen la columna vertebral de la aplicación. Salvo CtrlDB que se encarga del acceso a la BD, las classes de control se encargan de coordinar cada uno de los subsistemas de la aplicación. De hecho tienen muy poco código ya que la mayor parte del trabajo consiste en abrir ventanas y pasar las consultas de la BD a CtrlDB. En el diagrama he incluido ScrInicial ya que es la que inicia la aplicación.

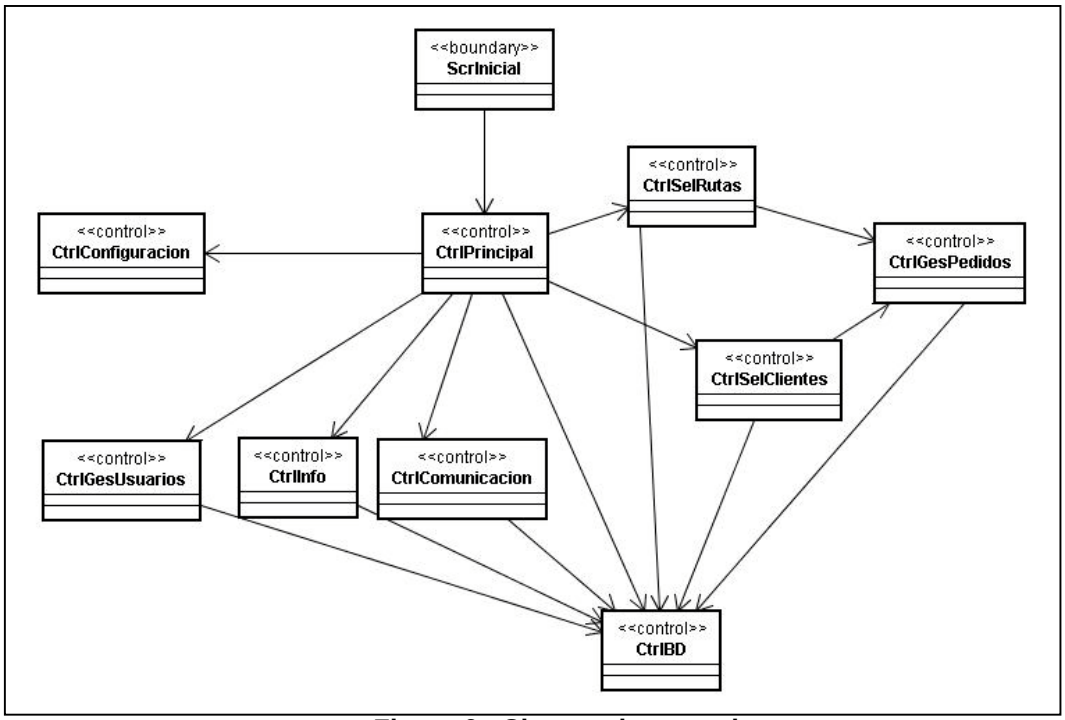

Figura 6 - Classes de control

CtrlDB implementa toda la manipulación de la BD: creación, consultas y manipulación de los datos de las diferentes tablas. Cuando el resultado de una consulta es una colección de registros el método devuelve los datos en forma de DataReader o de DataSet. Por el contrario cuando únicamente se quiere un registro, el resultado será una entidad.

2.2.2 Classes frontera

Las classes de frontera son las pantallas de la aplicación. El VS sólo tiene una classe de tipo ventana para el CF por lo que todas las pantallas descienden de ella. Una característica de la plataforma Pocket PC es que las ventanas siempre aparecen maximizadas. Únicamente las ventanas sin marco pueden ser más pequeñas por que lo cual algunos diálogos se han diseñado así.

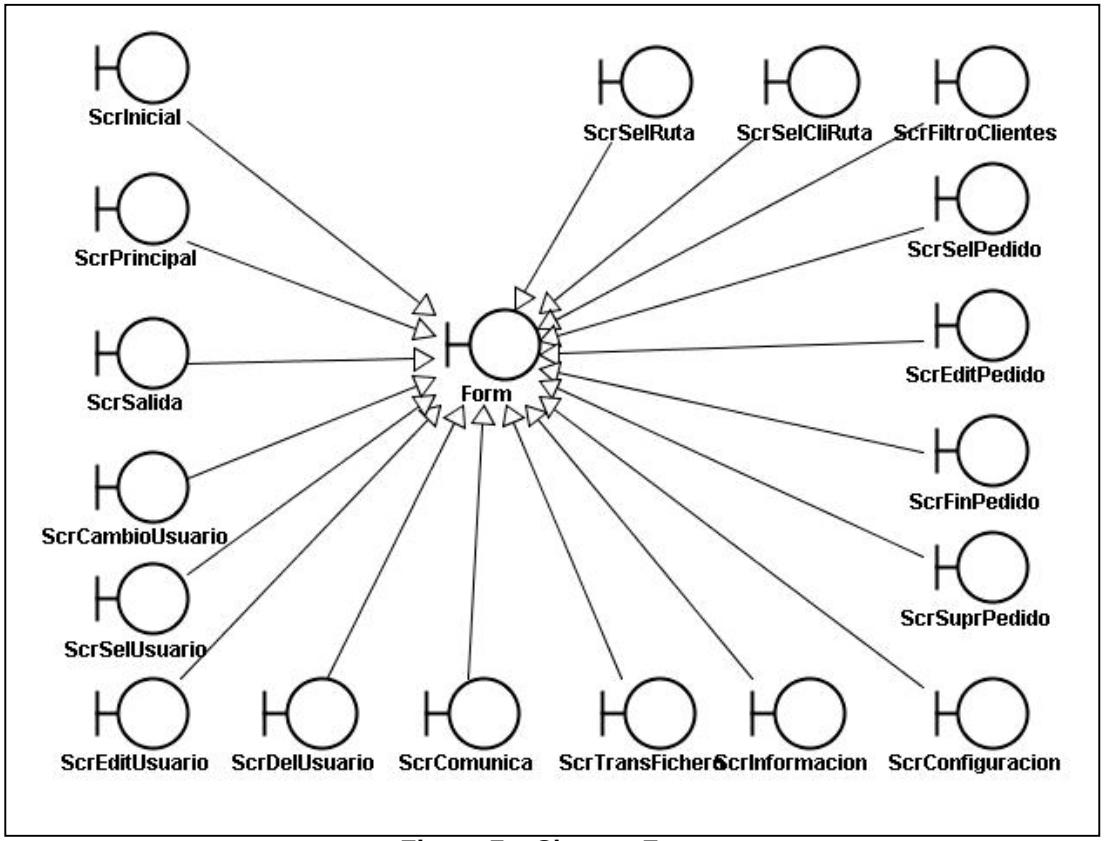

Figura 7 – Classes Frontera

2.2.3 Entidades

Las entidades representan los datos que utiliza la aplicación. En realidad hay más entidades que las que se han diseñado, lo que ocurre es que algunas de ellas se obtendrán como resultado de consultas a la BD y se guardarán en DataReader o DataSet. De forma similar las relaciones entre las diferentes entidades vienen representadas por un identificador, un campo clave de la tabla correspondiente, en vez de utilizar una referencia a un objeto. Las

relaciones de las diferentes entidades se visualizan mucho mejor viendo el esquema ER y las relaciones entre las diferentes tablas.

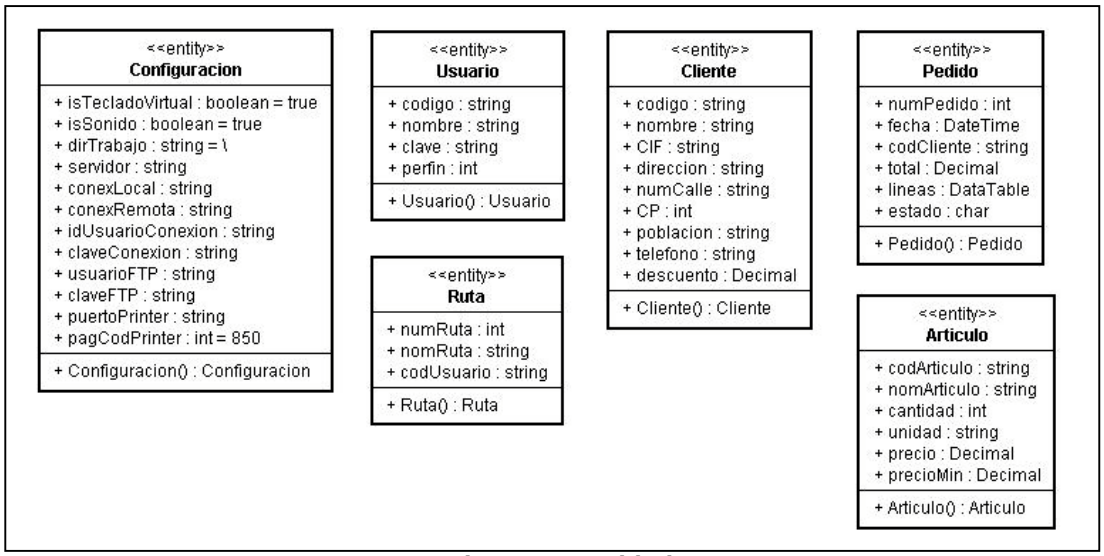

Figura 8 - Entidades

En Pedido tenemos dos propiedades que hay que comentar. Lineas es una propiedad de tipo DataTable que contiene las líneas del pedido. Cada línea contiene a demás de los campos de la tabla LinPed, la descripción del artículo, las tarifas, etc. La propiedad estado indica el estado del pedido. Un pedido cuando se crea está abierto y se queda así hasta que el comercial indica que el pedido se ha finalizado, pasando al estado Cerrado. Un pedido cerrado puede volverse a abrir. Cuando el terminal se conecta con el servidor, la aplicación envía los pedidos cerrados únicamente, si un pedido está abierto no se envía. Cuando un pedido cerrado es enviado al servidor pasa al estado enviado. Un pedido enviado no puede modificarse ni reabrirse pero si se puede consultar.

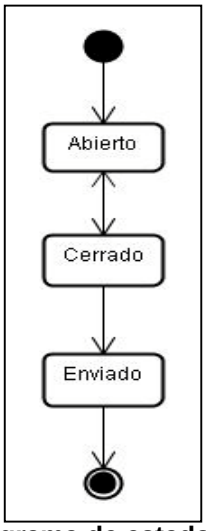

Figura 9 - Diagrama de estado de Pedidos

2.2.4 La librería estándar de Compact Framework

Los terminales Windows CE utilizan una versión reducida de la Framework del PC, la Compact Framework 1.0. La librería también se ha reducido tanto en número de classes como en las propiedades y métodos de las classes (esto obliga a veces a hacer auténticas piruetas ya que en algunos casos se le ha ido un poco la mano y faltan elementos totalmente necesarios). Todas las classes de la aplicación heredan o contienen alguna de estas classes. Únicamente citaremos algunas de las classes más importantes:

Entre los tipos básicos: int, decimal, string, DateTime.
Entre los objetos y controles gráficos: Form, Button, TextBox, DataGrid, etc.
Entre los objetos de BD: SqlCeConnection, SqlCeCommand, SqlCeDataReader, etc.

2.2.5 La librería Util

Por último en el desarrollo de la aplicación se han utilizado una serie de classes que forman parte de la aplicación pero que se han desarrollado a parte de ella. Estos elementos están agrupados en el espacio de nombres Util. Algunos de ellos provienen de la Smart Device Framework 1.4 (www.opennetcf.org) y otros han sido desarrollados por el autor en su trabajo. Hacemos una pequeña relación de las classes más importantes:

Senkeys: Permite enviar teclas al buffer del teclado.
Registry: Permite acceder al registro de Windows.
Dial: Permite efectuar conexiones y llamadas telefónicas.
Dispositivos: Permite trabajar con los puertos serie y Bluetooth.
Print: Permite imprimir texto en pequeñas impresoras.
clsFTP: Implementa un cliente FTP.

2.3 Diseño de la BD

La BD de la aplicación de Preventa se ha diseñado atendiendo a sus necesidades, intentando mantener una sencillez de diseño.

2.3.1 Diagrama ER

Cuatro son las entidades fundamentales que estructuran la BD: los Usuarios de la aplicación, los Clientes, los Articulos y los Pedidos. A demás hay tablas diseñadas para facilitar al usuario su trabajo: Rutas, PersonaContacto, Familia, ArtFrec. Por último tenemos un tercer grupo de tablas que son entidades débiles y que completan la información de la entidad principal: LinRutas y Rutas, LinPed y Pedidos, Tarifas y Articulos.

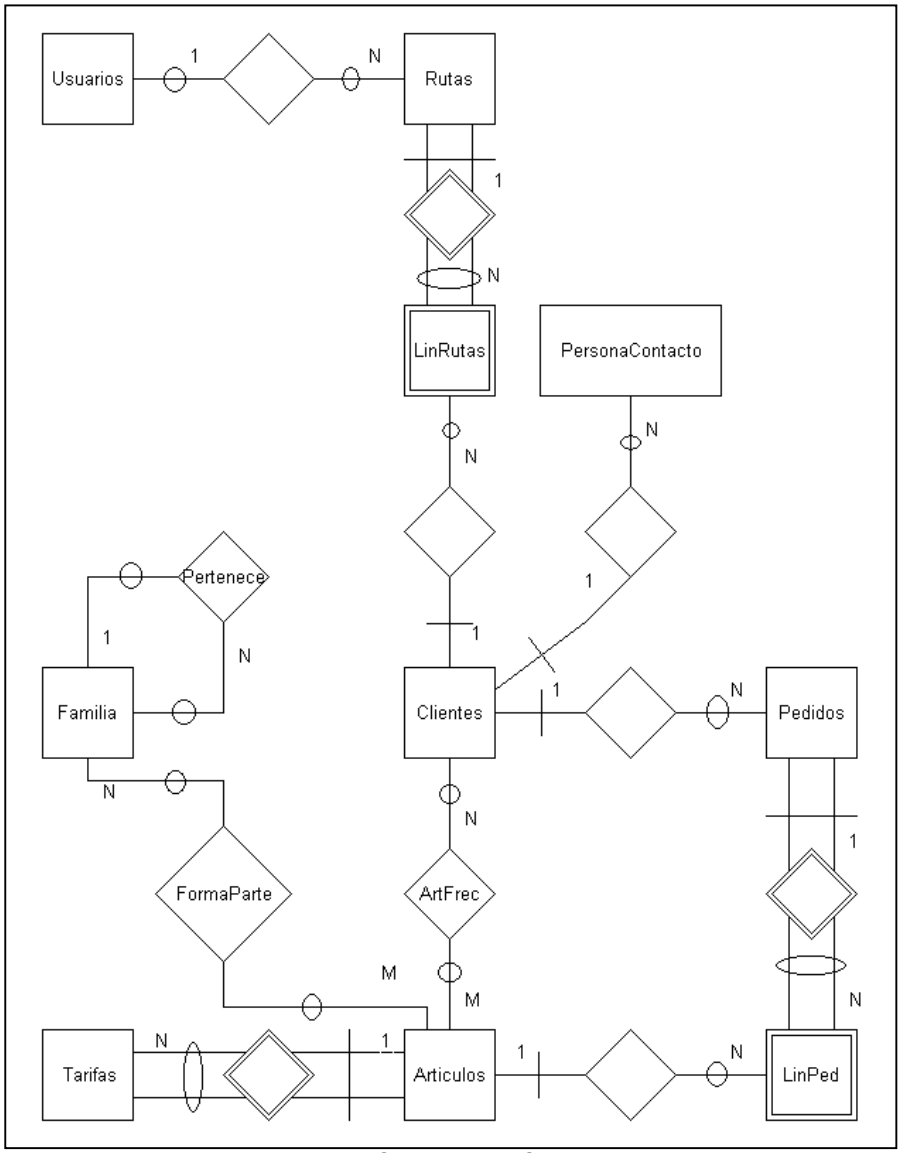

Figura 10 - Diagrama ER

Hay que hacer algunas puntualizaciones:

- Los usuarios pueden tener asignadas rutas o no. Las rutas asignadas a un usuario solamente las podrá utilizar él, mientras las que no estén asignadas a ninguno las podrá realizar cualquiera.
- Una ruta podría no tener ningún cliente aunque no tendría mucho sentido. Se podría haber indicado la relación entre Rutas y Clientes como tal pero de esta forma se recalca que las líneas de las rutas forman parte de las rutas.
- Un cliente puede tener o no personas de contacto.
- Las familias pueden contener artículos y otras familias. Estas a su vez pueden otras familias y así sucesivamente. La estructura que forman las familias y subfamilias es un árbol aunque no hay nada que impida en la BD que la estructura sea un grafo.
- Un cliente puede tener unos artículos que consuma de forma habitual. En ese caso cuando se crea un nuevo pedido se incluyen en el pedido con el fin que el usuario deba introducir únicamente la cantidad.
- Puede haber pedidos sin líneas de pedido.
- Aunque en el esquema ER un artículo puede tener o no una tarifa, siempre debe tener al menos una. Las tarifas tienen una fecha a partir de la cual entran en vigor y dejan de estar vigentes cuando otra tarifa las sustituye. De esta forma las tarifas se pueden cargar en la BD del terminal con antelación. La implementación de la aplicación obliga que las tarifas de todos los artículos se actualicen al mismo tiempo.

2.3.2 Diagrama lógico

Podemos ver la BD con más detalle con un siguiente diagrama donde están representadas las tablas y las relaciones entre estas, pero a demás los campos de cada tabla. La relación recursiva entre familias no queda reflejada debido a una limitación de la herramienta utilizada para hacer el diagrama.

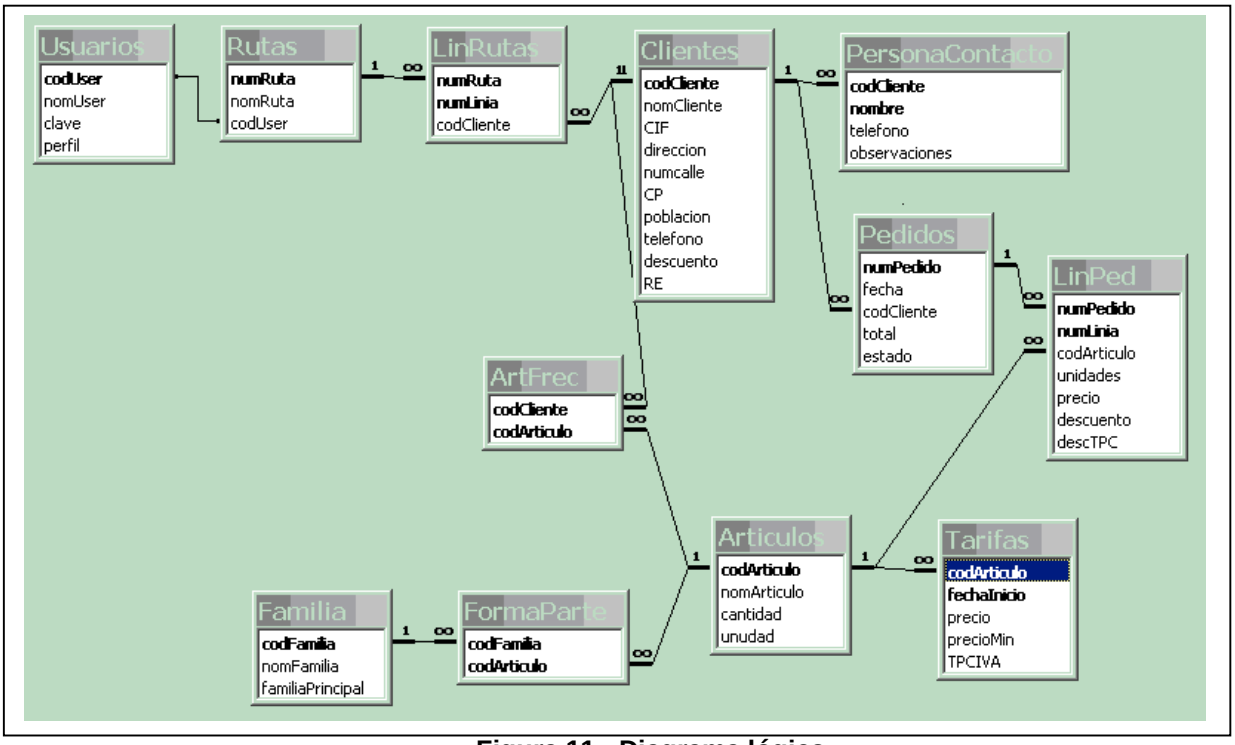

Figura 11 - Diagrama lógico

3- Descripción de la aplicación

En este capítulo describimos la aplicación analizando sus requerimientos para pasar a describir su funcionamiento desde el punto de vista del usuario.

3.1 Análisis funcional

En el sector de la distribución se denomina “Preventa” a la labor de visita de clientes y recogida de pedidos que realizan los comerciales. Una vez recogidos los pedidos se transmiten a la central donde se cargan los camiones que posteriormente entregarán la mercancía. Una actividad similar a esta es la “Autoventa ” donde es el mismo camión de reparto el que va cliente por cliente tomando los pedidos y entregando la mercancía al mismo tiempo.

Nuestra aplicación es una aplicación de preventa para terminales con Sistema Operativo Windows CE 4.2 y plataforma Pocket PC 2003. La aplicación permitirá al terminal conectarse con un ordenador central (PC) para descargar la información de los clientes, artículos, rutas, etc. Una vez haya recibido la información, el comercial podrá ir visitando los diferentes clientes siguiendo una ruta e introduciendo los pedidos. En cualquier momento del día el comercial podrá enviar los pedidos a la central mediante una conexión remota o bien mediante una comunicación directa.

Habrá dos tipos de usuarios: Los comerciales que introducirán los pedidos y los administradores que se encargarán de configurar la aplicación. A los comerciales se les presupone escasos conocimientos informáticos por lo cual la aplicación ha de tener una interfaz sencilla y los mensajes han de ser poco técnicos y claros. La aplicación tendrá opciones accesibles únicamente a los administradores donde los mensajes podrán ser más técnicos.

La primera vez que se ponga en marcha la aplicación el administrador configurará la interfaz, la comunicación y la impresión. También habrá de dar de alta a los usuarios del terminal. La aplicación creará por defecto un usuario administrador con el fin de que se pueda configurar.

Cuando un usuario utilice el terminal habrá de introducir su identificador y su clave. Una vez le aparezca la pantalla principal podrá acceder a las diferentes opciones. Inicialmente no habrá ningún dato por lo cual lo primero que ha de hacer es conectarse con el servidor con el fin de recibir las rutas, los clientes, los artículos, etc. Una vez reciba la información podrá trabajar normalmente.

La forma habitual de un comercial será la de elegir una ruta e ir visitando a los clientes de la ruta. Si fuese necesario el comercial podrá buscar un cliente fuera de ruta. Para cada cliente la aplicación mostrará sus datos y las personas de contacto. También podrá ver los pedidos que haya hecho anteriormente que

estén en el terminal, y podrá crear nuevos pedidos. Al crearse un nuevo pedido la aplicación añadirá a este los artículos frecuentes, esto es, los artículos que el cliente consume de forma habitual. De esta forma el comercial únicamente tendrá que introducir las cantidades.

Cuando el comercial esté editando un pedido deberá introducir el código de un artículo y la cantidad. La aplicación obtendrá los datos correspondientes al artículo y realizará los cálculos oportunos. Cada artículo tendrá un precio que será el resultado de calcular el PVP menos el descuento del cliente. El comercial podrá modificar el precio pero nunca podrá rebajarlo por debajo del precio mínimo del artículo.

Dentro de la edición de un pedido, el usuario podrá buscar un artículo por diferentes criterios (código, descripción, etc.). Los artículos podrán estar agrupados por familias con el fin que los usuarios puedan buscar los artículos dentro de las familias. Una familia puede contener otras familias y diferentes artículos. Un artículo puede pertenecer a una familia, a varias o a ninguna.

Al terminar la edición de un pedido el comercial podrá ver el importe total e imprimir un ticket. Un pedido puede está abierto mientras se está editando y cuando terminamos se puede dejar abierto o cerrarlo. Un pedido abierto es un pedido que no se ha terminado, que se ha suspendido su edición para retomarla posteriormente. Únicamente se enviaran al servidor los pedidos cerrados. Los pedidos cerrados cuando se envíen al servidor pasan a ser pedidos “enviados”, que se pueden editar pero que no se pueden modificar.

En cualquier momento del día el usuario puede enviar los pedidos al servidor incluso lo puede hacer varias veces al día. Normalmente enviará los pedidos cerrados pero en ocasiones especiales se podrá enviar los pedidos enviados anteriormente (por ejemplo sí ha habido algún problema de comunicación o en la incorporación de datos).

Los usuarios podrán obtener información y estadísticas acerca de los pedidos. También información del sistema acerca de la memoria y la batería que son puntos a tener en cuenta en los terminales.

3.2 Descripción funcional

En este apartado haremos un rápido repaso al funcionamiento de la aplicación desde el punto de vista de un usuario (comercial). La instalación y las funcionalidades de configuración propias del administrador se repasan en el Manual de Instalación. Seguiremos el orden habitual de trabajo: Entrada en la aplicación, Comunicación, Rutas, Clientes, Información.

3.2.1 Inicio (Entrada en la aplicación)

El inicio de la aplicación es muy sencillo, basta con introducir el código de usuario, la clave y pulsar el botón Aceptar. En este caso entramos Código:2 y Clave:2.

Figura 12 – Pantalla de identificación

Si los datos son correctos la aplicación nos muestra la pantalla principal. En caso de error, muestra el mensaje correspondiente y hay que volver a identificarse. La pantalla principal se ajusta al perfil del usuario, en este caso un comercial (no se muestran Gestión de Usuarios, Opciones y Salir).

Figura 13 – Pantalla principal

3.2.2 Comunicación

La comunicación nos permite conectarnos con el ordenador central y recibir los datos: Rutas, Clientes, Artículos, etc. La misma opción nos permite enviar los pedidos del terminal al PC. Todo funciona de forma automática, el usuario únicamente ha de seleccionar el tipo de comunicación y pulsar el botón Enviar.

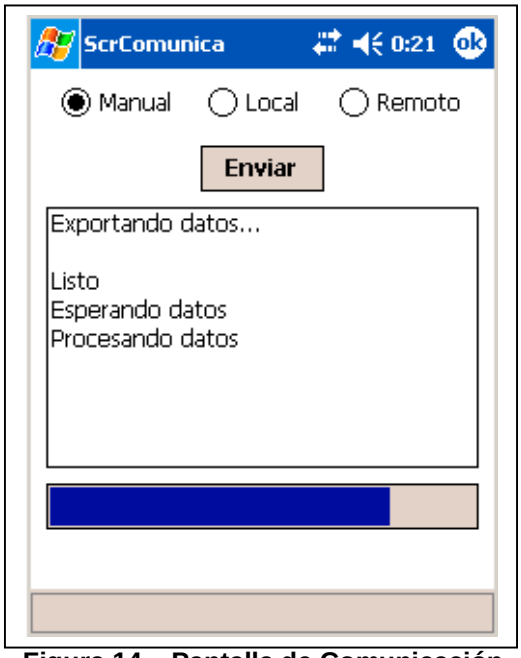

Figura 14 – Pantalla de Comunicación

Se han previsto 3 tipos de comunicación diferentes:

- **Manual**: La comunicación manual está diseñada para los casos en los que no hay un servidor FTP disponible, porque una avería o porque se están haciendo pruebas y todavía no se ha configurado. En este caso al pulsar el botón Enviar la aplicación genera el fichero con los pedidos y muestra el mensaje “Listo”. Entonces se han de copiar los pedidos del terminal al PC y los nuevos datos del PC al terminal. El programa espera hasta que aparezca un fichero Fin.txt en el directorio raíz del terminal. El contenido del fichero Fin.txt no tiene importancia, únicamente sirve para indicar que hemos terminado de copiar los datos. La transferencia entre el terminal y el PC se puede hacer con el ActiveSync o en el caso de trabajar con el emulador, con el explorador de archivos. Consultar el Manual de instalación para más detalles.

- **Local**: La comunicación local es automática y está diseñada para los casos en el que el terminal se conecta de forma directa al PC. Hay multitud de combinaciones posibles: ActiveSync y cable USB o serie, Wi-Fi, etc.

- **Remoto**: La comunicación remota es automática y está diseñada para los casos en el que el terminal se conecta a distancia al servidor. Lo habitual es que se trate de una comunicación GSM o GPRS.

3.2.3 Rutas

Una vez que se han cargado los datos en el terminal, disponemos de Rutas, Clientes y Artículos. La forma normal de trabajar es escoger una ruta e ir visitando a los clientes, tomando sus pedidos. El usuario únicamente ha de entrar en Rutas, seleccionar una ruta (hay que pulsar 2 veces sobre la ruta) y dentro de la ruta, ir seleccionando a los clientes.

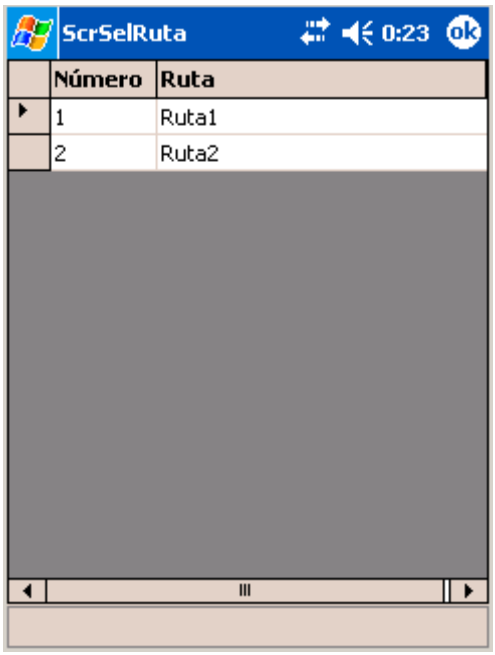

Figura 15 – Pant. Rutas

Figura 16 – Pant. Clientes de la ruta

Los controles del VS2003 no soportan doble clic por lo que la primera vez que nos posicionamos sobre un registro se selecciona. Cuando pulsamos una segunda vez sobre el mismo registro es cuando pasamos a la siguiente pantalla.

En Pocket PC el botón OK de la esquina superior derecha sirve para cerrar la pantalla, en la aplicación sirve para cerrar la ventana y volver a la anterior.

Cuando seleccionamos un cliente nos aparece la ventana de selección de pedidos. En esta pantalla vemos los datos del cliente con el fin que el comercial sepa su dirección, teléfono, etc. En la pestaña de contactos podemos ver en caso de que estén registrados, diferente personal del cliente con las que poder ponerse en contacto.

También podemos ver los pedidos del cliente y su estado: A-Abierto, C-Cerrado y E-Enviado. Si queremos editar un pedido no hay más que seleccionarlo dos veces. Para borrar o imprimir un pedido basta con seleccionarlo una vez y pulsar el botón correspondiente.

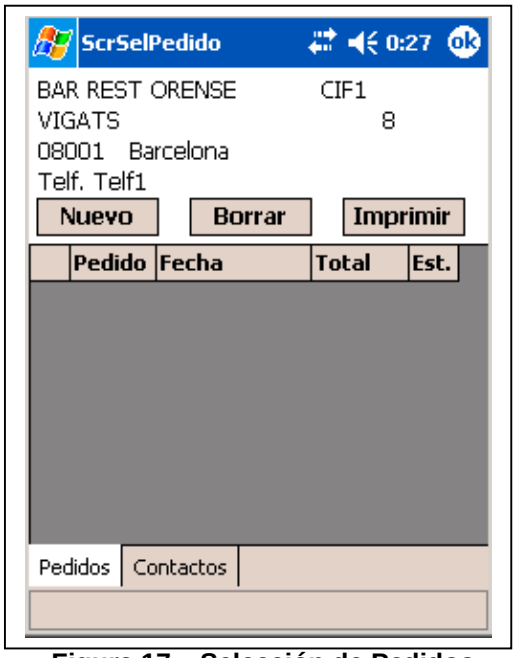

Figura 17 – Selección de Pedidos

Para crear un pedido pulsamos el botón Nuevo y pasamos a la ventana de edición de pedidos. Esta ventana está diseñada para simplificar el trabajo del usuario al máximo, únicamente hay que introducir el código del artículo, pulsar Enter, introducir la cantidad (si es diferente de 1) y pulsar Enter de nuevo. Si no recordamos el código de un artículo podemos buscarlo por el nombre en la pestaña Artículos, o por la Familia a la que pertenece en la pestaña Familias. Si buscamos por familia primero hay que ir al árbol de familias hasta que encontremos la que nos interesa y en la parte inferior nos aparecen los artículos pertenecientes a la familia. Al seleccionar un artículo volvemos a la pestaña General y podemos continuar confirmando la selección.

Los precios de los artículos son calculados de forma automática por la aplicación en función de la tarifa vigente y del descuento del cliente. El comercial puede modificar ese precio entrando en la pestaña de detalle. En ningún caso el precio del artículo puede ser inferior al precio mínimo marcado por la tarifa.

Según vamos entrando el pedido podemos ver en la parte inferior el importe total (sin IVA ni RE) del pedido. Si queremos ver el total o terminar el pedido debemos pulsar el botón OK e ir a la ventana de fin de pedido.

La ventana de fin de pedido nos muestra un resumen del pedido: El nombre del cliente, su CIF, el número de pedido, la fecha y la hora en que se creó el pedido y los diferentes importes.
Es importante indicar el estado del pedido. Un pedido abierto es un pedido sin terminar al que pensamos volver a editar más adelante. Un pedido cerrado es un pedido terminado. Por defecto la aplicación señala al pedido como cerrado. Los pedidos abiertos no se envían a la central cuando nos conectamos. Una vez se ha enviado un pedido su estado pasa a ser “Enviado”.

Figura 18 – Pantalla de fin de pedido

Podemos imprimir el albarán del pedido pulsando el botón Imprimir. También podemos regresar a la pantalla de edición pulsando el botón Anterior. Por último podemos guardar el pedido con el botón Grabar o descartar los cambios pulsando el botón Descartar. Cuando se descarta un pedido nuevo el pedido desaparece completamente. Cuando grabamos o descartamos un pedido volvemos a la pantalla de selección de pedidos. Para volver a la pantalla principal basta con ir cerrando las ventanas pulsando el botón OK.

3.2.4 Clientes

La función Clientes nos permite buscar un cliente por diferentes aspectos: Código del cliente, CIF, Nombre, Dirección, etc. Una vez introducido el criterio de búsqueda pulsamos el botón Buscar y nos aparece la lista de clientes. Cuando seleccionamos un cliente vamos a la pantalla de selección de pedidos. Los siguientes pasos son los mismos que los de Rutas que hemos descrito anteriormente.

3.2.5 Información

La función de información nos permite conocer el estado del terminal. Unicamente nos informa y el usuario no ha de introducir ningún dato. La pestaña pedidos nos muestra un resumen de los pedidos que hay en el terminal, los pedidos listos para enviar (pedidos cerrados) y los pedidos abiertos. En la pestaña Sistema podemos ver el estado de la batería así como la cantidad de memoria.

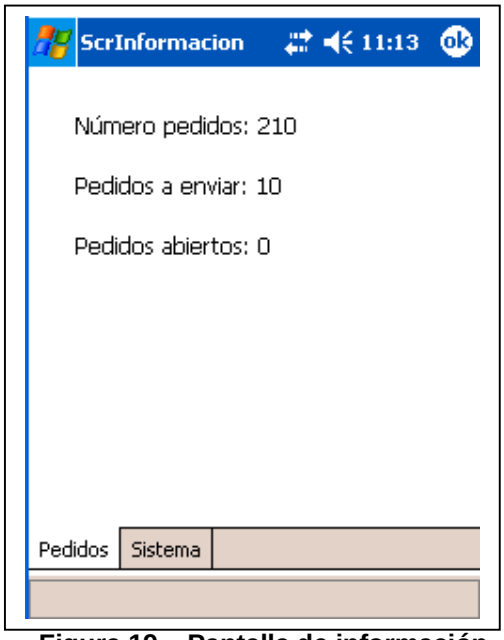

Figura 19 – Pantalla de información

4- Valoración económica

Al no tratarse de una aplicación hecha a medida para un cliente ni estar destinada a la venta, la valoración económica en términos monetarios no tiene demasiado sentido. En cambio tiene importancia conocer que recursos han sido necesarios para llevar a cabo el desarrollo y de esta forma poder estimar el esfuerzo necesario para futuros desarrollos.

Empecemos por revisar las horas dedicados al proyecto.

Etapa	Horas
Análisis	6
Diseño	16
Implementación	64
Depuración	25
Total	**111**

Hay que hacer algunas observaciones:

- Unicamente se han contado las horas de trabajo real, sin incluir pausas ni interrupciones. Si lo queremos traducir a jornadas de trabajo hay que tenerlo en cuenta ya que es prácticamente imposible estar 8 horas seguidas trabajando sin interrupción. Para traducirlo a jornadas de trabajo quizás sería más conveniente utilizar 7h/jornada o incluso 6h/jornada.
- En las horas de análisis únicamente se han contabilizado las horas empleadas en hacer los diagramas de casos de uso y los puntos clave de cada uno. No se incluye por tanto las horas destinadas a hacer el documento completo de análisis que es naturalmente muy superior. La razón de contabilizar únicamente las horas dedicadas a los casos de uso es porque se intenta tener una estimación de los costes de desarrollos reales, donde se tiende a documentar poco o incluso nada. Los casos de uso son piezas clave del análisis pero también del diseño en la POO por lo cual se ha de contabilizar el tiempo dedicado.
- En el caso del diseño pasa algo similar, únicamente se ha contabilizado el tiempo dedicado al diseño de los diagramas de colaboración y el esquema ER, sin incluir el tiempo dedicado al resto del documento de diseño.
- En el tiempo de la implementación se incluye también parte del tiempo dedicado a la depuración ya que es muy difícil desarrollar código sin probarlo de vez en cuando, y por la misma razón es difícil separar el cómputo de los tiempos de desarrollo sin incluir parte de los tiempos de depuración.

El proyecto tiene aprox. 7500 líneas de código (sin contar la librería Util). Esto nos da una media de 68 líneas código/hora de proyecto o si se prefiere 117 líneas/hora de implementación. Aunque este dato nos puede indicar que el VS es una buena herramienta de desarrollo ya que permite obtener una productividad elevada, la verdad es que no nos sirve para poder hacer estimaciones de costos de futuros

desarrollos. Para ello utilizaremos métodos clásicos de estimación como son los Puntos de Función, aprovechando el hecho de tratarse de una aplicación de gestión formada principalmente por pantallas y una BD SQL. De forma resumida y adaptando un poco la tabla a nuestras necesidades tenemos:

Puntos de función sin ajustar (TUFP)					
Tipo	**Descripción**	**Complejidad**			**Total**
		Simple	**Media**	**Complejo**	
EO	Pantallas de Salida	6x3	0x4	0x6	18
EI+EQ	Pantallas de E/S	9x4	2x5	1x7	53
EIF	Tablas BD de Lectura	9x5	0x7	0x10	45
LIF	Tablas BD de Lec/Esc	3x7	0x10	0x15	21
Total de puntos de función sin ajustar (TUFP)					137

Aquí abría que calcular el PCA para poder ajustar los puntos en función. Para no complicar los cálculos apliquemos un factor corrector de 0,8 teniendo en cuenta que la aplicación no se ha refinado mucho en cuanto facilidades, actualización on-line, etc.

PF= TUFP*PCA = 137*0,8 = 109,6

Combinando los PF con las horas de desarrollo tenemos aproximadamente que 1PF = 1h de desarrollo, o si se prefiere 1PF = 35’ de implementación.

Supongamos ahora que a la aplicación le queremos añadir una pantalla que nos muestre el resultado de una consulta a la BD. Sería:

1 Pantalla de Salida x 3 TUFP/pantalla * 0,8 PF/TUFP * 35 min Impl/PF = 84’ Impl.
1 Pantalla de Salida x 3 TUFP/pantalla * 0,8 PF/TUFP * 1h Des/PF = 1h24’ Des.

Nos llevaría aprox. 1h24’ de implementar y 2h24’ de desarrollo total. Evidentemente no todas las pantallas son iguales y puede haber muchos factores que hagan que el resultado final varíe tanto a favor como en contra. Lo importante es que podamos tener una primera idea del esfuerzo que requiere el desarrollo de una aplicación o la ampliación de una ya existente.

Insistir una vez más que es una primera aproximación, al final ha de ser la experiencia de cada uno la que nos diga cuales son valores correctos para hacer la estimación. En realidad según aumenta la experiencia del programador lo normal es que los tiempos de desarrollo se vayan acortando, aunque muchas veces esto se ve contrarrestado por el hecho que los programadores experimentados hacen implementaciones más refinadas.

5- Conclusiones

A lo largo de 3 meses se ha desarrollado esta aplicación de preventa. Conviene ahora al final de todo el proceso extraer una serie de conclusiones. Para ello analizamos el desarrollo desde diferentes puntos de vista que nos permiten obtener múltiples conclusiones.

5.1 Conclusiones sobre la aplicación de preventa

La aplicación funciona bien y cumple con las funcionalidades que nos habíamos marcado al principio del trabajo, así que por esa parte bien. Lo que ocurre que una vez terminada se observan ciertas carencias y se piensa en múltiples formas de mejorarla. Hay que decir que a la aplicación le falta mucho para ser una aplicación profesional con la que realmente los usuarios puedan trabajar, pero tampoco era ese el objetivo. Por contra, la aplicación es una magnífica herramienta de trabajo con la que se pueden experimentar nuevas técnicas, desde ese punto de vista no tiene precio.

Una vez entregada la aplicación y después de un pequeño descanso se desarrollarán dos nuevas versiones:

La primera (Preventa 1.1) sería simplementemente convertir el proyecto al VS2005. Gracias al nuevo CF 2.0 y al SGBD SQL SERVER 2005 CE la aplicación correrá un 50% más rápida. Puntualmente se puede introducir algún cambio en el código para aprovechar las posibilidades que proporcionan las estructuras genéricas, el ResultSet, etc. Al mismo tiempo hay reformar a fondo la librería Util ya que muchas de sus funcionalidades ya forman parte de las librerías estándar del CF 2.0.

La segunda versión (Preventa 2.2) constituiría un desarrollo totalmente renovado, con nuevas funcionalidades, nuevas tablas en la BD, más pantallas y en general una mayor sofisticación de la aplicación. Entre otras cosas:

- **Mejorar la interfaz gráfica**: Hay que mejorar el tema de los mensajes y sobre todo que se pueda hacer todo o casi todo a través del teclado.
- **Añadir una opción de pedidos**: Ahora se pueden dejar los pedidos abiertos (es un pedido no terminado) pero si se quiere volver a editarlo y no se recuerda del cliente, hay que ir uno por uno buscándolo. La idea es poder buscar los pedidos de forma directa.
- **Añadir la gestión de cobros**: Ahora no se ha hecho por no complicar la aplicación. El problema es que cuando hay dinero de por medio hay que asegurarse que todo funciona perfectamente so riesgo que a uno le corten el pescuezo.
- **Mejorar la introducción de los descuentos**: Actualmente sólo se puede modificar el precio.

- **Contemplar las ofertas**: Al menos las ofertas más simples (2x1, etc)
- **Añadir más opciones de configuración**: Al haber nuevas funcionalidades se necesitarán más opciones.
- **Informar del progreso del recorrido de la ruta**: Utilizar algún tipo de indicador que informe al usuario de sí se ha visitado un cliente, si tiene pedidos o cobros pendientes, etc.
- **Ampliar la opción de información**: Hay que incorporar informes, estadísticas etc.
- **Posibilidad de enviar los pedidos de forma automática**: Con una conexión GPRS se podría monotorizar la cobertura y cuando hubiese suficiente enviar los pedidos que hubiese hasta el momento.
- **Soportar más perfiles de usuarios**: Un perfil de programador (la empresa que instala el software) que permita un control total de la aplicación, un perfil administrador (el informático de la empresa), un perfil de usuario avanzado al que se le permitiría algunas opciones de configuración, y un perfil de usuario normal.
- **Multi-idioma**: El VS permite trabajar con recursos y gestionar diferentes idiomas. En España muchas comunidades son bilingües y con un poco de trabajo se puede incorporar un segundo idioma. Una vez que contemplas 2 idiomas añadir un 3º o un 4º ya sólo es cuestión de ir traduciendo.
- **Aplicación actualizable a través de la conexión**: Tanto en las instalaciones con elevado número de terminales como en las que los terminales casi nunca van a la central esta posibilidad puede ser de gran ayuda para los administradores.
- **Modo Kiosco**: Son las aplicaciones que funcionan de forma exclusiva. Una aplicación profesional normalmente no permite a los usuarios acceder al SO.

El ser muy elevado el número de cambios no se harían de golpe sino de forma progresiva. Lo que sí se puede es diseñar la aplicación desde un principio para soportar todas estas funcionalidades y así cuando una nueva opción esté lista no haga falta tocar nada del resto de la aplicación.

5.2 Conclusiones sobre las herramientas de desarrollo

La principal herramienta de desarrollo es naturalmente el VS2003. A pesar de ser un magnífico IDE de desarrollo casi desde el comienzo fue necesario instalar diferentes plug-ins para poder desarrollar para dispositivos móviles. Después de 2 años se impone una puesta al día y en ese sentido el VS2005 es un digno sucesor.

En la actualidad los procesos de análisis y diseño son tan importantes como la misma implementación. De hecho con las herramientas de desarrollo actuales se puede pasar de los diagramas UML y ER directamente a implementar con un importante ahorro de tiempo. En este desarrollo se ha empleado el Jude para hacer los esquemas UML. Es una magnífica herramienta pero está pensada para trabajar con código JAVA por lo que habría que buscar una nueva

herramienta que soporte C#. Con los diagramas ER pasa lo mismo, se impone una nueva herramienta.

A parte de todo esto se han utilizado otras herramientas:

- Al ActiveSync para comunicar con el terminal.
- MS Remote Display para capturar las pantallas.
- El Office para la documentación.

5.3 Conclusiones sobre el método de desarrollo

Como ya se ha indicado anteriormente el método de desarrollo utilizado es el clásico: Análisis, Diseño, Implementación y Depuración. Lo cierto es que ha funcionado muy bien y cuando ha llegado el momento de la verdad (la implementación) apenas ha habido que retocar el diseño, mas que nada por alguna omisión. Hay que reconocer que ha habido un poco de suerte pero sobre todo los estudios de la UOC sobre la POO + BD y la experiencia de 2 años trabajando con el VS2003 me han permitido enfocar el diseño de forma adecuada.

Entre las cosas que mejor han ido el empleo de Casos de Uso y Diagramas de Colaboración para el análisis y el diseño. Por contra el empleo de un prototipo ha sido poco productivo ya que en la mayoría de casos ha habido que rehacer las pantallas. Esto se debe al extenso uso de DataGrid ligados a DataTable en la aplicación que en el prototipo se han reemplazado por ListViews. Sinceramente creo que si se puede conviene evitar el empleo de prototipos, o si no hay que buscar una manera de aprovechar las pantallas generadas y así ahorrar trabajo.

Un punto importante para llegar a buen término el desarrollo es hacer una buena planificación y cumplirlo. El tiempo de desarrollo de una aplicación de gestión se pueden estimar bien pero hay que evitar en la medida de los posible hacer experimentos ya que los tiempos de desarrollo no son previsibles y pueden desbaratar toda la planificación. No quiere decir que no se puedan utilizar nuevas técnicas, sino que conviene que se investiguen en una aplicación a parte y cuando estén a punto incorporarlas a la aplicación principal.

Por ejemplo para la impresión se emplean plantillas utilizando un método capaz de interpretar pequeñas expresiones y que utiliza la Reflexión para extraer los datos del pedido. Su desarrollo se hizo en paralelo al de la aplicación y por suerte fue todo bastante bien. Si no hubiese salido a tiempo se habría utilizado una técnica más sencilla y así se habría respetado la planificación.

5.4 Conclusión general

A modo de resumen, el trabajo ha sido bastante duro pero los resultados han sido estupendos. No quiero decir que la aplicación no sea mejorable, sino que ha sido una magnífica oportunidad para combinar los conocimientos aprendidos de múltiples asignaturas de la carrera así como la experiencia adquirida en el trabajo. Es más el trabajo continuará probablemente a lo largo de 6 meses donde se aprovechará para experimentar nuevas técnicas y seguir practicando el análisis y el diseño.

6- Glosario

.Net: Siglas utilizadas por Microsoft para identificar una familia de productos caracterizados por trabajar con el Framework (o Compact Framawork).
Autoventa: La autoventa es la actividad comercial que realizan los empleados de una empresa de distribución donde se toma el pedido al cliente y se le entrega a la vez. Por los tanto los empleados son comerciales y repartidores al mismo tiempo, llevando la mercancía con ellos.
BD: Base de Datos, conjunto de datos de diferente tipo organizados y estrechamente relacionados entre sí.
Bluetooth:
C#: Lenguaje de programación orientado a objetos desarrollado por Microsoft. Comparte muchos aspectos del Java e incorpora algunas características del C. El código generado es código intermedio que se ejecuta por el Framework (o Compact Framework).
CLI: Código intermedio generado por los compiladores .Net (C#, VB.Net, etc). Este código es compilado a su vez a código máquina y ejecutado por el CLR.
CLR: Runtime del Framework encargado de ejecutar el código intermedio. Entre otros elementos contiene un compilador JIT y un recolector de basura.
Compact Framework (CF): Máquina virtual de Microsoft encargada de ejecutar los programas que contienen código intermedio, en su versión reducida para dispositivos móviles.
EPOC: Sistema Operativo desarrollado originalmente por Psion y actualmente por Symbian. Especialmente diseñado para PDAs actualmente se encuentra en los teléfonos móviles. Fundamentalmente se programa en C++ y en Java.
Esquema ER: Esquema Entidad Relación que permite diseñar una Base de Datos.
eVB: Visual Basic Embebed, una Basic derivado del Ms VB Script para los terminales Windows CE 3.0
eVC++: Visual C++ Embebed, es el C++ de Microsoft en su versión para dispositivos móviles con SO Windows CE.
FTP: File Transfer Protocolo, protocolo de transferencia de archivos estándar de Internet.
GPRS: General Packet Radio Service.El GPRS es un sistema de transmisión de datos sin cable basado en la división de los datos a transmitir en paquetes de información. Se caracteriza por ritmos de transferencia de datos muy elevados, de 56 hasta 114 Kbs, y por la garantía de un enlace a Internet sin cable continuo - 'always connected' - ya sea para teléfonos móviles como para ordenadores portátiles o cualquier otro tipo de dispositivo que incorpore esta tecnología. La característica de conexión continua hace que desaparezca el concepto de tiempo de conexión, facturándose sólo por la cantidad de información transmitida. El GPRS fue diseñado primeramente como una mejora del estándar GSM pero puede ser adoptado por otros operadores que utilicen estándares diferentes al GSM. Siempre trabaja a través de una conexión a Internet y no siempre hay cobertura.

GSM: Global System for Mobile Communications.El GSM es un estándar internacional para las comunicaciones radio digitales creado por el Instituto Europeo para los Estándares en Telecomunicaciones. Este estándar ha sido desarrollado para permitir las comunicaciones móviles en todos los mercados y empezó a utilizarse en el año 1991. En Europa se aplica en dos bandas de frecuencia, a 900 MHz y a 1800 MHz (GSM 900 y GSM 1800 respectivamente), mientras que en EUA se aplica en la banda de 1900 MHz (GSM 1900) y en el Sureste Asiático y Japón a 850 MHz (GSM 850). Cuando se utiliza para transmitir datos tiene unas velocidades limitadas (9.6/14.4 Kbits/s) y la comunicación es punto a punto.

IrDA: Protocolo de comunicaciones a través de luz infrarroja.
JIT: Just In time Translator, compilador incorporado sobre todo en las máquinas virtuales (JVM y Framework) que permite compilar a código máquina en el momento de la ejecución.
Kiosco: Modo de trabajo de las aplicaciones en las que el usuario no puede acceder al SO proporcionando así una mayor seguridad.
MVC: Modelo Vista Controlador, arquitectura de las aplicaciones donde se distribuye el código entre las “Vistas” (ventanas o clases frontera) y el controlador. De esta forma el código de las Vistas se encarga de la interacción con el usuario mientras que el código del controlador se encarga de los procesos y la lógica de la aplicación.
Palm: Es a la vez una marca de fabricante de PDAs como un SO. Se han vendido (y se siguen vendiendo) millones de PDAs cada año pero como capturadores de datos industriales no son muy empleados.
PDA: Personal Digital Assistant, asistente de datos personal originalmente pensado para contener una agenda y una pequeña BD de contactos, con el tiempo se ha ido ampliando hasta convertirse en ordenadores completos.
Plataforma: Microsoft define una plataforma como una serie de especificaciones tanto de Hardware como de Software. En el mundo de los terminales con SO Windows CE hay 2 plataformas: Windows CE y Pocket PC.
Pocket PC: Plataforma para terminales Windows CE diseñada originalmente para los PDAs. En la actualidad se utiliza tanto en dispositivos industriales como en PDAs y se va actualizando de forma periódica. Ha habido diferentes versiones como Pokect PC 2002, PPC 2003, Mobile 2003 y Mobile 2005.
POO: Programación Orientada a Objeto, sistema de programación caracterizado por modelizar tanto el código como los datos en objetos. Es un sistema bastante diferente al utilizado en los lenguajes tradicionales de programación (C, Pascal, etc). Lenguajes Orientado al Objeto son el Java, C#, etc.
Preventa: Es la actividad comercial que realizan los empleados de una empresa de distribución donde se toma el pedido al cliente y se envía a la central para que unos camiones de reparto hagan posteriormente la entrega. Los empleados únicamente hacen una labor comercial.
Prototipo: Es una versión reducida de la aplicación que permite tanto a los desarrolladores como a los clientes hacerse una idea de la aplicación final.
Puntos de Función (PF): Sistema desarrollado por A.J. Albrecht que permite medir la complejidad de una aplicación y estimar su costo.

Ruta: Una ruta es una relación de clientes que ha de visitar un comercial.
SGBD: Sistema Gestor de Base de Datos, en nuestro caso el SQL Server CE 2.0
Smart Device: Identifica a diferentes dispositivos caracterizados por ser portátiles, tener un SO (Windows CE) con requerimientos reducidos. Básicamente son los terminales industriales con Windows CE, los PDAs y los SmartPhones (teléfonos móviles con SO Windows CE).
SQL: Structured Query Language, lenguaje utilizado fundamentalmente en los SGBD. Aunque hay un estándar, cada fabricante introduce sus propias variantes.
Terminal: Dispositivo móvil de uso industrial, normalmente con pantalla y teclado. Son ordenadores completos y tienen ciertas características comunes a los PDAs, aunque normalmente son más grandes, más robustos y también mas caros.
UML: Unified Modeling Lenguage, lenguaje de modelado para la construcción de aplicaciones orientado a objetos. Al ser un estándar permite construir modelos y esquemas que pueden comprender diferentes desarrolladores.
UMTS: Universal Mobile Telecomunications System **.**El UMTS (Sistema Universal de Telecomunicaciones Móviles) es una tecnología que permitirá la implantación de la telefonía de tercera generación (3G). El sistema ha sido desarrollado por ETSI (European Telecommunications Standard Institute) y las especificaciones que lo definen se engloban en el 3GPP (Third Generation Partnership Project). UMTS se caracteriza por ofrecer un gran ancho de banda y elevadas velocidades de transferencia de 2 Mbps en adelante. La nueva tecnología abrirá paso a una simbiosis entre Telefonía móvil, Internet, Tecnología de la Información e Industria, creando todo un abanico de nuevas aplicaciones. El UMTS hizo su entrada en el mercado en el año 2003 y aunque en el plano de operadores el sistema ya está totalmente implementado, a fecha del 2005 aún hay muy pocos teléfonos móviles que la incorporen. El destino final del UMTS es desplazar a los actuales sistemas GSM y GPRS. La conexión es TCP/IP y trabaja a través de Internet, la cobertura todavía es muy limitada.
USB: Universal Serial Bus, conexión serie síncrona de propósito general. Utilizada ampliamente en la actualidad, acostumbra a ser el medio de conexión entre los PDAs y los terminales con el PC.
VB.Net: Visual Basic .Net, versión del lenguaje Visual Basic orientado a objetos. La herramienta de desarrollo es el Visual Studio y el código generado es código intermedio que debe ser ejecutado por el Framework (o el CF).
Visual Studio (VS): Herramienta de programación y entorno de desarrollo de Microsoft. Soporta diferentes lenguajes de programación, para dispositivos móviles: C#, VB.Net y ASP.
Windows CE: Es a la vez el nombre de una familia de Sistemas Operativos como el nombre de una plataforma. Como SO proviene del Windows NT, evolucionado y adaptado para estar embebido en los dispositivos móviles. Como plataforma se caracteriza por tener un SO Windows CE y un entorno gráfico similar a los Windows de los PC de escritorio (98, 2000 y XP).

7- Bibliografía

Las principales fuentes bibliográficas tienen su origen en su vertiente más teórica en el material de la carrera. En la vertiente más práctica ligada al lenguaje y a las herramientas de programación, en la ayuda del VS y múltiples enlaces sobre todo de MSDN. Por último una serie de páginas web de diferentes orígenes que complementan algún punto expuesto en la memoria.

Fonaments de programació II
Programació orientada a l'objecte
Julià Minguillón i Alfonso (coordinador)
XP03/05063/00942

Estructura de la informació
Xavier Franch Gutiérrez (coordinador)
XP00/05001/00651

Enginyeria del programari
Benet Campderrich Falgueras
Recerca informàtica, SL
XP03/05060/02078

Bases de dades I
Jaume Sistac Planas (coordinador)
XP05/05002/00492

Bases de dades II
Jaume Sistac Planas (coordinador)
XP03/05053/02047

Tècniques de desenvolupament de programari
Fatos Xhafa
XP02/05049/00099

Gestió d'organitzacions i projectes informàtics
Miquel Barceló García
Joan Antoni Pastor i Collado
XP03/05069/02064

Historia de Psion
http://3lib.ukonline.co.uk/historyofpsion.htm

Historia de Palm
http://www.xbitlabs.com/articles/mobile/display/palm.html

Historia de Symbol

http://www.symbol.com/about/overview/overview_hist_highlights.html

Historia de Windows CE
http://www.hpcfactor.com/support/windowsce/

Arquitectura del Compact Framework 1.0
http://msdn2.microsoft.com/en-us/library/9s7k7ce5.aspx

Introducción a la programación para Windows CE y PPC
http://msdn.microsoft.com/library/en-us/dnhcvs03/html/vs03i1.asp

Múltiples formularios
http://msdn.microsoft.com/library/en-us/dv_vstechart/html/vbtchWorkingWithMultipleFormsInVisualBasicNETUpgradingToNET

Diferencias del GUI entre Windows CE .Net y PPC
http://msdn.microsoft.com/library/en-us/dnnetcomp/html/netcfPPCtoCE.asp

Comportamiento de los controles en Windows CE .Net
http://msdn.microsoft.com/library/en-us/dv_evtuv/html/etconWindowsCENETPlatformBehavior

Comportamiento de los controles en PPC
http://msdn.microsoft.com/library/en-us/dv_evtuv/html/etconPocketPCPlatformBehavior.asp

Imagen inicial
http://msdn.microsoft.com/library/en-us/dnhcvb03/html/vb03i11.asp

Estructura de datos: Listas, Colas, Arboles, TablasHash, Conjuntos
http://msdn.microsoft.com/library/shared/deeptree/asp/deeptree.asp?stcfg=/library/searchtabconfig.xml&dtcfg=/library/deeptreeconfig.xml&url=/library/en-us/vblr7/html/vaconbuildingfromcommandline.asp?frame=true

Estructura de datos (C# 2.0): Listas, Colas, Arboles, TablasHash, Conjuntos
http://msdn.microsoft.com/vcsharp/default.aspx?pull=/library/en-us/dnvs05/html/datastructures20_1.asp

http://msdn.microsoft.com/vcsharp/default.aspx?pull=/library/en-us/dnvs05/html/datastructures20_6.asp

Acceso a directorios y archivos
http://msdn.microsoft.com/library/en-us/dnhcvb04/html/vb04l4.asp

Ayuda del SQL Server CE 2.0 (se instala con el VS 2003)
..\Microsoft SQL Server CE 2.0\sqlce.chm

Uso SQL Server CE
ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.3082/dv_evtuv/html/etconWalkthroughCreatingSQLServerCEDatabase.htm

Optimización SQL
http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/ssceqpop.mspx

Información adicional del SQL Server CE
http://msdn.microsoft.com/library/en-us/dv_evtuv/html/etconSQLServerCE.asp

DataReaders
http://msdn.microsoft.com/library/en-us/dnhcvs04/html/vs04g6.asp

Filtrar y Ordenar
http://msdn.microsoft.com/library/en-us/dnhcvb03/html/vb03g9.asp

http://msdn.microsoft.com/library/en-us/dnnetcomp/html/PISAPICF.asp

Consideraciones
http://msdn.microsoft.com/library/en-us/dv_evtuv/html/etconNetworkAccessProgrammingConsiderations

Comunicaciones IrDA
http://msdn.microsoft.com/library/en-us/dv_evtuv/html/etconInfraredConnections.asp
http://msdn.microsoft.com/library/en-us/dnhcvs04/html/vs04b6.asp

P-Invoke: Introducción
http://msdn.microsoft.com/library/en-us/dnnetcomp/html/netcfintrointerp.asp

P-Invoke: Avanzado
http://msdn.microsoft.com/library/en-us/dnnetcomp/html/netcfadvinterop.asp

Creación de una librería
http://msdn.microsoft.com/library/en-us/dnnetcomp/html/PInvokeLib.asp

Ejemplo de aplicación PPC
http://msdn.microsoft.com/mobility/default.aspx?pull=/library/en-us/dnppc2k3/html/fieldsales_ppc.asp

Novedades en .NET Compact Framework 2.0
http://msdn2.microsoft.com/library/hlek3akf.pf

Mejoras en Microsoft SQL Server CE 2005
http://msdn2.microsoft.com/library/ayee3tzx.pf