PROYECTO DE SISTEMAS INFORMÁTICOS

CURSO 2010/2011

Desarrollo de un sistema de consultas web para sistemas SAT

Profesor director: Miguel Ángel Blanco Rodríguez

Autores: Pedro Pablo García

Curso académico: 2010-2011

Índice

Autorización a la UCM

Autorizo a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

Fdo.: Pedro Pablo García García

1. Resumen

El proyecto presentado en esta memoria consiste, fundamentalmente, en una solución informática que proporciona soporte automatizado para un Servicio de Asistencia Técnica o SAT de sencillo manejo para los operarios encargados de gestionarlo.

En la práctica, éste servicio lo llevará a cabo un operario al que se le exigirán plazos de ejecución, teniendo en todo momento la capacidad de reparar, instalar o sustituir un TPV.

Asimismo, el sistema permite al operario controlar la gestión de recursos propios, es decir que tendrá acceso a los datos reales tanto de la compra del producto como la cantidad en stock que hay de dicho producto en almacén.

SUMMARY:

This project consists in an IT solution to provide an automated support for customer technical assistance service, and easy working for the technical managers handle.

In the practice, this service will carry out by managers with the responsibility of obtain the executions periods, having the capacity to repair, to install and to substitute a TPV (Point Sale Terminal)

Likewise, the system allows the technical managers to control their own management resources, it means, they will be constant access at true information from the items sales and the stock store data base.

2. Palabras clave

2.1. Microsoft.NET

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el software en forma de servicios que puedan ser suministrados remotamente y que puedan comunicarse y combinarse unos con otros de manera totalmente independiente de la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados. Ésta es la llamada plataforma .NET, y a los servicios antes comentados se les denomina *servicios web.*

Para crear aplicaciones para la plataforma .NET, tanto servicios web como aplicaciones tradicionales (aplicaciones de consola, aplicaciones de ventanas, servicios de Windows NT, etc.), Microsoft ha publicado el denominado kit de desarrollo de software conocido como .NET Framework SDK, que incluye las herramientas necesarias tanto para su desarrollo como para su distribución y ejecución y el IDE Visual Studio.NET, que permite hacer todo lo anterior desde una interfaz visual muy cómoda basada en ventanas.

El concepto de Microsoft.NET también incluye al conjunto de nuevas aplicaciones que Microsoft y terceros han (o están) desarrollando para ser utilizadas en la plataforma .NET. Entre ellas se pueden destacar aplicaciones desarrolladas por Microsoft tales como Windows.NET, Visual Studio.NET, MSN.NET, Office.NET, y los nuevos servidores para empresas de Microsoft (SQL Server.NET, Exchange.NET, etc.)

2.2. Lenguaje C#

El lenguaje C# es el nuevo lenguaje diseñado por Microsoft para su plataforma .NET. En concreto, ha sido diseñado por Scott Wiltamuth y Anders Hejlsberg, éste último también conocido por haber sido el diseñador del lenguaje Turbo Pascal y la herramienta RAD Delphi.

Aunque en realidad es posible escribir código para la plataforma .NET en muchos otros lenguajes, como Visual Basic.NET o JScript.Net, C# es el único que ha sido diseñado específicamente para ser utilizado en esta plataforma, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes. Por esta razón, Microsoft suele referirse a C# como el lenguaje nativo de .NET, y de hecho, gran parte de la librería de clases base de .NET ha sido escrito en este lenguaje.

C# es un lenguaje orientado a objetos sencillo, moderno, amigable, intuitivo y fácilmente legible que ha sido diseñado por Microsoft con el ambicioso objetivo de recoger las mejores características de muchos otros lenguajes, fundamentalmente Visual Basic, Java y C++, y combinarlas en uno sólo en el que se unan la alta productividad y facilidad de aprendizaje de Visual Basic con la potencia de C++.

Quizás el más directo competidor de C# es Java, lenguaje con el que guarda un enorme parecido en su sintaxis y características. En este aspecto, es importante señalar que C# incorpora muchos elementos de los que Java carece (sistema de tipos homogéneo, propiedades, indexadores, tablas multidimensionales, operadores redefinibles, etc.) y que según los benchmarks realizados la velocidad de ejecución del código escrito en C# es ligeramente superior a su respectiva versión en Java.

A continuación desarrollamos de manera resumida las **principales características de C#:**

- Dispone de todas las características propias de cualquier lenguaje orientado a objetos: **encapsulación, herencia y polimorfismo**.

- **Ofrece un modelo de programación orientada a objetos homogéneo**, en el que todo el código se escribe dentro de clases y todos los tipos de datos, incluso los básicos, son clases que heredan de System.Object (por lo que los métodos definidos en ésta son comunes a todos los tipos del lenguaje)

- **Permite definir estructuras**, que son clases un tanto especiales: sus objetos se almacenan en pila, por lo que se trabaja con ellos directamente y no con referencias al montículo, lo que permite un acceso mucho más rápido. Sin embargo, esta mayor eficiencia en sus accesos tiene también sus inconvenientes, fundamentalmente que el tiempo necesario para pasarlas como parámetros a métodos es mayor (hay que copiar su valor completo y no sólo una referencia) y no admiten herencia (aunque sí implementación de interfaces)

- **Es un lenguaje fuertemente "tipado"**, lo que significa que controla todas las conversiones entre tipos que se realicen de forma compatible, asegurando que nunca se acceda fuera del espacio de memoria ocupado por un objeto. De esta forma se evitan frecuentes errores de programación y se consigue que los programas no puedan poner en peligro la integridad de otras aplicaciones.

- **Dispone de un recolector de basura** encargado de descargar al programador de la tarea de tener que limpiar las referencias a objetos que dejen de ser útiles. Esta herramienta asume el trabajo de eliminar automáticamente los elementos no necesarios evitando agotar la memoria ante posibles olvidos del programador, y descarta la posibilidad de que se produzcan errores porque el programador libere áreas de memoria ya liberadas y reasignadas.

- **Incluye soporte nativo** para eventos y delegados. Los delegados son similares a los punteros a funciones de otros lenguajes como C++ aunque más cercanos a la orientación a objetos, y los eventos son mecanismos mediante los cuales los objetos pueden notificar de la ocurrencia de sucesos. Los eventos suelen usarse en combinación con los delegados para el diseño de interfaces gráficas de usuario, con lo que se proporciona al programador un mecanismo cómodo para escribir códigos de respuesta a los diferentes eventos que puedan surgir a lo largo de la ejecución de la aplicación (pulsación de un botón, modificación de un texto, etc.)

- Incorpora propiedades basadas en un mecanismo que **permite el acceso controlado a miembros de una clase**, tal y como si de campos públicos se tratasen. Gracias a ellas se evita la pérdida de legibilidad que en otros lenguajes provoca la utilización de métodos Set() y Get() pero sin embargo permanecen todas las ventajas de un acceso controlado por dichos métodos.

- Permite la definición del significado de los operadores básicos del lenguaje (+, -, *, &, ==, etc.) para nuestros propios tipos de datos, lo que **facilita enormemente tanto la legibilidad de las aplicaciones como el esfuerzo necesario para escribirlas**. Es más, se puede incluso definir el significado del operador [] en cualquier clase, lo que permite acceder a sus objetos como si fuesen tablas. A la definición de éste último operador se le denomina indizador, y es especialmente útil a la hora de escribir o trabajar con colecciones de objetos.

- **Admite unos elementos llamados atributos** que no son miembros de las clases sino información sobre éstas que podemos incluir en su declaración. Por ejemplo, indican si un miembro de una clase ha de aparecer en la ventana de propiedades de Visual Studio.NET, cuáles son los valores admitidos para cada miembro en ésta, etc.

2.3. Internet Information Service

Internet Information Services o IIS es un servidor web y un conjunto de servicios para el sistema operativo Microsoft Windows. Originalmente era parte del *Option Pack* para Windows NT. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003. Windows XP Profesional incluye una versión limitada de IIS. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS.

Este servicio convierte a una PC en un servidor web para Internet o una intranet, es decir que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente.

Los servicios de Internet Information Services proporcionan las herramientas y funciones necesarias para administrar de forma sencilla un servidor web seguro.

El servidor web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas. Por ejemplo, Microsoft incluye los de Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes, como PHP o Perl.

2.4. ADO.NET

ADO.NET proporciona acceso coherente a orígenes de datos como Microsoft SQL Server y XML, así como a orígenes de datos expuestos mediante OLE DB y ODBC. Las aplicaciones para usuarios que comparten datos pueden utilizar ADO.NET para conectar a estos orígenes de datos y recuperar, manipular y actualizar los datos contenidos.

ADO.NET separa el acceso a datos de la manipulación de datos y crea componentes discretos que se pueden utilizar por separado o conjuntamente. ADO.NET incluye proveedores de datos de .NET Framework para conectarse a una base de datos, ejecutar comandos y recuperar resultados. Los resultados se procesan directamente o se colocan en un objeto DataSet de ADO.NET con el fin de exponerlos al usuario para un propósito específico, combinados con datos de varios orígenes, o de utilizarlos de forma remota entre niveles. El objeto DataSet de ADO.NET también puede utilizarse independientemente de un proveedor de datos de .NET Framework para administrar datos que son locales de la aplicación o que proceden de un origen XML.

Las clases de ADO.NET se encuentran en el archivo System.Data.dll y están integradas con las clases de XML que se encuentran en el archivo System.Xml.dll. Cuando se compila un código que utiliza el espacio de nombres System.Data es necesario hacer referencia a los archivos System.Data.dll y System.Xml.dll. Para obtener un ejemplo de una aplicación de ADO.NET que se conecta a una base de datos, recupera datos de ésta y, a continuación, los muestra en el símbolo del sistema.

ADO.NET proporciona funcionalidad a los programadores que escriben código administrado similar a la funcionalidad que los objetos ADO (ActiveX Data Objects) proporcionan a los programadores de modelo de objetos componentes (COM) nativo para obtener más información sobre las diferencias entre ADO y ADO.NET.

2.5. DataSet

El **DataSet de ADO.NET** es una representación de datos residente en memoria que proporciona un modelo de programación relacional coherente independientemente del origen de datos que contiene. Un **DataSet** representa un conjunto completo de datos, incluyendo las tablas que contienen, ordenan y restringen los datos, así como las relaciones entre las tablas.

Hay varias **maneras de trabajar con un DataSet**, que se pueden aplicar de forma independiente o conjuntamente. Puede:

- Crear mediante programación una **DataTable**, **DataRelation** y una **Constraint en un DataSet** y rellenar las tablas con datos.
- Llenar el **DataSet** con tablas de datos de un origen de datos relacional existente mediante **DataAdapter**.
- Cargar y hacer persistente el contenido de **DataSet** mediante **XML**.

Un DataSet con establecimiento inflexible de tipos también se puede transportar mediante un servicio web XML. El diseño del DataSet hace que sea ideal para transportar datos mediante servicios web XML.

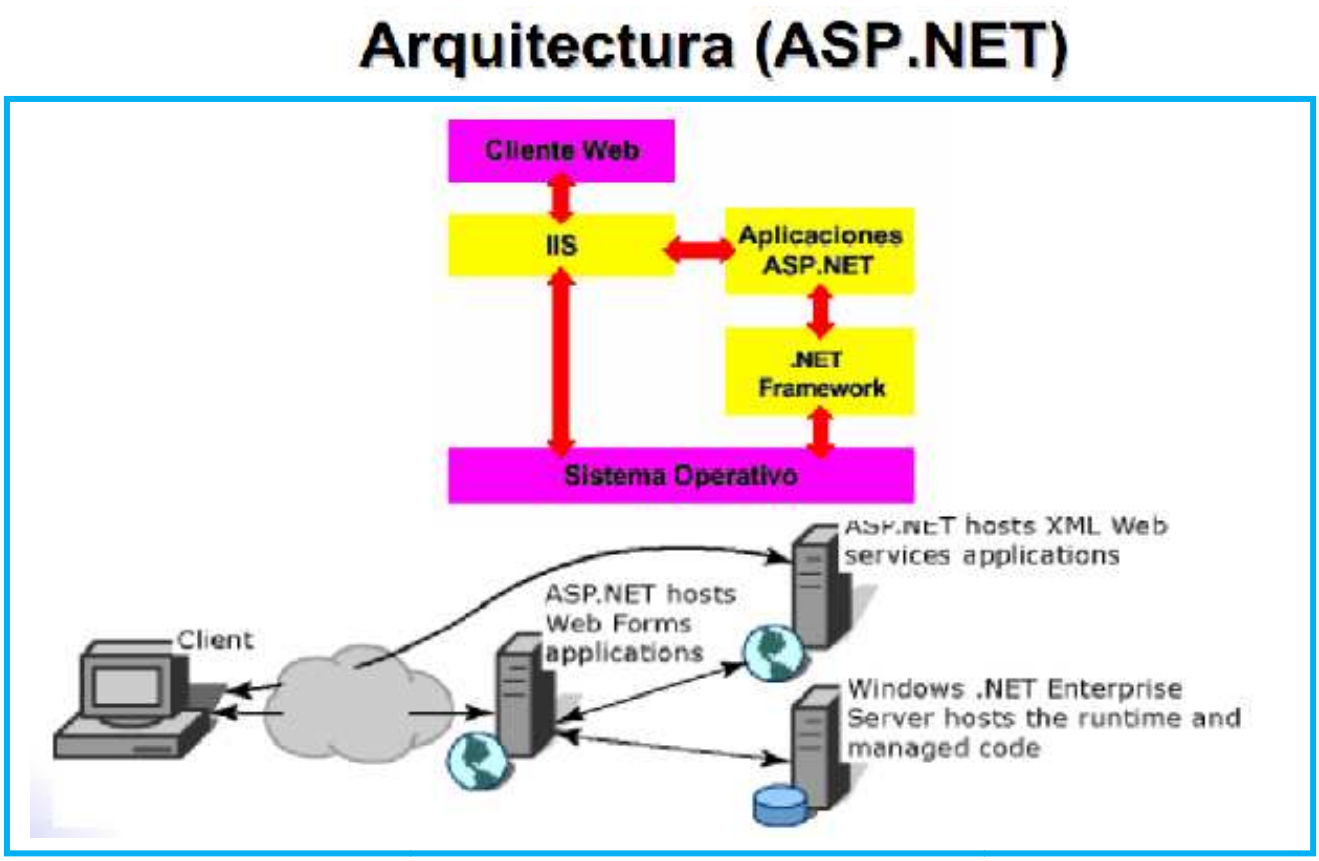
Con ADO.NET es posible llenar un DataSet a partir de una secuencia o un documento XML. Se puede utilizar la secuencia o el documento XML para suministrar datos al DataSet, información de esquema o ambas cosas. La información suministrada desde la secuencia o el documento XML puede combinarse con datos o información de esquema existente ya presente en el DataSet.

ADO.NET también permite crear una representación XML de un DataSet, con o sin su esquema, para transportar el DataSet a través de HTTP con el fin de que lo utilice otra aplicación u otra plataforma compatible con XML. En una representación XML de un DataSet, los datos se escriben en XML y el esquema, si está incluido en línea en la representación, se escribe utilizando el lenguaje de definición de esquemas XML (XSD). XML y el esquema XML proporcionan un formato cómodo para transferir el contenido de un DataSet a y desde clientes remotos.

El DataSet también puede utilizarse junto con datos existentes en un origen de datos cuando se usa con un proveedor de datos de .NET Framework. Éste utiliza un DataAdapter para rellenar el DataSet con datos e información de esquema, así como para resolver cambios relacionados con los datos en el origen de datos.

2.6. ASP.NET

- Es una implementación completamente nueva de **ASP**, escrita de cero en **C#**.
- **ASP.NET** utiliza lenguajes de programación compilados como Visual **Basic.Net, C#, incluso COBOL** (es "lenguaje-neutral"), para escribir aplicaciones Web.
- Las aplicaciones son compiladas en el servidor, y las páginas son generadas en **HTML** específicamente para el browser que hizo la invocación.
- Es un lenguaje compilado común que se ejecuta en el servidor.
- Aplica conceptos de **"early binding"**, compilación **"just-in-time"**, optimización de código nativa y **"caching services"**
- Tiene un conjunto de herramientas completo y un **IDE** común para diseño (VisualStudio.Net).
- La **.NET Framework class library**, la mensajería, y las soluciones de acceso a datos son accesibles completamente desde el Web en forma transparente.
- Es independiente al lenguaje, ya que permite elegir el lenguaje que más se aplique al problema o particionar el mismo e implementar la solución con múltiples lenguajes.
- Emplea una configuración a nivel de archivos de texto a nivel jerárquico que simplifica aplicar los **"set"** al entorno del servidor y las aplicaciones Web.
- El deployment de una aplicación **ASP.NET** implica simplemente copiar los archivos necesarios al servidor.
- Se integra en la autenticación del sistema operativo Windows y permite una configuración a nivel de aplicación.



2.7. JQuery

jQuery es una biblioteca o framework de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

La librería jQuery en resumen nos aporta las siguientes **ventajas**:

- Nos **ahorra muchas líneas de código**
- Nos **hace transparente el soporte de nuestra aplicación** para los navegadores principales
- Nos **provee de un mecanismo para la captura de eventos**.
- Provee un conjunto de funciones para **animar el contenido de la página en forma muy sencilla**
- Integra funcionalidades para **trabajar con AJAX**

2.8. TPV

TPV es el acrónimo de terminal punto de venta (en inglés "*POS terminal*" o "*Point of sale terminal*"). Hace referencia al **dispositivo y tecnologías que ayudan en las tareas de gestión de un establecimiento comercial** de venta al público.

Estos locales pueden contar con sistemas informáticos especializados que ayudan en las tareas de gestión del punto de venta mediante una interfaz accesible para los vendedores que se denomina "terminal de punto de venta" o "TPV".

Los TPV permiten la creación e impresión del ticket de venta mediante las referencias de productos, realizan diversas operaciones durante todo el proceso de venta, así como cambios en el inventario. También generan diversos reportes que ayudan en la gestión del negocio. Los TPV se componen de una parte hardware (dispositivos físicos) y software (sistema operativo y programa de gestión)

Software:

El TPV tiene su programa de gestión o software. Puede ser:

- A medida: Contienen software específico para una única empresa. Suelen ser mucho más caros y las modificaciones o actualizaciones van siempre ligadas a la disponibilidad de la empresa que desarrolla ese software.

- Comerciales: Dentro de este grupo pueden estar predefinidos para tiendas de ropa, hostelería, ferreterías, farmacias, videoclubs o TPV de carácter general. Suelen estar diseñados para un establecimiento tipo del sector al que va dirigido y no admite cambios específicos. Suelen ser mucho más económicos.

- Específicos: algunas empresas fabricantes de TPV desarrollan un software específico para un tipo de negocio en concreto. Este software puede ir junto con un sistema operativo propio del fabricante embebido en la propia memoria del terminal (no tienen discos duros ni trabajan con sistemas operativos como los PC) o bien instalados en el disco duro del terminal, como cualquier otro PC. Este tipo software cuenta con múltiples opciones de configuración en función de las necesidades de un cliente concreto, siempre que trabaje dentro de este sector en concreto.

Hardware

Los tipos de TPV actual son:

- Compacto: se trata de los equipos más modernos que integran todos los elementos necesarios en el terminal de un solo aparato. Es decir, que **integran la CPU, la impresora, la pantalla y el teclado en una sola máquina**. Normalmente llevan integradas pantallas táctiles, aunque también permiten la conexión de otras interfaces de usuario y periféricos, como teclados, cajón portamonedas etc.

En apariencia son similares a cualquier PC corriente, pero con la diferencia de que estos equipos de última generación **incluyen una caja de reducidas dimensiones que ocupa poco espacio**, ubicada generalmente encima del cajón portamonedas.

Los componentes internos son también muy parecidos a los de un PC corriente, no obstante, los terminales modernos eliminan elementos mecánicos como el disco duro consiguiendo **más fiabilidad y menor consumo del terminal**. Además, estos equipos suelen sufrir menos averías provocadas por la desconexión de los diversos cables que en los TPV modulares conectan los diferentes elementos que componen el TPV. Tengamos en cuenta que hasta ahora la labor de las CPU la realizaba una memoria con un programa y una memoria de trabajo en una caja registradora.

- Modular: son equipos basados en un PC corriente con un software instalado sobre un sistema operativo convencional. Todos los componentes del TPV se conectan a un CPU a través de sus diferentes cables e interfaces.

Permiten el uso de múltiples componentes de distintos fabricantes y el uso del TPV para otras funciones típicas en un PC. Tienen una gran versatilidad ya que se pueden utilizar para diversos tipos de negocio en función del software instalado en el terminal.

2.9. Crystal Report

Crystal Reports es una aplicación de inteligencia empresarial utilizada para **diseñar y generar informes desde una amplia gama de fuentes de datos** (bases de datos)

Varias aplicaciones, como Microsoft Visual Studio, incluyen una versió de Crystal Reports como una herramienta de propósito general del informes/reportes. Crystal Reports se convirtió en el escritor de informes estándar cuando Microsoft lo liberó con Visual Basic.

Ventajas:

- El reporte forma parte del proyecto, de manera que no se tiene que agregar ningún archivo de reporte al empaquetado.
- Al usar el diseñador existe la posibilidad de capturar algunos eventos que el reporte dispara, por ejemplo a la hora de leer los registros y a la hora de imprimir el reporte.
- Existe una mayor flexibilidad para manejar la jerarquía de objetos y acceder a ella.
- Se tiene mayor control sobre los objetos que forman parte del reporte.

Desventajas:

- El uso de diseñadores hace que el proyecto crezca en tamaño.
- Un paquete de instalación con diseñadores de Crystal Reports es más propenso a fallar (según mi experiencia) que si se lee un reporte desde un archivo externo.

2.10. XML

El código HTML permite insertar menús, tablas, imágenes o bases de datos en los documentos, pero no permite al usuario que maneje esos elementos como mejor le convenga con la poderosa ayuda del ordenador. Esa es la principal novedad que aporta XML.

Con HTML se pueden hacer accesos a información comparativa en diferentes tiendas, por ejemplo, pero nada más. Con XML el usuario podrá ordenar los datos o actualizarlos en tiempo real o realizar un pedido.

La información que manejan las empresas es uno de sus principales activos, pero lo normal es que esa información esté fragmentada en diferentes departamentos, ordenadores conectados o no, etc. El reto ahora está en **interrelacionar toda esa información para obtener un máximo rendimiento y aprovechar todo su potencial**, y que de esta forma se **incremente la productividad aumentando los beneficios o reduciendo los costes.** Para dar viabilidad a esta premisa se necesita un estándar de almacenamiento estructurado que es precisamente lo que nos ofrece XML.

XML, es el estándar de Extensible Markup Language. XML en definitiva no es más que un conjunto de reglas para definir las etiquetas semánticas que nos organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados.

En primer lugar, para asimilar correctamente en qué consiste hay que mantener relativamente al margen lo que entendemos como código HTML.

- En teoría, HTML es un subconjunto de XML especializado en presentación de documentos para la web, mientras que XML es un subconjunto de SGML especializado en la gestión de información para la web.

- En la práctica XML contiene a HTML aunque no en su totalidad. La definición de HTML contenido totalmente dentro de XML y por lo tanto que cumple a rajatabla la especificación SGML es XHTML (Extensible, Hypertext Markup Language).

Los comentarios a partir de los que el compilador generará la documentación han de escribirse en XML, por lo que han de respetar determinadas reglas comunes a todo documento XML bien formado.

3. Desarrollo del Proyecto

El proyecto se ha desarrollado siguiendo las instrucciones y explicaciones que el director proporcionaba en las reuniones mantenidas. En cada una de estas reuniones se presentaba el trabajo realizado durante la semana y se discutía sobre él, valorando qué debía mantenerse y qué debía cambiarse. Tras la valoración del trabajo presentado, el director indicaba el siguiente paso a seguir. Cada reunión quedaba registrada en un acta1 donde constaba la fecha de la reunión, los asistentes, los temas tratados, los acuerdos alcanzados y el trabajo a realizar durante la siguiente semana.

En la primera reunión mantenida se decidió dividir el proyecto en cuatro fases para que su seguimiento y desarrollo fueran más sencillos. Dichas fases han sido:

- **Fase 1 - Análisis funcional:** en esta fase definimos el funcionamiento interno de nuestro proyecto.

- **Fase 2 - Diseño de la base de datos:** en esta fase diseñamos la base de datos basándonos en los elementos analizados en la fase anterior.
- **Fase 3 - Creación de un prototipo:** en esta fase del proyecto, implementamos una parte de la aplicación. El prototipo sirve para que tanto el desarrollador como el cliente tengan una idea de qué es lo que hay que hacer y cómo hay que desarrollarlo, de esta forma el cliente podrá hacerse una idea del funcionamiento del sistema final.
- **Fase 4 - Creación de la memoria:** en la última fase del proyecto, se ha redactado esta memoria.

El trabajo realizado en las diferentes fases se desarrolla a continuación especificado en cada uno de los sucesivos puntos de esta memoria.

3.1. Análisis Funcional

Durante esta fase definimos el funcionamiento interno de nuestro sistema.

3.1.1. Funcionamiento del sistema

El proyecto se enmarca dentro de los Sistemas denominados Servicio de Asistencia Técnica o SAT, que gestionan y registran las actividades que se realizan sobre productos, generalmente identificados con un identificador único (Número de Serie). Las actividades a realizar, son iniciadas ante la necesidad de realizar una actuación de reparación, sustitución o instalación de los dispositivos. La actividad, es planificada conforme a una fecha estimada de ejecución que el operario encargado de realizarlo debe intentar cumplir.

El escenario propuesto es de una **Organización que recibe peticiones de actuación para ejecutar la orden de trabajo en determinados clientes.** Al recibir la orden de actuación, lanza una orden de trabajo que debe ejecutarse. La ejecución lleva consigo, el registro por parte del recurso (empleado) de la actuación (fecha y hora), junto con una grabación del resultado. A veces, al resultado es acompañado de comentarios que indican las incidencias encontradas. Tanto los comentarios como los resultados, son susceptibles de ser tratados estadísticamente posteriormente.

Previamente a la actuación, el sistema realiza un re-aprovisionamiento de dispositivos, realizando pedidos de compra a los proveedores. Los proveedores, suministran los dispositivos conforme a una tarifa, y son guardados en un almacén para su posterior instalación. La gestión de la compra se realiza conforme al stock disponible y a las previsiones de actuaciones.

3.1.2. Interacción con el usuario

El usuario deberá, en primer lugar, introducir sus datos en la base de datos usando una aplicación que no forma parte de nuestro prototipo.

Una vez que ha introducido sus datos, el usuario puede consultar los servicios y la información asociada a ellos como por ejemplo operario que realiza el servicio, facturación, etc.

Además el usuario podrá generar informes y descargarse esa información en ficheros Excel.

3.2. Diseño de la Base de Datos

Para una mejor compresión, dividiremos la Base de Datos en dos partes bien diferenciadas. Por un lado se refleja la ***Compra del Producto*** a un vendedor y por otro representamos el ***Servicio de Asistencia Técnica*** ofrecido a un cliente.

Compra Producto:

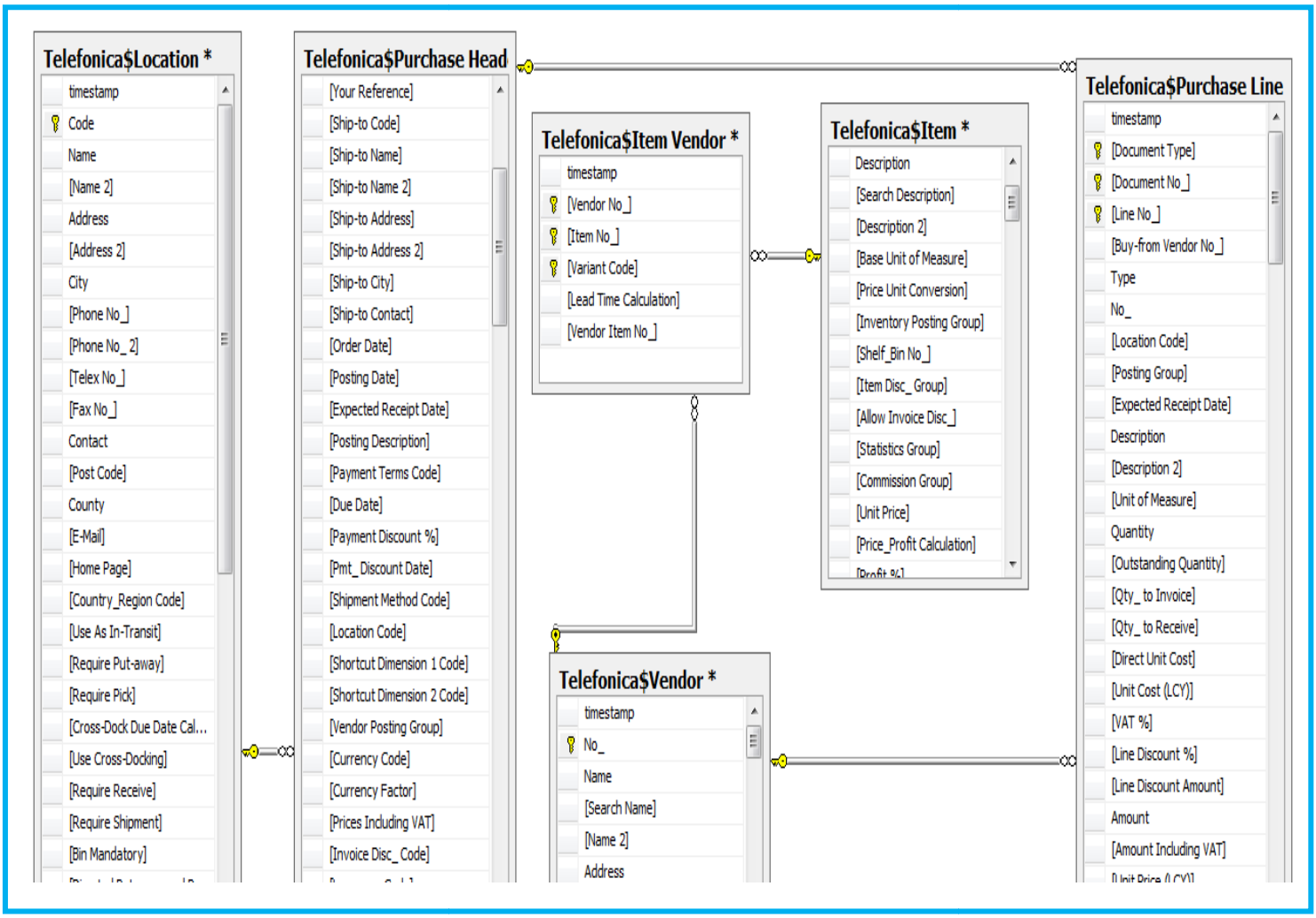


Fig. 3.1 Diseño Base de Datos "Compra Producto"

A continuación se describe información de las tablas relacionadas con la *compra del producto*:

- Telefonica$Purchase Head

Ésta tabla guarda la información que contiene la cabecera de una factura, por ejemplo contiene atributos como nº factura (que será un número sencuencial), proveerdor, etc.

- Telefonica$Purchase Line

Cada factura tiene un conjunto de líneas que ofrecen información de los productos comprados. Contiene campos como descripción del producto, código, nombre, precio e importe total, etc.

- Telefonica$Vendor

Guarda información general (nombre, dirección, ciudad, contacto, etc.) del proveedor

- Telefonica$Item Vendor

Representa los productos que puede vender el proveedor. Estará relacionada tanto con *Telefonica$Vendor* como con *Telefonica$Item,* realmente nos sirve para establecer una relación de muchos a muchos entre un proveedor y un producto. De esta forma representamos que un vendedor puede vender muchos productos y un producto puede ser vendido por varios vendedores

- Telefonica$Item

Como se ha explicacdo anteriormente representa la información sobre los productos, guardará entre otras cosas, el precio unitario, código producto, comisión, etc.

- Telefonica$Location

Cuando un producto se compra hay que guardarlo en un almacén, la información de este almacén está en esta tabla. Cuando se emite una factura utilizando la información de *Telefonica$Purchase Head* debe indicarse el almacén en el que se guardó el producto, por lo que se establece una relación entre estas dos tablas.

El diseño correspondiente al Servicio de Asistencia Técnica SAT es el siguiente:

Servicio de Asistencia Técnica (SAT)

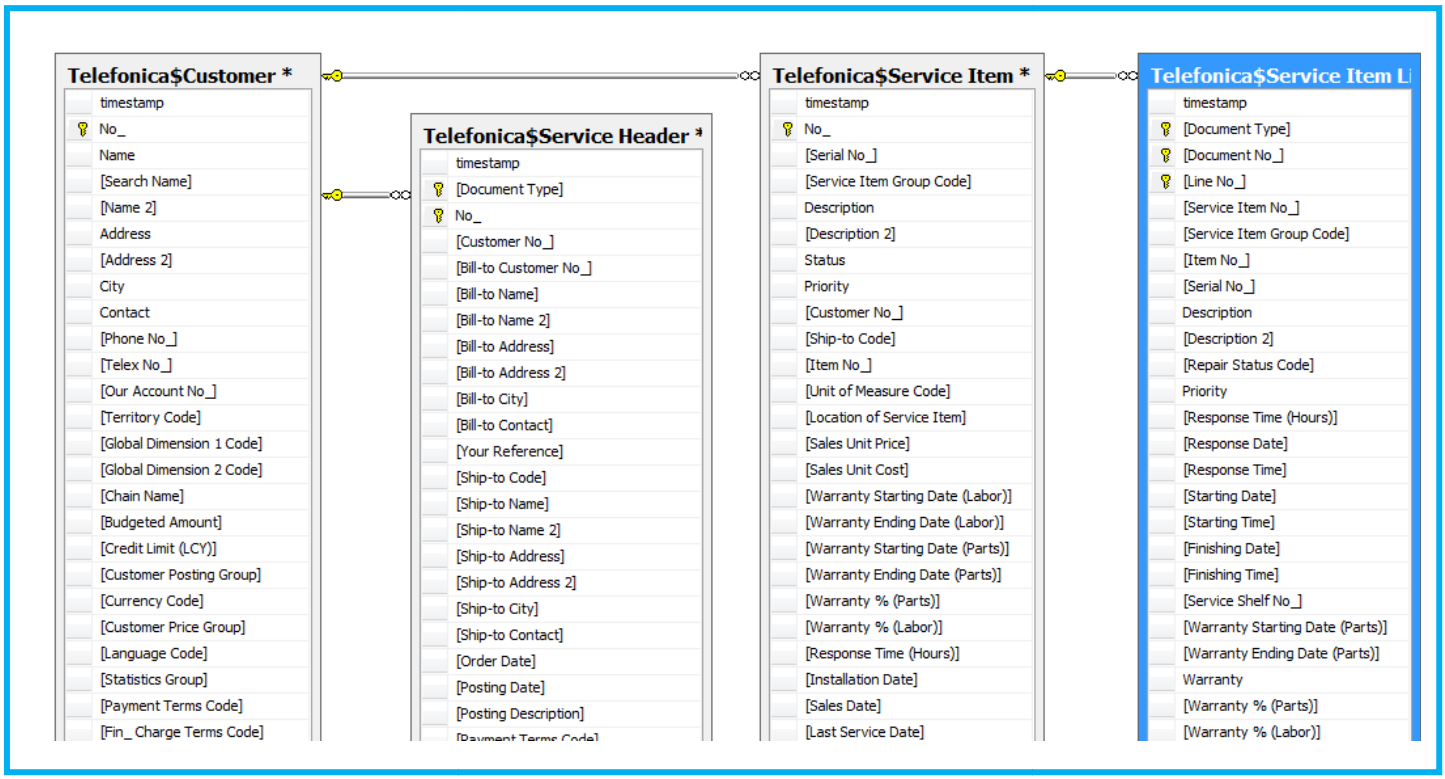


Fig. 3.2 Diseño de Base de Datos "Servicio de Asistencia Técnica"

A continuación se describe información de las tablas relacionadas con la el Servicio de Asistencia Técnica o SAT:

- Telefonica$Customer

Contiene información sobre los clientes del servicio. Un cliente puede solicitar varios servicios (reflejados en la tabla Telefonica$Service Header). En el fondo los clientes son tiendas, por ejemplo Ahorramas o Carrefour)

- Telefonica$Service Header

Cada solicitud de servicio que hace un cliente queda reflejado en ésta tabla que a su vez esta compuesta por varios Items o elementos. Es la parte correspondiente a una actuación que puede implicar varias actuaciones sobre los dispositivos. Un ejemplo podría ser: Ir a la tienda y realizar una actuacción ya sea reparar, cambiar, etc. La cabercera de esta solicitud sería *Ir a la tienda*.

- Telefonica$Service Item

Un servicio está compuesto por un conjunto de elementos que conforman el servicio ofrecido al cliente. Esta tabla guarda esa información. Es la parte correspondiente al producto, podría ser un TPV sobre el cual hay que realizar una actuación, entonces sobre el TPV, por ejemplo 10, hay que realizar una actuación, ese TPV 10 representa una línea de Service Item. Un cliente puede tener varios *ítem¸* esto debe ser reflejado en la base de datos, de ahí la relación que se establece a través del campo *Customer No_*

- Telefonica$Service Item Line

Cada *Item* lo componen un conjunto de líneas que reflejan información que es necesario guardar para ese elemento. Service *item* es por ejemplo reparar un TPV y un *ítem line* podría ser toda la información que es necesario guardar sobre ese TPV en concreto.

3.3. Creación de un prototipo

El siguiente paso es el desarrollo de la aplicación. Se empezará por una descripción de los ficheros que componen la solución, con una breve descripción de la funcionalidad de algunos de ellos.

Ficheros de la aplicación:

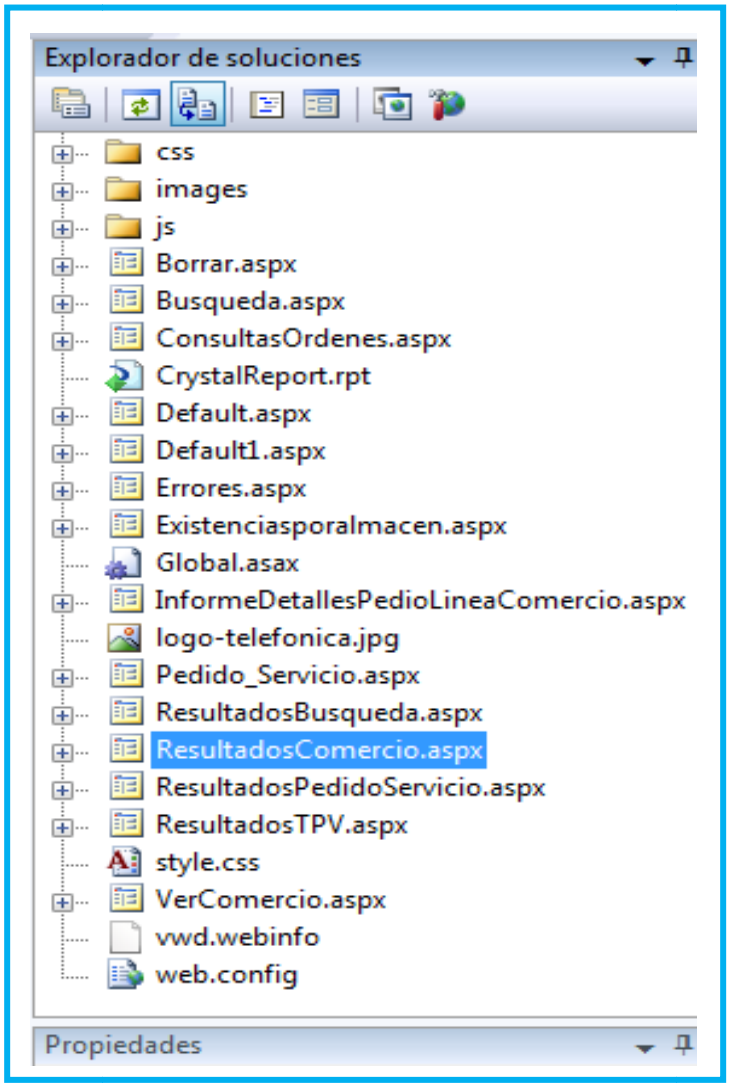


Fig. 3.3 Ficheros aplicación

Con el explorador de soluciones de Visual Studio.NET., se puede observar fácilmente una descripción general de los ficheros

Tenemos tres carpetas: **"images, css y jQuery"**, que contienen respectivamente las imágenes de la aplicación, los estilos utilizados y por último los controles jQuery utilizados.

Los ficheros **".aspx"** son ficheros que contienen, por un lado el código ASP.NET de la parte servidor, y por otro el código "HTML" de la parte cliente.

Por eso aparece el signo **"+"** a la izquierda de estos ficheros. Al expandirlo aparecen dos ficheros, uno con extensión **"aspx"** y otro con extensión **"cs"**, el primero contiene código del cliente: "HTML" y "JavaScript", y el segundo escrito en lenguaje "C#" que forma parte del código que se ejecuta del lado del servidor.

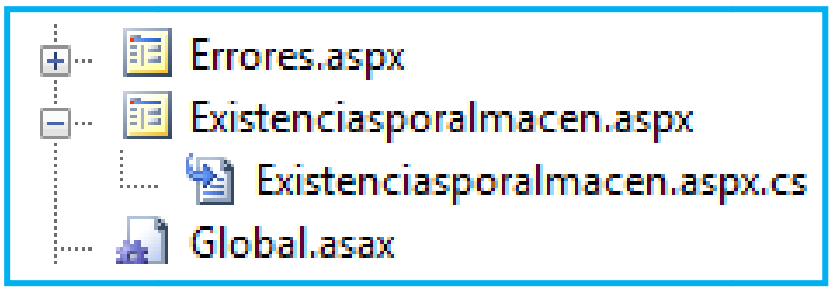


Fig. 3.4 Ficheros aspx y cs

Ejemplo 1: código "HTML" del fichero **"Existenciasporalmacen.aspx"**

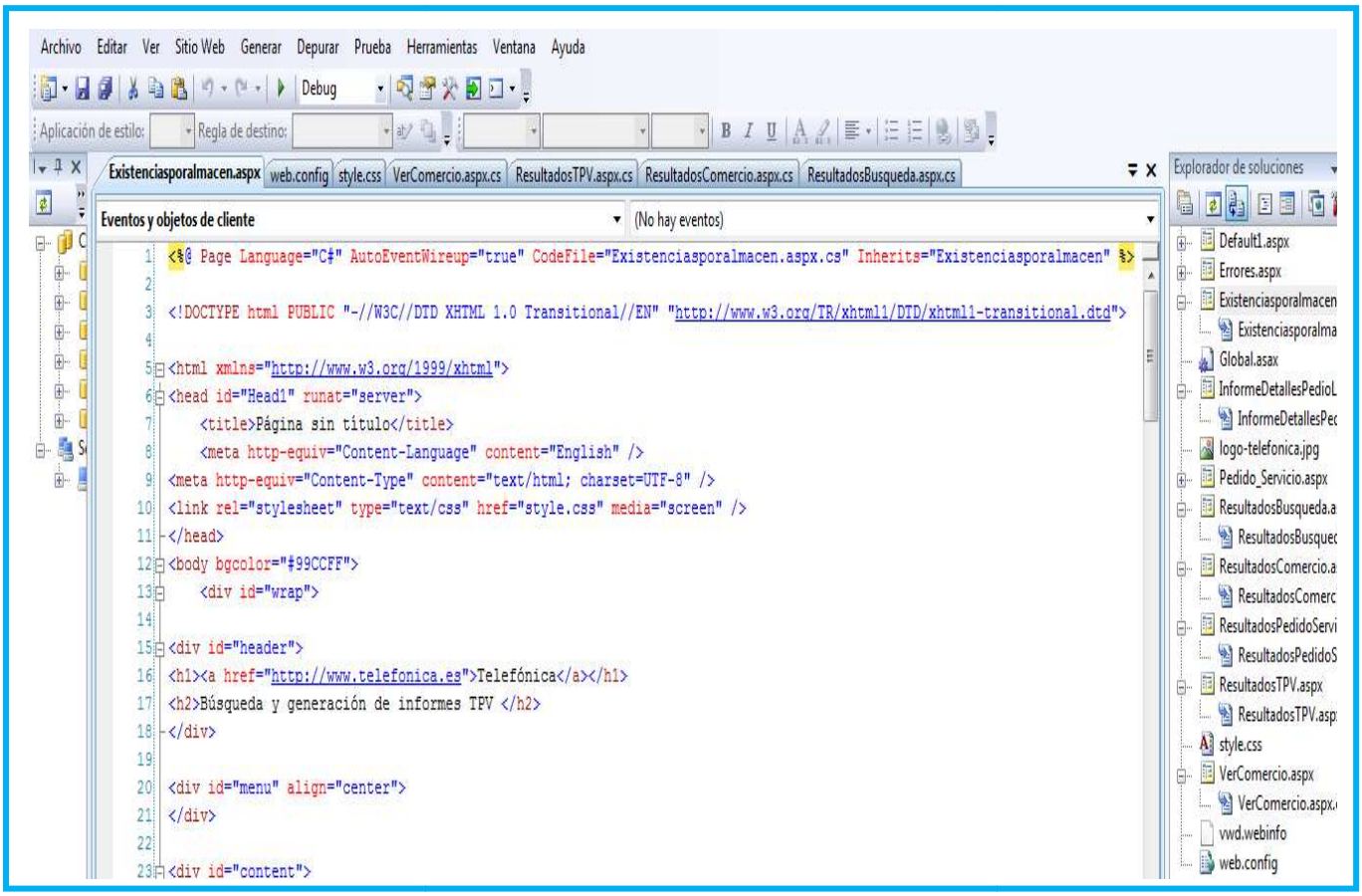


Fig. 3.5 Fichero Existenciasporalmacen.aspx

Ejemplo 2: código Servidor del fichero **"Existenciasporalmacen.cs"**

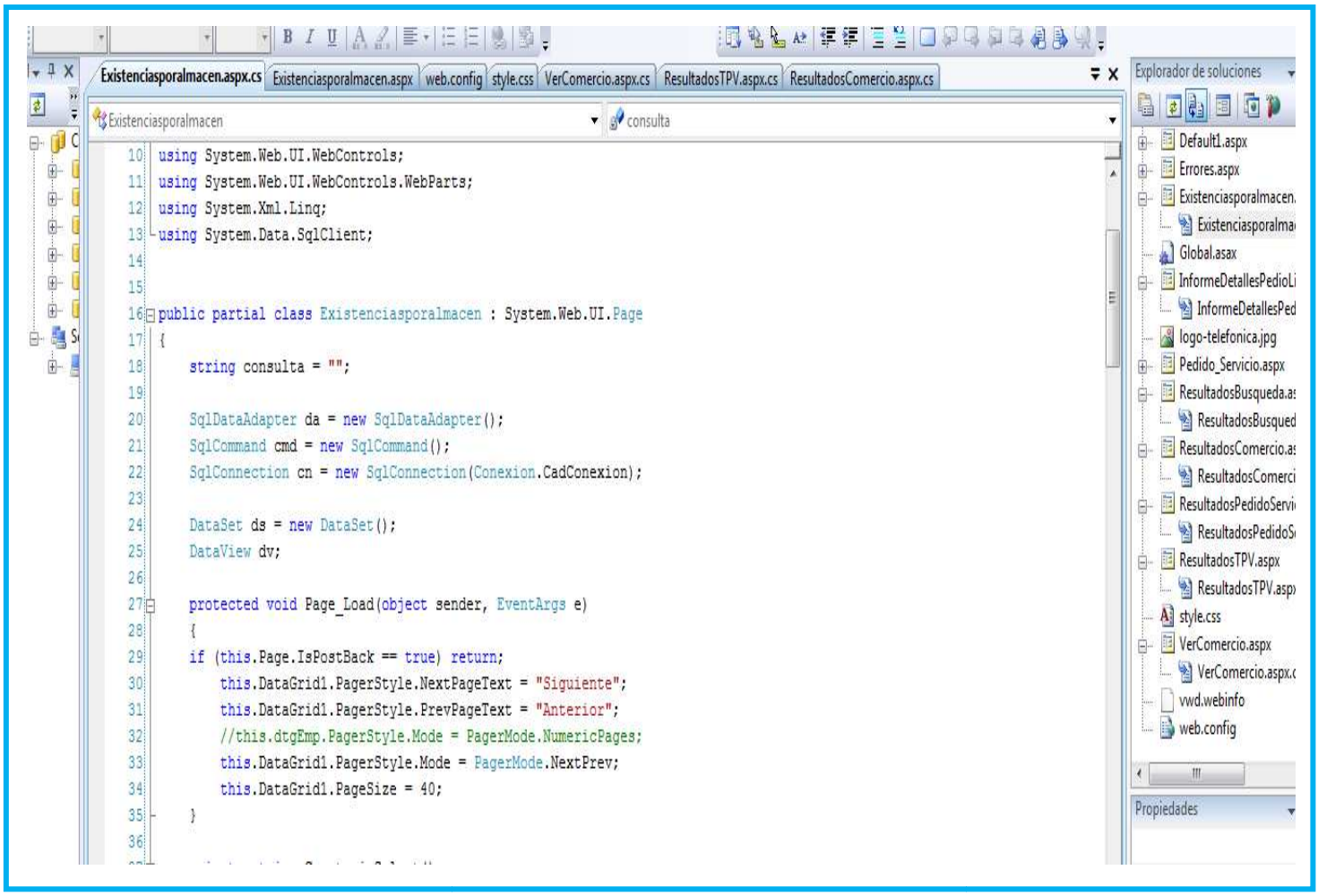


Fig. 3.6 Fichero Existenciasporalmacen.cs

El fichero **Web.Config**, es un fichero con formato **"XML"** lo utilizaremos como fichero de configuración inicial y también para la autenticación de usuarios.

A continuación se muestra parte **del código de autenticación** de un usuario:

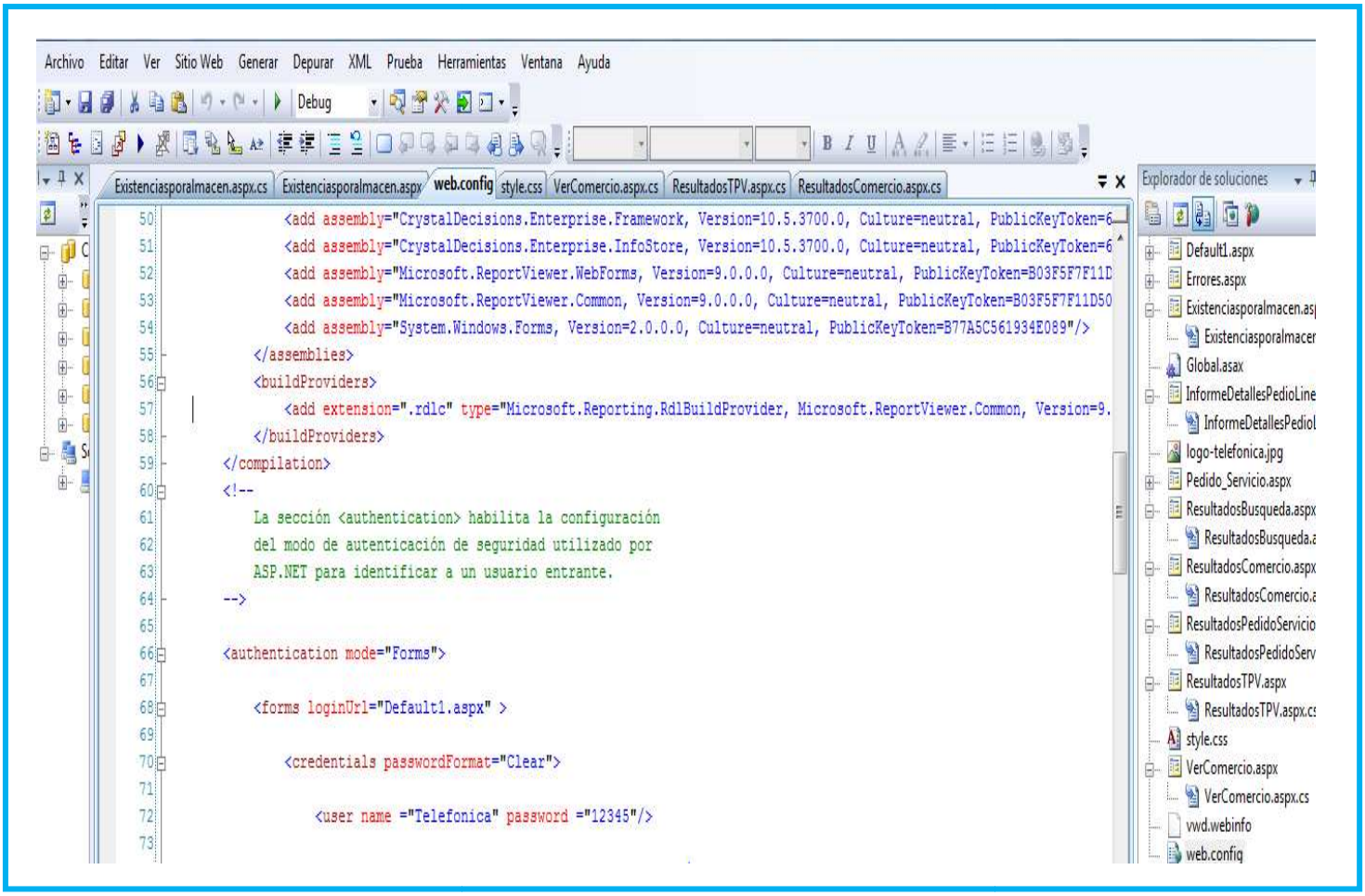


Fig. 3.7 Fichero web.config

Este tipo de validación de usuarios presenta algunas ventajas y desventajas que se explican más adelante.

En el fichero **Global.asax**, guardaremos las variables **Session**, que permitirán guardar información de los usuarios mientras éstos estén conectados:

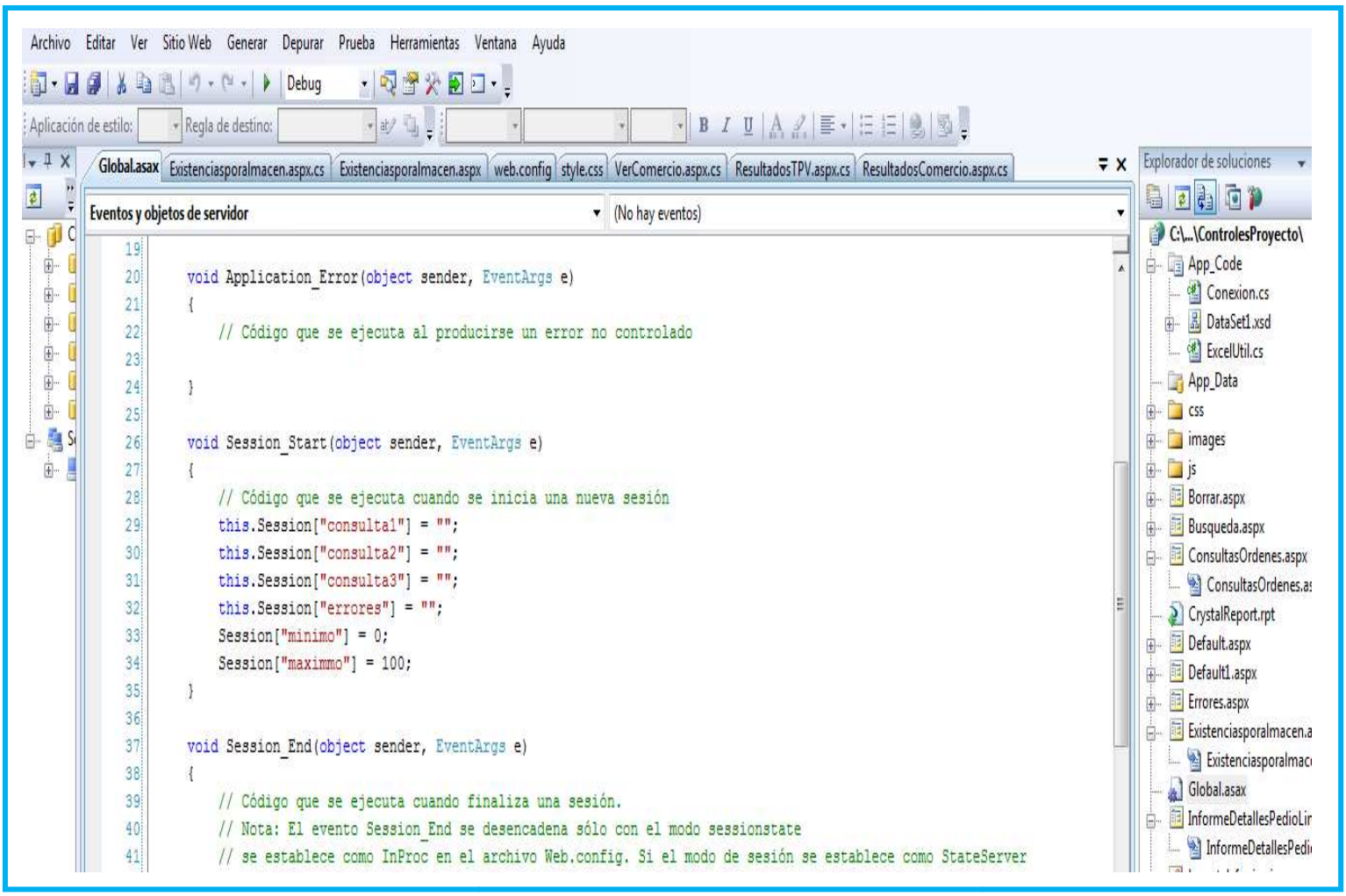


Fig. 3.8 Fichero global.asax

La carpeta **App_Code** contiene los siguientes ficheros:

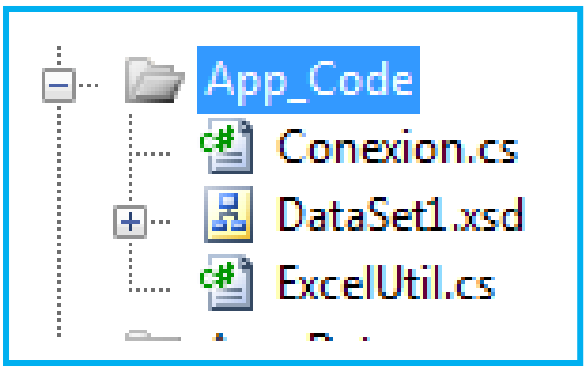


Fig. 3.9 Carpeta App_Code

"Conexion.cs" es una clase que contiene información sobre la conexión a la Base de datos, como por ejemplo la cadena de conexión.

El fichero **DataSet1.xsd** es un fichero que lo utilizaremos en **Crystal Report** para la generación de informes.

El fichero **ExcelUtil.cs** contiene el código que nos permitirá generar documentos **Excel** en lado del cliente. Este fichero se explicará en detalle más adelante.

3.3.1. Interacción con el usuario

El usuario deberá en primer lugar "loguearse" en la página **"Default1.aspx"**:

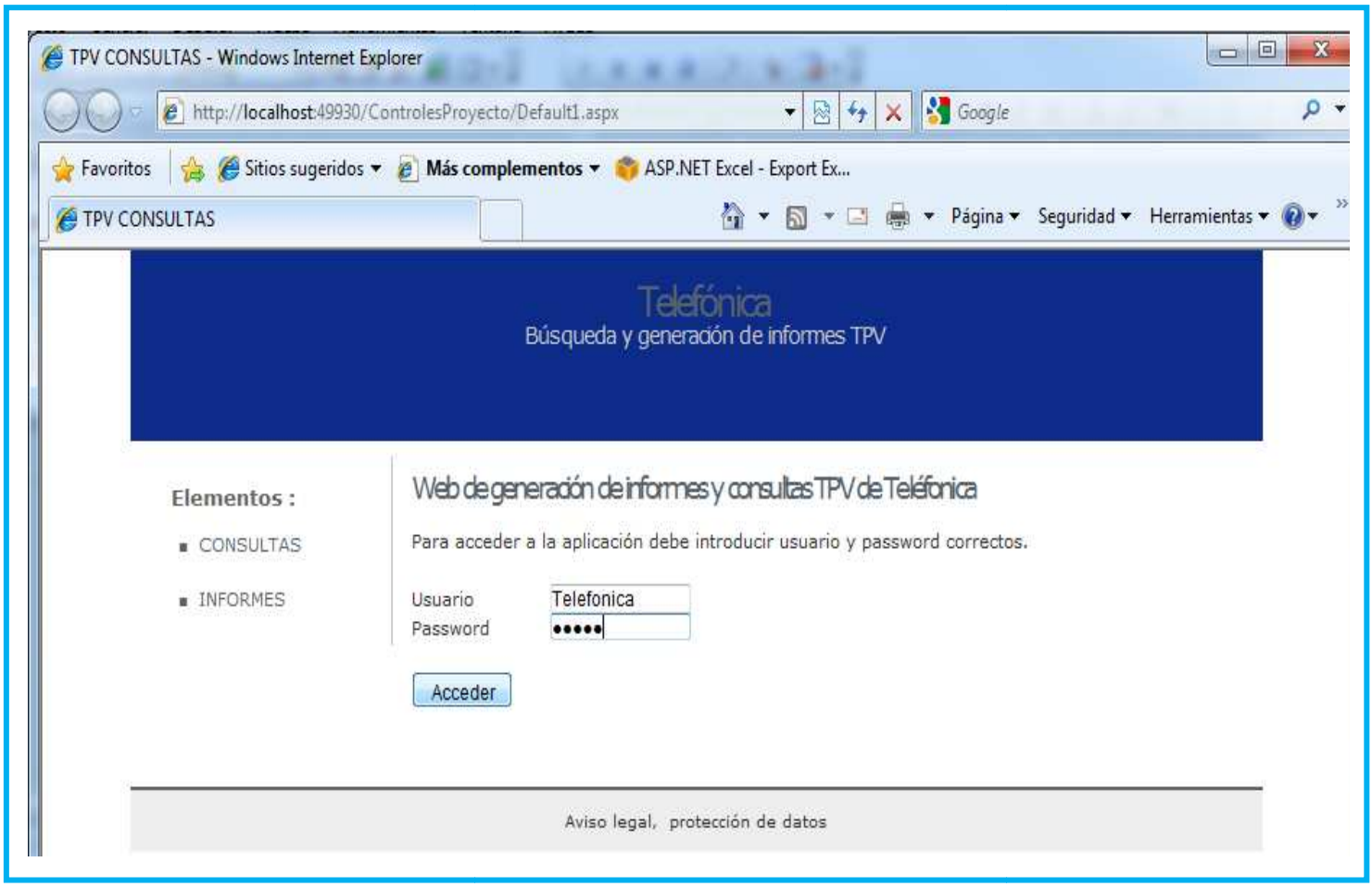


Fig. 3.10 Formulario Validación de usuarios

El código por parte del servidor comprueba que los datos introducidos concuerdan con los datos que hay en el fichero **"web.config"**. No se hace "logueo" en la Base de Datos ya que el cliente no lo solicitó como requisito funcional.

Una vez "logueados" se accede a la web que ofrece la funcionalidad de la aplicación.

Formulario de **menú de entrada**:

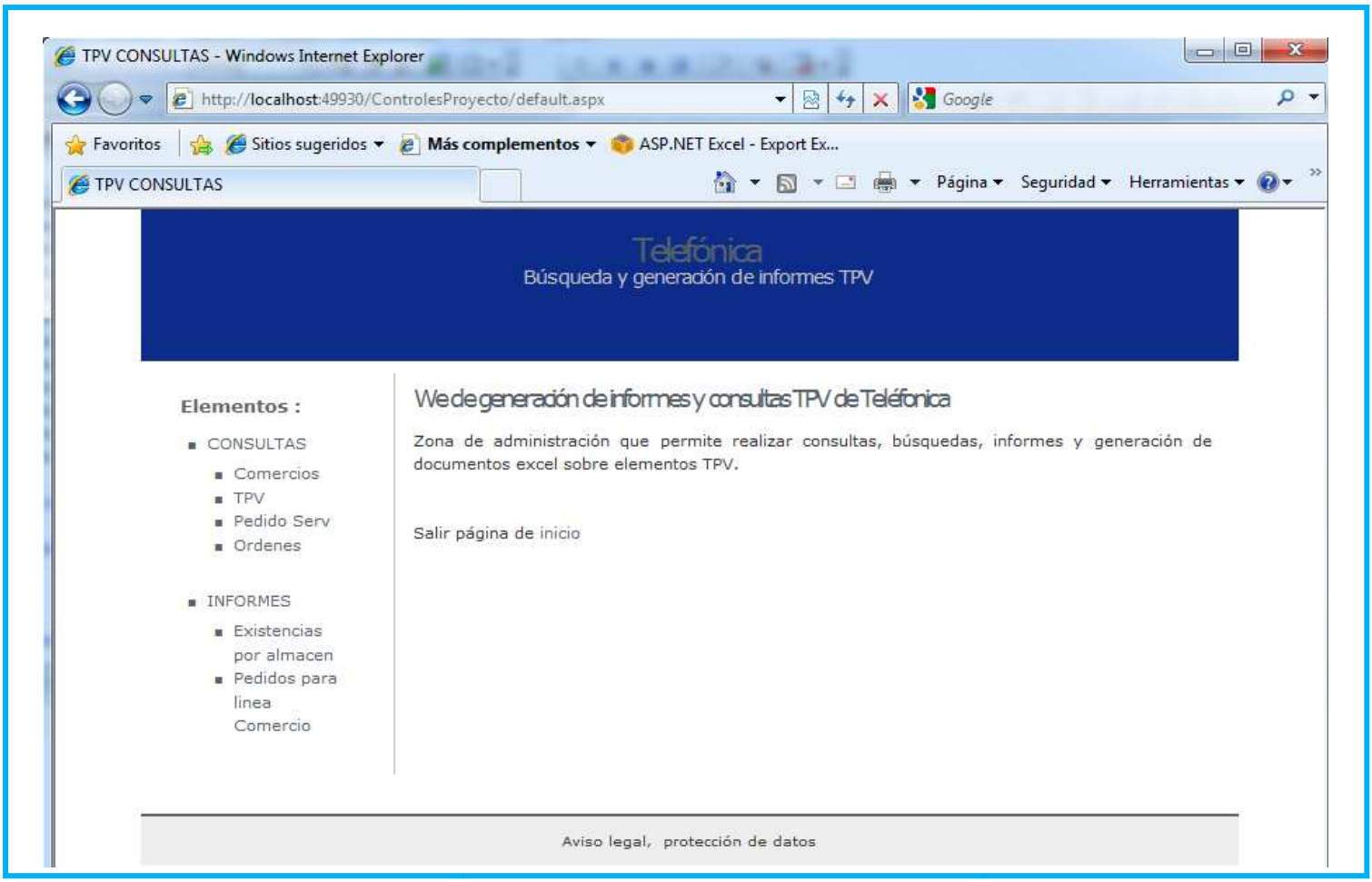


Fig. 3.11 Formulario del usuario "Telefónica"

Este formulario **permite al usuario elegir una acción** reflejada en el menú de la izquierda de la imagen.

Si pulsamos sobre ***Comercios*** accederemos a la web que nos permite hacer consultas sobre los comercios que tienen **TPV´S**

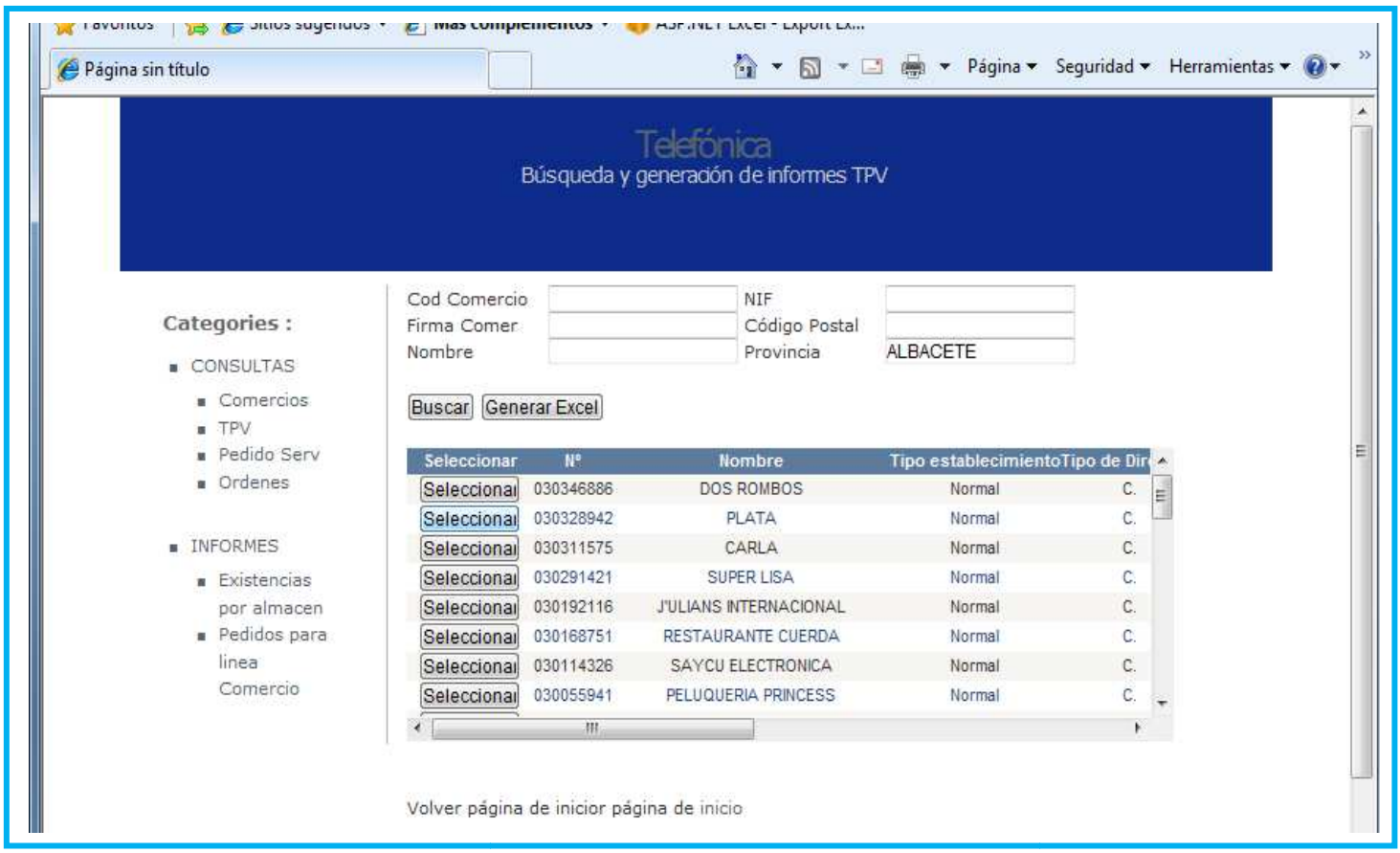


Fig. 3.12 Formulario "Comercios"

A continuación se muestra una parte del código del formulario que se ejecuta al pulsar el botón **"Buscar"**:

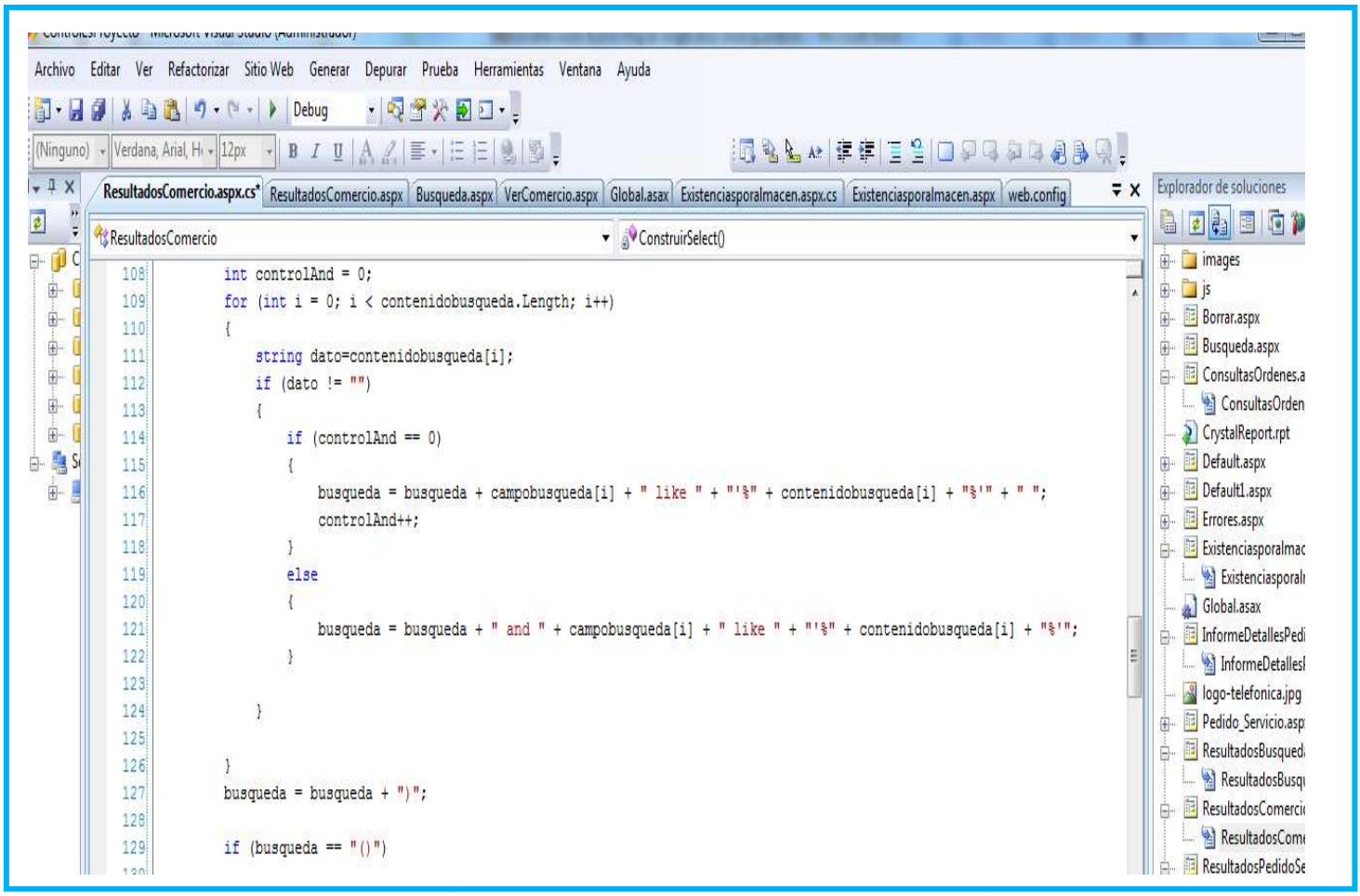


Fig. 3.13 Código formulario "Comercios"

El problema de construir una ***Select*** de esta forma es que permite que un usuario malintencionado pueda lanzar código ***SQL inyectado***, esto hace que pueda borrar la base de datos, o peor, borrar o alterar datos individuales, e incluso acceder como administrador del sistema con todo lo que conlleva. Por ejemplo al construir la **sentencia "*where*"** mediante la concatenación de cadenas utilizando *dobles comillas y comillas simples* podría hacerse un **"*Where '1=1' "*** y añadiendo otra sentencia, se podrían consultar datos no permitidos o crear mayores problemas.

Esto se puede evitar de varias formas, una de ellas consiste en utilizar la clase ***SqlParameter*** que al menos evita que en un campo se puedan poner comillas simples o dobles. Otra es haciendo que cuando un usuario introduce en un campo de búsqueda una comilla simple o doble, sustituir ese carácter por cualquier carácter válido o anularlo.

Otro botón interesante en este formulario es ***Generar Excel*** con el siguiente código asociado:

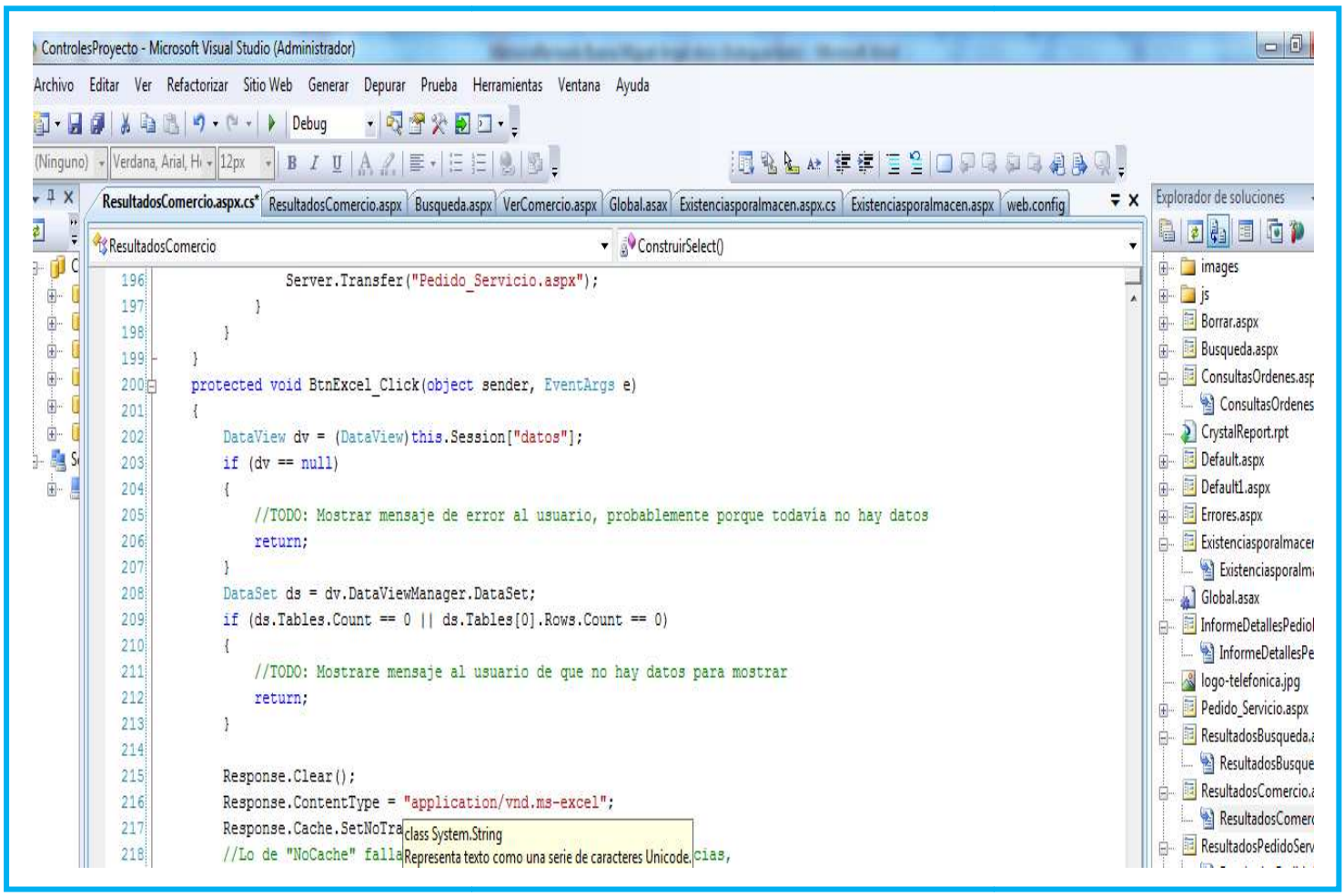


Fig. 3.14 Código botón "Generar Excel"

El código del botón anterior utiliza la funcionalidad que nos ofrece el fichero **ExcelUtil.cs**, lo que hace básicamente es crear un fichero de texto **XML** e indicarle al cliente que debe abrirlo con **Excel** de la siguiente manera:

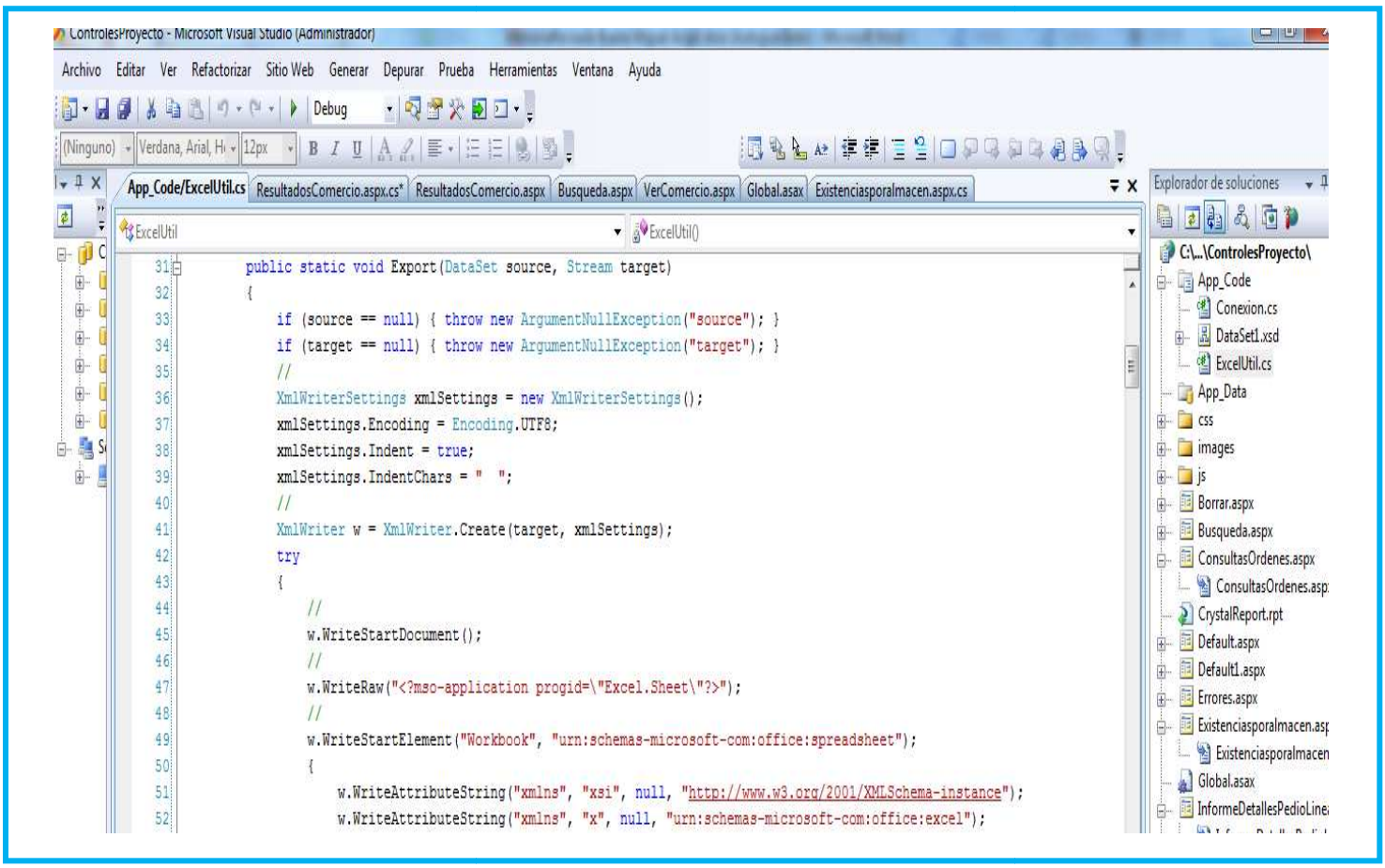
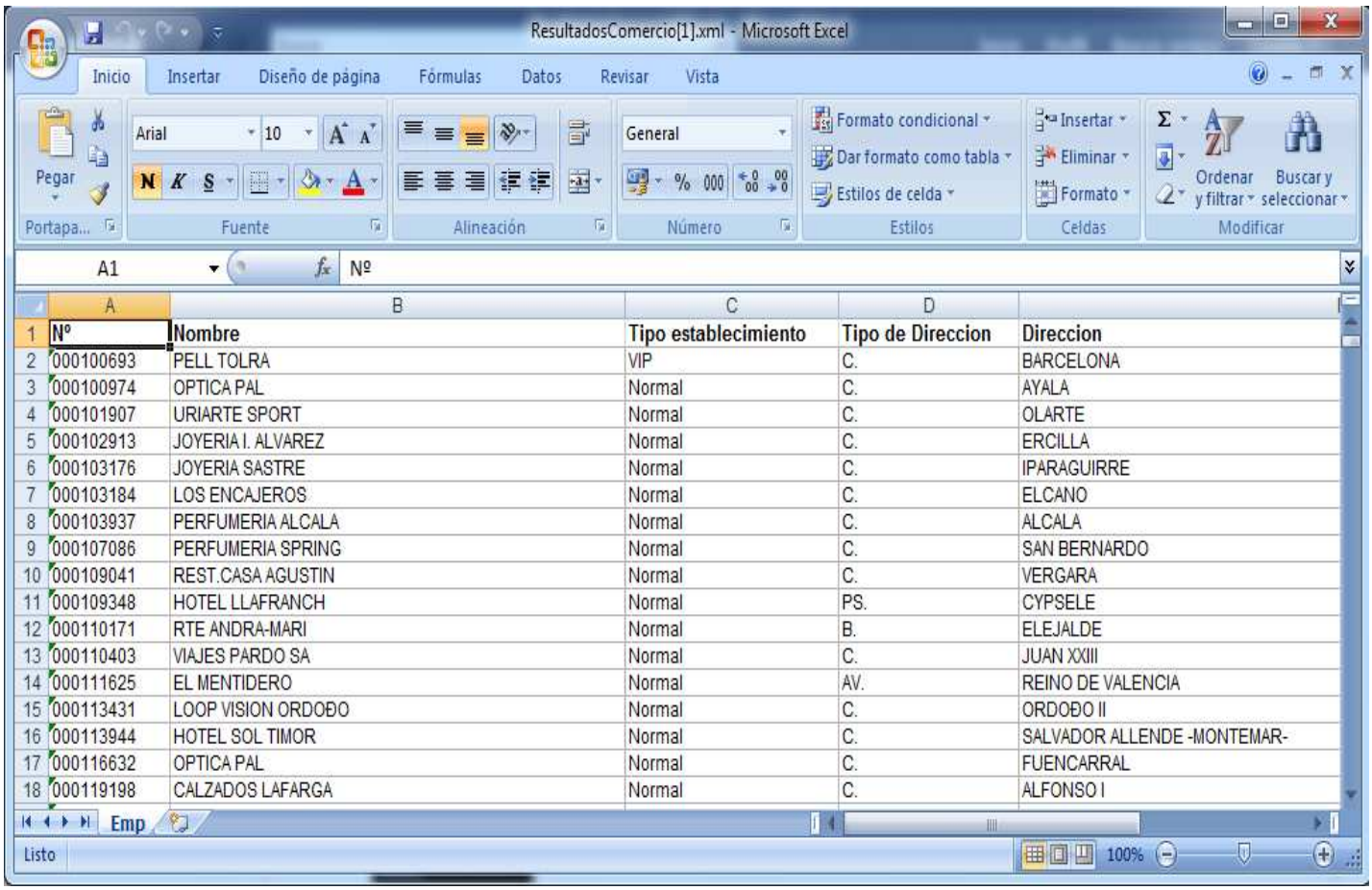


Fig. 3.15 Código fichero "ExcelUtil"

Este código que se encuentra en el fichero ***"ExcelUtil.cs"*** permite abrir o guardar un fichero Excel en el cliente, siempre que el cliente tenga instalado **Microsoft Office**. Si el cliente no lo tiene instalado el explorador lo intentará abrir y se podrá visualizar la información del fichero en formato ***XML***.

Captura de un ejemplo de **generación de un fichero "Excel"**:



Nº	Nombre	Tipo establecimiento	Tipo de Direccion	Direccion
000100693	PELL TOLRA	VIP	C.	BARCELONA
000100974	OPTICA PAL	Normal	C.	AYALA
000101907	URIARTE SPORT	Normal	C.	OLARTE
000102913	JOYERIA I. ALVAREZ	Normal	C.	ERCILLA
000103176	JOYERIA SASTRE	Normal	C.	IPARAGUIRRE
000103184	LOS ENCAJEROS	Normal	C.	ELCANO
000103937	PERFUMERIA ALCALA	Normal	C.	ALCALA
000107086	PERFUMERIA SPRING	Normal	C.	SAN BERNARDO
000109041	REST.CASA AGUSTIN	Normal	C.	VERGARA
000109348	HOTEL LLAFRANCH	Normal	PS.	CYPSELE
000110171	RTE ANDRA-MARI	Normal	B.	ELEJALDE
000110403	VIAJES PARDO SA	Normal	C.	JUAN XXIII
000111625	EL MENTIDERO	Normal	AV.	REINO DE VALENCIA
000113431	LOOP VISION ORDOÐO	Normal	C.	ORDOÐO II
000113944	HOTEL SOL TIMOR	Normal	C.	SALVADOR ALLENDE -MONTEMAR-
000116632	OPTICA PAL	Normal	C.	FUENCARRAL
000119198	CALZADOS LAFARGA	Normal	C.	ALFONSO I

Fig. 3.16 Fichero Excel generado

Nota: para generar documentos **Excel ASP.NET** ofrece varias posibilidades, se ha elegido esta última por considerar que es la más rápida, ligera y eficiente del lado del servidor.

A continuación se describen algunas de las formas de generar **documentos Excel** junto con los problemas que podemos encontrarnos:

Para generar un archivo de Excel desde un Servidor Web con **ASP.NET** hay varias alternativas:

- Hacerlo **generando un simple archivo de texto XML** compatible únicamente con las últimas versiones de Office (probado en 2007, y 2003)
- **Utilizar un componente** o librería específicos asegurándose que sea indicado para su uso dentro de un servidor web
- **Usar automatización COM/OLE de Excel** para que sea el propio Excel el que genere el archivo. Ésta forma de hacerlo está totalmente desaconsejada para un servidor Web, puesto que al procesar las distintas peticiones web de forma simultánea, en varios "hilos" independientes, seguro que da problemas puesto que Excel es una aplicación de usuario que no está optimizada para trabajar en múltiples hilos independientes dentro del mismo proceso de **ASP.NET**. En cualquier caso también supone una sobrecarga excesiva en el servidor, en cuanto a memoria, uso de disco y CPU entre otros recursos, por lo que múltiples peticiones simultáneas de éste tipo (bastarían muy pocas) podrían dejar el servidor fuera de combate.

Captura del ***Formulario Ordenes***:

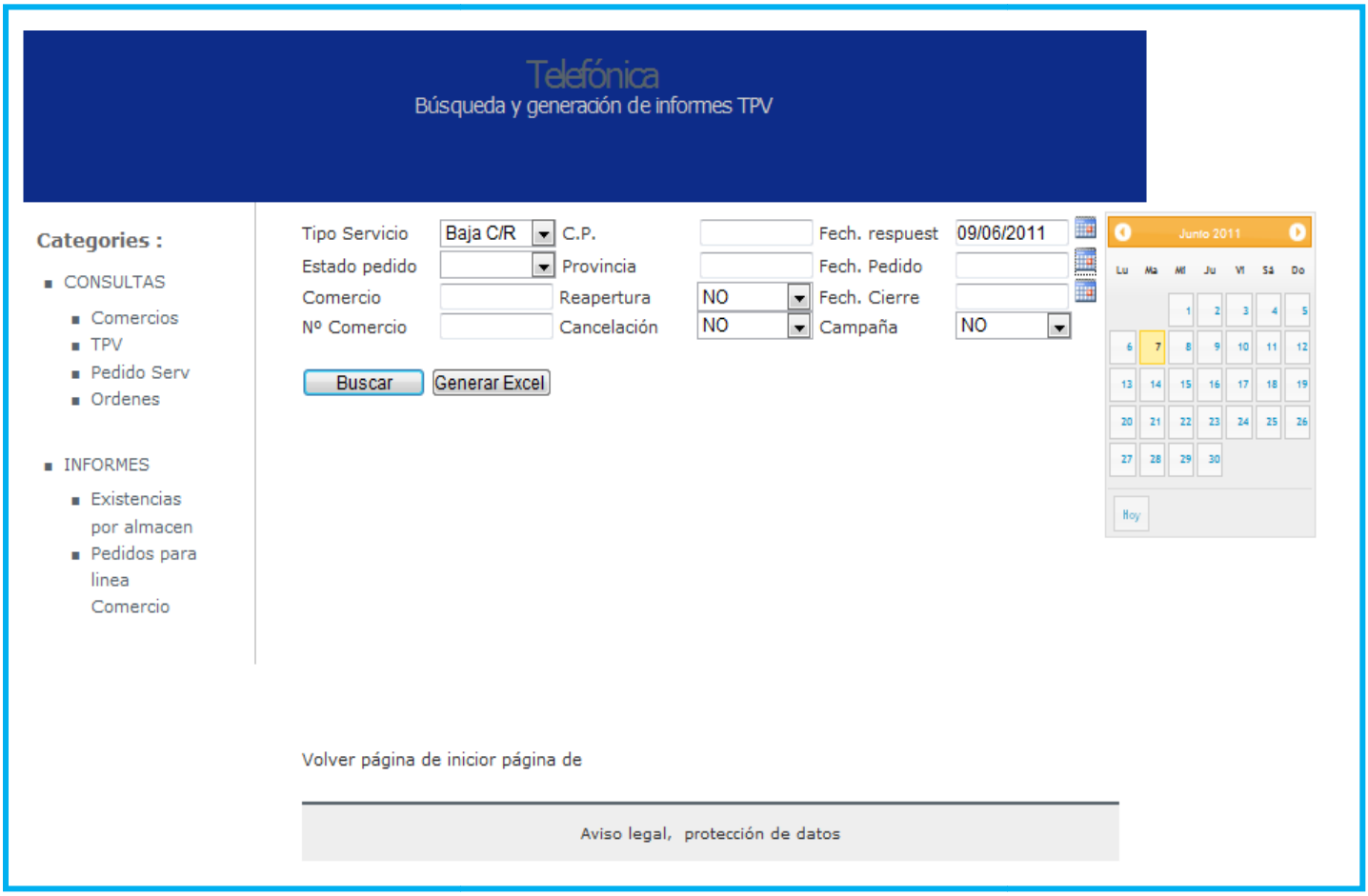


Fig. 3.17 Formulario "Ordenes"

Del menú que aparece a la izquierda, los enlaces ***TPV y Pedido Serv*** son análogos al primero.

Aparece un formulario en el cual hay **un calendario**, para que el usuario pueda seleccionar distintas fechas de búsqueda.

En un principio se utilizó el control ***Calendar*** que es nativo de **Visual Studio.NET**, el problema de utilizar este control es que al elegir una fecha, el control envía la fecha seleccionada al servidor para que la trate. Para los propósitos de esta aplicación esto es absurdo puesto que tan solo se necesita hacer un **"submit"** al pulsar el botón *Buscar*.

Por lo tanto se hizo necesario utilizar un control menos pesado y que escribiera la fecha seleccionada en el control **"Caja de texto"**, sin necesidad de ir al servidor. Por eso se utilizó el control ***datepicker*** **de la librería jQuery** que se trata en el cliente y no hace necesario ir al servidor.

El **código JavaScript** para el tratamiento de este control se encuentra en **la carpeta *js,*** dentro del fichero ***clientadminUtil.js***, el funcionamiento es el siguiente:

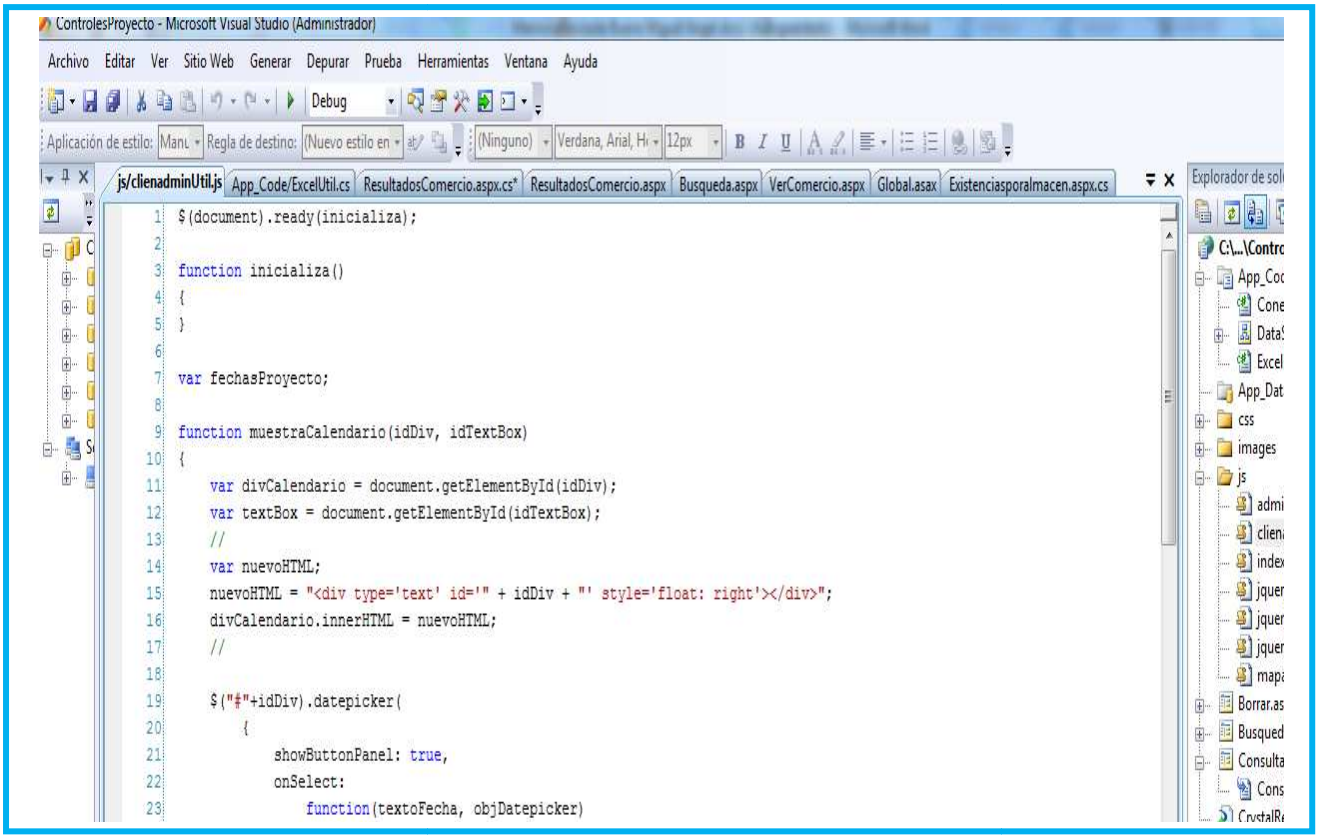


Fig. 3.18 Fichero "clienAdminUtil.js"

Básicamente, consiste en poner en la caja de texto la fecha seleccionada y destruir el control una vez se ha seleccionado una fecha. Este control hace mucho más ligero el funcionamiento del formulario.

Se puede observar que todas las páginas responden al patrón de diseño **"Modelo Vista Controlador" o MVC**, puesto que tienen un **código HTML**, un **código JavaScript** que se encuentra en un fichero que ofrece la funcionalidad deseada, y por último el Modelo lo ofrece la página *style.css* que contiene los estilos de todas las páginas.

También tenemos el fichero el formulario **"ResultadosPedidoServicio.aspx"** que nos permite hacer búsquedas de los distintos servicios ofrecidos a un cliente en concreto sobre elementos TPV.

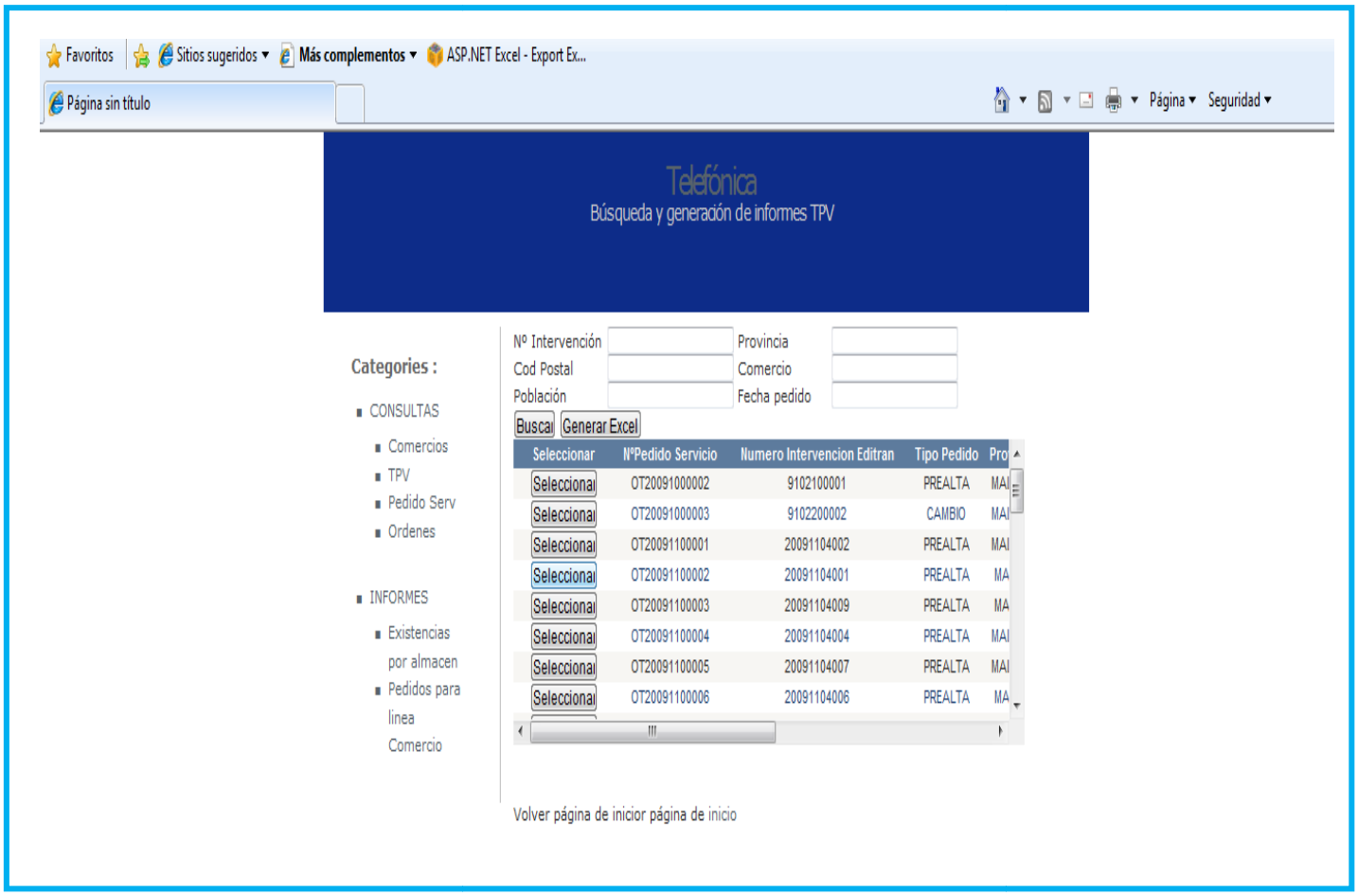


Fig. 3.19 Formulario "ResultadosPedidoServicio"

Si sobre el formulario anterior seleccionamos un servicio, pulsando el botón **"Seleccionar"** del control **"Datagrid"**, accederemos al siguiente formulario.

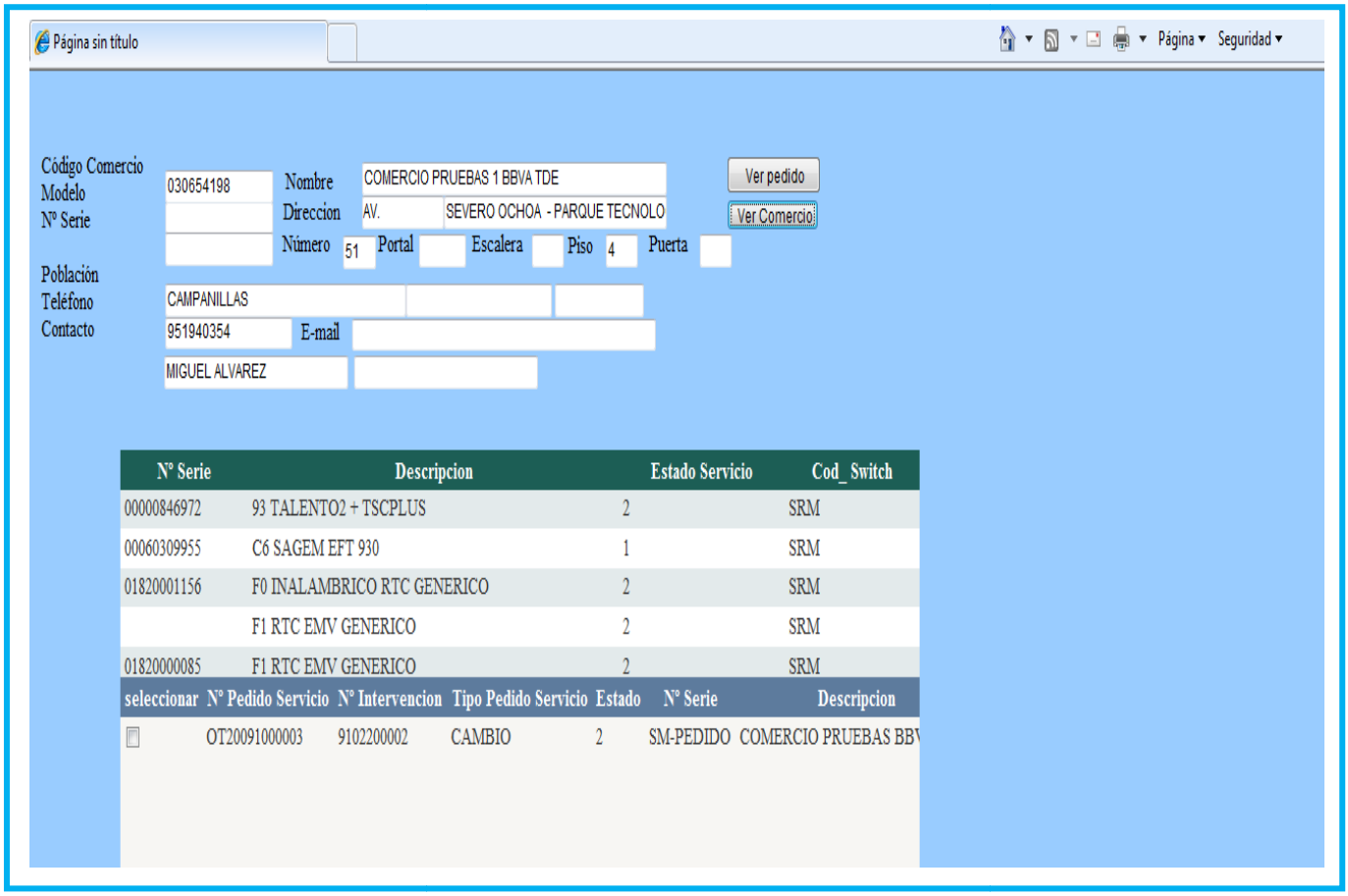


Fig. 3.20 Formulario "ResultadosBusqueda"

Podemos observar que **el formato del formulario cambia**, esto es **debido a petición expresa del cliente**.

En este formulario se muestra por un lado información sobre el comercio elegido, los distintos **elementos TPV** que tiene ese comercio y por último los **servicios ofrecidos sobre esos TPV**.

Si pulsamos sobre el **botón "Ver pedido"** accederemos al siguiente formulario:

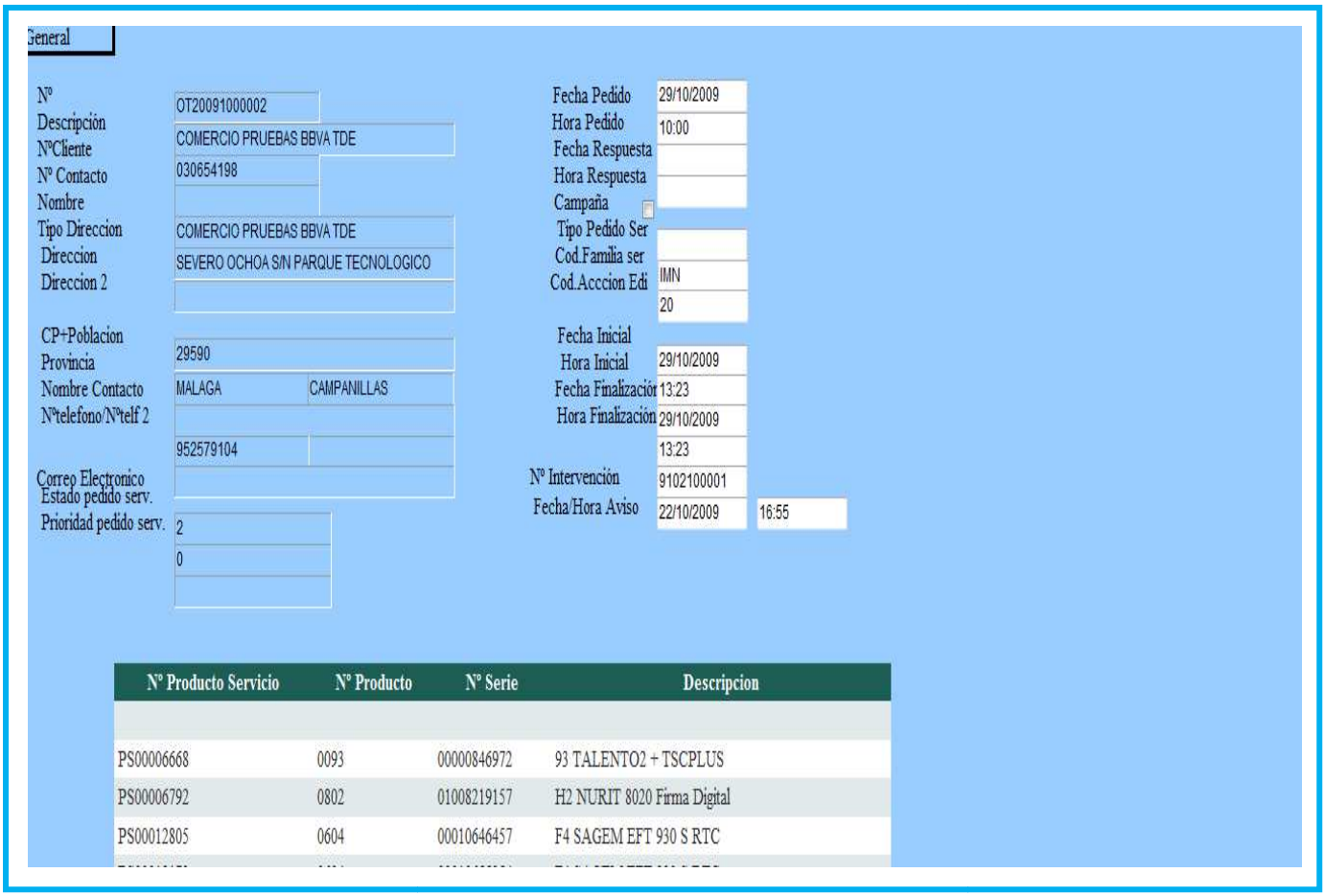


Fig. 3.21 Formulario "PedidoServicio"

Este formulario muestra información sobre el pedido realizado.

Si pulsamos sobre el **botón "Ver comercio"** podremos ver información del comercio que realiza el pedido.

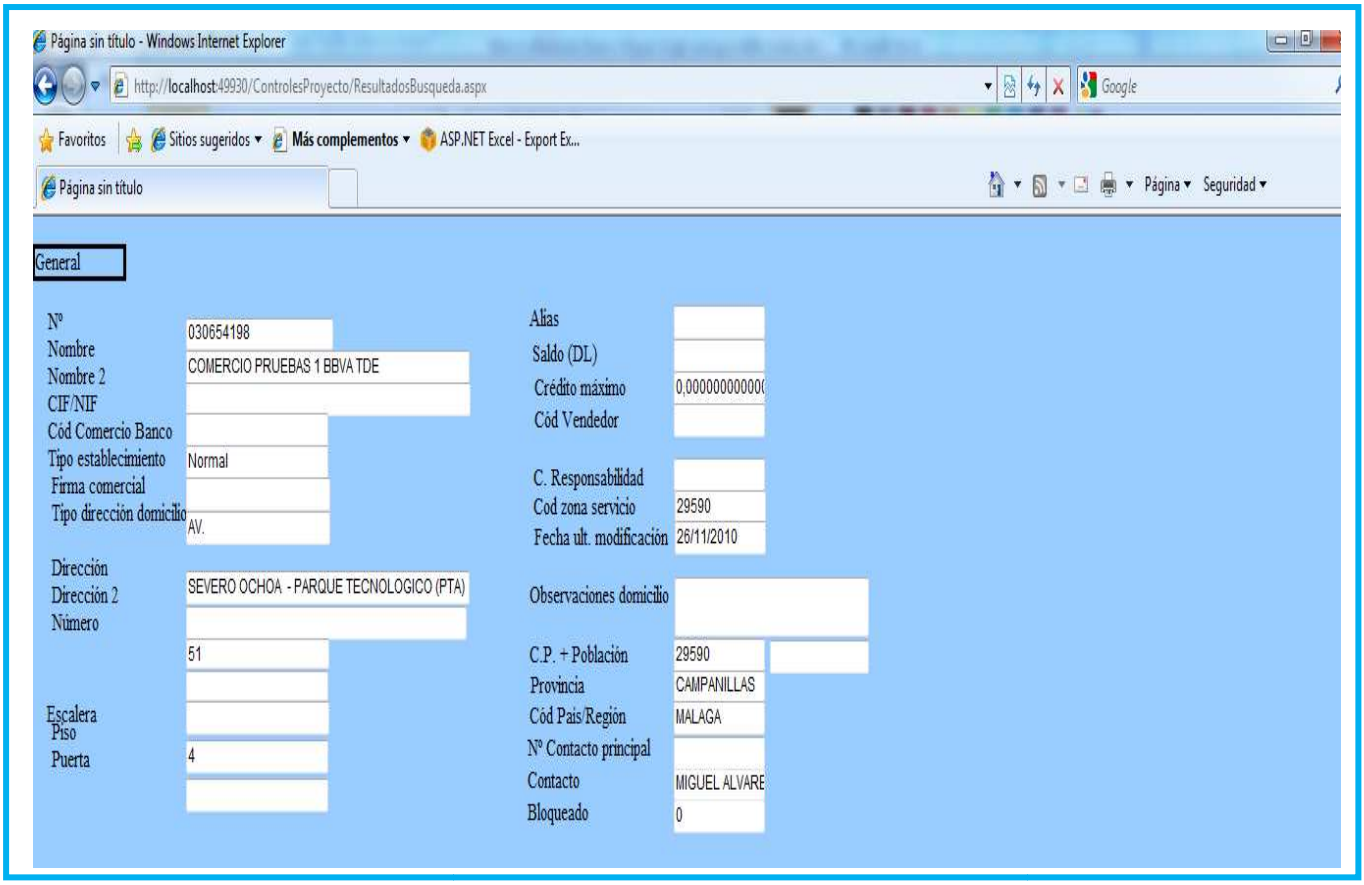
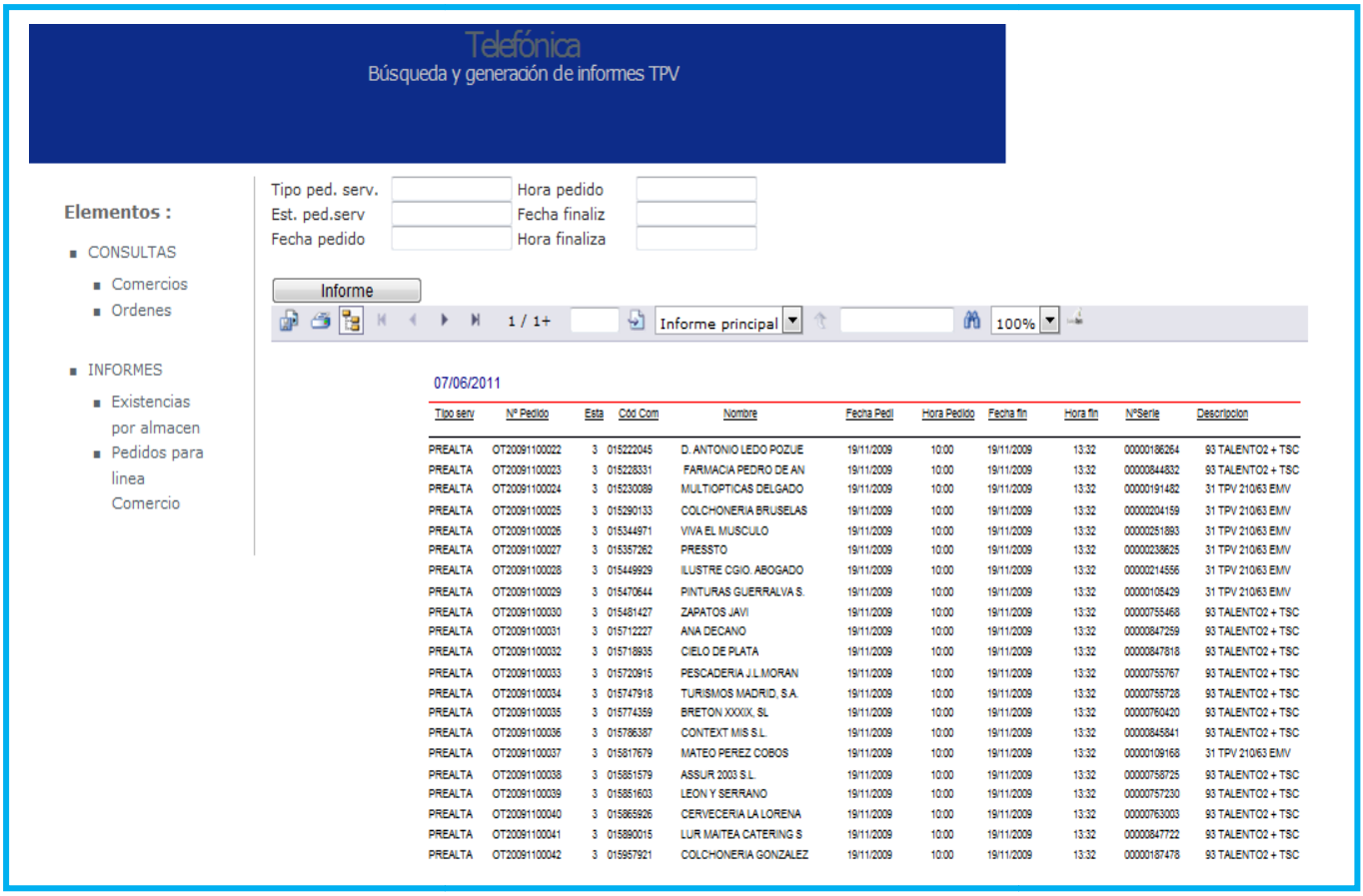


Fig. 3.22 Formulario "InfoComercio"

Aquí podemos ver información adicional y concreta sobre el **"comercio"** referido.

Siguiendo con el menú de la izquierda, nos quedan los informes, si pulsamos sobre el enlace ***"Pedidos para Linea Comercio"*** obtenemos el siguiente formulario:



Tipo serv	Nº Pedido	Esta	Cód Com	Nombre	Fecha Pedi	Hora Pedido	Fecha fin	Hora fin	NºSerie	Descripcion
PREALTA	OT20091100022	3	015222045	D. ANTONIO LEDO POZUE	19/11/2009	10:00	19/11/2009	13:32	00000186264	93 TALENTO2 + TSC
PREALTA	OT20091100023	3	015228331	FARMACIA PEDRO DE AN	19/11/2009	10:00	19/11/2009	13:32	00000844832	93 TALENTO2 + TSC
PREALTA	OT20091100024	3	015230089	MULTIOPTICAS DELGADO	19/11/2009	10:00	19/11/2009	13:32	00000191482	31 TPV 210/63 EMV
PREALTA	OT20091100025	3	015290133	COLCHONERIA BRUSELAS	19/11/2009	10:00	19/11/2009	13:32	00000204159	31 TPV 210/63 EMV
PREALTA	OT20091100026	3	015344971	VIVA EL MUSCULO	19/11/2009	10:00	19/11/2009	13:32	00000251893	31 TPV 210/63 EMV
PREALTA	OT20091100027	3	015357262	PRESSTO	19/11/2009	10:00	19/11/2009	13:32	00000238625	31 TPV 210/63 EMV
PREALTA	OT20091100028	3	015449929	ILUSTRE CGIO. ABOGADO	19/11/2009	10:00	19/11/2009	13:32	00000214556	31 TPV 210/63 EMV
PREALTA	OT20091100029	3	015470644	PINTURAS GUERRALVA S.	19/11/2009	10:00	19/11/2009	13:32	00000105429	31 TPV 210/63 EMV
PREALTA	OT20091100030	3	015481427	ZAPATOS JAVI	19/11/2009	10:00	19/11/2009	13:32	00000755468	93 TALENTO2 + TSC
PREALTA	OT20091100031	3	015712227	ANA DECANO	19/11/2009	10:00	19/11/2009	13:32	00000847259	93 TALENTO2 + TSC
PREALTA	OT20091100032	3	015718935	CIELO DE PLATA	19/11/2009	10:00	19/11/2009	13:32	00000847818	93 TALENTO2 + TSC
PREALTA	OT20091100033	3	015720915	PESCADERIA J.L.MORAN	19/11/2009	10:00	19/11/2009	13:32	00000755767	93 TALENTO2 + TSC
PREALTA	OT20091100034	3	015747918	TURISMOS MADRID, S.A.	19/11/2009	10:00	19/11/2009	13:32	00000755728	93 TALENTO2 + TSC
PREALTA	OT20091100035	3	015774359	BRETON XXXIX, SL	19/11/2009	10:00	19/11/2009	13:32	00000760420	93 TALENTO2 + TSC
PREALTA	OT20091100036	3	015786387	CONTEXT MIS S.L.	19/11/2009	10:00	19/11/2009	13:32	00000845841	93 TALENTO2 + TSC
PREALTA	OT20091100037	3	015817679	MATEO PEREZ COBOS	19/11/2009	10:00	19/11/2009	13:32	00000109168	31 TPV 210/63 EMV
PREALTA	OT20091100038	3	015851579	ASSUR 2003 S.L.	19/11/2009	10:00	19/11/2009	13:32	00000758725	93 TALENTO2 + TSC
PREALTA	OT20091100039	3	015851603	LEON Y SERRANO	19/11/2009	10:00	19/11/2009	13:32	00000757230	93 TALENTO2 + TSC
PREALTA	OT20091100040	3	015865926	CERVECERIA LA LORENA	19/11/2009	10:00	19/11/2009	13:32	00000763003	93 TALENTO2 + TSC
PREALTA	OT20091100041	3	015890015	LUR MAITEA CATERING S	19/11/2009	10:00	19/11/2009	13:32	00000847722	93 TALENTO2 + TSC
PREALTA	OT20091100042	3	015957921	COLCHONERIA GONZALEZ	19/11/2009	10:00	19/11/2009	13:32	00000187478	93 TALENTO2 + TSC

Fig. 3.23 Formularo "InformesPedidoLineaComercio"

En este formulario al pulsar sobre *buscar,* se genera un informe creado con ***Crystal Report***, el parte de este código es el siguiente:

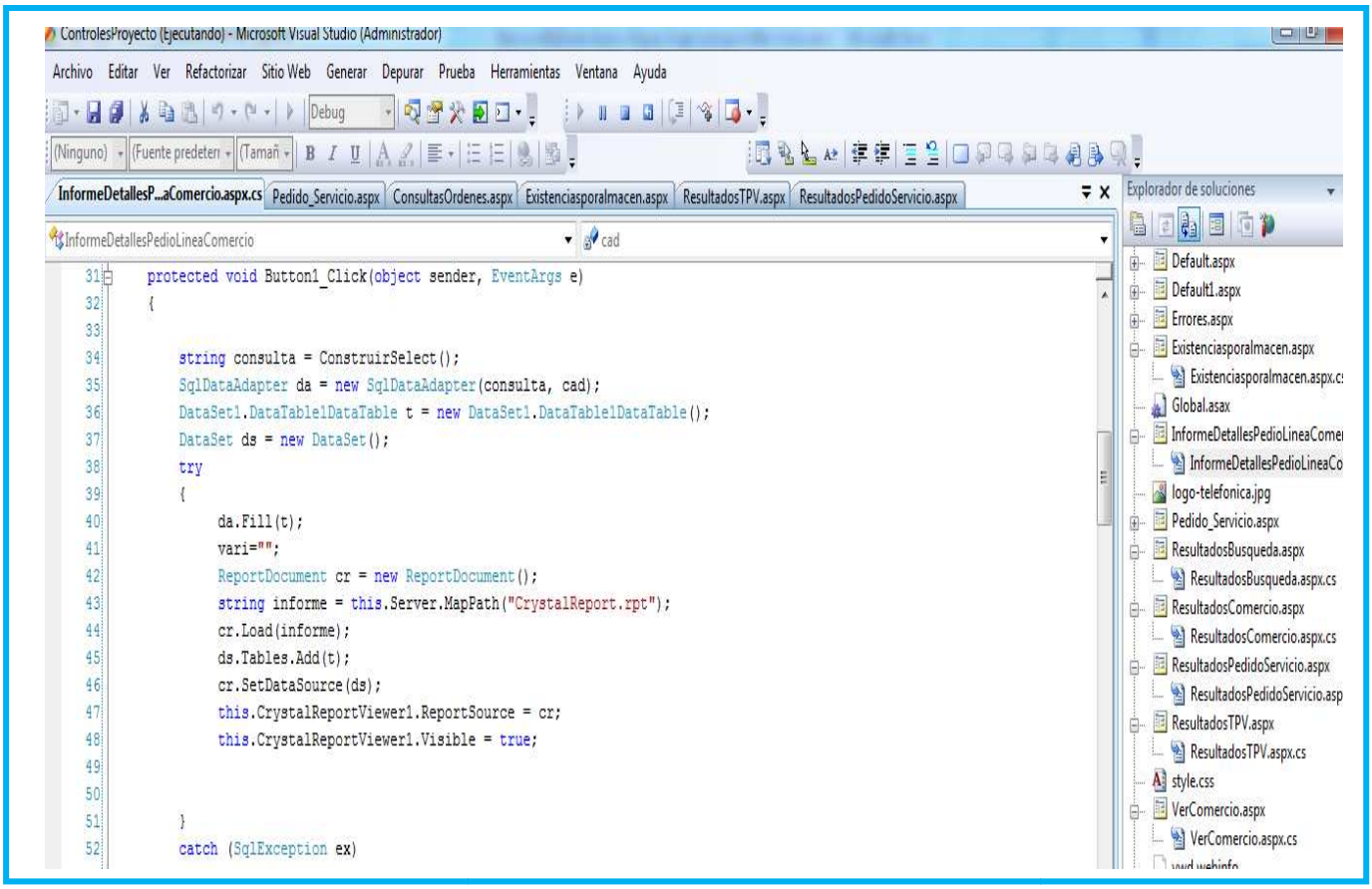


Fig. 3.23 Informe "Crystal Report"

3.3.2. Consideraciones adicionales

ASP.NET permite varias formas de autenticación de usuarios:

- **Utilizando el fichero de configuración "Web.config".** Esta es sencilla y rápida, pero ofrece varias debilidades por ejemplo cualquier administrador de sistema que tenga acceso al fichero puede ver las password, además en principio no se hace Hash de las password.
- **Autenticación mediante acceso a servidor de dominio**. En este caso se utilizan usuarios del dominio. Es aconsejable cuando existe un servidor de dominio y los usuarios pertenecen al mismo, como es en el caso de una Intranet. El problema es (a parte del pago de licencias que hay que realizar por cada usuario) es el problema de la propia infraestructura y la necesidad de un administrador que de de alta/baja a los usuarios.
- **Utilizando la propia base de datos de usuarios que crea ASP.NET** que se llama ASPNETDB, es una base de datos separada de la propia base de datos de la aplicación lo cual puede traer varias desventajas ya que podría haber tablas de la base de datos propia de la aplicación que utilicen claves foráneas de los usuarios logueados.
- **Personalizada**. En este caso se podrían utilizar las propias tablas de la aplicación y guardar únicamente el Hash de la password, además se podrían crear rutinas personalizadas como por ejemplo si está al corriente de pago, etc.

Por otro lado se hace necesario indicar que se ha elegido la **tecnología ASP.NET frente a otras tecnologías como por ejemplo PHP**, por las siguientes razones:

1. ASP.NET tiene un **tipado fuerte de datos frente a PHP**, esto permite compilar el código y verificar errores sintácticos antes de su puesta en producción, mientras que en PHP es más difícil saber si se está utilizando un tipo de datos incorrecto.
2. En ASP.NET te **permite trabajar con varios lenguajes de programación** mientras que PHP sólo puede trabajar en un tipo de lenguaje y esto restringe el número de desarrolladores.
3. ASP.NET es un **lenguaje totalmente Orientado a Objetos**, con todas las ventajas que ello implica, mientras que si bien PHP en sus últimas versiones es Orientado a Objetos, sus librerías no lo son.
4. **PHP es interpretado mientras que ASP.NET es compilado a un código ejecutable** por la máquina virtual.
5. La documentación existente en ASP.NET así como el número de profesionales dedicados a labores docentes es mucho mayor que en PHP.
6. ASP.NET **es mucho más rápido de desarrollar que PHP**, debido al patrón de diseño utilizado de ASP.NET, frente al patrón clásico utilizado por PHP o las antigüas versiones de ASP.NET.
7. **ASP.NET como es una tecnología relativamente novedosa** se ha desarrollado intentando solventar las carencias que se observaban en PHP o Java.

4. Actas

A continuación incluimos las actas redactadas durante el desarrollo del proyecto:

Fecha: 16 de noviembre de 2010
Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

Se propone al alumno por parte del tutor una aplicación de gestión de un SAT (Servicio de Asistencia Técnica), que sirva para gestionar y registrar las actividades que se realizan sobre productos TPV. Estas actividades son iniciadas ante la necesidad de realizar una actuación de reparación, sustitución o instalación de los dispositivos TPV.

Se indica que sería interesante que esta aplicación pudiera soportar consultas via Web. Se indica que es necesario estudiar qué tecnología es más adecuada para la realización de la aplicación. Se propone realizar un estudio de los pros y contras que tienen las tecnologías propuestas. El director de proyecto propone ASP.NET o PHP con Bases de datos Microsoft SQL Server o MySql.

Durante la realización del proyecto se harán reuniones semanales, las cuales quedarán registradas en actas que servirán para la fase de análisis funcional. En las reuniones se discutirá el trabajo realizado durante la semana y se decidirán los puntos a trabajar durante la siguiente semana.

También se ha decidido dividir el proyecto en cuatro fases:

1. Análisis funcional.

2. Diseño técnico.

3. Creación de un prototipo.

4. Redacción de la memoria.

Fecha: 23 de noviembre de 2010

Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

El alumno presenta una lista de pros y contras de las tecnologías propuestas. Haciendo hincapié en que para él sería más sencillo realizar el prototipo de la aplicación en ASP.NET con sistema gestor de bases de datos Microsoft SQL Server, ya que conoce la tecnología.

Es necesario realizar el diseño de la base de datos para lo cual se le da una documentación de la empresa en la que se explica como funciona la gestión de un SAT.

Se propone para la siguiente reunión la realización del diseño de la base de datos.

Fecha: 30 de noviembre de 2010

Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

El alumno presenta las listas categorizadas de las distintas entidades que conforman la base de datos. El director de proyecto realiza un estudio del modelo de datos presentado por el alumno e indica algunas correcciones al modelo propuesto.

Se propone para la próxima reunión la implementación de éste modelo de datos en SQL Server. Con datos ficticios para poder probar algunas consultas.

Fecha: 14 de diciembre de 2010

Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

El alumno presenta una base de datos con algunas consultas que demuestran el funcionamiento de la misma.

Para la siguiente reunión se solicita la realización de una web con formularios que muestren información sobre algunas consultas frecuentes por los operarios de la empresa. Se indica que esta web debe ser ligera y que debe funcionar en un servidor web IIS (Internet Information Service). Se hace necesario aprender a configurar correctamente un IIS.

Fecha: 17 de enero de 2011

Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

El alumno muestra un prototipo operativo de una web que contiene controles sobre los cuales recaen los datos de las distintas consultas realizadas por los operarios de la empresa.

Tras algunas modificaciones sobre estas consultas se solicita al alumno que realice un diseño más atractivo de la web ya que esta presenta una funcionalidad adecuada, pero la presentación de los datos no es muy atractiva. Se indica que se realice para la próxima reunión una hoja de estilos CSS adecuada.

Por otro lado el director de proyecto observa que algunas consultas que incluyen campos de fecha. No funcionan correctamente por lo que se pide que se utilice un control adecuado para la selección de las fechas.

Fecha: 10 de febrero de 2011

Asistentes: Miguel Ángel Blanco Rodríguez y Pedro Pablo García

Se presenta una web con las correcciones solicitadas por el director de proyecto. El alumno indica que ha añadido a la web un control "calendario" jsp, ya que éste se gestiona en cliente y es más rápido que el control "calendar" nativo que tiene ASP.NET. Ahora las consultas funcionan correctamente.

Se indica al alumno que la próxima reunión se realizará en el cliente, al cual habrá que mostrarle el funcionamiento de la aplicación.

Fecha: 21 de febrero de 2011

Asistentes: Miguel Ángel Blanco Rodríguez, Pedro Pablo García y representantes de Telefónica.

Se hace una presentación de la aplicación y de su funcionamiento. El cliente solicita algunas modificaciones, por ejemplo solicita que las consultas realizadas por los operarios además de volcarse sobre controles web, puedan ser descargadas en formato Excel.

También solicita que un operario tenga que logearse antes de poder utilizar la aplicación.presenta una web con las correcciones solicitadas por el director de proyecto.

Fecha: 10 de abril de 2011

Asistentes: Miguel Ángel Blanco Rodríguez, Pedro Pablo García

Una vez realizadas las modificaciones solicitadas por el cliente final. Se indica que se realice un estudio sobre la seguridad del sistema ya que se ha detectado que puede haber código inyectado en la consultas y se podría atacar la base de datos.

Fecha: 18 de abril de 2011

Asistentes: Miguel Ángel Blanco Rodríguez, Pedro Pablo García

El alumno indica que para poder proteger de código inyectado es necesario modificar algunas consultas, por ejemplo se hace necesario la utilización de la clase SqlParameter, para minimizar el posible daño producido por un usuario malintencionado, además de crear rutinas propias de seguridad.

Se indica al alumno que esto último queda fuera del alcance del proyecto y que de momento no se va a realizar.

Fecha: 21 de abril de 2011

Asistentes: Miguel Ángel Blanco Rodríguez, Pedro Pablo García

Se observa que sobre las consultas sólo devuelven un número determinado de registros y esto podría plantearse como un problema para el cliente. Se pide al alumno que estudie una posible solución para la siguiente reunión.

Fecha: 10 de mayo de 2011

Asistentes: Miguel Ángel Blanco Rodríguez, Pedro Pablo García

El alumno indica que hay varias soluciones, entre ellas está el uso de procedimientos almacenados que reciban parámetros en los que se indique el número de registros y el número a partir del cual se debe empezar a devolver información.

Otra opción es en la misma "select" hacer una "select" de una "select", pero debido a que es una aplicación web se concluye que debe ser el usuario el que especifique mejor los criterios de búsqueda.

5. Bibliografía

- Básica

Libros:

"Profesional C# 2ª Edición". S. Robinson, K. Scott Allen, ... Ed: Wrox.Press, 2002.

"Microsoft Visual C# .NET Aprenda ya" J. Sharp, J. Jagger. Ed: McGrawHill, 2002.

"El lenguaje de programación C#" J.A. González Seco, 2003.

Diferentes artículos de MSDN de distintos autores.

Enlaces C#:

- http://www.programacion.com/tutorial/csharp/
- http://www.csharphelp.com
- http://manowar.lsi.us.es/~csharp/
- http://www.csharpfriends.com/
- http://www.c-sharpcorner.com/
- http://www.dotnet.com

- Complementaria

Enlaces

- http://www.elguille.info/NET/indice.asp
- http://www.programacion.com/
- http://msdn.microsoft.com/
- http://www.microsoft.com/spanish/msdn/botica.asp