Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de software ERP para empresa de montajes

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Alejandro Manuel Albert Casañ

Tutor: Eliseo Jorge Marzal Calatayud

Tutor externo: Elena Saláis López

2020-2021

Resumen

En este proyecto se aborda el desarrollo de un software de tipo ERP para una empresa de montaje de muebles. El resultado será una aplicación de escritorio que gestionará la información relacionada con los servicios que prestan y automatizará el trabajo relacionado con los aspectos operativos o productivos de la empresa. El programa utilizará una base de datos en SQL server y será programado sobre el framework .NET, en concreto en el lenguaje Visual Basic.

Palabras clave: ERP, aplicación, gestión, .NET

Abstract

This project addresses the development of ERP-type software for a furniture assembly company. The result will be a desktop application that will manage the information related to the services they provide and automate the work related to the operational or productive aspects of the company. The program will use a database on SQL server and will be programmed on the .NET framework, specifically in the Visual Basic language.

Keywords: ERP, application, manage, .NET

Tabla de contenidos

Tabla de ilustraciones

1. Introducción

El uso del mueble se remonta al principio de la humanidad. Desde el uso de objetos naturales para depositar objetos o dormir hasta la infinita variedad de los que disponemos hoy en día, ha habido multitud de evoluciones en este arte. En la etapa actual donde la inmediatez prima, muchos de los propietarios de nuevos muebles necesitan que alguien los monte por ellos y fruto de esta necesidad han surgido muchas empresas que se encargan de su montaje.

Desde hace años la digitalización de los procesos de negocio se ha vuelto necesaria para mantenerse competitivo en el mercado y desde la pandemia Covid-19 es algo prácticamente obligatorio para seguir funcionando de una forma segura para todos los empleados.

Para satisfacer estas necesidades en este proyecto se va a relatar el proceso de desarrollo de una solución ERP (Sistema de planificación de recursos empresariales) personalizada para la empresa Montajes García Campos. Esta empresa tenía una solución ERP ya implantada y necesita una nueva que se adapte a los cambios que ha habido. Este proyecto se enmarca en dos convenios de prácticas realizados en la empresa Tiatools SL. La empresa es una consultoría que se encarga de asesorar en materia de tecnologías de la información a empresas o particulares y vender software a la medida del cliente.

a. Motivación

La motivación principal del proyecto es obtener la mayor satisfacción posible del cliente y contribuir a su desarrollo digital y económico.

Una vez las necesidades del cliente hayan sido satisfechas obtendremos el consiguiente rédito económico por la venta de la solución que vamos a desarrollar.

Además, habremos mejorado nuestro uso del lenguaje y las tecnologías que vamos a emplear ya que manejaremos una base de datos SQL Server que se usará en un entorno empresarial real y emplearemos nuestros conocimientos sobre el framework .NET para crear un programa que se pondrá en práctica en una empresa funcional.

b. Objetivos

Una vez presentados los hechos que nos han llevado a este punto vamos a concretar los objetivos que debemos cumplir con el proyecto.

El objetivo principal es desarrollar una solución ERP que implemente el proceso de negocio de una empresa de montajes y que esta cumpla con las expectativas del cliente.

Por otro lado, tenemos una serie de objetivos secundarios:

- Estudiar el tipo de software ERP. Para este proyecto es necesario que conozcamos algo de historia de este tipo de software y su estado actual en el mercado.
- Analizar las necesidades del cliente. Para un resultado satisfactorio para el cliente deberemos conocer a fondo todos los procesos de negocio que vamos a manejar y todos los datos involucrados.

- Diseñar una base de datos que acepte los datos de la antigua. Para que la empresa pueda poner en marcha el programa deberá tener totalmente integrados los datos de los que disponía.
- Integrar la solución con otros programas de los que la empresa disponga.
- Dominar las tecnologías que se van a usar en el proyecto. Es importante mejorar nuestra capacidad de uso tanto del sistema de gestión de bases de datos SQL Server como del framework .NET, en concreto con el lenguaje Visual Basic.
- Involucrar al cliente en el proceso de desarrollo. Durante cada etapa del proyecto se le mostrarán al cliente los avances y se aceptarán sus sugerencias.

c. Estructura de la memoria

La memoria está compuesta de diez capítulos. A partir de la introducción que acabamos de hacer, podríamos agruparlos en tres bloques:

1. Bloque de recopilación de información.

   Este bloque contendría los capítulos 2, 3 y 4 donde recopilaremos toda la información que necesitemos para realizar el proyecto.

   En el capítulo 2, conoceremos la historia de los ERP, veremos su desempeño en el mercado del software actual y presentaremos nuestra solución. Más tarde, en el capítulo 3 veremos los requisitos funcionales y no funcionales que debe cumplir el programa. Por último, en el 4 recogeremos toda la información que el cliente nos pueda proporcionar y nos ayudaremos de la elaboración de diagramas para orientarnos en el diseño de la solución.

2. Bloque de diseño, implementación y pruebas.

   Compuesto de los capítulos 5, 6 y 7 en este bloque traduciremos la información recopilada en el bloque anterior a el software que desarrollaremos. En el capítulo 5 veremos todas las tecnologías que utilizaremos, como se van a compenetrar entre ellas y veremos el diseño tanto de la interfaz gráfica como del modelo de datos. A continuación, en el apartado 6 relataremos el proceso de implementación de la solución y en el 7 comentaremos las pruebas que hayamos hecho del software a la vista del cliente y su impacto en el desarrollo.

3. Bloque final.

   Este bloque incluye los capítulos 8, 9 y 10. En el capítulo 8 valoraremos los resultados obtenidos durante el desarrollo, nuestros trabajos futuros a partir de este proyecto y la relación de este proyecto con la formación recibida en la universidad. En el capítulo 9, encontraremos las referencias a los trabajos o informaciones que hayamos usado para la redacción de la memoria y el desarrollo. Para finalizar, en el último capítulo incluiremos información adicional que pueda resultar útil para el lector.

2. Estado del arte

En este apartado vamos a conocer a fondo qué son los ERP. Para empezar a conocer el tipo de software ERP tenemos que definirlo.

a. ¿Qué es un ERP?

Un ERP (Sistema de gestión de recursos empresariales) es un software que integra los procesos necesarios para el funcionamiento de una empresa como por ejemplo recursos humanos, finanzas, cadena de suministro, compras, ventas o inventarios.

Existen distintos tipos de software ERP:

- ERP en la nube. Software que funciona en una plataforma ubicada en la nube.
- ERP on-premise. Modelo tradicional en el que se tiene control total. El software de ERP se instala en un centro de datos de la empresa.
- ERP híbrido. En este modelo algunas de las aplicaciones y datos estarán en la nube y otros on-premise. También se conoce como ERP de dos niveles.

(SAP, s.f.)

b. Historia

Una vez conocemos una visión general de qué son los ERP vamos a ver su historia:

Origen (Años 40 y 50)

El origen de los ERP se remonta a finales de la segunda guerra mundial. En esa época el ejército de los Estados Unidos empezó a usar programas para gestionar procesos militares complejos. Durante este periodo eran prácticamente inaccesibles para los entornos empresariales porque solo el ejército tenía acceso a computadoras.

Década de los 60

Una vez ya se pudieron emplear en el mundo civil las tecnologías que unas décadas atrás usaban los militares, empezaron a aparecer las primeras computadoras comerciales. Gracias a este avance empezaron a surgir las primeras empresas dedicadas a la venta de software. Entre los primeros tipos de programas que se empezaron a comercializar se encontraban las listas de materiales (BOM) o los ya más sofisticados softwares de gestión y control de inventarios (IMC).

Década de los 70

Los 70 estuvieron caracterizados por la escasez de multitud de materias primas y fruto de esa necesidad se originó un nuevo tipo de programa, los MRP (Planificación de los requerimientos de material). Los MRP añadieron a los programas de planificación la capacidad de prever los momentos en los que iban a ser necesarias ciertos recursos y la cantidad demandada. Durante esta época se fundaron dos de los desarrolladores de ERP que conocemos actualmente, SAP (1972) y Oracle (1977).

Década de los 80

Los programas de planificación que usaban las empresas para evolucionaron para empezar a incluir otros ámbitos además de las materias primas, entre ellas, la introducción de la capacidad de gestión de procesos contables. Gracias a estas mejoras se implantarían los cimientos que darían lugar a los ERP.

Década de los 90

Es la década en la que nace el concepto de ERP de la mano de la consultora Gartner ya que el término MRP se quedaba corto para definir los nuevos programas de planificación empresarial. El nuevo software superaba ampliamente a los antiguos en ámbitos como la fabricación y las finanzas. Un ERP, más que un programa de planificación consistía en una herramienta capaz de respaldar las decisiones de una empresa. Por lo tanto, sus metodologías se abrieron paso entre todo tipo de negocios. Al final de la década, en 1999 nace Salesforce e inicia el modelo SaaS (Software como Servicio).

Siglo XXI

A principios de siglo los ERP ya empezaban a incluir funciones adicionales que realizaban otros programas anteriormente como la gestión de relaciones con clientes (CRM) o gestión de cadena de suministro (SCM). Estas inclusiones y su incremento de popularidad dieron lugar al término ERP extendido e hicieron el mercado de este tipo de software muy atractivo para grandes desarrolladoras como Microsoft que sacó al mercado su solución llamada ERP Dynamics.

A partir de 2005 la tendencia del mercado se decanta hacia soluciones de software en la nube (Cloud). Estas soluciones ofrecen funcionalidades equiparables a las de un ERP local con un coste mucho menor. Uno de los principales actores en este nacimiento es WorkDay, que ofrece soluciones en la nube de finanzas y RRHH. Además, nacen los primeros ERP de código abierto como Odoo (conocido anteriormente como OpenERP y TinyERP)

Con la popularización del software de gestión, el volumen de datos que tenían que manejar las empresas crecía a un ritmo exponencial por lo que cada vez más era necesaria su correcta gestión. Así empieza a surgir el término bussiness intelligence que hace referencia a un tipo de software que integre todas las fuentes de datos que pueda tener una empresa desde sus distintas aplicaciones para obtener información en la que basar sus decisiones. Tanta fue la importancia de este nuevo modelo que durante el 2007 los grandes (SAP, Oracle y IBM) compran empresas que desarrollaban este tipo de software llegando a desembolsar más de 10.500 millones de euros.

Presente

Hoy en día están establecidas las soluciones ERP en modo SaaS (Software como Servicio) se ha consolidado. Una de las ventajas de este tipo de servicio que gusta mucho a las empresas es la movilidad que proporciona ya que puedes acceder a tu puesto de trabajo desde cualquier dispositivo simplemente con una conexión a internet. Por ejemplo, SAP cuenta con más de 200 Millones de usuarios en cloud. Se ha vuelto imprescindible tener un mínimo de software de gestión implantado en la empresa para ser competitivo en el mercado. Aunque pueda parecer que ahora gestionar una empresa se vuelve muy caro por todo lo que puede necesitar, tenemos a nuestra disposición soluciones ERP de código abierto que democratizan el acceso a

herramientas de gestión como Adempiere, Apache OFBiz, Dolibarr, ERPNext, Metasfresh, Odoo, Openbravo o WebERP.

Futuro

Hacer predicciones del futuro es algo complicado, pero aún quedan muchas tecnologías que integrar en el software de tipo ERP como la inteligencia artificial o blockchain.

La inteligencia artificial probablemente será el tema de moda puesto que sigue siendo muy importante qué hacemos con los datos que tenemos y porque también se siguen manteniendo tareas repetitivas en los procesos de negocio. Desarrollar modelos predictivos que aporten a las empresas predicciones sobre las que basarse o automatizar muchos de esos procesos que siguen necesitando intervención humana son algunas de las muchas aplicaciones que podría tener la inteligencia artificial en los ERP.

Por último, dada la popularidad de blockchain; esta tecnología probablemente se haga un hueco en este tipo de software. La idea de incluir criptomonedas en las formas de pago o de que los módulos de compra y venta operen sobre cadenas de bloques son ideas en las que se trabaja para hacerlas realidad.

(ESAA, 2020)

(SoftwarePara, s.f.)

c. Solución propuesta

Basándonos en la información recopilada durante la investigación acerca de los ERP y considerando la información preliminar que tenemos, nuestro sistema consistirá en un ERP de on-premise. Diseñaremos una solución ERP cuyas bases de datos estarán almacenadas en el servidor de la empresa de forma centralizada y cada empleado accederá a ellas desde el programa que se desarrollará para satisfacer los casos de uso que necesiten. La empresa tendrá un control total de su información.

3. Especificación de requisitos

Una vez introducido el tema y analizado el contexto en el que trabajamos vamos a concretar los requisitos que deberá cumplir nuestra solución ERP. Con estos requisitos podremos obtener una primera idea de cómo será el diseño de la aplicación.

Dividiremos estos requisitos en funcionales y no funcionales. Los requisitos funcionales nos dirán que tiene que hacer el programa mientras que los no funcionales nos dirán como tiene que hacerlo.

a. Requisitos funcionales

Los requisitos funcionales son los siguientes:

- El sistema debe permitir gestionar la información de todas las entidades necesarias en el proceso de negocio (Las veremos en el punto 4).
- El sistema debe tener una autenticación de usuario y contraseña. Cada usuario tendrá una serie de permisos.
- El sistema debe implementar de la gestión de pedidos.
- El sistema debe implementar de la facturación de ventas.
- El sistema debe implementar de la gestión de recibos.
- El sistema debe integrar los listados de los que dispone la empresa.
- El sistema debe integrar con la aplicación de contabilidad de la empresa, es decir, debe permitir contabilizar las facturas que contenga el programa.

b. Requisitos no funcionales

Los requisitos no funcionales son los siguientes:

- No se utilizarán claves ajenas en el diseño de la base de datos. Las relaciones en modelo se implementarán por programa.
- Separar la base de datos del programa con la base de datos de los usuarios, es decir crear dos bases de datos distintas.
- Las bases de datos estarán alojadas en el servidor de la empresa y el programa se deberá conectar a él para insertar, modificar y obtener datos.
- La aplicación deberás ser desarrollada sobre el framework .NET Framework 4.5.
- No se utilizará ningún ORM (Asignación objeto-relacional) para el acceso a datos.
- Deberemos usar la arquitectura ADO.NET para el acceso a los datos.

4. Análisis

En este punto vamos a ver toda la información recopilada de la empresa de montajes a lo largo de la primera sesión de análisis y con ella vamos a diseñar el modelo de clases que utilizará el programa y los casos de uso que tendremos que implementar.

Para empezar, la empresa trabaja con cuatro agentes cuya información debe poder ser gestionada por sus empleados:

- Montadores – Trabajadores que montan los muebles en la casa del cliente.
- Vendedores – Contratan el servicio de la empresa. El vendedor es el que ofrece el producto al cliente y contrata a la empresa de montajes para montarlo.
- Fabricantes – Fabrican los productos ofrecidos por el vendedor.
- Proveedores – Empresas a las que nuestro cliente compra bienes o servicios que necesita para realizar sus labores.

Los tres primeros agentes tendrán un papel muy importante en el proceso de negocio pues están involucrados siempre. En cambio, los proveedores solo son almacenados en el sistema para consultar la información en el momento que se use el programa de contabilidad para introducir las facturas de compra. A estos agentes se les llama terceros.

La información de los terceros es la siguiente: razón social, nombre, provincia (no el nombre si no su código), localidad (depende de la provincia), código postal, direcciones, teléfonos, email, fax, web, persona de contacto y CIF. Además, los terceros tienen una cuenta contable en la base de datos de contabilidad, una forma de pago y un tipo de IVA.

Una forma de pago es la forma en la que la empresa de montajes recibirá el pago. En una forma de pago tendremos en cuenta los recibos que se generarán, los días a partir de los que vencerá el primer recibo, la cadencia (días que tardarán en vencer los recibos sucesivos), tipo de recibo que se generará (contado, reposición…), la situación en la que se genera el recibo. Además, una forma de pago tiene una descripción y su correspondiente abreviatura.

Un tipo de IVA nos dice el porcentaje que se pagará por este impuesto en cada venta. Cada tipo de IVA se enmarca en un ejercicio (tiene una fecha de inicio y otra de fin) y contiene el porcentaje que se paga, una descripción, la cuenta donde se contabiliza el IVA, el porcentaje de recargo y la cuenta donde se contabiliza el recargo. Las cuentas contables del IVA empiezan por 47.

Un tercero, además de la persona de contacto, puede tener otros contactos dentro de la organización y que se asocian a él. Un contacto tiene su nombre, su cargo, el departamento al que pertenece, sus teléfonos y sus correos.

Por otro lado, cada tercero puede tener distintos centros con sus respectivos departamentos. Por ejemplo, una marca de tiendas de muebles puede tener un centro en Valencia y otro en Alicante y dentro de ellos distintos departamentos como por ejemplo uno muebles de baño u otro de muebles de terraza. Un centro tiene entre sus datos una descripción, provincia, localidad, código postal, direcciones, teléfonos, un nombre de contacto y un email. Un departamento tiene una descripción, nombre de la persona de contacto, el teléfono y el email.

Por otra parte, un tercero puede tener una o varias cuentas bancarias asociadas por cada banco. Una cuenta tiene el banco al que pertenece, su número de cuenta (iban, banco, oficina, dígitos de control, número de cuenta), el código Swift, si es su predeterminada, el mandato, si está firmado y la fecha de su firma.

Por último, los servicios que la empresa de montajes da a los vendedores se llaman artículos. Cada vendedor puede tener más de un artículo asociado. Un artículo tiene un código que empieza por el identificador del vendedor y acaba con el número del artículo, una descripción, el porcentaje que se cobra de la venta, el importe, el porcentaje del vendedor, el importe del vendedor, el porcentaje que se le paga al montador y el importe que recibe. Los importes y porcentajes son orientativos pues después a la hora de realizar el pedido se introducen los que en ese momento estime el empleado. Los artículos tienen dos usos pues en unos casos puede solo utilizarse para indicar una mercancía y otras para saber la distribución del precio del servicio, como veremos más adelante en las líneas de los pedidos.

Cuando un vendedor contrata la empresa de montajes el empleado da de alta un pedido. Un pedido es una orden de trabajo que consistirá en el proceso de recepción y almacenaje de la mercancía y en el posterior montaje en la ubicación del cliente. Los pedidos se identifican por su año de realización y su código. Además, contienen la siguiente información: el vendedor con su centro y su departamento, una referencia de pedido (referencia por la que el vendedor identifica al cliente), una referencia de vendedor(referencia por la que el vendedor identifica al pedido), el nombre del cliente, sus direcciones, provincia, localidad, código postal, los teléfonos del cliente, el tipo de facturación del pedido (si se factura al vendedor, al fabricante o a los dos), a quién se le facturará, la fecha de creación, la fecha de previsión de recepción de la mercancía, la fecha de recepción final, la fecha de albaranado, la fecha de previsión de montaje, la fecha de montaje final, el albarán de la tienda, los bultos, el tiempo aproximado de montaje, si requiere coordinación, el coordinador, una nota, el montador que realizará el montaje, el año y las referencias a las facturas del montador y de la tienda o fabricante(más tarde veremos cuáles), si hay que devolverlo y la fecha de su devolución, si hay que facturar el almacenaje y la fecha de hasta cuando, si tiene alguna incidencia, si proviene de algún pedido anterior (veremos esto más claro cuando lleguemos a la parte de incidencias) y un historial. Además, un pedido tiene el estado de la comunicación con el cliente (veremos este proceso más adelante).

La empresa opera con dos tipos de facturación:

- FPROV – Facturación al proveedor. Se albarana y se cobra todo al fabricante.
- FVEND- Facturación del vendedor. Se albarana y se cobra todo al vendedor.
- FPV- Facturación al proveedor y al vendedor. Se cobra a los dos con los porcentajes acordados. Este tipo de facturación es poco común, pero debe mantenerse porque algunas cadenas de tiendas funcionan de esta forma.

Los pedidos están compuestos por una serie de líneas que les van asociadas. Una línea tiene su tipo, su proveedor (que puede ser el vendedor o el fabricante según el tipo de facturación), su fabricante, su departamento, su cargo, un artículo (del vendedor) y la descripción.

Las líneas pueden ser de dos tipos:

- Tipo 1. Normalmente una por cada mercancía que haya que recibir, aunque también puede tener solo valor informativo como para indicar que el pedido proviene de una incidencia. Una línea de este tipo tiene además información sobre si se ha recibido (puede no tener que recibirse) como la fecha de entrada, los bultos y su tiempo aproximado de montaje; su albarán individual (algunas veces vienen de distintos fabricantes),
- Tipo 4. Las líneas tipo cuatro son las que llevan la información del precio del servicio. Como norma general, solo hay una en la que el empleado selecciona el artículo y ajusta los importes. Cada línea de este tipo tiene una cantidad, un importe, el porcentaje que se factura, el importe que se factura, el porcentaje del vendedor y su importe (solo si se le facturase el pedido al fabricante y al vendedor simultáneamente), el porcentaje que cobra el montador y el importe correspondiente.

Además, los pedidos tienen un registro de las llamadas en las que se ven involucrados. Una llamada tiene como información la fecha en la que se realiza, el usuario que la hace, el teléfono al que llama, el código destinatario (0 si es al cliente y el código del tercero si es a al vendedor o al fabricante) y una nota. Las llamadas van asignadas al pedido que se va a montar.

Por último, un pedido puede tener asociadas una serie de incidencias que puedan ir ocurriendo durante el transcurso del proceso de negocio. Una incidencia tiene asociada el fabricante responsable, un texto donde se explica con detalle lo que ha podido pasar, un estado (resuelta o pendiente) y el pedido que la resolverá.

El empleado cuando da de alta un pedido introduce toda la información y sus líneas. Una vez es grabado el pedido se encuentra en la situación pendiente y puede comenzar a pasar por las etapas del proceso de negocio. El pedido en este estado se encuentra a la espera de recibir todas las líneas.

El empleado va notificando la recepción de las líneas conforme van llegando al almacén. Cuando notifica una recepción introduce el almacén donde se guarda, la ubicación, los bultos que finalmente se reciben y el tiempo aproximado de montaje en minutos (al final el pedido tendrá unos bultos y tiempo aproximado igual a la suma de los de todas sus líneas de tipo 1). Una vez se recibe alguna de ellas la situación del pedido pasa a ser parcial. Cada vez que una línea es recibida se crea un registro de ubicación con la información que se ha introducido. Si todas han sido recibidas el pedido pasa a ser recibido.

Cuando el pedido ha sido recibido el empleado asigna el albarán que le llega de la tienda y el pedido pasa a la situación de albaranado por la tienda. Un albarán es un documento que certifica que un servicio se ha prestado o que un producto se ha entregado. En este estado el empleado de la empresa de montajes comienza la comunicación con el cliente para concertar la fecha del montaje (día y horario). Este contacto se realiza por vía telefónica y se lleva un registro de cada llamada que se hace. Aquí entra en juego lo que veíamos antes pues si el empleado tiene éxito en la comunicación deberá actualizar la información del pedido para indicar que el cliente ha sido contactado, la fecha de montaje prevista y el horario en el que se realizará. Por otra parte, Si el cliente no contesta se dejará constancia de la llamada.

Una vez hemos contactado con el cliente el pedido pasa a estar en el estado contactado y falta asignarle un montador que esté disponible ese día. Cuando el empleado decide el montador al que designarle el encargo se lo asigna y entonces el pedido pasa a estar en ruta.

El día que el montador va a casa del cliente se imprime un listado llamado albarán de entrega de mercancía en el que queda constancia de que se le entrega la mercancía. Además, se le entrega al montador otro listado llamado hoja de ruta en el que aparecen todos los montajes que tiene que realizar en el día.

Durante todo este proceso puede haber incidencias como por ejemplo que la mercancía llegue en mal estado o que cuando se llegue a la casa del cliente las medidas no coincidan. Cuando esto sucede el empleado añade a la información de la incidencia en el pedido e imprime o guarda un listado de incidencia con la información que acaba de introducir. Una vez está registrada la incidencia el trabajador envía un correo a quien corresponda para resolver la incidencia. En este correo pueden adjuntarse fotografías que se guardan junto al listado en un directorio compartido en la red alojado en el servidor. Por programa el empleado genera un subdirectorio cuyo nombre viene dado por la referencia del pedido y como de la misma referencia puede haber varias incidencias se crea dentro otro con el código interno del pedido en el que ha ocurrido en el que estarán las fotos y el listado (si se considera oportuno). Cuando hay un acuerdo entre la tienda o el fabricante con la empresa de montajes el empleado valora si necesita generar otro pedido que la resuelva. Si opta por generar un pedido que la resuelva se copiará el original con un nuevo código y referenciando el pedido del que proviene. Este pedido contendrá una línea de tipo 1 informativa de que se trata de una incidencia y puede incluirse alguna línea de tipo 4 para cargar algún importe. Este pedido vuelve a empezar el proceso de los pedidos y el original se da como montado.

Por último, cuando vuelve el montador se da el pedido como montado y se generan los albaranes del servicio correspondientes al tipo de facturación.

Un albarán se identifica por su año, su serie y su número. Además, tiene la siguiente información: la fecha en la que se realiza, la descripción, el tercero involucrado con su centro y departamento (si es vendedor), el importe bruto, el porcentaje de descuento, el importe base (resta entre el importe bruto y el descuento), el tipo de IVA que se aplica, el porcentaje de IVA que se paga, el importe correspondiente al IVA y el importe neto que es la suma del importe base y el del IVA. Por último, un albarán puede estar pendiente de facturar o facturado.

Los albaranes se descomponen en líneas y cada línea tiene el artículo correspondiente al servicio, la descripción, la cantidad, el importe total, el porcentaje que se le cobra al tercero, el importe que se le cobra y la referencia al pedido del que proviene (año y código).

Cuando se genera un albarán al haber realizado el montaje primero se rellena la información de la cabecera (excepto los importes) y una línea de albarán por cada línea de tipo cuatro del pedido. Con todas las líneas generadas se suman los importes y se calculan los de la cabecera del albarán.

Una vez el albarán está registrado el empleado puede modificarlo o borrarlo, tanto sus líneas como su cabecera. Además, también puede crear albaranes nuevos si lo desea.

Las series de los albaranes y de posteriormente las facturas siguen esta numeración:

- 1 – Vendedores.
- 6 – Alquiler de furgonetas.
- 8 – Montadores.
- 10 – Fabricantes.

Dependiendo del tercero se querrá generar facturas mensuales, por pedido o incluso anuales. Los albaranes propios cumplen el propósito de certificar que un servicio se ha prestado y su correspondiente precio y se encuentran a la espera de ser facturados. Una vez el empleado considere que se debe realizar una factura la generará a partir de los albaranes pendientes que considere. Como los albaranes y los pedidos una factura tiene una cabecera y sus líneas.

Una factura se identifica por su año, serie y número de factura y contiene la siguiente información: tercero al que se le factura, la fecha de factura, tipo de IVA asociado, importe bruto, porcentaje descuento, importe de descuento, importe base, importe de IVA, importe de recargo y el importe neto. Además, según si una factura puede seguir añadiendo albaranes o no puede estar abierta o cerrada respectivamente. Las líneas de una factura disponen de los siguientes campos: artículo correspondiente al servicio, descripción del servicio, cantidad, importe total, porcentaje que se factura y el importe que se factura. Las líneas provienen de un albarán y pueden corresponderse con un pedido. Cuando las facturas se generan, cada línea de la nueva factura se corresponde con una de cada albarán seleccionado y al final se rellena la cabecera con la información del tercero y la suma de los importes de cada una de ellas. Solo se puede borrar la última factura ya que las auditorías de hacienda demandan que no haya huecos entre los números de las facturas para evitar fraudes y no se podrán borrar facturas contabilizadas. Un empleado puede contabilizar y des contabilizar una factura, es decir, generar un asiento contable o borrarlo (si es el último). Los asientos contables son los conjuntos de apuntes que se hacen en un libro de contabilidad o en este caso los que se hacen en la base de datos de contabilidad.

En el momento en el que una factura se cierra se generan los recibos según la forma de pago que el tercero tenga asignada. Los recibos sirven para llevar un registro de si se han realizado o no los pagos. La información que tiene un recibo es la factura de la que proviene, la fecha de la factura, la fecha de vencimiento del recibo, a quien va facturado, el importe del pago, la cuenta de la que proviene el pago y la cuenta donde se ingresa. Un empleado puede cambiar la situación de los recibos conforme desee. Para resumir el funcionamiento normal del proceso de negocio al completo desde que se da de alta un pedido hasta que se contabiliza vamos a ilustrarlo con un diagrama de flujo

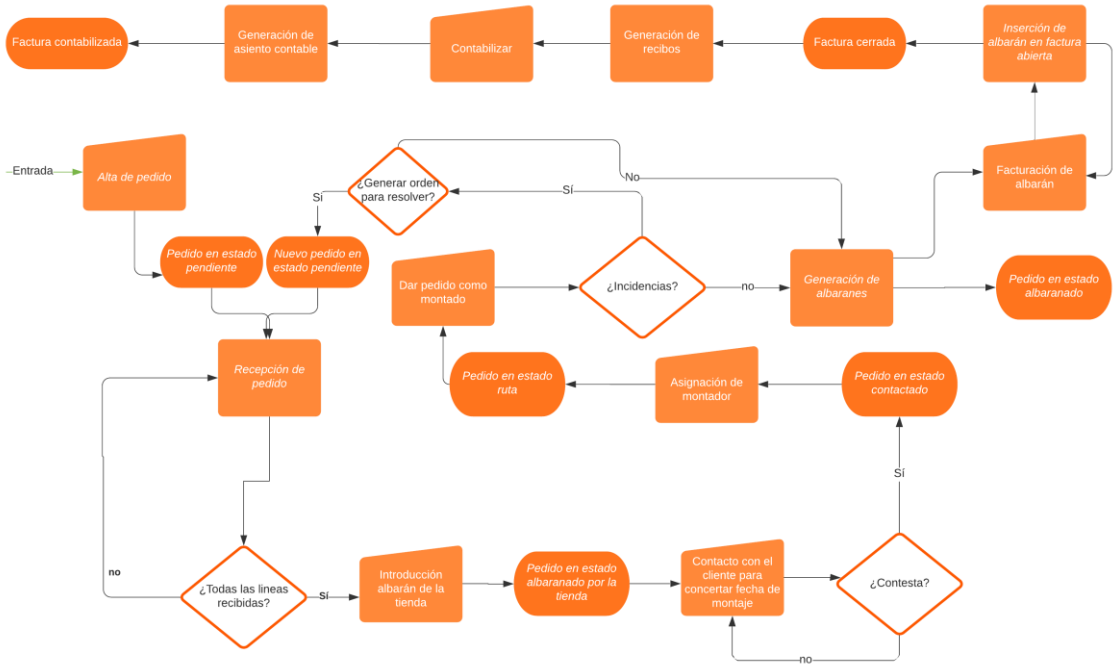

Ilustración 1- diagrama de flujo del proceso de negocio.

Con la información recopilada podemos pasar a realizar los diagramas. Vamos a realizar un diagrama de casos de uso y uno de clases. Un diagrama de casos de uso representa los procesos empresariales que tenemos que implementar y un diagrama de clases la estructura que vamos a seguir para implementarlos.

Un caso de uso es una representación de una actividad realizada por los actores que intervienen en los procesos de un programa y una clase es la abstracción de una idea de la realidad como por ejemplo un pedido de un cliente.

a. Diagrama de casos de uso

Los casos de uso que veremos en nuestro programa son los siguientes:

- Iniciar sesión: un usuario no autenticado debe ser capaz de iniciar sesión con sus credenciales.
- Buscar terceros: el usuario debe ser capaz de buscar la información de un tercero
- Crear tercero: el usuario debe ser capaz de crear un tercero.
- Editar tercero: el usuario debe ser capaz de editar la información de un tercero.
- Ver pedidos: el usuario debe ser capaz de ver todos los pedidos.
- Crear pedido: el usuario debe ser capaz de crear un pedido.
- Editar pedido: el usuario debe ser capaz de editar la información de un pedido.
- Borrar pedido: el usuario debe ser capaz de borrar un pedido.
- Recibir líneas: el usuario debe ser capaz de dar líneas de pedido como recibidas.
- Recibir pedido: si todas las líneas están recibidas el pedido deberá poder ser recibido
- Ver listados: el usuario debe ser capaz de ver los listados de la empresa.
- Imprimir listados: el usuario debe ser capaz de imprimir los listados de la empresa.
- Crear incidencia: el usuario debe ser capaz de crear una incidencia de un pedido.

- Ver incidencias: el usuario debe ser capaz de ver todas las incidencias en los pedidos.
- Resolver incidencia: el usuario debe ser capaz de dar como resuelta una incidencia.
- Introducir albarán de la tienda: el usuario debe ser capaz de introducir el albarán de la tienda en el pedido para cambiar su estado a ALBARANADO_TIENDA.
- Concertar fecha de montaje: el usuario debe ser capaz de concertar una fecha de montaje con el cliente para cambiar el pedido al estado CONTACTADO.
- Asignar montador: el usuario debe ser capaz de asignar un montador al pedido para ponerlo en situación de RUTA.
- Realizar montaje: el usuario debe ser capaz de notificar el montaje de un pedido por parte del montador.
- Ver albaranes: el usuario debe ser capaz de ver todos los albaranes en el sistema.
- Crear albaranes: el usuario debe ser capaz de crear un albarán
- Editar albarán: el usuario debe ser capaz de editar la información de un albarán.
- Borrar albarán: el usuario debe ser capaz de borrar un albarán
- Facturar albarán: el usuario debe ser capaz de introducir el albarán en una factura.
- Ver facturas: el usuario debe ser capaz de ver todas las facturas.
- Editar factura: el usuario debe ser capaz de editar la información de las facturas.
- Borrar factura: el usuario debe ser capaz de borrar una factura si es la última creada.
- Contabilizar factura: el usuario debe ser capaz de generar el asiento y apuntes contables a partir de una factura.
- Descontabilizar factura: el usuario debe ser capaz de borrar el asiento y apuntes contables de la última factura.
- Ver recibos: el usuario debe ser capaz de ver todos los recibos generados.
- Cambiar situación: el usuario debe ser capaz de cambiar la situación de un recibo.

Aunque no dependen del programa deberemos tener en cuenta que para crear pedidos o terceros necesitaremos que ellos nos proporcionen su información o la del cliente al que se le dará el servicio.

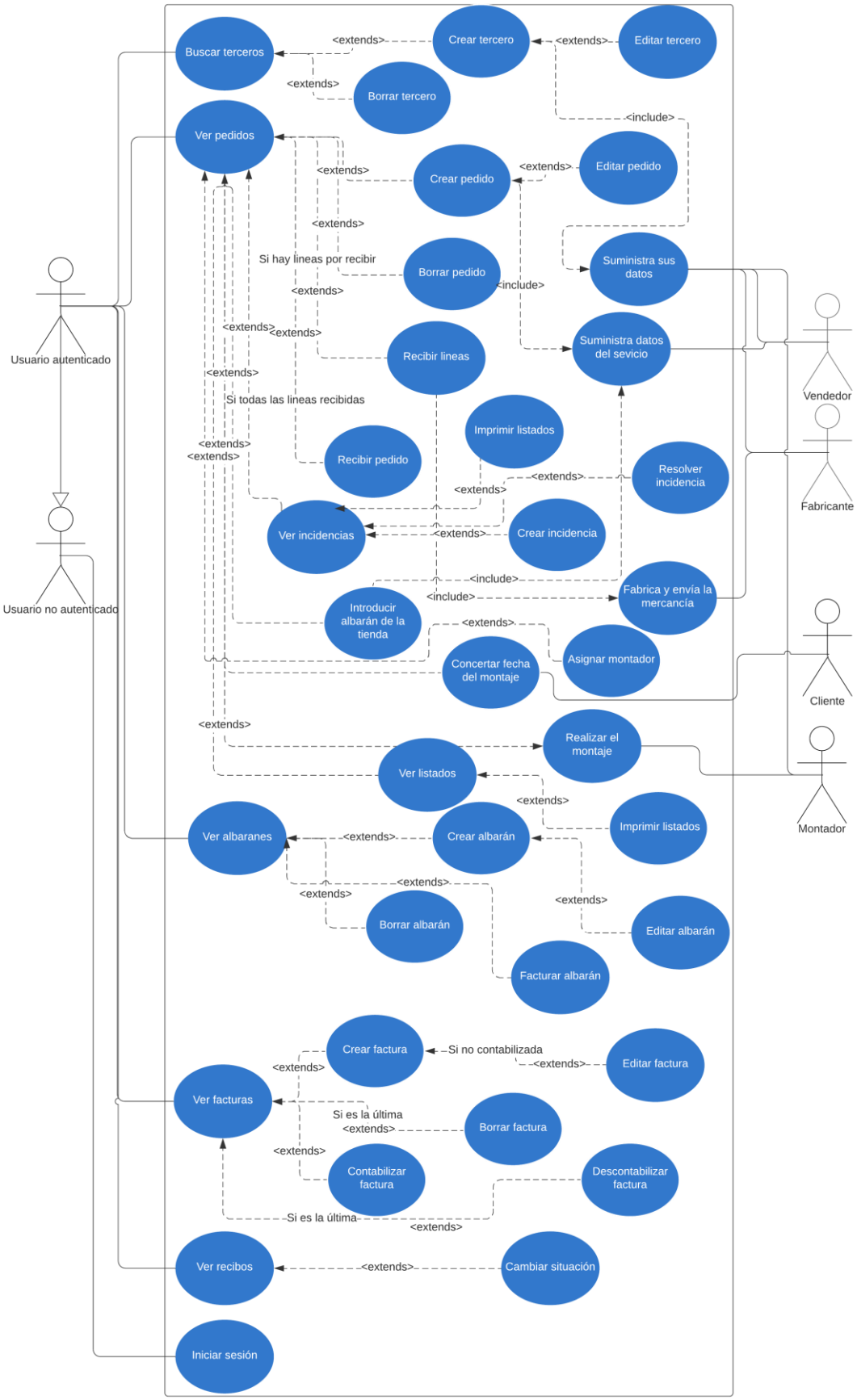

Ilustración 2 - Diagrama de casos de uso.

b. Diagrama de clases

El resultado después de identificar todas las clases involucradas en el funcionamiento que deberá tener el programa es el siguiente:

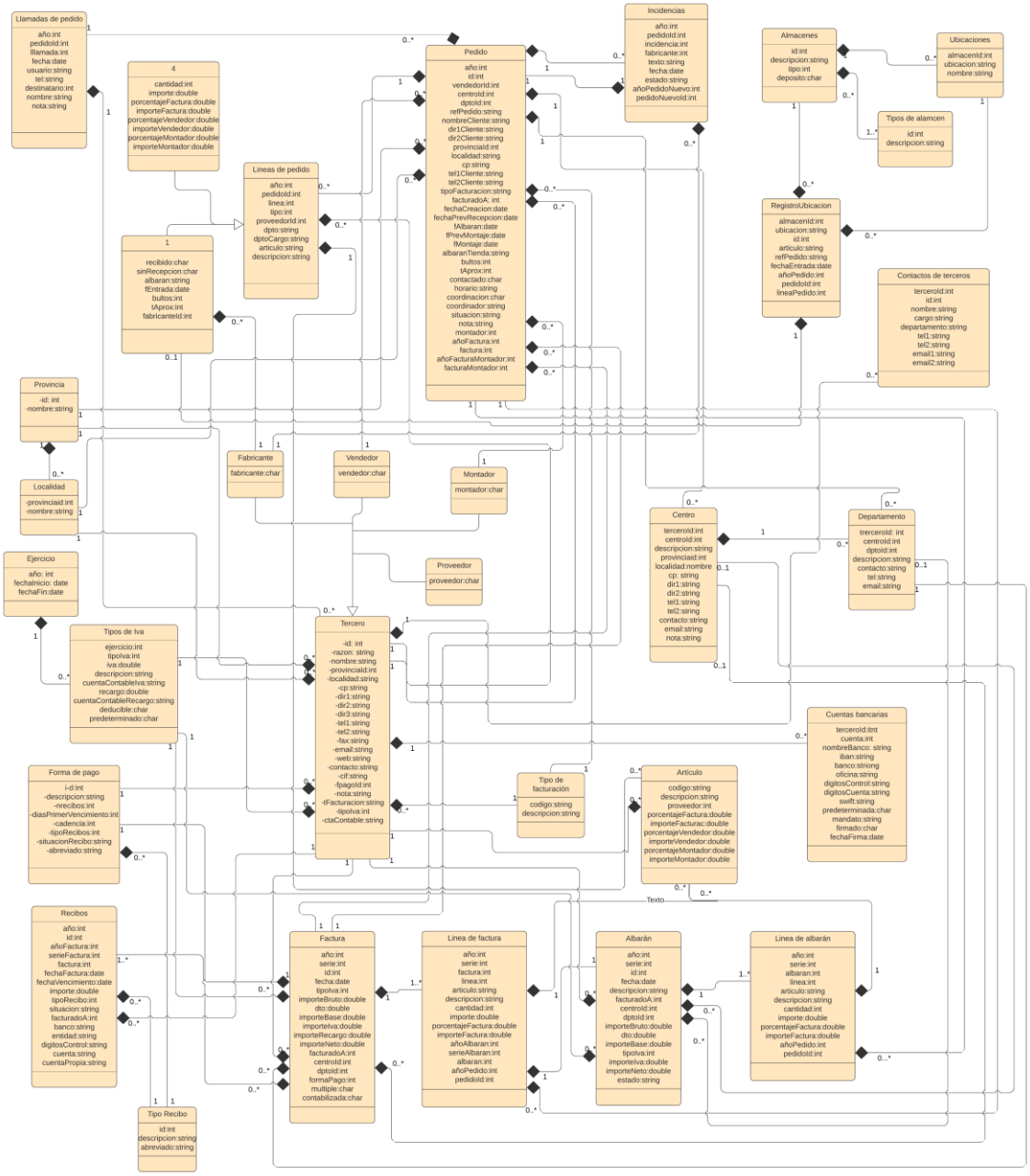

Ilustración 3 - Diagrama de clases.

5. Diseño

En este punto vamos a explicar el diseño de la solución que vamos a desarrollar según la información que hemos recogido en el análisis. Primero, veremos la arquitectura en la que se basará nuestra aplicación.

a. Arquitectura de la aplicación

El sistema que se va a desarrollar consta de una aplicación de escritorio desarrollada sobre el framework .NET (en concreto la versión 4.5). En .NET se puede desarrollar con los lenguajes de programación C# y Visual Basic que será el que vamos a utilizar. La interfaz gráfica será diseñada con el framework de interfaces de usuario Windows Forms. La aplicación se conectará a conectar a tres bases de datos SQL Server 2014 distintas (usuarios, contabilidad, programa) para realizar operaciones CRUD.

La arquitectura que mejor se adapta a las especificaciones y la que se va a seguir el proyecto se la conoce por las siglas MVC (Modelo-vista-controlador) ya que es la ideal para separar los datos de la interfaz gráfica. A continuación, vamos a analizar esta arquitectura en detalle

MVC es un patrón en el diseño de software utilizado para el diseño de interfaces gráficas que separa la lógica de negocio de su visualización. Este tipo de arquitectura nos permite dividir el trabajo de una forma muy eficiente. El sistema se divide en tres partes:

- Modelo: Define qué datos debe tener la aplicación. Asimismo, encontraremos la lógica de negocio, es decir, las reglas que deben seguir los datos para que el sistema tenga un estado coherente.
- Vista: Se encarga del diseño de la interfaz gráfica y de la representación del estado del modelo en un momento concreto. Las vistas deben tener elementos que permitan la interacción del usuario como campos de texto o botones para así dar órdenes al sistema.
- Controlador: se encarga de comunicar los modelos con las vistas. El controlador contiene la lógica que permite actualizar el modelo o la vista según las órdenes del usuario.

(Contributors, 2021)

(Aguilar, 2019)

Las vistas serán los distintos formularios que tendremos que ir diseñando a lo largo de la fase de implementación. Cada formulario Windows Forms tiene una serie de controles asociados donde se manejan los distintos eventos que puedan suceder y estos serán los que desempeñarán la función del controlador de esa vista. Por último, el modelo estará representado por las funciones que utilizaremos para obtener, insertar o eliminar los registros de las bases de datos conforme vayamos necesitándolo.

b. Tecnologías y herramientas

.NET Framework 4.5

Como hemos visto antes trabajaremos sobre el framework .NET 4.5 con el lenguaje Visual Basic. Este entorno proporciona los servicios necesarios para la ejecución de aplicaciones en Windows y una biblioteca de código probado y reutilizable a disposición de los desarrolladores. Sobre ella se ejecutará nuestro programa.

Windows Forms

Plataforma dedicada al diseño y desarrollo de interfaces gráficas para aplicaciones Windows. La utilizaremos para diseñar los formularios que actuarán como vistas en el sistema y para colocarles los controles que manejarán los eventos que vayan sucediendo en ellos como clics o pulsaciones de teclas.

Syncfusion

Biblioteca de componentes de interfaces gráficas que podemos añadir a nuestros formularios de Windows Forms. En este proyecto la usaremos para introducir el componente SfDataGrid en nuestros formularios cuando necesitemos visualizar una gran cantidad de registros pues ofrece un filtrado muy intuitivo.

ADO.NET

Conjunto de clases incluido en el framework .NET que permite el acceso a bases de datos y a servicios. Utilizaremos estas clases para acceder a las bases de datos SQL Server y así crear, modificar o eliminar registros desde la capa de negocio con las órdenes recibidas de las vistas.

Crystal Reports

Aplicación que permite el diseño y la generación de informes desde una fuente de datos. Esta aplicación nos permitirá incorporar los listados que el cliente pueda necesitar a nuestro programa. Desde su diseñador actualizaremos los listados para que se adecuen al nuevo modelo de datos.

SQL Server 2014

Sistema de gestión de bases de datos (SGBD) desarrollado por Microsoft que permite el almacenamiento y manipulación de datos. Los datos de nuestro programa se almacenarán en las bases de datos de una instancia SQL Server.

ODBC

Es un controlador que permite a las aplicaciones acceder a los datos de un sistema de gestión de bases de datos utilizando el lenguaje SQL.

Visual Studio 2013 Ultimate

Entorno de desarrollo integrado (IDE) que permite la creación de aplicaciones en el entorno .NET. Utilizaremos este entorno para el desarrollo de nuestro programa.

SQL Management Studio

Este programa nos ofrece la posibilidad de gestionar nuestra instancia SQL Server a nuestro gusto. Utilizaremos este software para administrar las bases de datos que intervienen en nuestro sistema.

Librería de componentes de Tiatools SL

Es una biblioteca propia de la empresa Tiatools SL que contiene componentes de interfaces gráficas que utilizaremos en el diseño de los formularios.

c. Modelo de datos

Un modelo de datos es la estructura que organiza la información almacenada en la base de datos. En las bases de datos que utilizará el programa se usa un modelo de datos relacional.

El modelo de bases de datos relacional se caracteriza por el uso de tablas para representar las relaciones. Una tabla es un conjunto de filas, cada fila es un registro de una entidad y está compuesta por uno o más campos que representan los atributos de esa entidad. Cada campo suele tener un valor del tipo determinado.

Nuestro programa hereda tres bases de datos que siguen el mismo patrón:

- Contabilidad. Accederemos para crear los registros correspondientes a los asientos contables y sus apuntes y para consultar las cuentas contables que pueden tener los terceros y los tipos de IVA.
- Usuarios. Solo accederemos para consultar las credenciales de los usuarios del programa.
- Programa antiguo. En esta base de datos están almacenados los datos de todas las entidades que han intervenido hasta ahora en el funcionamiento de la empresa. Nosotros tendremos que importar sus datos para que el cliente pueda seguir haciendo un funcionamiento normal.

Las tablas de estas bases de datos están diseñadas sin claves ajenas ni otro tipo de restricciones más allá de claves primarias y por consecuencia son susceptibles de ser bases de datos sucias. Cuando una base de datos es sucia significa que sus datos han perdido coherencia. Puesto que nuestro programa tiene que usar los mismos datos para cumplir los requisitos del cliente diseñaremos nuestra base de datos sin esta herramienta y dejaremos a la capa de negocio el mantener la máxima consistencia posible.

Durante el análisis hemos identificado todas las posibles entidades que va a utilizar el programa en la confección del diagrama de clases y en el apartado a del anexo se podrá encontrar el diseño de las tablas que utilizará el programa en SQL Server 2014.

d. Diagrama de componentes

En este punto vamos a identificar los distintos componentes de los que se compone el sistema en el que se enmarca nuestro programa y ver como se relacionan entre ellos mediante un diagrama de componentes que nos permita ver cómo funciona el sistema de una forma general.

Para empezar, en una aplicación Windows Forms intervienen principalmente los formularios. Un formulario es una representación de cada ventana que aparece en la aplicación y con la que el usuario interactúa. Necesitaremos distintos formularios para cumplir todos los casos de uso que requiere la aplicación, así que, estos serán el primer componente que podemos identificar.

Cada formulario tiene un control que permite ejecutar código cuando determinados eventos ocurren en el mismo. Por ejemplo, si tenemos un formulario con un campo de texto y un botón que hace que se muestre por pantalla un mensaje con el contenido de ese texto necesitaremos que haya un control que detecte el evento clic del botón y que cuando lo detecte abra ese cuadro

de diálogo. Los formularios contienen componentes de interfaz gráfica de la librería Syncfusion y de la librería de la empresa en la que se ha desarrollado el programa, Tiatools SL.

Si el controlador requiere comunicarse con las bases de datos necesitará una capa de acceso a datos que le proporcione una interfaz para obtener los datasets con los datos y para enviarlos de vuelta si se requiere hacer modificaciones en los registros de una tabla esa. Esta capa permitirá que el controlador pueda consultar registros para representarlos en el formulario, modificar esos registros o borrarlos si el usuario lo considera oportuno. Además, los controles si precisan generar listados usarán Crystal Reports para este fin.

Para que la capa de acceso a datos pueda hacer su función necesitará de la librería ADO.NET. La librería permite obtener datasets de nuestra base de datos y nos permite persistir los cambios que hagamos adaptando los cambios hechos en esos datasets a consultas SQL. El driver ODBC nos permitirá hacer esa conexión a las bases de datos que hemos visto anteriormente.

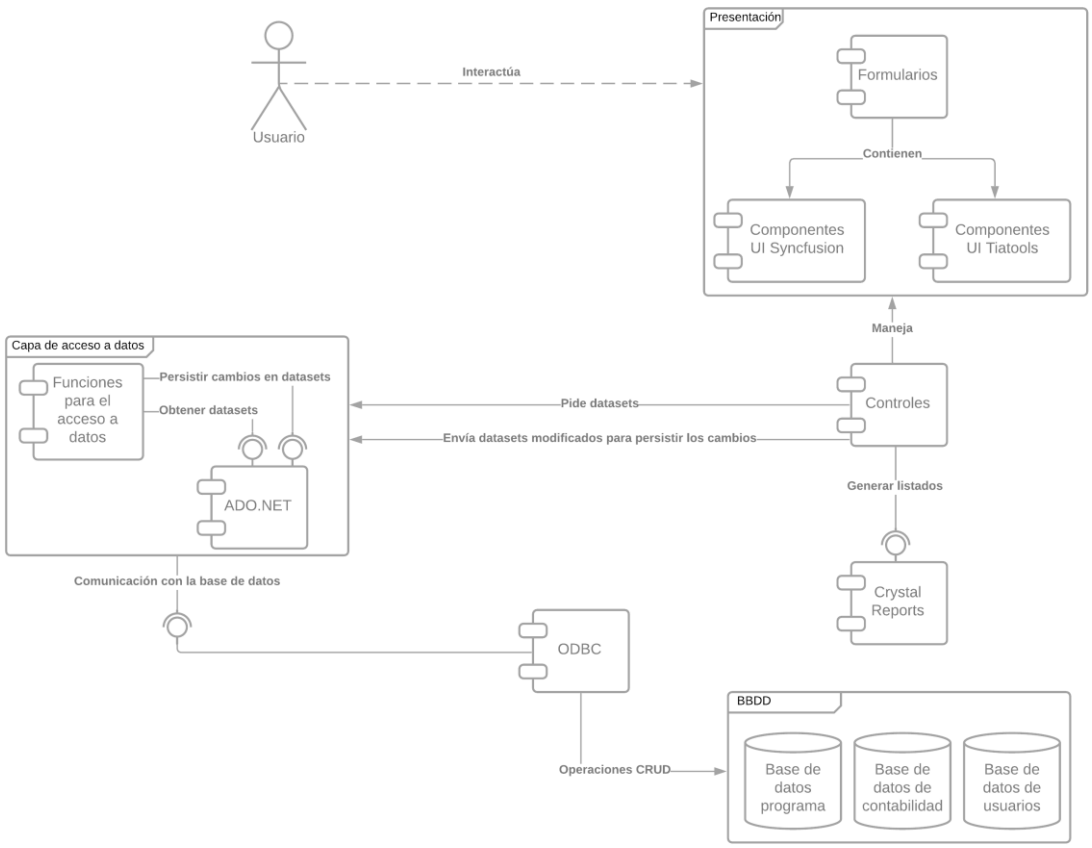

Ilustración 4 - Diagrama de componentes.

e. Diseño de la interfaz gráfica

Introducción

En este punto vamos a detallar el proceso de diseño de la interfaz gráfica con el diseñador de formularios de Visual Studio para Windows Forms.

Para empezar el diseño de la interfaz gráfica tenemos que decidir cómo va a ser la navegación de nuestro programa. Nuestro programa constará de un formulario contenedor que irá alojando el resto, es decir, un formulario MDI. Un formulario MDI permite trabajar con distintos formularios al mismo tiempo y actúa como padre de estos.

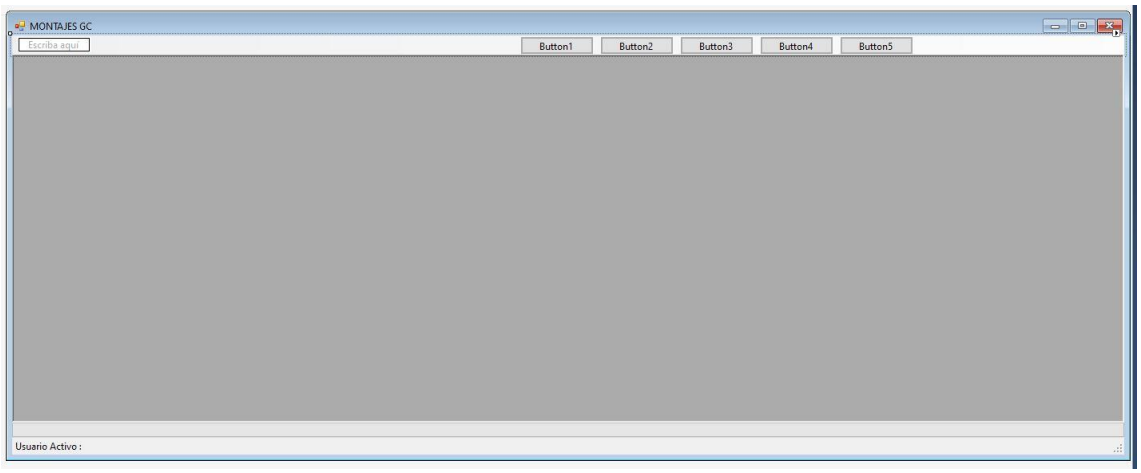

Ilustración 5 - Diseño MDI.

En la ilustración 5 podemos ver un cuadro gris que es donde se irán situando los formularios que el usuario vaya abriendo. En la parte de arriba hay dos secciones, en la de la parte de la izquierda se situará el menú desplegable con las distintas partes de la aplicación y en la del centro derecha se han incluido unos botones de acceso directo a las partes de la aplicación que el usuario use más a menudo. En la parte de abajo vemos una barra de progreso y una sección donde pone usuario activo. La barra indica el porcentaje de proceso en la apertura de un formulario y la etiqueta nos indica el usuario autenticado que está activo en ese momento.

Inicio de sesión

Dado que es necesario estar autenticado para poder usarlo primero tendremos que mostrar un formulario de inicio de sesión con dos campos de texto para el nombre de usuario y la contraseña.

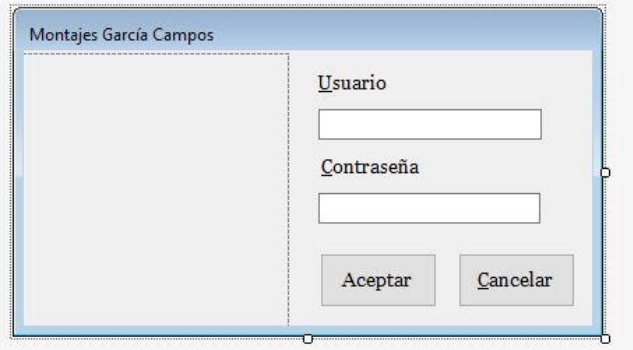

Ilustración 6 - Diseño formulario inicio de sesión.

El formulario de inicio de sesión que vemos en la ilustración 6 consta de un panel a la izquierda donde se situaría el logo de la empresa, dos campos de texto para introducir el usuario y la contraseña y dos botones para que el usuario decida si aceptar y entrar en el programa o cancelar y cerrarlo. Una vez el usuario ha rellenado los campos y clica en el botón de aceptar y si las credenciales son correctas el MDI empieza a mostrar su parte superior donde se encuentra el menú desplegable con las partes del programa. Empezaremos con los diseños de los mantenimientos. Un mantenimiento es un formulario donde se gestiona la información de una entidad concreta, es decir es donde se crea o desde donde se .

Mantenimiento de terceros

Una de las primeras partes del programa es el mantenimiento de terceros donde podremos dar de alta, modificar su información de estos.

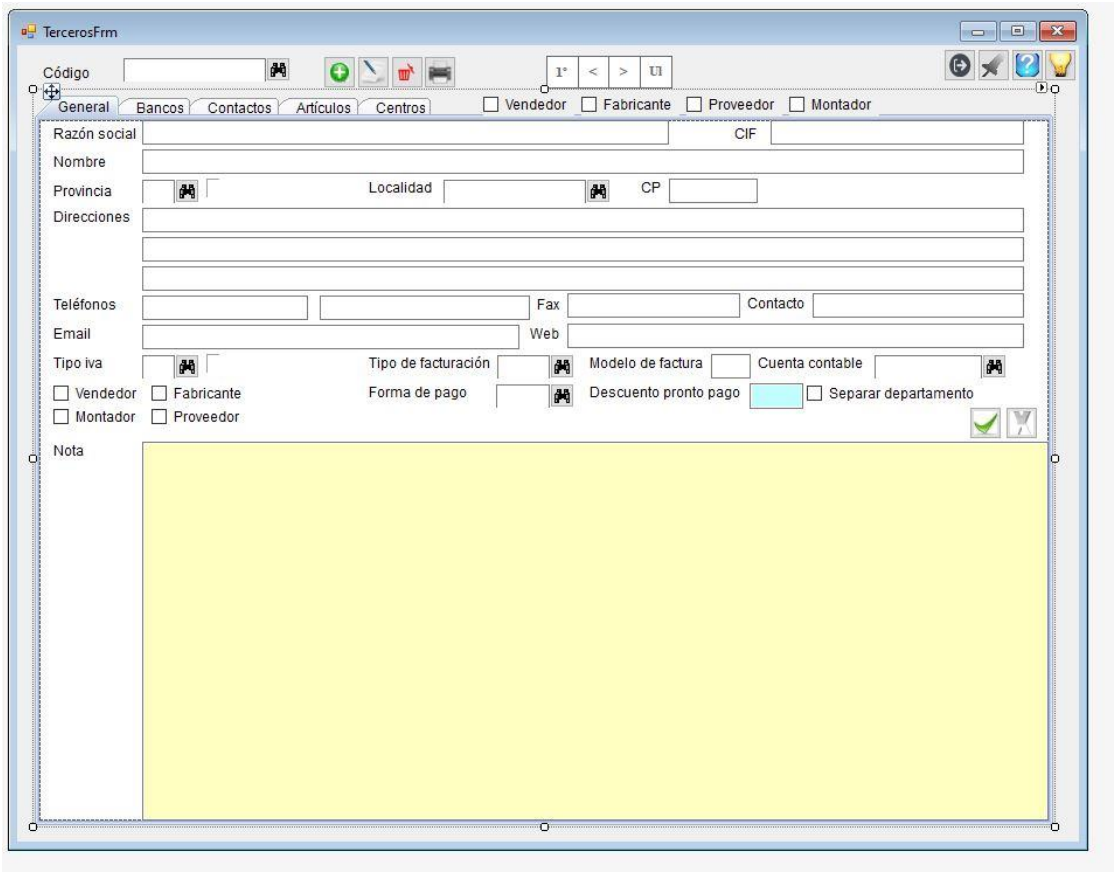

Ilustración 7 - Diseño mantenimiento de terceros.

En este formulario podemos ver ya la tónica que seguirán los formularios de mantenimiento. En la parte superior vemos un campo de texto acompañado de unos prismáticos donde irá el código del tercero. Este componente sirve para hacer una búsqueda por un campo del modelo de datos, en este caso, el nombre comercial. Más a la derecha se encuentran los botones de nuevo, modificar, borrar e imprimir (este no será visible nunca como veremos en la parte de implementación). A continuación, tenemos unos botones que nos permitirán ir al primer registro, al anterior, al siguiente o al último respectivamente y a la derecha tendremos otro conjunto de botones que se irá repitiendo en el resto de los formularios. El botón con la flecha cierra el formulario, el botón con la chincheta nos permite fijar este formulario en los accesos directos del mdi, el tercero nos da información sobre el formulario y el último nos deja escribir una nota que deseemos conservar en el formulario. Debajo de los botones blancos tenemos unos

checkbox que nos servirán de filtro a la hora de buscar en el campo código el registro que queremos discriminando por el tipo de tercero que es. Más abajo, tenemos las siguientes cinco pestañas:

1. General (ilustración 7): contiene los datos generales del tercero, en ella podremos modificar su información o introducirla por primera vez. Después de los campos de la información y antes del campo donde se podrá dejar una nota del tercero, tenemos un botón para guardar los cambios y otro para cancelar.
2. Bancos (ilustración 8): contiene la información sobre las cuentas bancarias del tercero. En ella vemos el mismo conjunto que en la parte superior, esta vez para introducir el código de la cuenta y poder crear una, modificarla o borrarla. Abajo hay un cuadro blanco que se corresponde con un grid donde podremos ver todas las cuentas del tercero y seleccionar una. Arriba tenemos un panel donde se mostrará la información de la cuenta seleccionada o se introducirá la de una nueva.

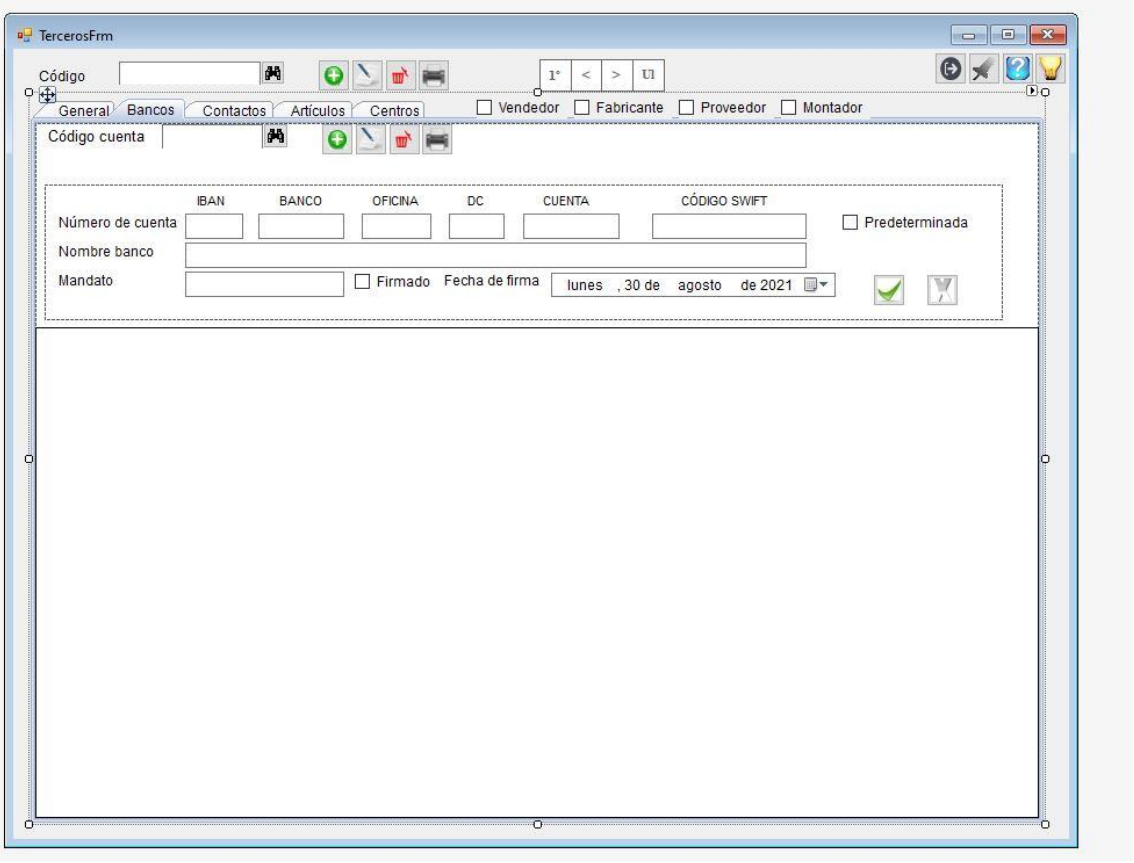

Ilustración 8 - Diseño pestaña bancos del mantenimiento de terceros.

3. Contactos (ilustración 9): contiene la información de los contactos asociados a ese tercero. El diseño es similar al anterior cambiando los campos por los correspondientes.

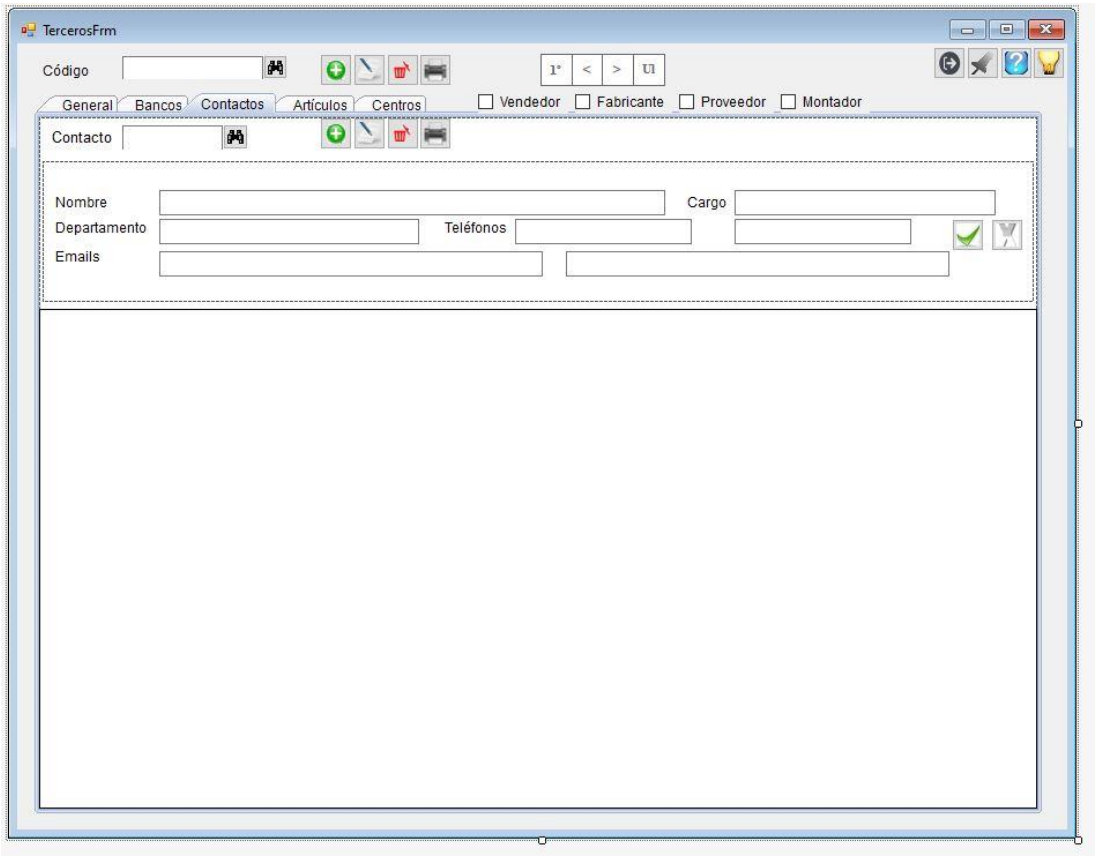

Ilustración 9 - Diseño pestaña contactos del mantenimiento de terceros.

4. Artículos (ilustración 10). En ella podemos ver los distintos artículos del tercero y podemos añadir, modificarlos o borrarlos. El diseño es similar a los dos anteriores.

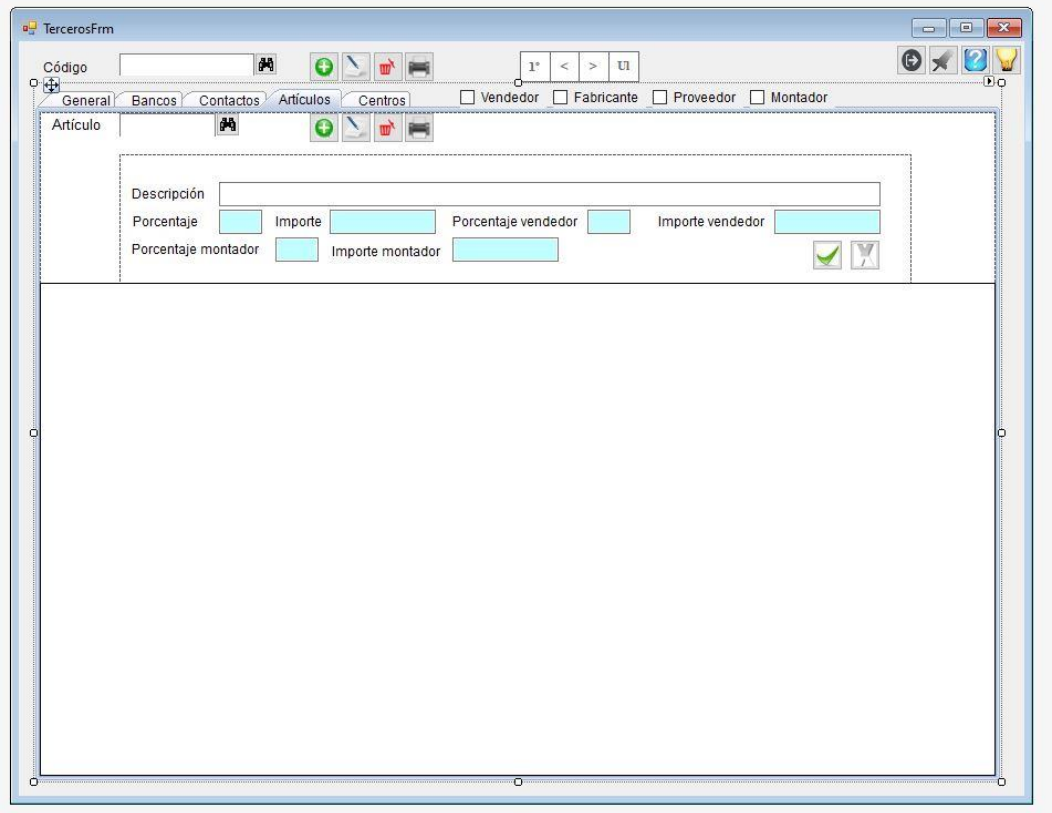

Ilustración 10 - Diseño pestaña artículos del mantenimiento de terceros.

5. Centros (ilustración 11). En esta pestaña encontraremos los centros de los que dispone un tercero y de los departamentos que contiene. En el diseño tenemos dos grids, uno arriba y otro abajo que quedan tapados por dos paneles que serán invisibles al inicio. En el grid de arriba tendremos los centros y si clicamos dos veces en uno se mostrará el panel con la información de este y podremos editarlo o borrarlo. Si le damos al botón de nuevo de más arriba podremos crear uno nuevo. El botón con la flecha ocultará el panel para volver a mostrar el grid con todos los centros. Con el grid de abajo pasará lo mismo con la única diferencia que su grid para mostrar los departamentos dependerá del centro que en el grid de arriba esté seleccionado. Para hacer más visible el estado de la selección se han usado dos etiquetas que mostrarán las selecciones en cada momento.

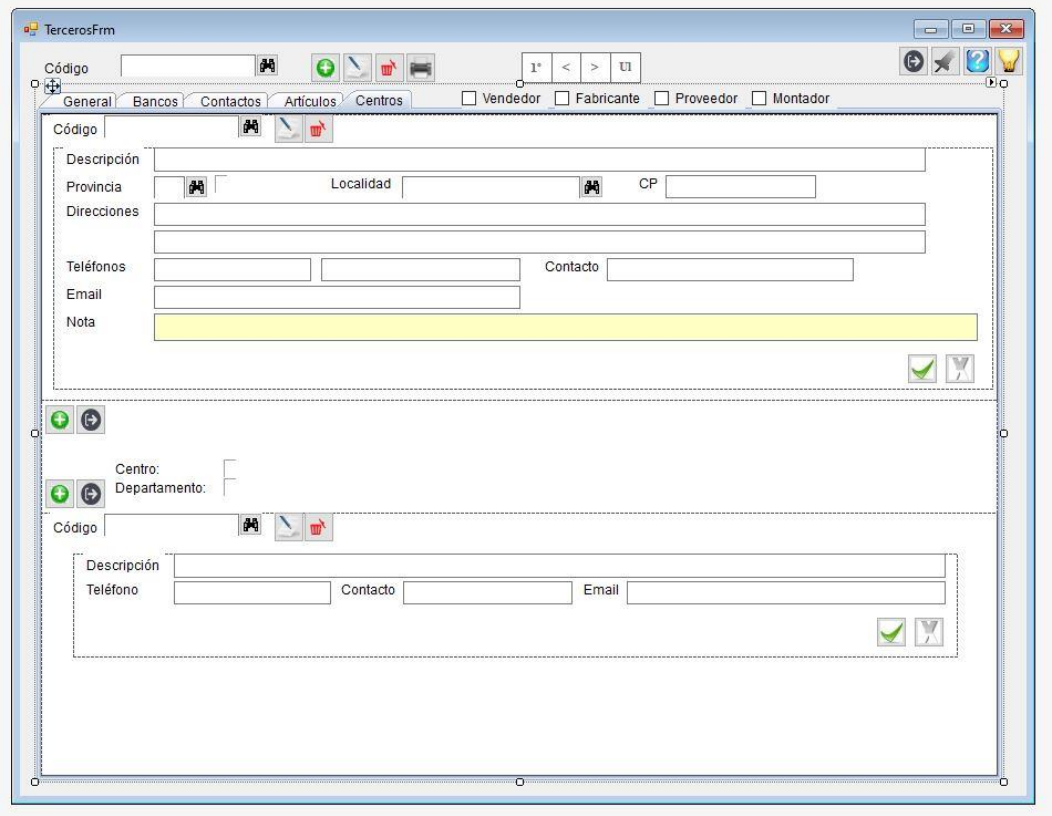

Ilustración 11 - Diseño pestaña centros del mantenimiento de terceros.

Y con esto tendríamos acabado el diseño del mantenimiento de los terceros. Los terceros tienen un tipo de IVA asociado y esta entidad puede ir cambiando conforme se vaya cambiando la ley así que también será necesario que hagamos un mantenimiento para los tipos de IVA de cada ejercicio.

Mantenimiento de tipos de IVA según ejercicio

Como vemos en la ilustración 12 podemos ver este mantenimiento sigue una estructura similar al anterior en la parte superior con los mismos conjuntos de botones solo que en este caso son para un ejercicio. Debajo de la primera fila de componentes tenemos un panel con los datos del ejercicio que estará deshabilitado hasta que el usuario no modifique o cree un registro nuevo. Más abajo, tenemos la pestaña de los tipos de IVA. Esta pestaña será similar a las anteriores con un grid donde se verán los tipos de IVA por cada ejercicio y un panel que se irá habilitando cuando el usuario quiera modificar o crear un tipo de IVA.

Por último, A la derecha se ha incluido un botón que servirá para generar un ejercicio para el año siguiente con los mismos tipos de IVA ya que de un año para otro se suelen mantener los mismos excepto si hay una nueva legislación.

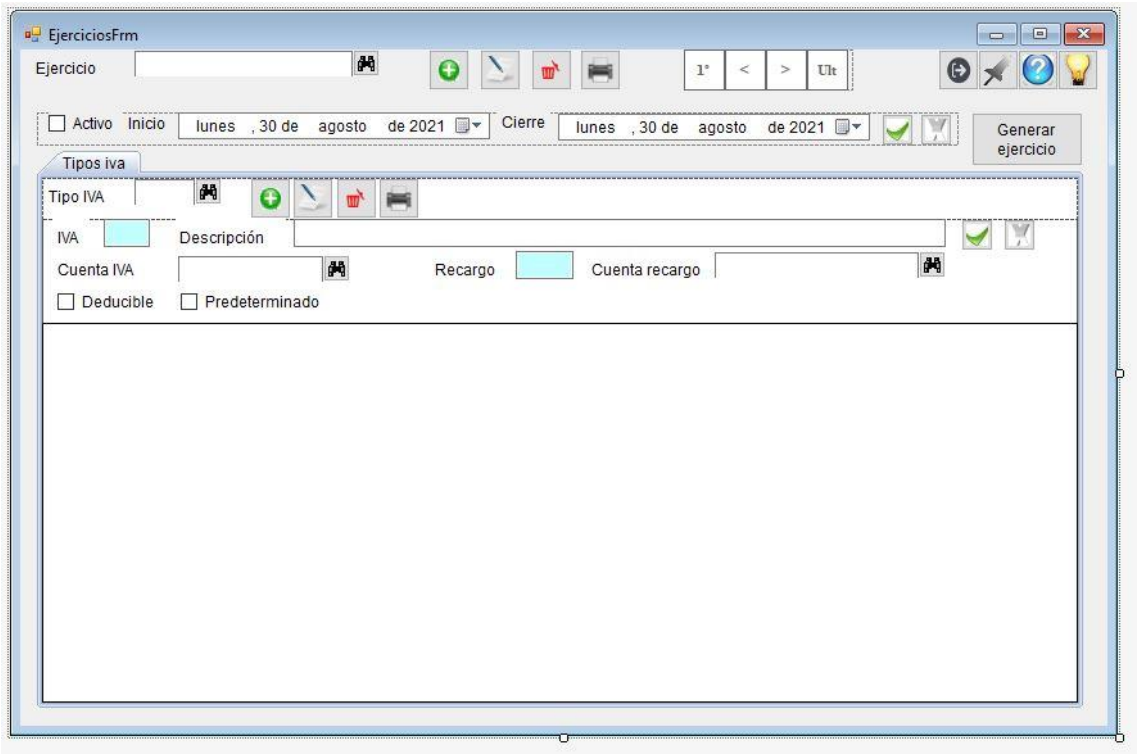

Ilustración 12 - Diseño del mantenimiento de tipos de IVA.

Mantenimiento de pedidos

A continuación, vamos a ver el diseño de una de las partes más importantes del programa como es el mantenimiento de pedidos. En esta ventana se gestionará toda la información que contenga un pedido.

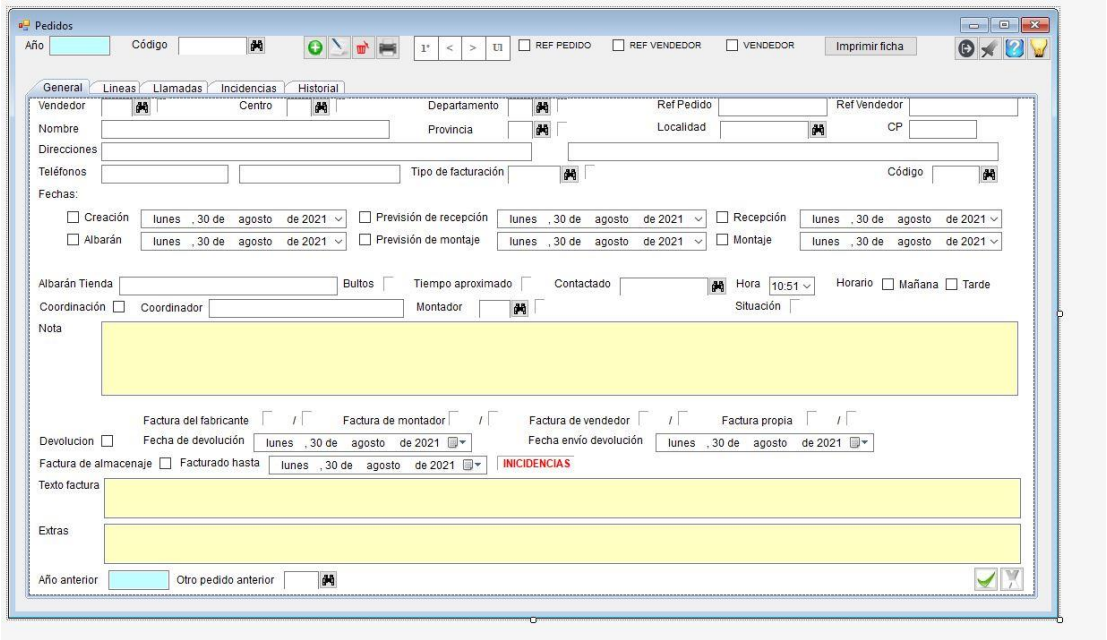

Ilustración 13 - Diseño del mantenimiento de pedidos.

En la ilustración 13 se ve que vamos a seguir el mismo patrón de diseño que el mantenimiento de terceros donde tendremos una línea arriba muy similar y una colección de pestañas de las distintas partes que componen un pedido. En la parte superior vemos que esta vez el código no es el único que sirve para identificar un pedido ya que necesitamos también saber su año, así qué, incluiremos un campo de texto azul (así indicamos que es un número lo que hay que poner). Entre los botones de navegación y los de la esquina superior derecha hemos colocado tres checkbox que servirán para cambiar el campo por el cual el usuario realizará la búsqueda en el componente del código (el que esté seleccionado será el campo por el que se realiza la búsqueda). Además, tenemos un botón para imprimir un listado del pedido con toda su información. Debajo, en la primera pestaña tenemos un panel con la totalidad de la información del pedido que el usuario podrá consultar, los dos botones que estamos utilizando para grabar el registro o cancelar y a destacar tenemos una etiqueta para señalar que el pedido tiene una incidencia (solo se mostrará si eso sucede).

Pasando a la pestaña de las líneas, tenemos un botón de nuevo y un grid que ocupará el resto de la pestaña para mostrar todas las líneas del pedido. Cuando se le dé al botón de nuevo o se haga click en alguna de las filas del grid se mostrarán los paneles que vemos en la ilustración 14. Primero, solo se mostrará un panel padre con la información común a todos los tipos de línea y cuando el usuario haya especificado el tipo se hará visible el panel hijo correspondiente al tipo de línea (panel con importes si es tipo 4 y panel con datos de recepción si es línea de tipo 1). Por último, tenemos los botones de grabar, cancelar y borrar registro y otro grid en el que iremos viendo el progreso de nuestra introducción de líneas ya que para facilitarle el trabajo al usuario cuando se genere una línea no se volverá a ver solo el grid de debajo (como pasará en el resto que siguen esta estructura) si no que se dejará preparado como si se fuera a crear otra línea.

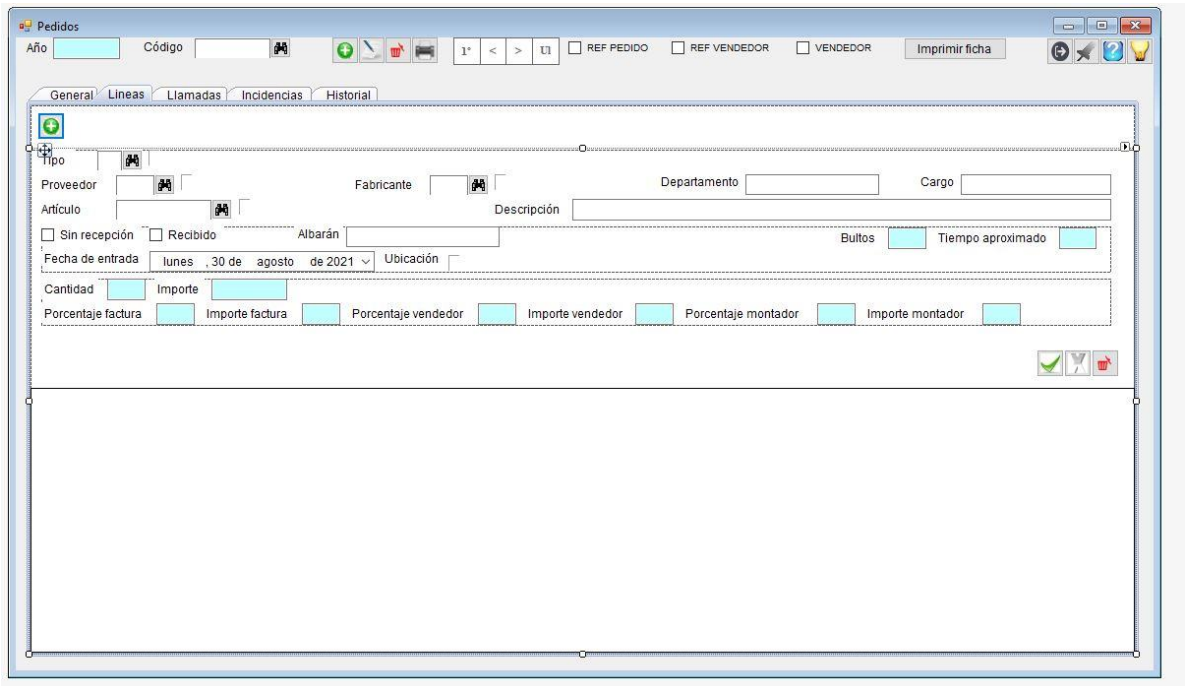

Ilustración 14 - Diseño de la pestaña líneas del mantenimiento de pedidos.

En la pestaña de las llamadas tenemos una disposición similar, un botón de nuevo registro y un grid donde se podrán ver todas las llamadas en las que se vea involucrado el pedido, pero además tenemos un botón para imprimir un listado con todas las llamadas que se han hecho. Encima del grid tendremos un panel que se mostrará igual que antes, cuando el usuario quiera registrar una nueva llamada, modificar una existente o borrarla. En el panel tendremos campos para rellenar con toda la información de la llamada, un campo de texto grande de color amarillo para dejar una nota (siguiendo el patrón de colores que estamos utilizando) y los botones de grabar, cancelar y eliminar.

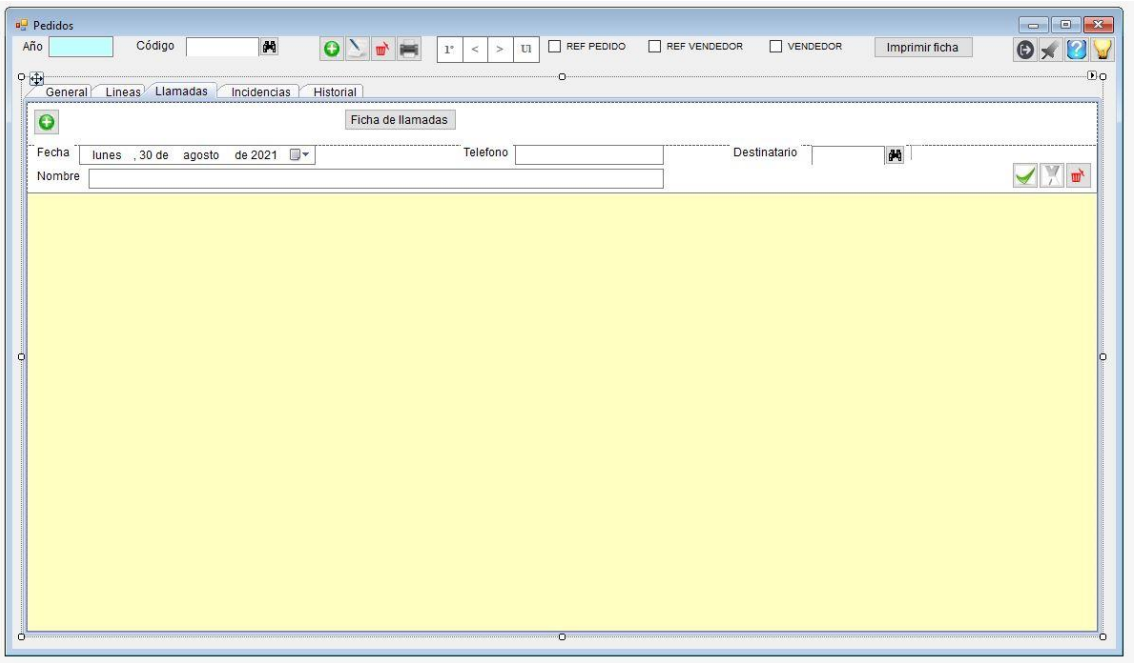

Ilustración 15 - Diseño de la pestaña de llamadas del mantenimiento de pedidos.

Por otra parte, tenemos la pestaña incidencias cuyo funcionamiento es idéntico a la de llamadas con la única diferencia de los campos y de que se han incluido tres botones: el botón de directorio de incidencia que le abrirá al usuario la carpeta específica de esa incidencia, el botón de generar una orden que el usuario utilizará para crear un pedido que resuelva la incidencia y el botón de hoja de reclamación que sirve para imprimir el listado con el mismo nombre cuya función hemos visto en el análisis.

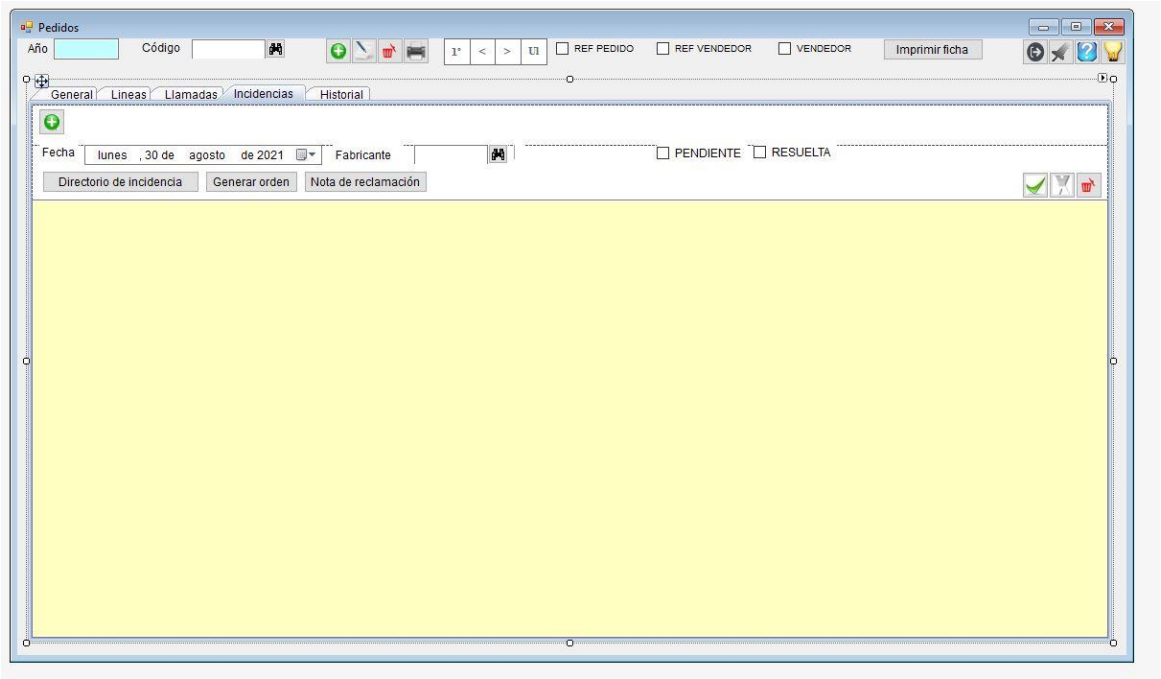

Ilustración 16 - Diseño de la pestaña de incidencias del mantenimiento de pedidos.

Por último, tenemos la pestaña de historial que solo tendrá un campo de texto deshabilitado cuya función será mostrar un histórico de las etapas que ha ido pasando el pedido.

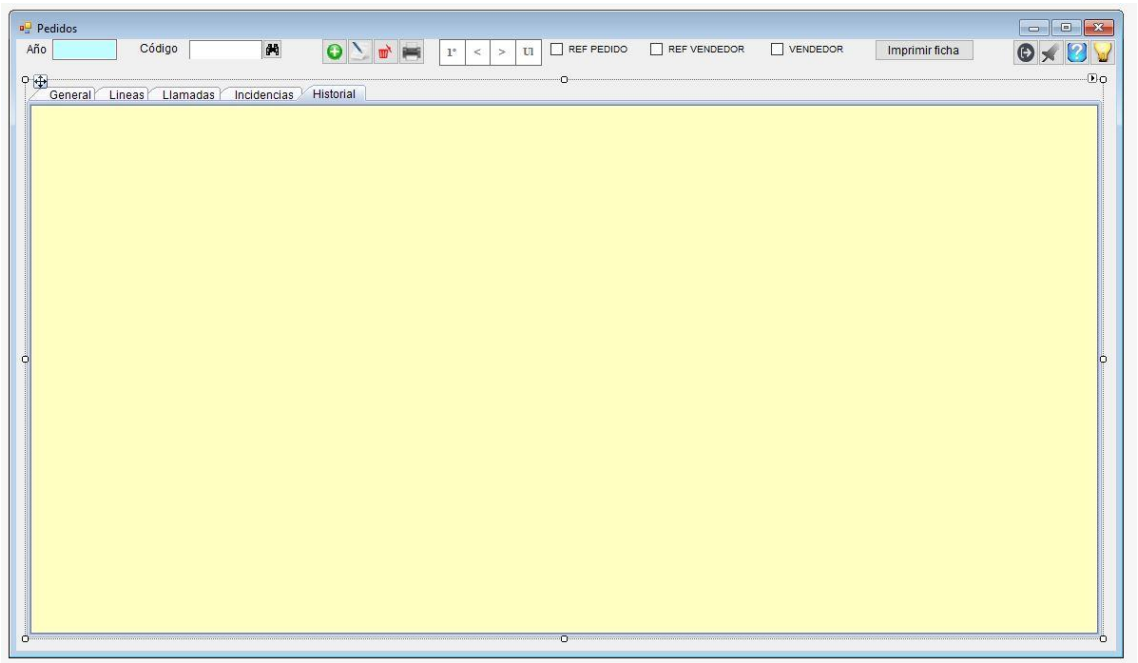

Ilustración 17 - Diseño de la pestaña de historial del mantenimiento de pedidos.

Como hemos visto en el análisis, de cada pedido se generan albaranes que el usuario deberá poder modificar o borrar y lo mismo con las facturas. Para este fin vamos a incluir un mantenimiento de albaranes y otro de facturas.

Mantenimiento de albaranes

Este mantenimiento seguirá un patrón similar al de los tipos de IVA. Ya que los albaranes también se identifican por el año en el que se realizan y la serie a la que pertenecen incluiremos dos campos de texto para que el usuario pueda especificarlo y ya pasar a seleccionar el albarán que quiera. El resto de los botones de la parte superior son iguales a los de otros mantenimientos. Debajo, tenemos un panel con los datos que lleva un albarán y como diferencia de otros mantenimientos aquí los importes son etiquetas ya que se calcularán por la suma de sus líneas. Las líneas las podemos ver en una pestaña debajo del panel donde encontraremos un botón de nuevo registro, un grid donde se pueden ver las líneas del albarán y encima el panel que se abrirá cuando se esté haciendo una nueva línea o modificando una.

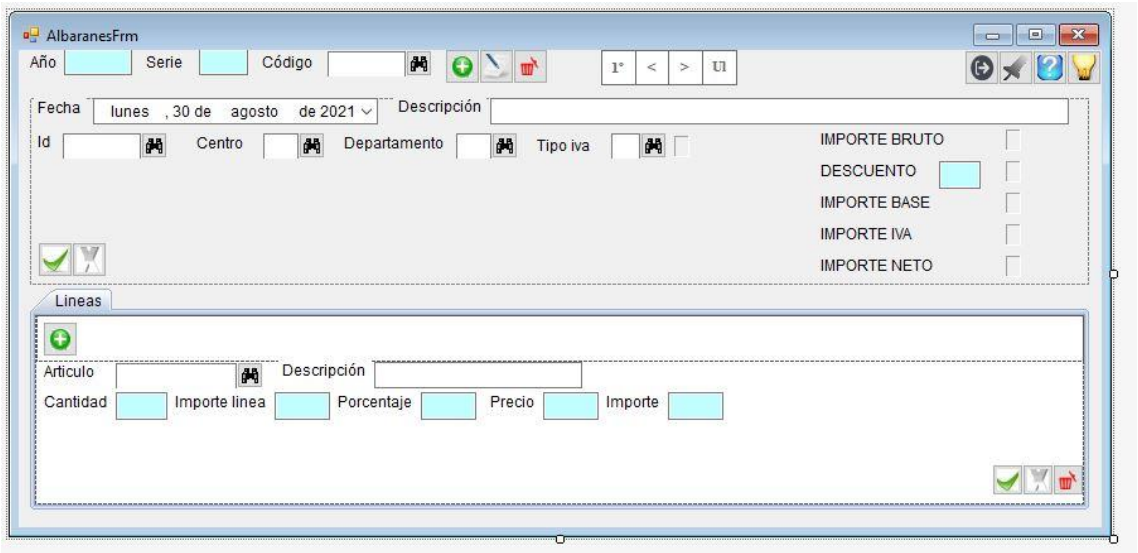

Ilustración 18 - Diseño del mantenimiento de albaranes.

Mantenimiento de facturas

Este mantenimiento será muy similar al de albaranes, pero se diferenciarán en que al ser la cabecera de una factura algo más importante que la de un albarán la pondremos en una pestaña y sus líneas en la siguiente pestaña. Además, como una vez los albaranes han llegado a la factura el importe tiene que quedar claro no se ha optado por indicar los importes con etiquetas si no con los campos de texto azules que quedarán deshabilitados, pero nos permitirán verlo mejor y evitar errores. Por último, también se diferencian en que en el de facturas se ha incluido un botón para contabilizar la factura que creará el asiento y apuntes contables en la base de datos de contabilidad, así como revertir el proceso descontabilizándola (solo si es la última). Una factura se podrá modificar y crear siempre pero nunca se podrá borrar excepto si es la última del año.

En la pestaña de los datos de la cabecera tendremos un panel con todos los campos de esta y a destacar tendremos un checkbox con fondo amarillo para resaltar si la factura está contabilizada. El funcionamiento será el mismo de siempre el panel estará deshabilitado hasta que el usuario entre a modificar o crear una factura desde los botones de modificar o nuevo respectivamente.

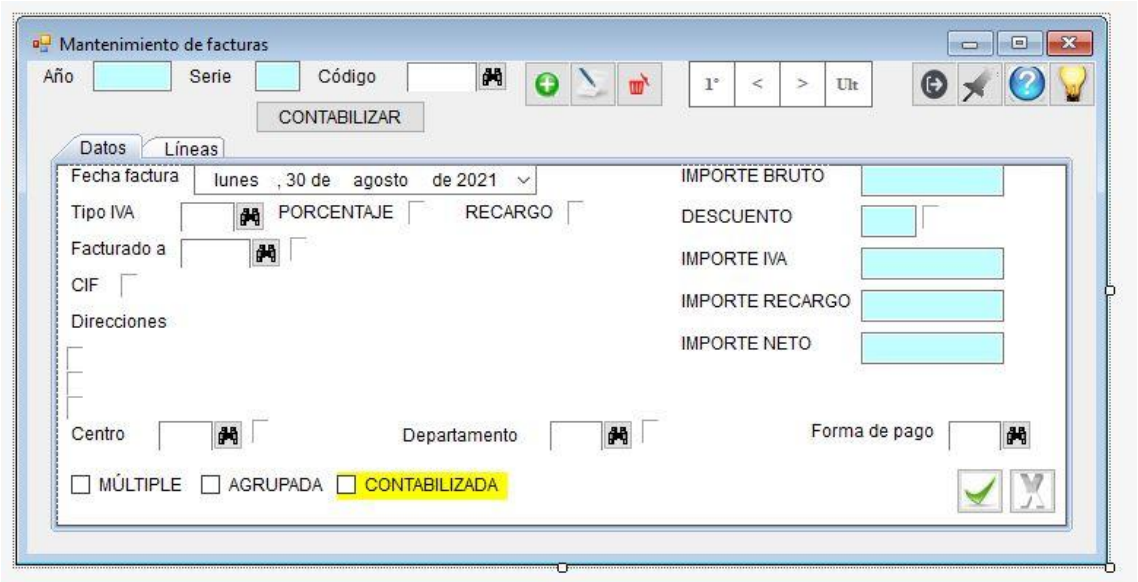

Ilustración 19 - Diseño del mantenimiento de facturas.

Las líneas son similares al resto de pestañas dedicadas a una colección de entidades donde tenemos un botón de nuevo y un grid donde se muestran todas las líneas. Cuando el usuario clica en una de ellas o quiere crear se hace visible un panel con los datos de la línea.

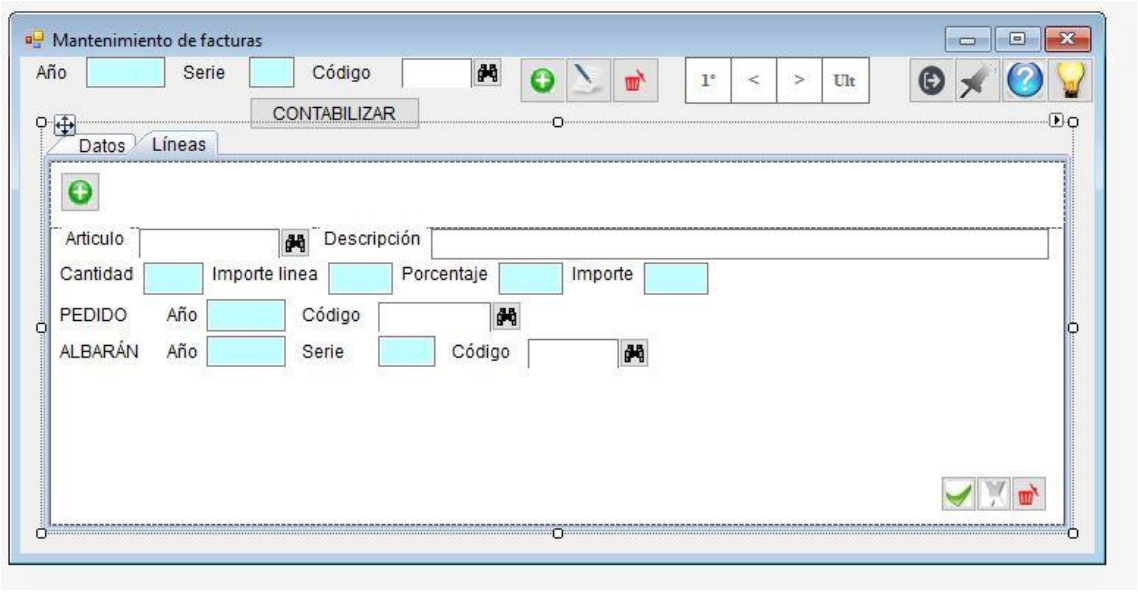

Ilustración 20 - Diseño de la pestaña de las líneas del mantenimiento de facturas.

Y con este último tendríamos todos los mantenimientos necesarios para ir gestionando las entidades involucradas en nuestro proyecto. Ahora, pasaremos a los formularios que llamaremos punteos donde se visualizan de una forma global las entidades principales del proceso de negocio y donde se realizarán muchos pasos de este. La idea es que los usuarios puedan acceder a los mantenimientos desde la barra de navegación del MDI, pero facilitarles el acceso desde los punteos como veremos más adelante conforme los vayamos viendo.

Punteo de líneas de pedido

En este punteo veremos todas las líneas de todos los productos con la posibilidad de filtrarlas.

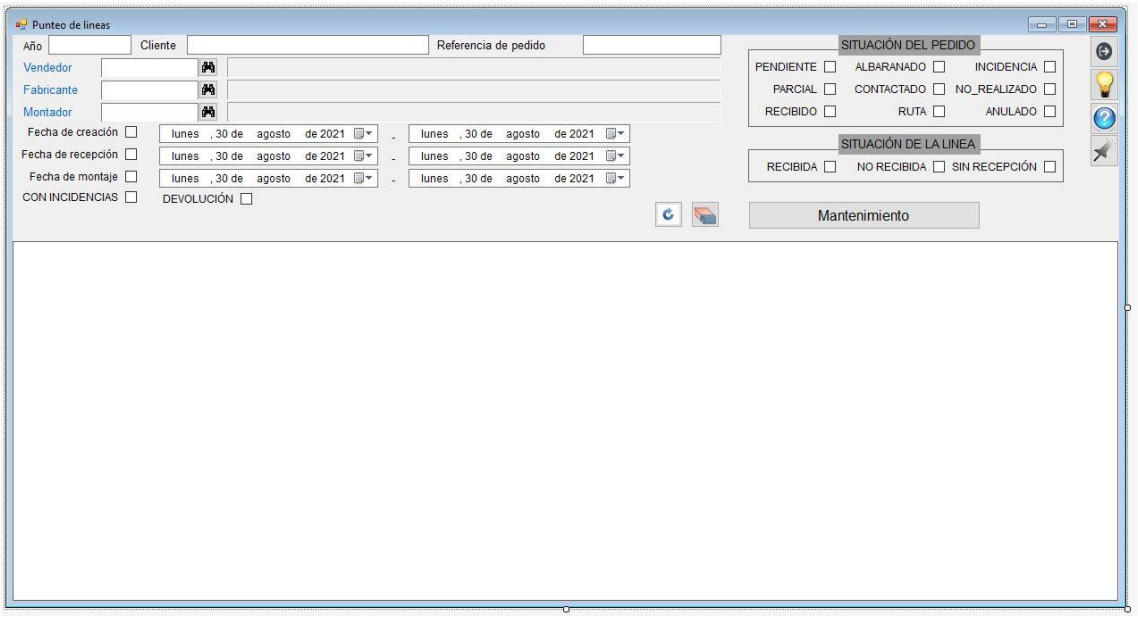

Ilustración 21 - Diseño del punteo de líneas de pedido.

Como vemos en la ilustración 21 tenemos una zona con todos los parámetros por los que se ha consensuado filtrar con el cliente, tanto del pedido del que forman parte como de la propia línea. Cuando el usuario haya rellenado o seleccionado los campos que considere oportunos podrá clicar en el botón con una flecha haciendo una forma de círculo que servirá para actualizar el grid de la parte inferior. A la derecha del botón se encuentra otro con una imagen de una goma de borrar que se encargará de limpiar los filtros y a la derecha de este encontraremos otro para abrir el mantenimiento de pedidos. Por último, tenemos a la parte superior derecha del todo los mismos botones auxiliares que en los mantenimientos, pero en los punteos los mostraremos en vertical. El grid que estamos incluyendo en los formularios es el SfDataGrid de la librería Syncfusion que hablábamos en el apartado de tecnologías y herramientas. Este grid nos permite herramientas de filtrado adicionales y la posibilidad de agrupar filas por el campo que se desee.

La función principal de este formulario es la del caso de uso de recibir mercancía. Esta recepción se hará de la siguiente forma:

Cuando el usuario seleccione el checkbox de situación de línea no recibida en el grid aparecerá un botón a la izquierda de cada fila con texto “Recibir” que abrirá una ventana de diálogo con otro formulario.

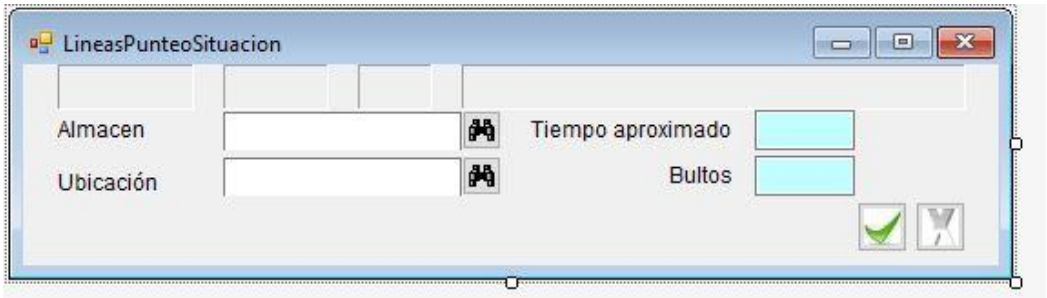

Ilustración 22 - Diseño de ventana de recepción de línea.

En el momento en el que el formulario se carga mostrará los datos de identificación de la línea (año, código de pedido, número de línea, cliente). En este formulario el usuario especifica el almacén y la ubicación donde se guardará la mercancía al entrar y una vez graba se hace el registro. En el caso de que se quiera registrar el movimiento de la mercancía de una ubicación a otra también se podrá si en lugar de seleccionar no recibida se selecciona recibida. El botón pasaría a tener como texto “mover”.

Punteo de pedidos

En el punto de pedidos será el formulario donde vamos a hacer pasar a un pedido por cada una de las etapas del proceso de negocio. En él, como en el punteo de líneas, el usuario puede consultar la información de todos los pedidos de la base de datos y filtrarla a su gusto.

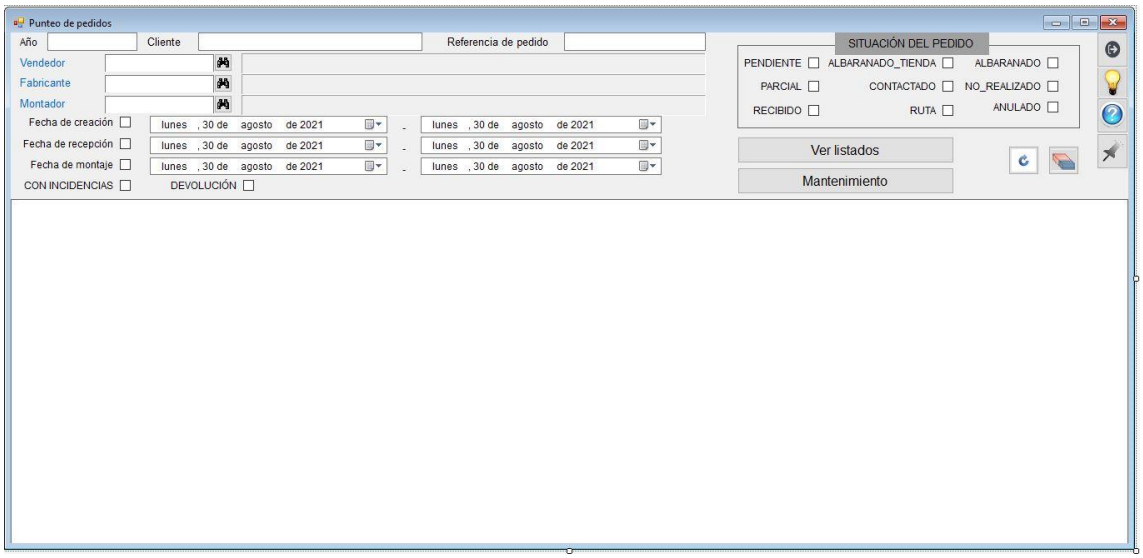

Ilustración 23 - Diseño del punteo de pedidos.

El diseño sigue la estructura de los punteos y solo cambia respecto en el que ya no podremos filtrar por la situación de una línea pues ahora estaremos tratando con el pedido en conjunto. Dispondremos del mismo botón para abrir el mantenimiento (también podremos acceder al mantenimiento de un pedido en específico mediante el doble clic en cualquier fila del grid) Además, tenemos un botón adicional con nombre “Ver listados” que utilizaremos para abrir una ventana con los distintos listados disponibles de los pedidos.

En la ventana podremos imprimir los tres listados principales de los pedidos que vimos en el análisis: el albarán de entrega de mercancía, el resumen de previsión de montadores y la hoja de ruta. Los filtros para cada uno y su correspondiente botón para imprimirlo estarán alojados en tres pestañas distintas, una para cada listado.

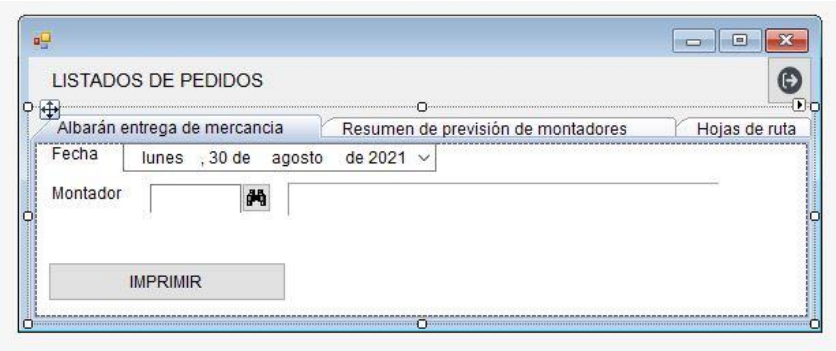

Ilustración 24 – Diseño pestaña albarán de entrega de mercancía del formulario ver listados.

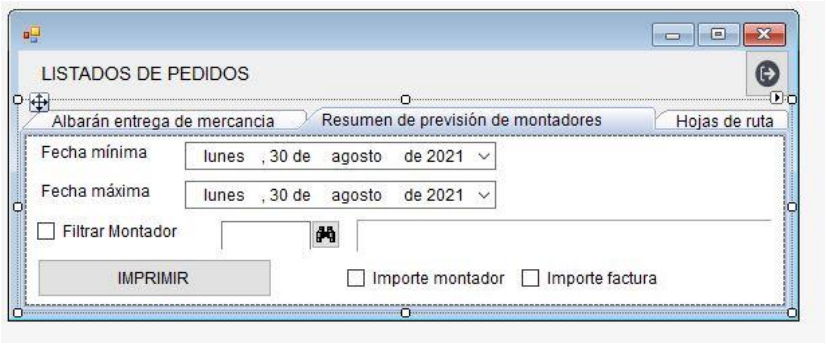

Ilustración 25 - Diseño pestaña resumen de previsión de montadores del formulario ver listados.

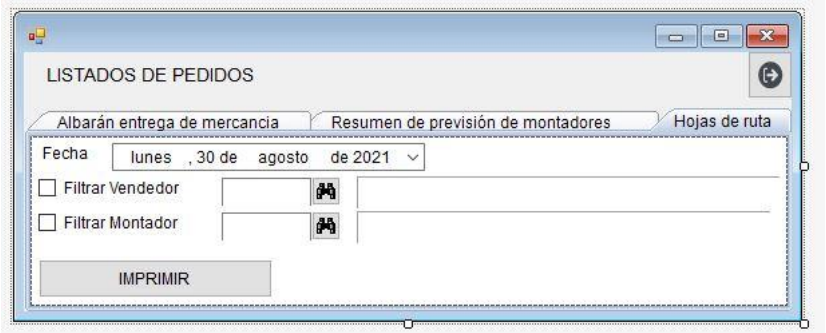

Ilustración 26 - Diseño pestaña hoja de ruta del formulario ver listados.

A destacar que cuando un parámetro de filtrado tiene un checkbox significa que es opcional ya que, por ejemplo, podemos querer sacar una hoja de ruta para un montador o para todos. El usuario tiene un botón en la esquina superior derecha que le permitiría salir de la ventana en el momento que quisiera.

Volviendo al punteo de pedidos, vamos a comenzar con como en el formulario se refleja el proceso de negocio. Como sabemos, cuando se han recibido todas las líneas de un pedido el pedido pasa al estado recibido. Si el usuario selecciona el checkbox “RECIBIDO” en los filtros y actualiza el grid podrá ir a la columna de albarán de la tienda del pedido que desee e introducirlo manualmente clicando dos veces en la celda y pulsando la tecla enter.

Una vez el pedido tiene el albarán de la tienda, el usuario llama al cliente para concertar fecha de montaje y se deja un registro de las llamadas. Esto se ve reflejado en el punteo de la siguiente forma: cuando el usuario selecciona el checkbox de situación del pedido

"ALBARANADO_TIENDA" y actualiza el grid aparece un botón a la izquierda de cada fila con la palabra contacto. Cuando el usuario pulsa el botón se abre una ventana de diálogo para registrar la llamada.

Ilustración 27 - Diseño de ventana de registro de llamada.

En esta ventana el usuario puede ver en las etiquetas de arriba el identificador del pedido (año y código) y el número de teléfono al que tiene que llamar. Debajo tiene un control donde podrá elegir el resultado de la llamada y si es que ha contactado se le mostrará el panel con la información que tiene que introducir. De las dos formas ya solo tendría que clicar en el botón de grabar y se registraría la llamada (si le da a cancelar saldría).

En el momento en el que un empleado haya concertado una fecha de montaje necesitará asignarle un montador. Para este fin tendrá que seleccionar el checkbox de situación de pedido "CONTACTADO" y como en el caso anterior aparecerá un botón a la izquierda de cada fila con esa condición, pero esta vez pondrá montador en el texto. Si el usuario clica el botón le llevará a la ventana de asignación de montador.

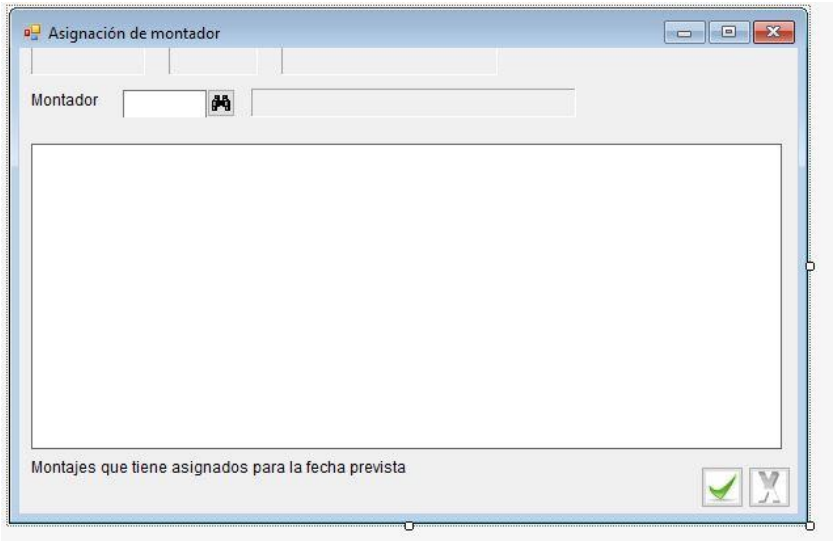

Ilustración 28 - Diseño ventana de asignación de montador.

En esta ventana, como en todas estas ventanas auxiliares que estamos viendo, tiene la identificación del pedido y información útil para la tarea que está haciendo (la fecha prevista de montaje). Cuando el usuario encuentra el montador que desea mediante el control se actualiza el grid de debajo y muestra todos los montajes que tiene el montador escogido programados para

ese día. Habiendo valorado el cliente el montador que debe escoger solo tendrá que pulsar el botón de grabar y el de cancelar si desea salir de la operación.

Por último, cuando un usuario quiere dar el pedido como montado seleccionará el checkbox "RUTA" y actualizará el grid para que aparezca un botón a la izquierda de cada fila con texto "MONTADO". En el momento en el que usuario da clic al botón se abrirá una ventana de diálogo auxiliar que solo tendrá un componente para escoger fecha en el que introducirá la fecha de realización del montaje. Con todo listo solo tendrá que clicar en el botón de grabar y ya se daría el pedido como montado. En esta ventana el usuario tendrá como información útil a quien se facturará el pedido.

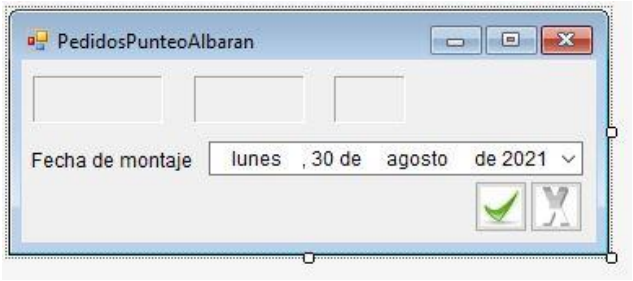

Ilustración 29 - Diseño ventana para dar pedido como montado.

Y con este último formulario tendríamos toda la parte del punteo de pedidos. En el transcurso de las etapas del pedido pueden producirse incidencias que habremos dado de alta en el mantenimiento de pedidos. El usuario debe ser capaz de monitorizarlas y en el momento oportuno resolverlas y para eso se ha diseñado un punteo de incidencias.

Punteo de incidencias

En este formulario podremos ver toda la información de las incidencias registradas en la base de datos y que tienen un pedido que la resuelve. Como todos los punteos hasta el momento tiene su grid en la parte inferior y sus filtros en la superior.

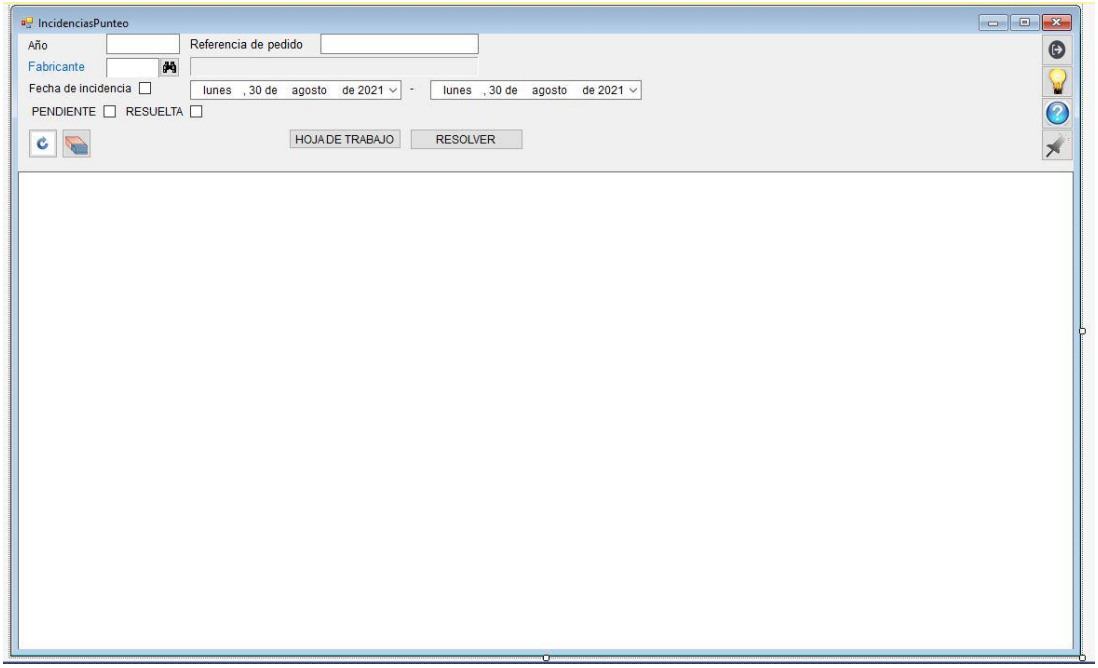

Ilustración 30 - Diseño del punteo de incidencias.

El punteo servirá para, además de ver la información de las incidencias, imprimir la hoja de trabajo de la resolución de la incidencia y resolverlas. El mecanismo para los dos usos es el mismo: el usuario selecciona en el grid la fila de la incidencia escogida y hace click en el botón de la operación que desee. Además, si el usuario hace doble click en las celdas de las columnas que identifican al pedido original o al que la resuelve se le abrirá el mantenimiento del respectivo.

Con esto finalizaría toda la lógica de negocio de los pedidos y pasaríamos a la facturación. Hemos visto que el caso de uso de contabilizar una factura se podrá hacer en el formulario de mantenimiento de estas, pero aún queda toda la parte de generación automática. El primer paso de este proceso se realiza cuando facturamos los albaranes provenientes de los pedidos y para ello utilizaremos un punteo de albaranes.

Punteo de albaranes

En este formulario el usuario podrá visualizar todos los albaranes registrados y podrá crear una factura a partir de los que desee. El formulario sigue la tónica de los punteos, grid en la parte de abajo, filtros en la de arriba y botones auxiliares en la derecha. Este formulario tiene en especial que su grid permite selección múltiple pues será necesaria para cuando el usuario quiera facturar varios albaranes a la vez. La selección múltiple se hará marcando la columna de elementos checkbox que el grid tendrá a la izquierda de cada fila. Además, permite especificar la cuenta bancaria a la que se ingresará lo cobrado.

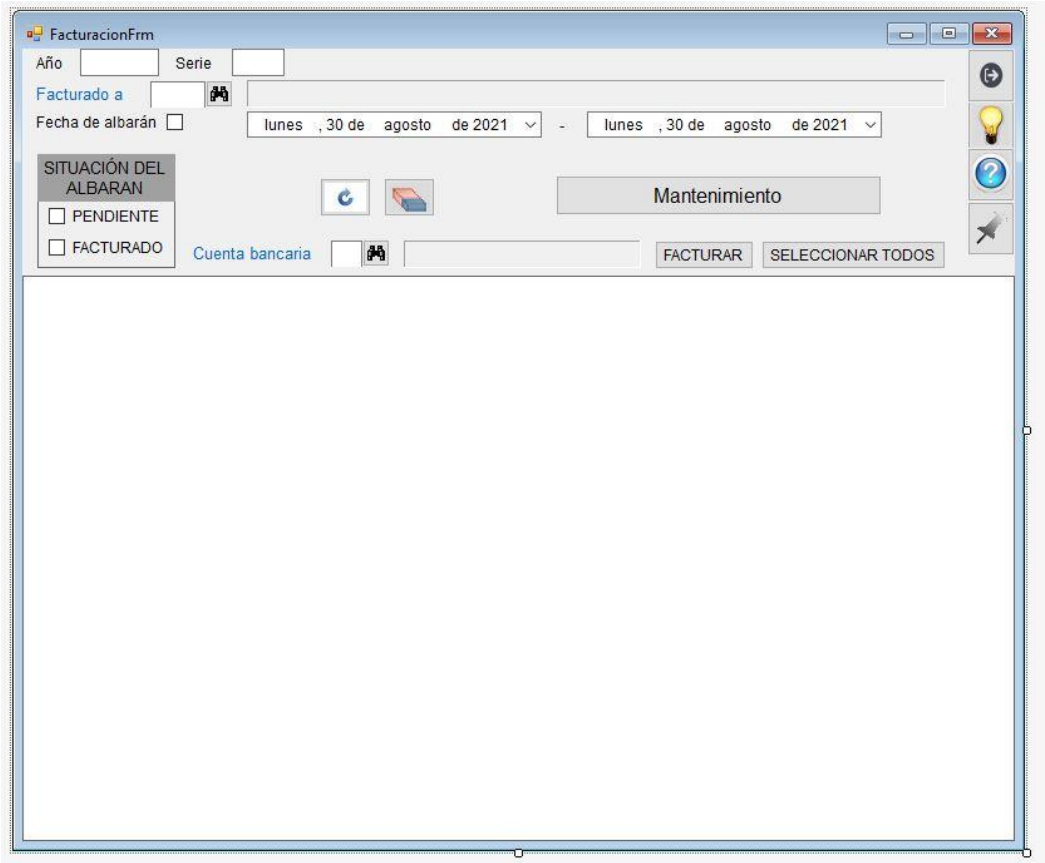

Ilustración 31 - Diseño del punteo de albaranes.

Cuando el usuario quiera facturar los albaranes solo tendrá que seleccionarlos marcando en la columna antes dicha los que quiera y después clicar en el botón con nombre “FACTURAR”. Si

quiere especificar la cuenta solo tendrá que hacer la búsqueda con el control situado al lado de la etiqueta con texto azul “Cuenta bancaria” y seleccionar la que desee antes de facturar. Una vez que acaba el proceso se actualiza el grid. Cuando los albaranes se hayan facturado se generarán automáticamente las facturas correspondientes y sus recibos.

Con las facturas y recibos generados ya solo faltarían formularios para ver la información de estos de forma conjunta.

Punteo de facturas

El usuario utilizará este formulario para ver todas las facturas en el sistema, acceder al mantenimiento de cualquiera de ellas e imprimirlas.

Lo único particular de este formulario en cuanto a diseño es el hecho de que los botones auxiliares vuelven a estar mostrados en horizontal. Esto se debe a que no necesitamos tantos filtros como en punteos anteriores y nos permite ahorrar espacio en el MDI. El grid de este punteo también permite selección múltiple para poder imprimir varias facturas a la vez.

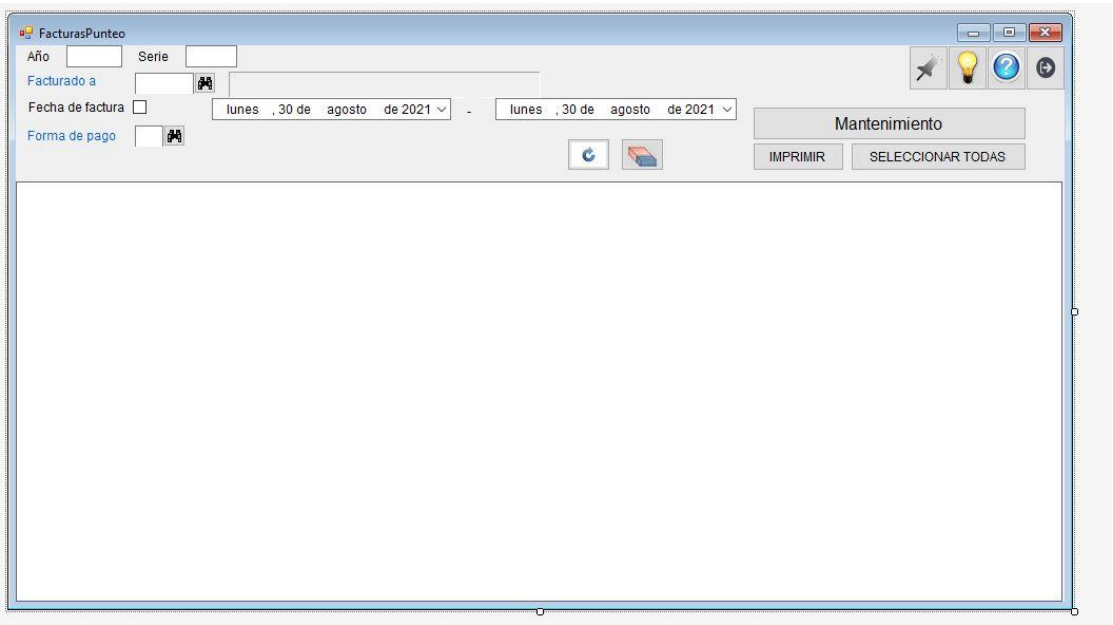

Ilustración 32 - Diseño del punteo de facturas.

El usuario solo tendría que seleccionar las facturas que desee imprimir y clicar en el botón respectivo. Si el usuario quiere acceder al mantenimiento de facturas podrá hacerlo mediante el botón con ese nombre o clicando dos veces en cualquier factura del grid. No se ha pensado en permitir contabilizar facturas de aquí para evitar errores que puedan suceder a causa de una selección errónea.

Punteo de recibos

Este formulario incluirá alguna función más que el de facturas ya que los recibos, aunque no requieran mantenimiento si que necesitan actualizarse sus situaciones y hemos dado la posibilidad de registrar pagos parciales de ellos.

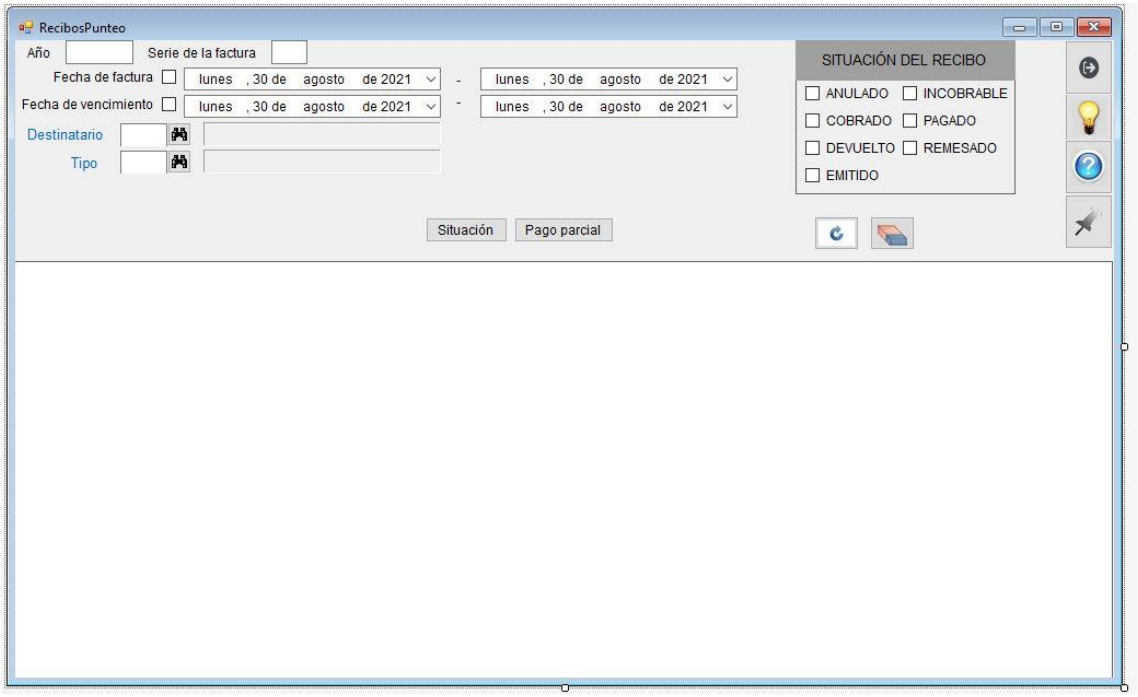

Ilustración 33 - Diseño del punteo de recibos.

El punteo sigue con el mismo diseño de la mayoría de los punteos y su diferencia principal es los dos botones correspondientes a las operaciones que vamos a poder realizar.

Con un clic en el botón de situación y un recibo seleccionado se abrirá una ventana que nos permitirá cambiar su situación a, por ejemplo, pagado o cobrado.

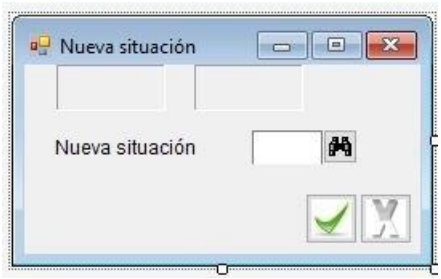

Ilustración 34 - Diseño ventana de cambio de situación de recibos.

En esta ventana solo necesitaremos hacer una búsqueda de la situación a la que queramos cambiar el recibo y clicar en el botón de grabar. También podemos clicar en el botón de cancelar si deseamos salir.

Con un clic en el botón de pago parcial y un recibo seleccionado se abrirá otra ventana que nos permitirá dar el recibo como cobrado con la cantidad que especifique el usuario y generar otro nuevo con la cantidad restante.

Ilustración 35 - Diseño ventana de pago parcial de recibos.

La ventana consta de las etiquetas que siempre usamos para saber el registro que estamos modificando, un campo de texto azul donde introduciremos la cantidad y los botones de grabar y cancelar. Cuando hay una cantidad y se pulsa el botón de grabar se completa la operación y se cierra la ventana. Con este último formulario ya tendríamos todos los punteos necesarios.

Impresión de listados

Para finalizar el apartado solo quedaría mostrar el formulario que se abre cada vez que vamos a imprimir un listado.

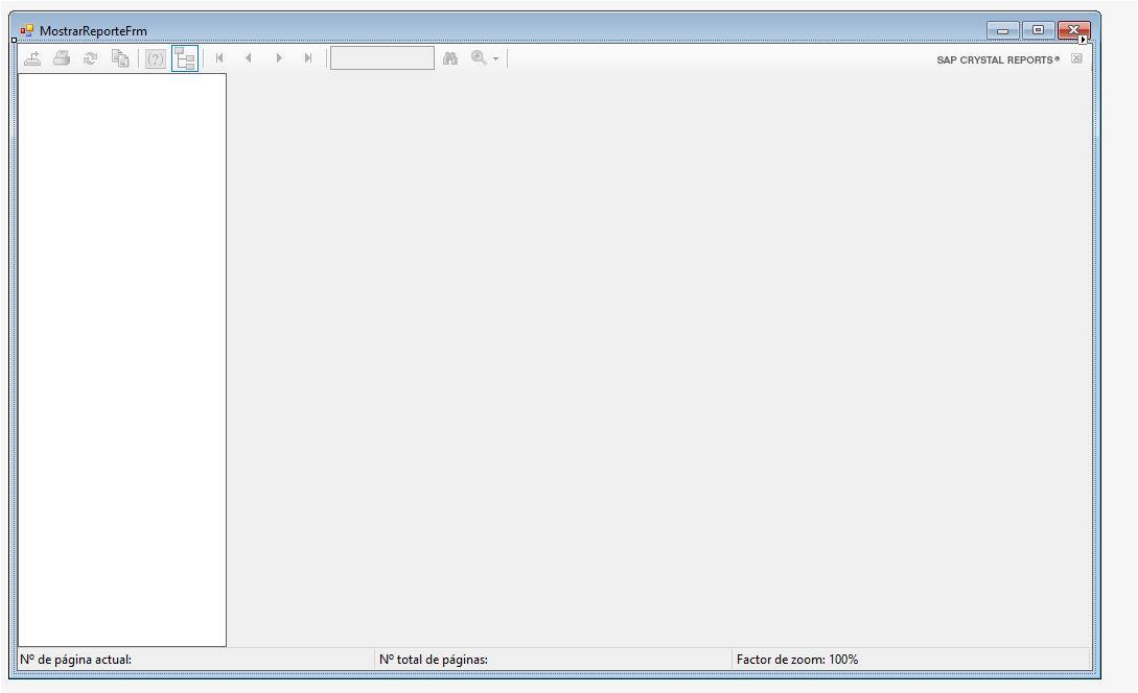

Ilustración 36 - Diseño formulario auxiliar para la impresión de listados.

Este formulario solo consta de un componente proveniente de la librería de Crystal Reports que muestra el formulario que se va a imprimir para poder leerlo antes de hacerlo y nos da información adicional como el número de páginas.

6. Implementación

En este apartado se va a relatar el proceso de implementación del proyecto:

El desarrollo del proyecto se sitúa en el comienzo del periodo de prácticas en diciembre del 2020 cuando se me dan unas lecciones de Visual Basic. Cuando el lenguaje ya resultaba familiar se me encargó el desarrollo de la solución sobre la que trata este proyecto. La metodología seguida iba a ser la de espiral. El proyecto iba a ir encadenando fases de análisis, diseño, desarrollo y evaluación. La primera iteración consistiría en hacer un prototipo del programa que incluyera el funcionamiento básico (inicio de sesión y navegación) y una versión primeriza de la gestión de pedidos.

Se hizo una copia del diseño de la base de datos de usuarios de la empresa y se usó como base de datos de usuarios en producción y se añadió algún usuario para ir probando el inicio de sesión. Una vez se creó esta primera base de datos se crea una preliminar capa de acceso a datos que iría evolucionando con el proyecto. Por el momento, nos permitiría obtener un dataset de un usuario.

Se consiguió que el programa iniciase en el MDI con el formulario de inicio de sesión activo. El control del formulario de inicio de sesión invocaba la capa de acceso para obtener la información del usuario y se comparaba con la obtenida en los campos de texto. Una vez completado el registro, el MDI mostraba el menú desplegable.

Con la parte básica completada se empezó a crear la base de datos del programa con las primeras tablas. Se importó de la base de datos antigua los datos de tablas que no iban a necesitar un mantenimiento que las controlase en el programa porque iban a permanecer siempre iguales como los tipos de facturación o las formas de pago. Más tarde se crearon unas tablas para proveedores, vendedores, montadores y fabricantes que serían las que darían pie a los primeros mantenimientos preliminares. Se creó un mantenimiento para cada uno que dejaba crearlos, modificarlos y borrarlos. Para empezar a hacer el mantenimiento fue necesario estudiar como funcionaba el componente desarrollado por mi empresa para realizar búsquedas en la base de datos (campo de texto con icono de prismáticos visto en el apartado de diseño) pues iba a ser utilizado en prácticamente la totalidad de los formularios. A este componente llamado MyControl se le asigna una consulta en la que se especifica el campo por el que se busca (nombre en el caso de los terceros) y el que se obtiene (el identificador).

Para finalizar, se creo la tabla de pedidos y la de artículos basándonos en la información recogida y en sus tablas anteriores que fueron difícile de interpretar porque tenía muchos campos que costaban de entender. Por ejemplo, en la tabla de pedidos había una columna que se llamaba mot y que viendo sus pedidos antiguos solo veíamos que había una n o una m. Ese campo acabaría teniendo sentido después pero no se utilizaría en el programa que se estaba desarrollando (era si el pedido estaba montado y en nuestro caso eso lo íbamos a controlar por la situación del pedido). Una vez creada la tabla se hizo un mantenimiento de pedidos con la pestaña de líneas y la de historial además de la principal y en él se empezaron a usar los grids. Fue muy importante leer la documentación de estos pues había varios eventos que podíamos controlar (aunque estuviera en C#). Al final se escogió el evento CellDoubleClick para que el

usuario pudiera acceder a la línea del pedido a partir del grid en lugar del que controla un simple clic.

(DataGridView.CellDoubleClick Event, s.f.)

Como veremos en el apartado 7, los clientes estuvieron contentos con el progreso en la primera prueba y después nos explicaron como iban a llevar el proceso de facturación. Hasta ese momento estaban facturando a las tiendas y fabricantes por pedido y a petición de sus clientes y como sugerencia nuestra decidieron introducir los albaranes como paso intermedio.

Cuando vi como iba a ir la facturación me di cuenta de que había que juntar los proveedores (aunque no participasen), los montadores, los vendedores y los fabricantes en una sola tabla para que la facturación estuviera integrada y no estuviera repartida en cinco tablas como tenían antes así que de las partes comunes de cada uno se creó la tabla terceros y su mantenimiento. Además, se añadieron también sus contactos, sus centros y sus departamentos porque nos comentaron que lo necesitarían esa especificación para determinados clientes muy importantes.

Para la siguiente iteración tendríamos que implementar el proceso de negocio de los pedidos (excepto las incidencias por el momento) y la generación de albaranes. Durante este periodo se actualiza la tabla de pedidos y la de líneas de pedido para incluir campos como la fecha de previsión de montaje o el albarán de la tienda que serían necesarios para satisfacer la demanda. Además, añadimos la tabla de llamadas de pedido para poder registrar principalmente las llamadas al cliente en el momento de concertar la fecha de montaje y la tabla para registrar las ubicaciones donde se guarda la mercancía que entra. Con las tablas referentes a los pedidos listas se crearon el punteo de líneas y el punteo de pedidos. En estos formularios se empieza usar el grid de Syncfusion que iba muy bien para este caso pues la posibilidad de agrupar las filas por cualquier campo y tener herramientas de filtrado adicionales a las que se iban a implementar iba a ir muy bien de cara a la usabilidad de la aplicación. A destacar está el momento en el que hay que implementar que un botón se situé como una columna más a la izquierda del todo pues hubo que buscar en la documentación de Syncfusion como hacerlo posible.

(How to add the Button and Text in GridTextColumn in WinForms DataGrid (SfDataGrid)?, 2020)

Por último, en esta iteración, se creo la tabla de albaranes y la de sus líneas para que cuando acabase el proceso de negocio se generaran dependiendo del tipo de facturación escogido. Además, se hizo una primera versión del punteo de albaranes para que se pudieran ver los albaranes generados.

La capa de acceso a datos iba avanzando en paralelo a los avances anteriores ya que siempre que un control iba a necesitar un dataset de un modelo que aún no se había contemplado iba a ser necesario implementar una función que lo hiciese.

Después de este punto, empieza la siguiente iteración en la que después de arreglar los fallos del estado en ese momento del proyecto nos piden que integremos sus listados en el programa, implementar el punteo de facturas y hacer formularios de mantenimiento de facturas y albaranes (lo que sorprendió pues pensaba que no deberían de poder modificarse). Como no íbamos a saber cuántos son hasta que no nos los fueran enviando todos se ha mantenido como un caso de uso general en todo el desarrollo.

Empezando la iteración se crea la tabla de facturas basándonos en las múltiples que tenían en la base de datos antigua y se implementa su mantenimiento. Llegados a este punto realizar un mantenimiento era bastante sencillo porque se iba repitiendo una estructura común, pero en este caso había que tener en cuenta que las facturas son algo delicado y había que minimizar los posibles errores como borrar una factura que no es la última ya que hacienda exige que las facturas tengan un orden consecutivo y no pueda haber saltos entre ellas. Una vez creado su mantenimiento, también se crea su punteo para ir preparando la posibilidad de imprimirlas.

Por otro lado, se crea también el mantenimiento de albaranes y se añade al punteo la capacidad de facturarlos. Este proceso fue complejo pues es donde tuve que poner en práctica todo lo que había ido aprendiendo en las reuniones sobre facturación. Explicándolo de forma general, cada vez que se factura un albarán se mira si hay una factura abierta y si no la hay se crea una que quedará abierta hasta que ya no queden albaranes que facturar. Este proceso debía tener en cuenta muchos factores y se tardó días en implementar.

Respecto a los listados, se creó el formulario auxiliar para poder visualizarlos en el programa y se hizo que se le llamara desde los sitios que hiciera falta un listado como por ejemplo en el punto de facturas se le llamaría para imprimir una o varias facturas. Al principio tenía la idea de usar el botón que se ve en los mantenimientos con un icono de impresora para imprimir listados, pero conforme se van añadiendo descarto esa idea y pondo esos botones sin visibilidad por si en algún momento quiero retomarla. Para utilizar los listados de Crystal Reports hubo que aprender a modificarlos para que se adaptasen a la nueva base de datos del programa y hubo que implementar en el programa funciones para servir de datasets a estos listados y así que Crystal no tuviera que interactuar con la base de datos directamente. Con esto se finalizaría el desarrollo de esta iteración. De esta forma se dio soporte a los listados de albarán de entrega de mercancía, resumen de previsión de montadores, hoja de ruta y facturas.

Para la última iteración debíamos incorporar la gestión de incidencias, la contabilización y la generación de recibos. Para ello se me entrega los campos que tendría que rellenar en las tablas de asientos y apuntes en la base de contabilidad y se me explica cómo funcionan las incidencias en dos reuniones.

Para dar soporte a las incidencias se crea una tabla para registrarlas. La forma en la que gestionaban antes las incidencias daba lugar a varios errores porque en su programa antiguo las incidencias se buscaban por su fecha y no se podía ver con claridad del pedido que provenían y el pedido que las resolvía.

Teniendo en cuenta la situación decido incorporar la creación de incidencias en el mantenimiento de pedidos para que siempre quede claro el pedido del que proviene y añado campos a la tabla pedidos para indicar si proviene de una orden anterior. En el mantenimiento también se incluye un botón para crear o acceder a la carpeta compartida en red de incidencias por sugerencia del responsable de incidencias de la empresa que perdía mucho tiempo en crearla y en encontrarla después. Por último, en esta parte también incluyo el listado de la nota de reclamación. Además, creo un punteo de incidencias que permitiría al usuario verlas en conjunto y buscarlas por los filtros adecuados. Este punteo también permitiría imprimir el listado de hoja de trabajo.

Con respecto a la contabilización de facturas, sigo las pautas de evitar errores y incluyo la opción de contabilizar y descontabilizar en el mantenimiento de facturas para que se vea de la forma más clara posible.

Por último, para la generación de recibos se crea primero su tabla y la de sus situaciones para poder guardarlos en la base de datos. Después, se modifica la generación de facturas para que, una vez generada, se generen los recibos necesarios en función de la forma de pago. Por último, se implementa el punteo de recibos para poder visualizarlos y para poder ir cambiando sus situaciones. Además, pienso en que se pueda pagar parcialmente un recibo (generando otro con la cantidad) y creo otra ventana.

Con estos cambios acaba la última iteración y comienza el proceso de implantación en el mes de agosto de 2021 en el que ya solo se van haciendo cambios menores para mejorar la usabilidad como el de insertar botones de acceso directo en el MDI o añadir botones para acceder a los mantenimientos desde los punteos para hacer más rápida e intuitiva la navegación mientras los usuarios trabajan en paralelo con el programa viejo y el desarrollado en este proyecto.

Por último, a finales del mes de agosto se nos propone añadir a los terceros un campo modelo de factura y otro de descuento de pronto pago. Llego a incluir estos campos en el mantenimiento de terceros, pero no se llegan a implementar por la finalización de mi contrato de prácticas.

7. Pruebas

En este apartado se van a explicar las pruebas que se han ido haciendo del proyecto. Al tratarse de un desarrollo en espiral, las pruebas son muy importantes pues nos ayudan a continuar el proyecto o a reconducirlo si es necesario.

Las pruebas se han hecho en un servidor instalado en la empresa con una copia de la base de datos de producción, pero manteniendo los datos al día importándolos de la base de datos antigua mediante consultas SQL.

En este proyecto hay una prueba por cada iteración:

1. Prueba primera iteración (enero 2021): En esta prueba se realiza por primera vez la instalación del programa y se crean las bases de datos que se irán usando en futuras pruebas. Los usuarios son capaces de iniciar sesión con un usuario de prueba y pueden empezar a navegar por el programa. Se prueba a crear varios vendedores y fabricantes en sus mantenimientos y estos se crean de forma satisfactoria. En el mantenimiento de montadores ocurre que un campo no se estaba guardando, pero ese es el único fallo hasta el momento. El mantenimiento de pedidos también se pone a prueba y la creación del pedido se hace bien. La prueba es un éxito y tras identificar el fallo en el mantenimiento de montadores se puede pasar a la siguiente iteración.
2. Prueba segunda iteración (marzo 2021): En esta prueba se examina la gestión del proceso de negocio de los pedidos en los punteos de pedidos y de líneas. El punteo de líneas da error al utilizar un filtro debido a que la confección de la consulta respecto al filtro da un fallo. Por suerte, utilizando pedidos que ya habían sido recibidos, se pueden probar todas las etapas de un pedido de forma exitosa. Se utiliza el punteo de albaranes para ver si estos se generan y también obtenemos un resultado positivo. A continuación, se prueba el cambio a terceros en el mantenimiento y se pueden crear, modificar y borrar, pero hace falta ajustar las dimensiones de las columnas del grid de los centros. La prueba es relativamente exitosa porque los fallos son sencillos de solucionar y en la siguiente prueba no ocurrirán.
3. Prueba tercera iteración (junio 2021): se vuelve a probar el punteo de líneas y esta vez se puede mostrar el proceso de recepción de mercancía. A continuación, se prueba otra vez el proceso de negocio de pedidos para generar albaranes y esta vez se facturan para probar la facturación. Las facturas se crean sin problemas en el punteo y se pueden gestionar desde el mantenimiento. Hay un fallo importante a la hora de imprimir los listados en el servidor de la empresa pues después de varios intentos nos damos cuenta de que la versión instalada de Crystal Reports del servidor no es la correcta para el servidor, así que, se reserva esta parte para la prueba siguiente.
4. Prueba iteración final (agosto 2021): En esta prueba se vuelve a hacer la instalación de Crystal Reports y esta vez al generar el directorio del programa se especifica que se guarden los listados. Se prueban todos los listados y esta vez van sin ningún problema. Entrando en el contenido específico de la última iteración, se prueba toda la gestión de incidencias y funciona de forma correcta. La ubicación de red creada para alojar las carpetas de las incidencias también responde correctamente. Además, se prueba otra vez el proceso entero desde la creación del pedido hasta contabilizar la factura y es un éxito. Por otra parte, se constata que los recibos han sido generados y que es posible cambiar

su situación y pagarlos parcialmente. La prueba ha sido un éxito y el equipo de la empresa puede empezar a trabajar con nuestro programa.

8. Conclusiones

Una vez finalizado el proyecto puedo estar satisfecho con los resultados obtenidos ya que el programa que he desarrollado ha cumplido las expectativas de los clientes. En varias reuniones el personal de la empresa me reconoció que el programa les iba a agilizar mucho el trabajo. Utilizar una metodología en espiral fue un acierto ya que el cliente sentía que sus opiniones eran tenidas muy en cuenta y, más importante aún, que se plasmaban en las siguientes pruebas. Por último, el objetivo de integrar el programa con los que ellos necesitasen se cumplió pues en la iteración en la que se implementa la contabilización se pudo ver en el programa de contabilidad reflejado de forma correcta. Además, pudimos integrar sus listados del programa Crystal Reports lo que ahorró mucho trabajo a la hora de no tener que volver a crearlos.

En cuanto al dominio de las tecnologías, ahora soy capaz defenderme manteniendo una base de datos SQL Server y entiendo el desarrollo de aplicaciones de escritorio en Windows Forms de .NET Framework. El dominio del lenguaje Visual Basic ha sido asequible pues ya había trabajado con C# y solo tuve que aprender las diferencias en la sintaxis. Este proyecto me ha servido también para entender una de las arquitecturas más populares de la actualidad como es MVC. Ahora me siento capaz de utilizarla en futuras aplicaciones.

Dejando de lado la tecnología, el desarrollo de esta solución me ha aportado muchos conocimientos de los campos de la contabilidad o la logística de los que no disponía. Además, me ha permitido desarrollar mi habilidad para hablar con los clientes, conocer sus necesidades y aconsejarles de la mejor forma posible teniendo en cuenta mi experiencia.

En definitiva, el proyecto ha sido un éxito tanto laboral por permitirme asentarme en la empresa para futuros proyectos como formativo pues he puesto en práctica los conocimientos que he ido adquiriendo a lo largo del grado.

a. Trabajos futuros

Después de acabar este proyecto se podría considerar añadir nuevas partes a la aplicación como una gestión de almacén más desarrollada con inventarios o automatizarla utilizando etiquetaje. Otra de los proyectos que se podrían abordar es integrar su contabilidad por completo y que no necesiten de otro programa adicional para manejarla como hasta ahora. Por último, si no fuera posible integrar el programa al completo podríamos introducir las facturas de compra en la aplicación para que los clientes no tengan que darlas de alta manualmente en su programa de contabilidad y así darles más protagonismo a los proveedores en el programa.

Por otra parte, me gustaría volver a desarrollar programas en C# porque se encuentra mucha más documentación y trabajaría en el mismo framework. Cuando he tenido que buscar soluciones a algún problema siempre encontraba la documentación en ese lenguaje y tenía que traducirlo al lenguaje utilizado. Además, Microsoft va a dar mucho más soporte a C# en el futuro y eso es muy importante ya que en Visual Basic no se podrán utilizar algunas nuevas herramientas o tecnologías que vayan surgiendo.

b. Relación del trabajo desarrollado con los estudios cursados

Este desarrollo me ha servido para poner en práctica muchos de los conocimientos aprendidos en el grado como veremos a continuación:

La asignatura ISW (Ingeniería del software) me ha ayudado mucho en este proyecto pues mucha de la materia que aprendí la he puesto en práctica. Entre los conocimientos puestos en práctica destaca la parte del diseño de la interfaz pues en el trabajo que se realiza en la asignatura ya utilicé Visual Studio para el diseño e implementación de formularios. Otra de las cosas en las que me ha ayudado la asignatura es a reconocer el modelo de datos y establecer los casos de uso del programa pues dedicamos toda la primera parte teórica de ella a aprender a hacerlo. Por último, la adaptación a la metodología en espiral fue sencilla ya que la conocía por el temario de la asignatura.

Las asignaturas BDA (Bases de datos y sistemas de información) y TBD (Tecnologías de bases de datos) me dieron las herramientas necesarias para trabajar con bases de datos y entender cómo funcionan. Cuando llegué a la empresa y empecé el proyecto solo tuve que aprender las diferencias entre las bases de datos de Oracle y las de SQL Server pues ya conocía la sintaxis SQL y hacer operaciones básicas como crear y modificar tablas.

Por último, las asignaturas IIP (Introducción a la Informática y a la Programación), PRG (Programación) y EDA (Estructuras de Datos y Algoritmos) me han proporcionado los conocimientos base en programación sin los que este proyecto no hubiera sido posible.

9. Bibliografía

Aguilar, J. M. (15 de Octubre de 2019). *¿Qué es el patrón MVC en programación y por qué es útil?* Obtenido de Campus MVP: https://www.campusmvp.es/recursos/post/que-es-el-patron-mvc-en-programacion-y-por-que-es-util.aspx

Contributors, M. (27 de Agosto de 2021). *MVC*. Obtenido de MDN Web Docs: https://developer.mozilla.org/es/docs/Glossary/MVC

DataGridView.CellDoubleClick Event. (s.f.). Obtenido de Microsoft Docs: https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview.celldoubleclick?view=net-5.0

ESAA. (19 de Julio de 2020). *Historia del ERP: pasado, presente y futuro*. Obtenido de Velneo: https://velneo.es/historia-de-erp-pasado-presente-y-futuro/

How to add the Button and Text in GridTextColumn in WinForms DataGrid (SfDataGrid)? (20 de Júlio de 2020). Obtenido de Syncfusion: https://www.syncfusion.com/kb/11754/how-to-add-the-button-and-text-in-gridtextcolumn-in-winforms-datagrid-sfdatagrid

SAP. (s.f.). *¿Qué es ERP?* Obtenido de SAP: https://www.sap.com/spain/insights/what-is-erp.html

SoftwarePara. (s.f.). *60 años de Historia y Evolución de los Sistemas ERP*. Obtenido de SoftwarePara: https://softwarepara.net/erp-historia/

10. Anexos

a. Diseño de la base de datos

Estas son todas las tablas de la base de datos SQL Server que pertenecen al modelo de datos del programa.

Pedidos	
Columna	Tipo SQL Server
año {Clave Primaria}	int
id {Clave Primaria}	int
vendedorId	int
centroId	int
dptoId	int
refPedido	nvarchar(50)
nombreCli	nvarchar(100)
dir1Cli	nvarchar(150)
dir2Cli	nvarchar(150)
provinciaId	int
localidad	nvarchar(100)
cp	nvarchar(10)
tel1Cli	nvarchar(10)
tel2Cli	nvarchar(10)
tipoFacturacion	nvarchar(5)
facturadoA	int
fechaCreación	datetime
fechaPrevRecepcion	datetime
fechaAlbaran	datetime
fechaPrevMontaje	datetime
fechaMontaje	datetime
albaranTienda	nvarchar(25)
bultos	int
tAprox	int
contactado	char(1)
horario	nvarchar(5)
coordinación	char(1)
coordinador	nvarchar(50)
situación	nvarchar(20)
nota	nText
montador	int
añoFactura	int
factura	int
añoFacturaMontador	int
facturaMontador	int
historial	ntext

PedidosLineas	
Columna	Tipo SQL Server
año {Clave Primaria}	int
pedidoId {Clave Primaria}	int
línea {Clave Primaria}	int
tipo	int
proveedorId	int
dpto	nvarchar(10)
dptoCargo	nvarchar(10)
articuloId	nvarchar(5)
descripción	nvarchar(100)
recibido	char(1)
sinRecepcion	char(1)
albaran	nvarchar(75)
fechaEntrada	datetime
bultos	int
tAprox	int
fabricanteId	int
cantidad	int
importe	float
pFactura	float
iFactura	float
pVendedor	float
iVendedor	float
pMontador	float
iMontador	float

PedidosLlamadas	
Columna	Tipo SQL Server
año {Clave Primaria}	int
pedidoId {Clave Primaria}	int
llamada {Clave Primaria}	int
fecha	datetime
usuario	nvarchar(10)
tel	nvarchar(10)
destinatarioId	int
nombre	nvarchar(50)
nota	ntext

PedidosIncidencias	
Columna	Tipo SQL Server
año {Clave Primaria}	int
pedidoId {Clave Primaria}	int
incidencia {Clave Primaria}	int
fabricanteId	int
texto	ntext
fecha	datetime
estado	nvarchar(10)
añoPedidoNuevo	int

pedidoNuevoId	int

Almacenes

Columna	Tipo SQL Server
id {Clave Primaria}	int
descripción	nvarchar(50)
tipo	int
deposito	char(1)

Ubicaciones

Columna	Tipo SQL Server
almacenId {Clave Primaria}	int
ubicación {Clave Primaria}	nvarchar(5)
nombre	nvarchar(50)

Tipos de almacen

Columna	Tipo SQL Server
id {Clave Primaria}	int
descripcion	nvarchar(50)

RegistrosUbicacion

Columna	Tipo SQL Server
almacenId {Clave Primaria}	int
ubicación {Clave Primaria}	nvarchar(5)
id {Clave Primaria}	int
articulo	nvarchar(5)
refPedido	nvarchar(10)
fechaEntrada	datetime
añoPedido	int
pedidoId	int
lineaPedido	int

Provincias

Columna	Tipo SQL Server
id {Clave Primaria}	int
nombre	nvarchar(50)

Localidades

Columna	Tipo SQL Server
provinciaId {Clave Primaria}	int
nombre {Clave Primaria}	nvarchar(100)

Ejercicios	
Columna	Tipo SQL Server
año {Clave Primaria}	int
fechaInicio	datetime
fechaFin	datetime

TiposIva	
Columna	Tipo SQL Server
ejercicio {Clave Primaria}	int
id {Clave Primaria}	int
iva	float
descripción	nvarchar(50)
cuentaContable	nvarchar(50)
recargo	float
cuantaContableRecargo	nvarchar(50)
deducible	char(1)
predeterminado	char(1)

Terceros	
Columna	Tipo SQL Server
id {Clave Primaria}	int
razon	nvarchar(50)
nombre	nvarchar(50)
provinciaId	int
localidad	nvarchar(100)
cp	nvarchar(10)
dir1	nvarchar(150)
dir2	nvarchar(150)
dir3	nvarchar(150)
tel1	nvarchar(10)
tel2	nvarchar(10)
fax	nvarchar(25)
email	nvarchar(50)
web	nvarchar(100)
contacto	nvarchar(50)
cif	nvarchar(10)
fpagoId	int
nota	ntext
tipoFacturacion	nvarchar(5)
tipoIva	int
ctaContable	nvarchar(50)
esMontador	char(1)
esProveedor	char(1)
esFabricante	char(1)
esVendedor	char(1)

Artículos	
Columna	Tipo SQL Server
código {Clave Primaria}	nvarchar(5)
descripción	nvarchar(50)
proveedor {Clave Primaria}	int
pFactura	float
iFactura	float
pVendedor	float
iVendedor	float
pMontador	float
iMontador	float

TercerosCuentas	
Columna	Tipo SQL Server
terceroId {Clave Primaria}	int
cuenta {Clave Primaria}	int
banco	nvarchar(50)
iban	nvarchar(5)
banco	nvarchar(5)
oficina	nvarchar(5)
dControl	nvarchar(5)
dCuenta	nvarchar(15)
swift	nvarchar(20)
predeterminada	char(1)
mandato	nvarchar(20)
firmado	char(1)
fechaFirma	datetime

TercerosCentros	
Columna	Tipo SQL Server
terceroId {Clave Primaria}	int
id {Clave Primaria}	int
descripción	nvarchar(50)
provinciaId	int
localidad	nvarchar(100)
cp	nvarchar(10)
dir1	nvarchar(150)
dir2	nvarchar(150)
tel1	nvarchar(10)
tel2	nvarchar(10)
contacto	nvarchar(25)
email	nvarchar(50)
nota	ntext

TercerosCentrosDptos	
Columna	Tipo SQL Server
terceroId {Clave Primaria}	int
centroId {Clave Primaria}	int
id {Clave Primaria}	int
descripción	nvarchar(50)
contacto	nvarchar(25)
tel	nvarchar(10)
email	nvarchar(50)

FormasPago	
Columna	Tipo SQL Server
id {Clave Primaria}	int
descripción	nvarchar(50)
nRecibos	int
diasPrimerVenc	int
cadencia	int
tipoRevibos	int
situacionRecibo	nvarchar(5)
abreviado	nvarchar(10)

TiposFacturacion	
Columna	Tipo SQL Server
código {Clave Primaria}	int
descripcion	nvarchar(50)

Albaranes	
Columna	Tipo SQL Server
año {Clave Primaria}	int
serie {Clave Primaria}	int
id {Clave Primaria}	int
fecha	datetime
descripcion	nvarchar(50)
facturadoA	int
centroId	int
dptoId	int
iBruto	float
dto	float
iBase	float
tipoIvaId	int
iIva	float
iNeto	float
estado	nvarchar(10)

AlbaranLineas	
Columna	Tipo SQL Server
año {Clave Primaria}	int
serie {Clave Primaria}	int
albaranId {Clave Primaria}	int
línea {Clave Primaria}	int
articulo	nvarchar(5)
descripción	nvarchar(50)
cantidad	int
importe	float
pFactura	float
iFactura	float
añoPedido	int
pedidoId	int

Facturas	
Columna	Tipo SQL Server
año {Clave Primaria}	int
serie {Clave Primaria}	int
id {Clave Primaria}	int
fecha	datetime
tipoIvaId	int
iBruto	float
dto	float
iBase	float
iIva	float
iRecargo	float
iNeto	float
facturadoA	int
centroId	int
dptoId	int
formaPagoId	int
multiple	char(1)
agrupada	char(1)
contabilizada	char(1)

FacturasLineas	
Columna	Tipo SQL Server
año {Clave Primaria}	int
serie {Clave Primaria}	int
facturaId {Clave Primaria}	int
línea	int
articulo	nvarchar(5)
descripción	nvarchar(50)
cantidad	int
importe	float
pFactura	float
iFactura	float
añoAlbaran	int
serieAlbaran	int

albaranId	int
añoPedido	int
pedidoId	int

Recibos	
Columna	Tipo SQL Server
año {Clave Primaria}	int
id {Clave Primaria}	int
añoFactura	int
serieFactura	int
facturaId	int
fechaFactura	datetime
fechaVencimiento	datetime
importe	float
tipoReciboId	int
situacion	nvarchar(5)
facturadoA	int
banco	nvarchar(5)
entidad	nvarchar(5)
dControl	nvarchar(5)
dCuenta	nvarchar(20)
cuentaPropia	nvarchar(50)

TiposRecibos	
Columna	Tipo SQL Server
id {Clave Primaria}	int
descripción	nvarchar(50)
abreviado	nvarchar(10)

b. Glosario

En este apartado se han incluido breves descripciones a palabras de fuera del dominio común:

Término	**Descripción**
Blockchain	Es una tecnología que divide un registro común en una red distribuida de nodos que facilita el registro de transacciones.
Clave ajena	Es el conjunto de atributos de una tabla que hacen referencia a un registro de otra tabla.
Dataset	Es una colección de datos que sigue un modelo común.
Framework	Es un entorno de programación con código ya desarrollado que proporciona funcionalidades para el desarrollo de software.
Grid	Un grid es una cuadrícula en la que todas sus filas comparten una estructura común
ORM	Es una tecnología que permite mapear estructuras presentes en una base de datos