ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERIA DEL SOFTWARE

CONFIGURADOR GRÁFICO DE FORMATOS PARA AMBAR7

CONFIGURATIONS FORMATS FOR APPLICATION AMBAR7

Realizado por
D. Jose Manuel Cordero Irles
Tutorizado por
D. Antonio J. Nebro
Departamento
LENGUAJE Y CIENCIAS DE LA COMPUTACIÓN

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Enero 2016

Fecha defensa:
El Secretario del Tribunal

Resumen

El presente trabajo fin de grado consiste en el desarrollo de una aplicación para la creación y manipulación de los formatos gráficos del programa de facturación Ambar7. A diferencia del configurador de formatos del que dispone la aplicación actualmente, este configurador realiza esta tarea dibujando y moviendo los elementos del formato directamente sobre la pantalla, con la forma de trabajar WYSIWYG-What You See Is What You Get. De tal forma usando el ratón se podrá realizar la casi totalidad del formato. El objetivo es incrementar la productividad de esta tarea, que actualmente lleva mucho tiempo incluso para simples cambios en un formato concreto y facilitar el uso a todo tipo de usuarios haciendo que este proceso sea intuitivo, ya que la forma actual es demasiado compleja y laboriosa.

Para la realización de esta aplicación se ha usado el compilador Visual Basic versión 6 de Microsoft, así como los motores de bases de datos SQL Server 2005 Express edition y MySql.

Palabras clave: Ambar7 – formatos – gráficos – dibujar – productividad – visual – basic – sqlserver - mysql

Abstract

This final degree project consists on the developing of an application for the creation and manipulation of graphics formats for the application Ambar7. In contrast to the current format configurator, the new one accomplishes its task drawing and moving the format´s elements directly on the screen, using the foundations of WYSIWYG-What You See Is What You Get. In this way, the user can create almost the total format just using the mouse. The main goal is to increase the productivity of this task, because nowadays it is very time consuming even for simple item changes in a specific format. In this way, the application will be accessible to all kind of users, creating a “use friendly” application, due that the present application is very complex and arduous.

For the creation of this application the compiler Visual Basic version 6 from Microsoft has been used, as well as the database engine SQL Server 2005 Express edition and MySQL.

Keywords: Ambar7 – formats – graphics – paint – productivity – visual – basic – sqlserver - mysql

Índice

1.- Introducción

El objetivo del presente trabajo fin de grado es la creación de una aplicación para la realización de manera eficaz, sencilla, rápida y sobre todo fácil e intuitiva, de los formatos de impresión de los documentos de la aplicación de facturación Ambar7.

Con esta herramienta se pretende conseguir una reducción drástica en los costes (en tiempo, y por tanto dinero) en la creación y manipulación de los formatos de la aplicación, que actualmente requieren de una gran cantidad de tiempo para realizar simples modificaciones, lo que de otra forma se haría de forma mucho más rápida. En el caso de realizar un formato de cero el tiempo necesario para ello se vuelve excesivamente elevado, provocando que casi nunca se haga de cero y siempre partamos de la modificación de un formato ya creado previamente.

Además, al hacer esta herramienta más fácil e intuitiva, lograremos que el usuario final pueda realizar estas tareas sin necesidad de recurrir a la empresa que le provee el software. La relación contractual de los clientes con la empresa suministradora del software se establece por un contrato de mantenimiento, que conlleva poder recurrir a nosotros sobre todo para consultas sobre la aplicación, pero también para ayuda en ciertos temas. Los clientes suelen solicitar ayuda para la modificación de sus formatos con mucha frecuencia. Aunque el mantenimiento sólo se incluye ayuda verbal para explicar cómo funciona la herramienta, finalmente siempre suele ser algún trabajador de la empresa de software el que realiza la tarea de configurar estos formatos, sin un coste adicional para el cliente final, lo que conlleva una inversión no remunerada de tiempo y dinero. El cliente también malgasta tiempo, pues tras intentar hacerlo él mismo, la mayoría de las veces acaba recurriendo a su proveedor debido a la complejidad de esta herramienta. Muchos de los clientes ni lo intentan y solicitan directamente nuestra ayuda.

Además, la actual herramienta del software para realizar esta tarea, al ser tan compleja de usar, provoca una cierta mala imagen en el cliente. El cliente se queda con la sensación de aridez de este software, cosa que no incrementa la

buena imagen de la empresa. Conseguir que la aplicación para este tema sea más fácil e intuitiva también conlleva una mejora de la imagen de software y de la empresa.

Para el desarrollo de esta herramienta gráfica se utilizará el lenguaje Visual Basic para el que se usará el entorno de desarrollo Microsoft Visual Basic 6.0 Service Pack 6. La elección de esta herramienta, ya antigua, es obligada debido a que la herramienta en que se basa el software actual está realizada en este compilador y tenemos que mantener la compatibilidad con el resto de la aplicación.

Las bases de datos sobre las que se puede trabajar para el almacenamiento y uso de los datos pueden ser tanto Microsoft SQL Server Express Edition en cualquiera de las versiones que comprenden desde la versión 2000, hasta la versión 2012, o bien la base de datos MySql. Nosotros usaremos Microsoft SQL Server 2005 Express Edition.

2.- Descripción del programa Ambar7

Ambar7 es un programa de gestión de empresas para PYMES, que contempla los aspectos principales en la gestión de una empresa. Las principales áreas que integra el programa son las ventas y facturación, stock y control de almacén, compras, gestión de cobros, gestión de tesorería...

Figura 1: Pantalla principal del programa Ambar7

Una de las características más interesantes de esta aplicación es la posibilidad de configurar, por parte del usuario, la forma en que se imprimen los documentos de venta de la empresa (facturas, albaranes de entrega, pedidos y presupuestos), es decir la estética de las facturas, albaranes y pedidos de venta (y de compra) que se imprimirán, adaptándolas totalmente a las necesidades de cada usuario de la aplicación, y permitiendo distintos formatos de cada documento según las necesidades de la empresa.

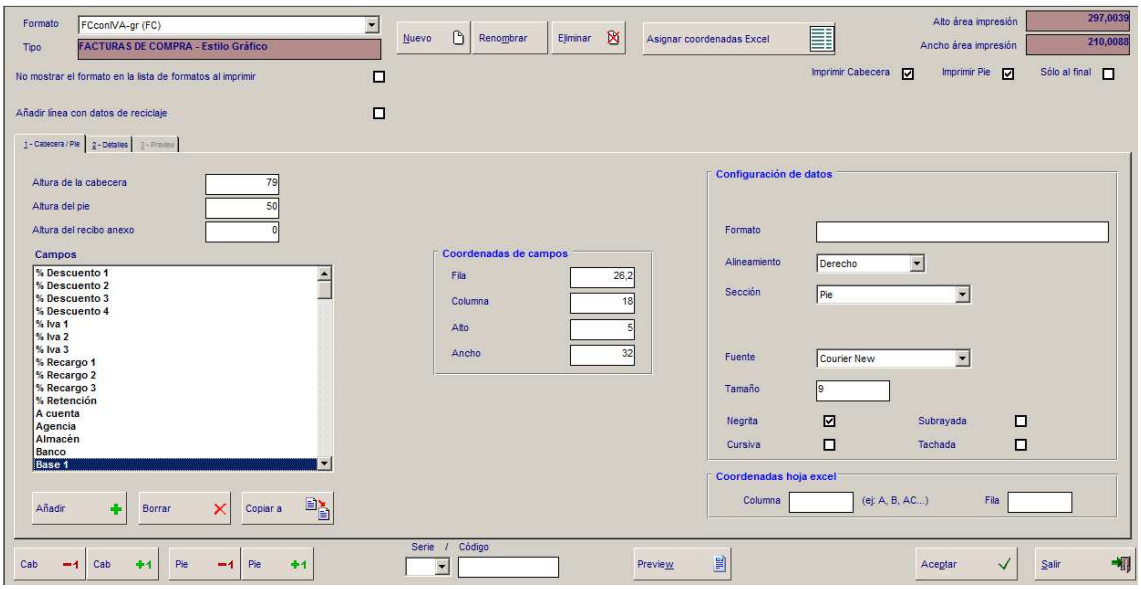
La forma en que se imprimen las facturas de la empresa, que solemos llamar “formatos” es importante para el usuario de la aplicación, pues define en cierta medida la identidad de la empresa de cara a sus clientes.

Además, con los constantes cambios que se producen relativos a temas de facturación, como los recientes cambios en el Impuesto sobre el valor añadido (I.V.A.), o el régimen especial de criterio de caja, provocan que los formatos que se tenían definidos dejen de cumplir con las normativas que había establecidas hasta hace poco tiempo. El usuario necesita cambiar sus formatos para adaptarse a estas normativas que cambian constantemente para poder reflejar estas variaciones en el papel. Esto hace prácticamente imprescindible pode disponer de la capacidad en el software de facturación de poder alterar cuando sea necesario, y de forma rápida y eficaz, estos cambios que impone la

administración, que son datos tan relevantes como por ejemplo el nuevo I.V.A a aplicar en los documentos o informar de los nuevos cambios fiscales que afectan a las facturas que imprimimos.

Para poder realizar estos cambios en los formatos, la aplicación Ambar7 dispone de una herramienta que permite configurar dichos formatos. A continuación mostramos la herramienta para configurar los formatos en el sistema actual.

Figura 2: Pantalla principal del configurador actual de formatos



Este configurador, si bien cumple con esta funcionalidad, por lo que explicaremos más adelante, resulta demasiado compleja para un usuario medio de la aplicación. Además, aunque se pueden hacer todos los cambios que sean necesarios por parte del usuario, debido a la forma en que permite realizar dichos trabajos, llevarlos a cabo se vuelve una tarea lenta y un tanto tediosa.

3.- Objetivos

El objetivo del presente software es por tanto poder realizar de forma más rápida y mucho más sencilla esta tarea de configuración de formatos. Con ello pretendemos acercar esta herramienta al usuario final para que pueda hacerlo directamente él mismo sin necesidad de recurrir a la empresa de software,

reduciendo costes y tiempo tanto para el cliente como para la empresa de software.

Para entender lo que se desea realizar, primero describiremos cómo es el funcionamiento actual de la aplicación en relación a esta característica de configurar formatos.

4.- El configurador de documentos actual de la aplicación

El funcionamiento del configurador, se basa en la idea de definir los objetos que queremos que aparezcan en el formato, y dibujarlos. Para ello agrupamos estos objetos en diferentes tipos. Todos los objetos tienen en común que se les pueden establecer las coordenadas, expresadas en milímetros, donde se desea que se impriman en la hoja de papel.

Los campos que se van a imprimir en su mayoría corresponden a la factura que se emite y que la empresa que adquiere el software le entrega a sus clientes. La aplicación permite diseñar formatos para facturas, albaranes de entrega de mercancía y pedidos de venta, además de sus equivalentes para las compras. También se pueden crear formatos para presupuestos y recibos de facturas.

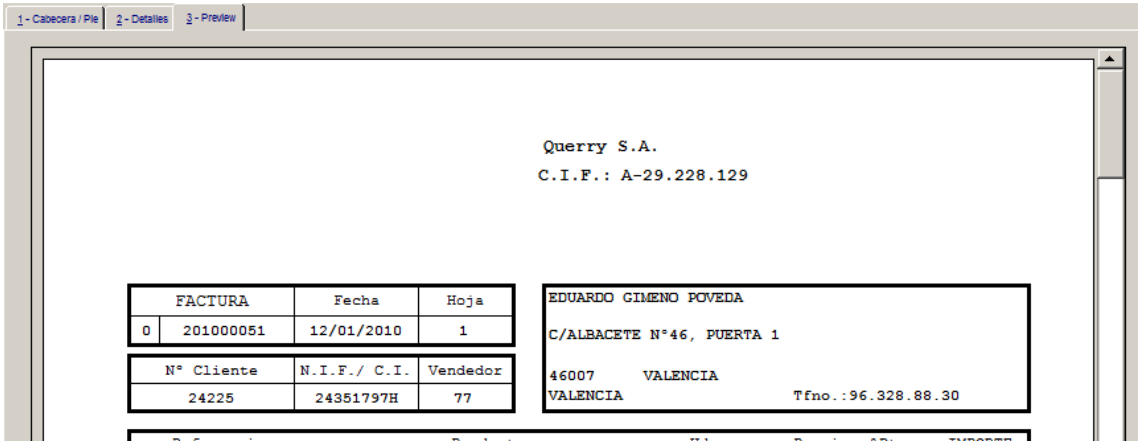
Nos centraremos en explicar los formatos basados en documentos de venta en general (documento para abreviar). Todos los formatos son casi idénticos a la hora de crearlos e imprimirlos. Lo único que cambia son ciertas peculiaridades de cada uno de ellos. Por ejemplo una factura de venta es un documento oficial que debe llevar la identificación fiscal. Un albarán de entrega o un pedido no. En los pedidos de venta puede aparecer la fecha prevista de entrega, y en la factura no, etc. Salvo por estos pequeños detalles, como decimos, la información es casi la misma y sólo hay diferencias entre ellos a la hora de la información que se puede imprimir o no.

La estructura de un documento, y por tanto del formato que se puede configurar, consta de una cabecera, un pie del documento, y los detalles del mismo.

En la cabecera suele aparecer la información correspondiente a la empresa que emite el documento, incluyendo en el caso de ser una factura la

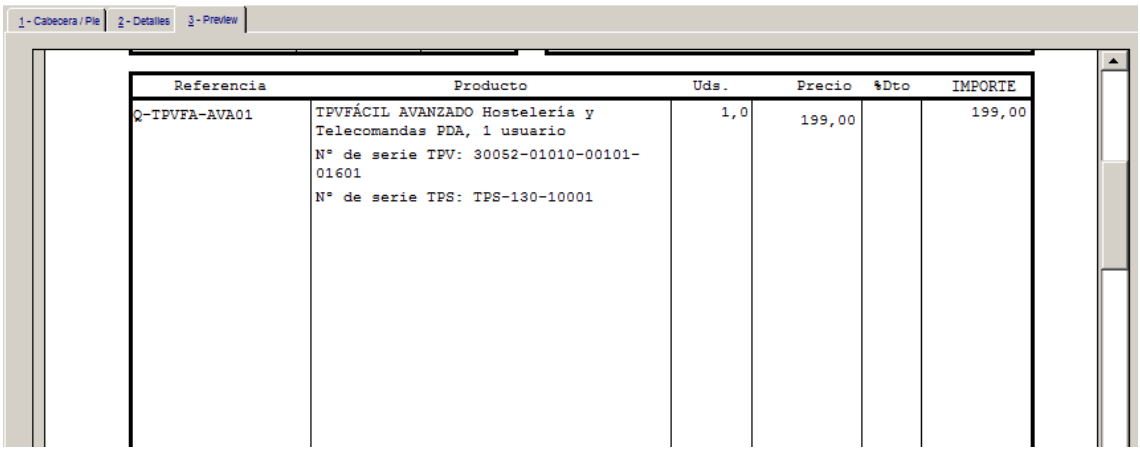
información fiscal. También suele llevar la información del cliente al que se le realiza la factura, y los datos propios del documento más generales, como el número de documento, la fecha de emisión, el vendedor que ha realizado la venta, así como el nombre del mismo, etc.

Figura 3: Ejemplo de vista previa de la cabecera de una factura



A continuación viene la sección de los detalles del documento, en la que se detallan, en forma de lista, los diferentes productos que el cliente ha adquirido en la empresa. A estos productos los solemos llamar artículos, y se suele imprimir en el papel el código que tiene el mismo en la aplicación, la descripción, la cantidad de productos que se lleva el cliente, el precio y posibles descuentos, y el total (cantidad por precio menos descuento de la línea).

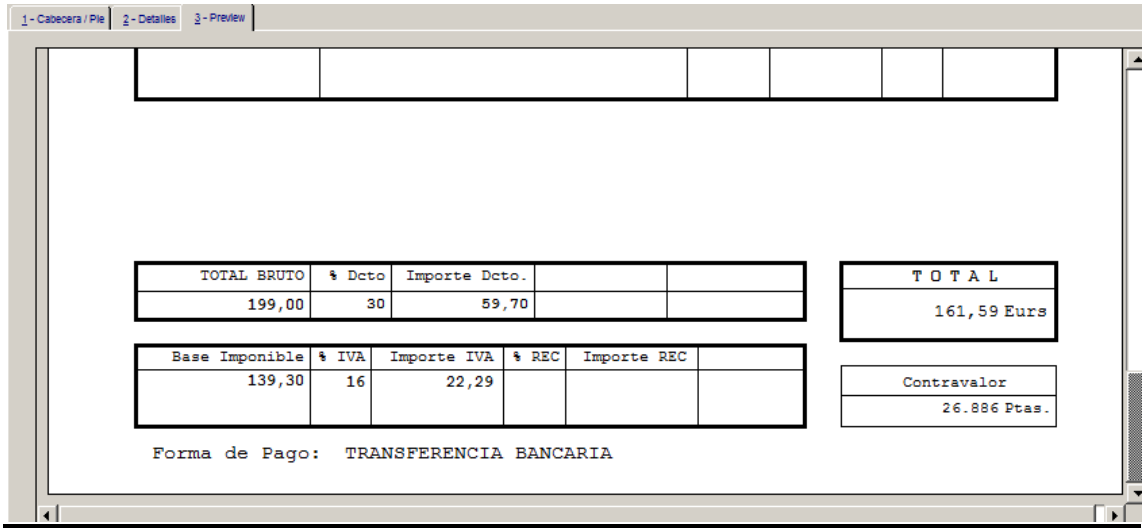
Figura 4. Ejemplo de los detalles de una factura



La última sección es el pie del documento. En este apartado viene la

información del total de la factura, desglosada en las diferentes bases imponibles e impuestos que tenga el documento realizado. La aplicación Ambar7 permite aplicar descuentos al total del documento, que también se suelen detallar en esta sección. Es habitual que los clientes impriman en el pie información de la empresa, como por ejemplo el típico texto de si se admite la devolución de mercancía y hasta que tiempo etc. Como explicaremos después este tipo de campos en el formato son textos de observaciones que el usuario puede configurar tantos como desee y en cualquiera de las secciones.

Figura 5: Ejemplo del pie de una factura



5.- Objetos que se pueden configurar para imprimir en el formato

A continuación explicaremos los diferentes objetos que se pueden configurar e imprimir. Como se puede ir apreciando, dichos objetos se corresponderán con la información que lleva el documento de venta, así como otros adicionales necesarios para el formato. Por tanto tenderemos objetos de la propia base de datos, y objetos gráficos que sirven para definir la estética del formato.

Datos de la empresa emisora de la factura

Son los datos de la propia empresa que emite los documentos de venta o compra con la aplicación. Estos datos pueden ser por ejemplo el nombre

comercial de la empresa, el nombre fiscal de la empresa, dirección de la empresa, identificación fiscal, etc. No se almacenen en la propia tabla de la base de datos de la factura, pues suelen ser datos comunes a toda la aplicación. Se suelen imprimir en la sección de cabecera del documento.

Datos del cliente

Son los datos identificativos del cliente al que se le realiza la venta (o del proveedor al que se hace la compra de mercancía en caso de una factura de compra, por ejemplo). Son datos similares a los de la empresa, es decir, el nombre fiscal o comercial, la dirección, población, provincia, etc. Estos datos sí se encuentran almacenados en la tabla donde se guardan los datos propios de la factura. Estos datos también se suelen imprimir en la sección de cabecera del documento.

Datos del documento de venta

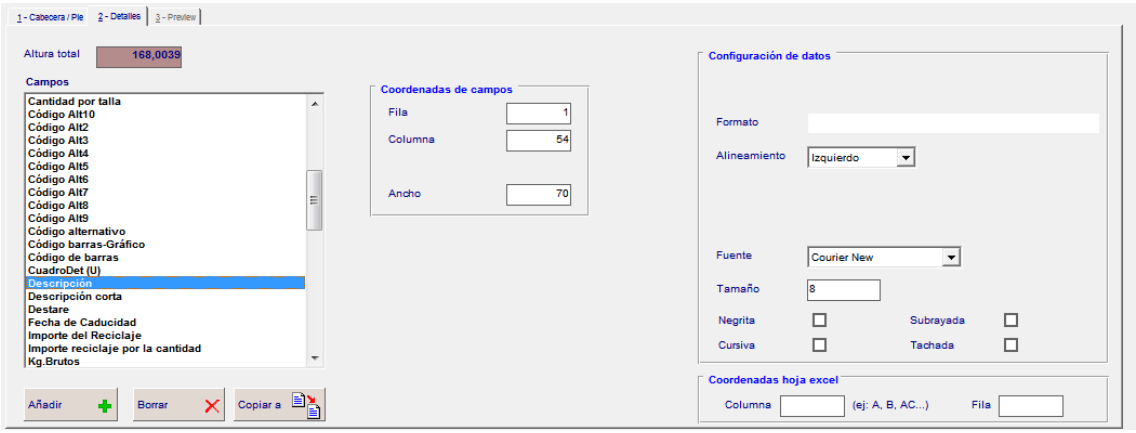
Estos datos son los más significativos del propio documento de venta, pues se trata de los datos relativos a los importes totales de los productos que se están vendiendo o comprando. Estos pueden ser por ejemplo la fecha del documento, numeración, nombre del vendedor, importes totales y sus desgloses en las diferentes bases imponibles, sus impuestos y cuotas respectivas, posibles descuentos aplicados, etc. Estos datos suelen imprimirse en la sección de cabecera y en el pie.

Entre los datos del documento se encuentran algunos de ellos que tienen la peculiaridad de no se trata de datos que están en si almacenados en la base de datos, sino que son cálculos generados en base a varios de estos datos que sí están almacenados, y que son necesarios poder permitir el imprimirlos. Por ejemplo el campo ‘suma de líneas’, establece el importe total de los artículos que se venden en la factura en cuestión. Evidentemente este valor es la suma de las líneas que contenga la factura, que es distinto para cada factura. A la hora de imprimir un formato, se calculan in situ estos valores. Por lo demás es como cualquier otro dato que estuviese almacenado en la base de datos y perteneciese al documento, el cliente, etc.

Detalles del documento

Son los datos de los productos que se venden o compran, como puede ser el código y la descripción del producto de venta o compra, su precio unitario, cantidad que se sirve, posible descuento al artículo, I.V.A. aplicado, total de ese artículo servido (cantidad por precio menos descuento), etc. Estos datos se imprimen en la sección de detalles del documento.

Figura 6: Sección de detalles del documento



Objetos de texto u objetos fijos

Son campos para imprimir un texto fijo en cualquiera de las secciones del documento, como por ejemplo los títulos o descripciones de los datos variables, como puede ser por ejemplo las palabras “Cliente” o “Proveedor”, “Artículo”, “Precio”, “Fecha”, “Número de factura”, etc.

Cualquier dato de texto fijo deseado se puede configurar para poder imprimirse en la sección de cabecera o pie, o en la de de detalle. Si se configura en la sección de cabecera/pie, se puede cambiar la propiedad sección cuando se desee y pasarlo de una sección a otra.

Objetos gráficos

Son objetos que se pueden dibujar, de tipo gráfico, como líneas rectas, de

orientación horizontal o vertical, recuadros y logotipos o imágenes, como marcas de agua. Se suelen imprimir en cualquier sección del documento, algo evidente por otro lado pues es lo que más define la estética del formato.

Como se puede ir viendo, el formato consta de tres secciones donde distribuir la información. Algunos datos, por la lógica intrínseca a lo que es un documento de venta o compra, se podrán imprimir en cualquiera de las secciones, pero otros sólo tendrán sentido en una u otra sección.

Las secciones de que consta un formato son la cabecera, el pie del documento y los detalles. Cada sección tiene sus dimensiones, pero todo se basa en primera instancia en la dimensión del formato en sí. Habitualmente se usa papel de tamaño A4 (210 mm x 280 mm) aunque se puede definir cualquier dimensión de papel soportada por cualquier impresora, lo que en realidad se traduce en que se puede definir prácticamente cualquier dimensión que se pueda desear.

Una vez definida la dimensión del papel con el que vamos a trabajar, hay que especificar el alto de la sección de cabecera, y el de la sección del pie. La dimensión del detalle será la diferencia entre el alto del papel y la suma de estados secciones:

[Alto sección detalle] = [Alto del papel] – ([Alto sección cabecera] + [Alto sección pie]).

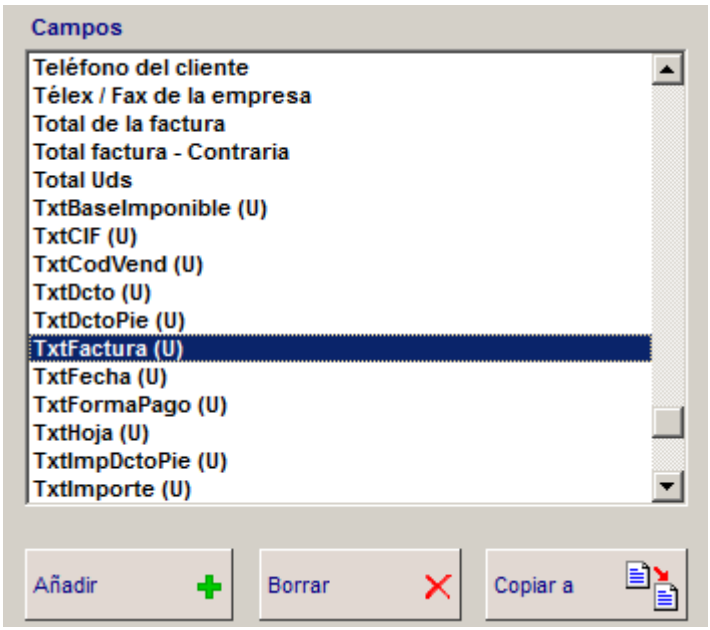
Cada objeto a la hora de definirlo tiene sus propiedades, algunas comunes a todos, y otras propias de cada tipo de objeto. Se explican a continuación.

Nombre del objeto

Es el nombre identificativo del dato a imprimir. Se establece al crear el objeto, y aparece en el programa en la sección “Campos”. En lo relativo a los

objetos basados en datos del documento en sí, es decir los datos de la empresa emisora de la factura, datos del cliente, y datos del documento de venta, son nombres descriptivos fijos y ya están creados por la aplicación, y el usuario no los puede alterar. Los objetos gráficos y los textos fijos, que son definibles por el usuario, y se les puede asignar el nombre que se desee en el momento de crearlos, y una vez creados no se podrán alterar tampoco.

Figura 7: lista de campos del formato



En el apartado de “Configuración de datos” se definen las siguientes propiedades.

Fórmula

En este apartado se establece en el caso de los textos fijos, el valor que se desea imprimir. El resto de objetos no dispone de esta propiedad, pues el dato a imprimir ya se obtiene de la base de datos, o bien no tiene sentido en caso de por ejemplo tratarse de un cuadro o recta.

Formato

Esta propiedad da la posibilidad de poder aplicar un determinado formato al texto a imprimir. Por ejemplo en caso de datos numéricos, se pueden

establecer puntos de miles, comas para decimales, etc. Para fecha, el formato de fecha corta o larga, etc. Este dato en ciertos casos a nivel interno se obvia, pues determinados datos deben imprimirse de forma pre-establecida, sin hacer caso a lo que pueda aparecer aquí. Por ejemplo, el total de una factura tiene que tener un máximo de dos decimales siempre.

Alineamiento

Se define para los campos que son textos imprimibles, ya sean textos fijos o datos leídos de la base de datos. Establece la alineación del mismo (izquierda, centrado o derecha). El dato se alinea en función del ancho definido para imprimir el dato.

Sección

Es la sección donde queremos imprimir el dato. Puede ser la cabecera, el pie o los detalles, como se explica anteriormente. No obstante sólo se pueden seleccionar los valores de cabecera o pie. Esto se debe a que la peculiaridad de los formatos, al tener una sección de detalles, implica que ciertos datos sólo pueden imprimirse en la sección de detalles (por ejemplo los detalles de una factura, como son los artículos que se venden). Por ello, en la definición de los formatos, los campos que se configuran en esta sección de detalles se establecen en su propio apartado de detalles, separado de la sección para los campos de cabecera o pie.

Fuente

Es el tipo de fuente para los campos de tipo texto. Tomará cualquiera de las instaladas en el ordenador donde se ejecute la aplicación.

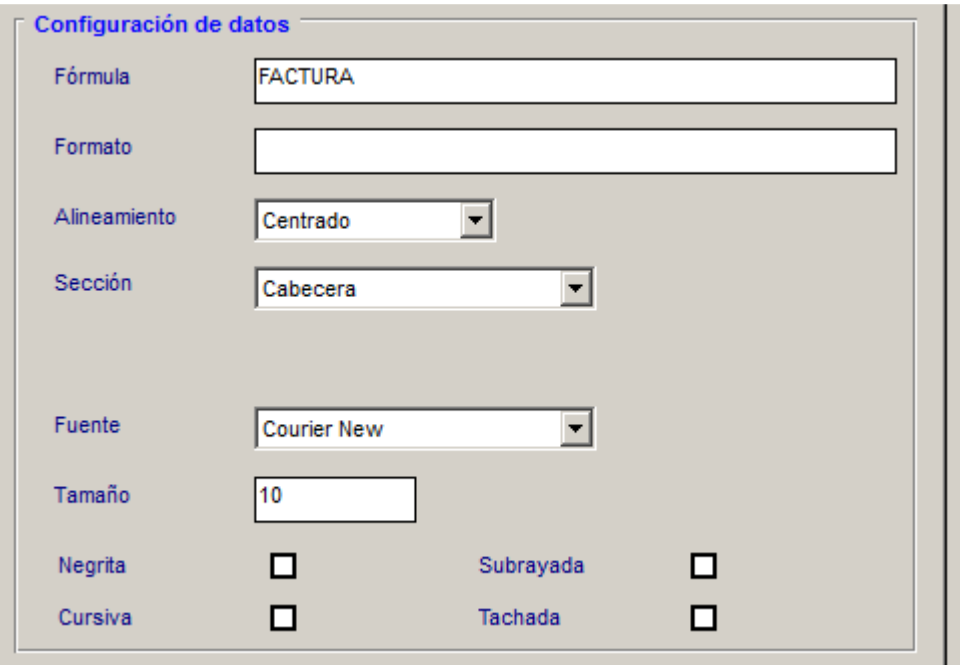
Tamaño

Es el tamaño de letra de los campos de tipo texto.

Negrita, Subrayada, Cursiva, Tachada.

Para aplicar a los campos de tipo texto.

Figura 8: detalle de propiedades de los campos



Coordenadas de campos

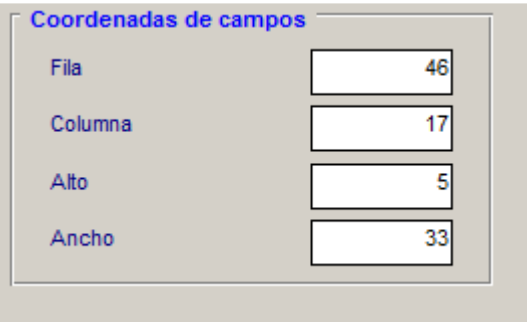
Aquí se definen las coordenadas para imprimir los datos. Se definen en milímetros, y varían según el tipo de dato que se desea imprimir. Estas coordenadas se basan en un eje (x,y) partiendo de la posición (0,0) situada en la parte superior izquierda del documento.

Para el caso de los campos que se imprimen de tipo texto, son:

Fila, columna: coordenada superior izquierda donde se empieza a imprimir el dato.

Alto, Ancho: las dimensiones de la región donde se va a imprimir el dato.

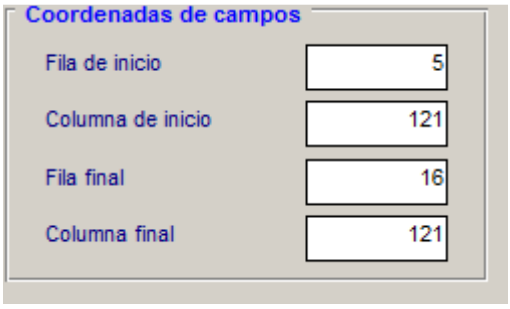
Figura 9: coordenadas de campos



Estas coordenadas de campos son importantes para el sistema de dibujo que se desarrolla, pues nos definen las dimensiones de los objetos a la hora de imprimirlos en papel, pero también a la hora de dibujarlos y arrastrarlos a la pantalla cuando expliquemos el sistema de diseño gráfico desarrollado. Es importante hacer notar que estas coordenadas establecen una subregión para cada campo, teniendo en cuenta que si a la hora de imprimir un dato, estas coordenadas establecen una subregión no lo suficiente grande para imprimir el dato que se desea que aparezca en el papel, el dato en unos casos imprimirá lo que quepa en dicha región, pero en otros casos, por motivos estéticos, no imprimirá nada al no caber en su totalidad. Esta peculiaridad a veces confunde al usuario de la aplicación, y con el sistema nuevo queda resuelta como explicaremos más adelante.

Para el caso de los objetos gráficos, las coordenadas cambian de nombre para el usuario, aunque en la base de datos se almacenen de igual forma:

Figura 10: detalle de coordenadas de campos para campos gráficos



Fila de inicio, columna de inicio: se corresponden con la fila y columna de los campos de texto, y definen las coordenadas del punto inicial donde se empieza a dibujar una recta o cuadro.

Fila final, columna final: se corresponden con el alto y ancho de los campos de texto, y definen las coordenadas del punto final donde se empieza a dibujar una recta o cuadro.

Como se puede observar, el hecho de que un objeto gráfico sea una recta

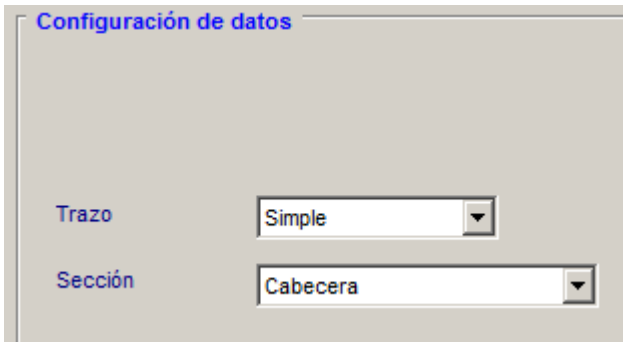
o un cuadro depende simplemente de si las coordenadas x e y coinciden en valor, por lo que se tratará de una recta vertical u horizontal respectivamente, y si los cuatro valores son distintos, estaremos definiendo un cuadrado. No obstante, para dar mayor claridad al usuario, al crear estos campos se pregunta si es una recta o un cuadro.

Si estamos definiendo una recta o cuadro, las propiedades propias de los datos de tipo texto, como son fórmula, formato, alineación etc, no se establecen, pues obviamente no son aplicables a este tipo de objetos gráficos. Para estos tipos de objetos sólo se pueden establecer dos propiedades:

Trazo: para definir el grueso de la recta o cuadro. Puede ser simple o doble.

Sección: Podrá ser cabecera o pie, o directamente no se podrá establecer si el objeto pertenece a la sección de detalle, pues pertenecerá a esta sección por defecto, obviamente.

Figura 11: detalle de propiedades de campos tipo rectas o cuadros

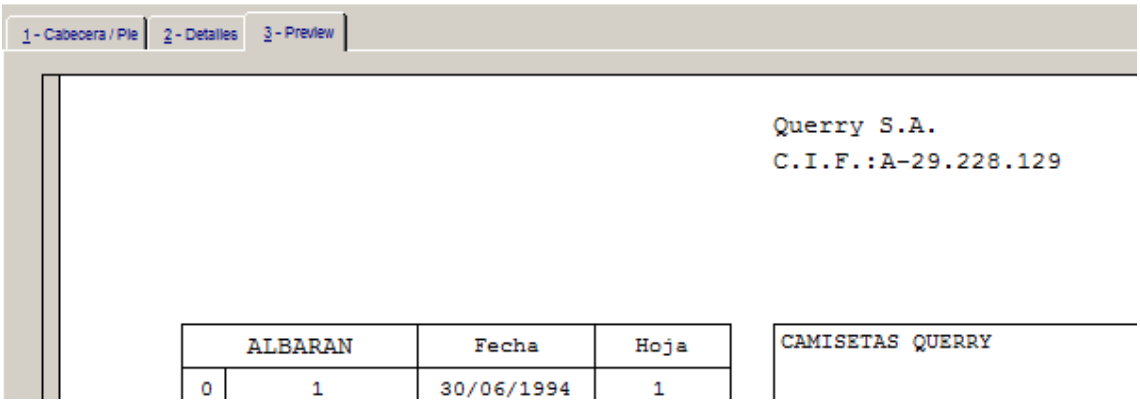


6.- Funcionamiento del configurador actual

Como hemos explicado hasta ahora, un formato se define estableciendo las coordenadas donde se puede imprimir cada dato. Los datos que provienen de la base de datos ya viene creados por la aplicación, y sólo debemos establecer donde queremos imprimirlo. Luego tenemos los objetos gráficos que los creamos para conformar la estética del documento.

A la hora de trabajar con la herramienta, una vez que vamos definiendo los campos, se hace imperativo el poder saber cómo va siendo el resultado final del formato. Es fundamental tener la referencia de cómo va el trabajo. Lo habitual podría ser imprimir en papel el trabajo que se va realizando. Pero debido al coste que esto supone, y al desperdicio de recursos que supondría, se usa un apartado adicional en la aplicación, llamado “vista previa”, en el que podemos ver en pantalla el resultado final de nuestra configuración. Cada vez que queramos ver cómo va quedando el formato, sólo tenemos que pulsar el botón ‘Preview’ y veremos el resultado en pantalla.

Figura 12: ejemplo de pestaña preview



Cada vez que pulsamos este botón, el programa recorre la base de datos para leer todos los objetos imprimibles del formato con el que estamos trabajando, y los va dibujando en un objeto picture de Visual Basic. Este objeto se ha definido con la peculiaridad de que las dimensiones del mismo se corresponden en tamaño real con las dimensiones del papel que hayamos establecido para el formato. Tal es así que si ponemos un folio en blanco encima del monitor, debe cuadrar exactamente con lo que vemos en pantalla.

La concordancia entre lo que se define en coordenadas y el resultado final no es exacta debido a la forma de trabajo con las dimensiones que se usan en estados unidos. Visual Basic usa pulgadas para sus dimensiones. De hecho a nivel informático se usan los twips, que viene del inglés “twentieth of a point”, y se define como 1/1440 de una pulgada. En Europa al trabajar en milímetros, nos

obliga a establecer una adaptación de una medida a otra. La relación que hay entre una pulgada y un milímetro viene a ser de 0,0393700787401575 pulgadas 1 milímetro. Al no ser algo exacto, genera ciertas diferencias por los redondeos. Las diferencias finalmente no suponen un gran problema al usuario final, pues no son muy apreciables visualmente.

La herramienta picture de Visual Basic ofrece la posibilidad de obtener las dimensiones equivalentes usando las funciones Picture.ScaleX y Pisture.ScaleY. Estas funciones nos permiten convertir un valor de una dimensión a otra, usando los parámetros vbTwips y vbMillimeters como sigue:

Dimension_En_mm = Picture.ScaleX(Dimension_En_Twips, vbTwips, vbMillimeters)

donde

Dimension_En_mm es el valor en milímetros que queremos obtener

Dimension_en_twips es el valor en twips que queremos convertir a milímetros

vbTwips y vbMillimeters son constantes para establecer los tipos de medidas a convertir

Con esta herramienta y estas funciones tenemos resuelto el problema de la conversión de una dimensión a otra, aunque con la consabida pérdida de precisión por el redondeo, aunque como ya hemos dicho no es muy apreciable.

Esta forma de trabajar, aunque nos facilita la posibilidad de visualizar el resultado final en la pantalla, ofrece algunos inconvenientes. El principal de ellos es la velocidad, que al tener que recorrer los objetos a imprimir cada vez que queremos pre-visualizar el resultado, hace que esta tarea sea lenta. Si se desea por ejemplo ajustar un valor y necesitamos ir viendo varias visualizaciones consecutivas para ir apreciando el ajuste final de cualquier campo que queramos afinar en la impresión, por cada vez que hacemos una pre-visualización se pierde bastante tiempo. Aunque trabajemos con datos en memoria, sigue siendo un tiempo demasiado alto, pues la necesidad de hacer muchas pre-visualizaciones es muy habitual al configurar el trabajo de esta manera. Esto es

una de las principales mejoras que aporta el nuevo sistema, y que detallaremos posteriormente: al ver directamente el resultado en pantalla, sólo recorremos los valores a imprimir una vez, en la primera carga de información, pues durante todo el proceso de diseño estaremos trabajando directamente en el resultado final, viendo todo cambio que hacemos in situ en pantalla.

Además, en la herramienta de trabajo, debido a la necesidad de configurar todas estas propiedades de los objetos, nos vemos obligados a separar lo que son las pantallas de trabajo a la hora de configurar los campos de la pantalla de visualización. Esto lo hacemos mediante otro objeto de Visual Basic que es el control Tab, que nos permite definir varias pestañas donde ubicar u agrupas diferentes grupos de objetos. Debido a la limitación de espacio que tenemos al trabajar con un monitor estándar, debemos usar tres pestañas diferentes para todo el proceso de configuración:

- una pestaña alberga los objetos que se definen en la cabecera o el pie del documento
- una segunda pestaña, similar a la primera, contiene todos los objetos que integran la sección de detalles
- la tercera y última pestaña contiene el objeto Picture que se encarga de mostrar el resultado final en pantalla del formato

Esta necesidad de tener estas tres pestañas de trabajo también generan otro inconveniente, provocado por la incomodidad de tener que estar trabajando con estos tres apartados. Y sobre todo el problema más importante consistente en que no podemos ver el resultado de cada cambio que hacemos in situ, pues una vez cambiadas las propiedades de un objeto, debemos volver a la pestaña de visualización para ver el resultado final.

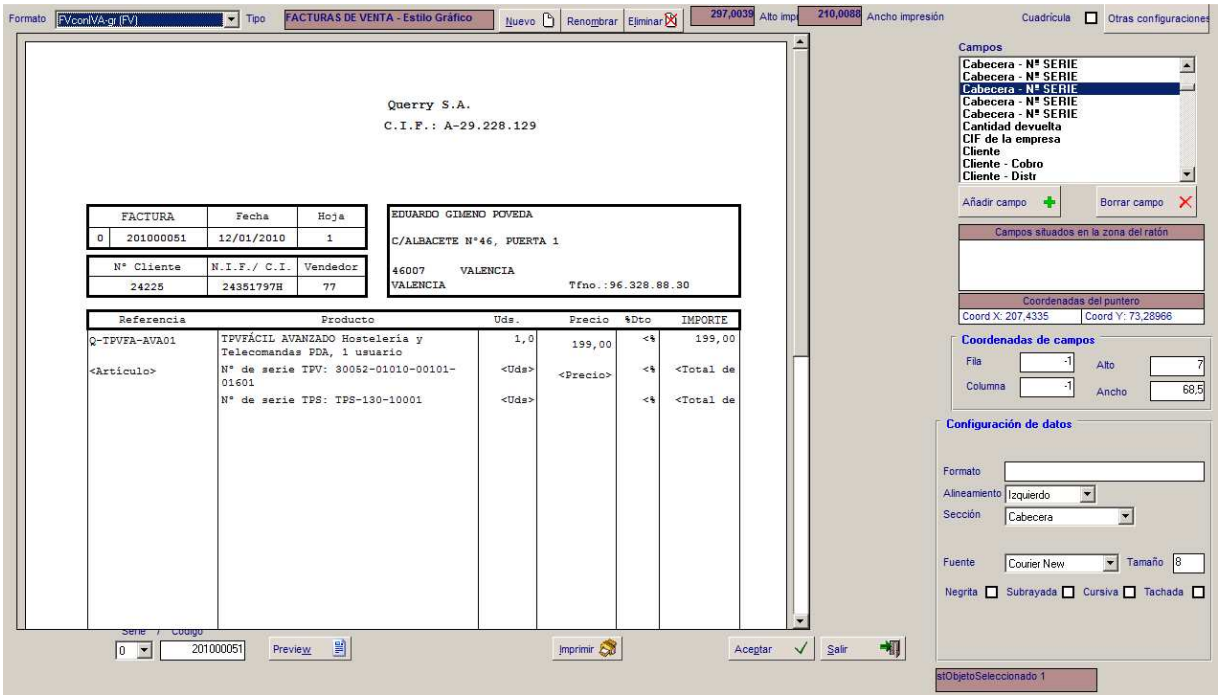
7.- Funcionamiento del nuevo configurador gráfico de formatos

Viendo la problemática del diseño de formatos con la herramienta actual, la idea de poder trabajar directamente sobre el resultado en pantalla del formato se hace muy interesante y necesaria para incrementar la productividad a la hora de realizar estos trabajos.

No obstante, desarrollar una aplicación de diseño gráfico en Visual Basic 6 no es aparentemente la mejor opción posible. Desarrollar una herramienta de estas características quizás es algo más propio para compiladores más modernos, como Visual Basic.Net o Java. No obstante, siempre existen unas limitaciones por parte de la empresa, sobre todo a nivel económico. No es factible en las circunstancias actuales una inversión económica para adquirir un compilador más moderno, ni para la formación correspondiente del personal para usar esta herramienta. La opción más factible siempre pasa por usar el compilador actual. Además, se pretende crear una herramienta que siga siendo compatible con el resto de la aplicación actual de facturación, y al estar esta realizada con el compilar actual, Visual Basic 6, es casi obligado tener que desarrollar esta herramienta en Visual Basic 6.

Sin embargo, aunque podría parecer que este compilador se pudiera quedar atrás para realizar esta tarea, no ha sido el caso. Visual Basic ofrece el control Picture, el mismo que usamos ya para visualizar los resultados gráficos, con una serie de ventajas que nos van a permitir abordar la tarea de poder diseñar directamente el formato gráfico, usando el ratón en la mayor parte del trabajo.

Figura 13: aspecto del nuevo configurador gráfico de formatos



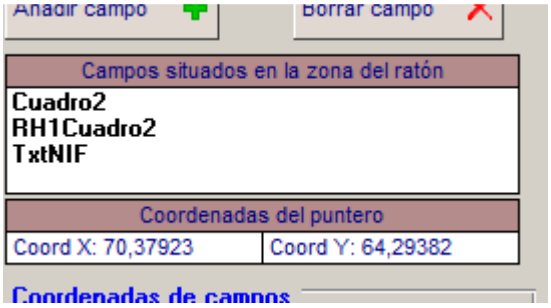
Principales cambios en la nueva herramienta respecto a la anterior.

El nuevo configurador gráfico de formatos, como principal ventaja tiene la inherente a una aplicación de estas características, que es el hecho de trabajar directamente sobre el resultado final. No es necesario pulsar el botón 'Preview' cada vez que queremos ver cómo queda cualquier modificación que se haga, ya que se hace la modificación directamente sobre el resultado.

Además, como se puede observar en la imagen, nos aporta la comodidad de tener toda la información sobre la misma pantalla sin tener que recurrir a las diferentes pestañas que agrupan la información. Estas pestañas desaparecen y se agrupa toda la información en una única lista de campos. Si bien esta forma de tener la información es más incómoda, queda suplida dicha incomodidad del la forma de trabajar en la que ya no es necesario recurrir a esta lista de campo salvo casos puntuales.

Otra ventaja que se ha aportado a la aplicación es la de poder ver in situ qué campo es el que se encuentra en la posición del ratón. Con el sistema anterior, al visualizar la vista previa del formato, si querías localizar un determinado valor para modificarlo, tenias que saber qué campo era exactamente, algo que era muy incómodo y provocaba malgastar mucho tiempo, pues es habitual que el campo que estamos viendo impreso no tenga un nombre claro asociado, o que simplemente no sepamos exactamente a qué campo se refiere ni cómo localizarlo. Con el nuevo sistema, al pasar el ratón por encima del dibujo directamente nos muestra qué campo o campos hay en esa posición del ratón. Esta información se muestra en el apartado que se denomina "Campos sitiados en la zona del ratón", que es una lista con todos los posibles campos que haya en la zona del ratón. Como se aprecia en la siguiente imagen, en este caso sólo hay dos objetos en la posición del ratón. Además de esta información, también se muestran las coordenadas actuales del ratón, algo que puede sernos útil para saber cómo ajustar los valores en caso de querer hacerlo manualmente, para por ejemplo lograr un ajuste más fino del que nos pueda proporcionar la herramienta gráfica.

Figura 14: información del campo que está en la zona del ratón



Otra ventaja aportada por el nuevo configurador surge de la necesidad del diseño intrínseco. Al tener que trabajar dibujando 'in situ' en pantalla, se hace imprescindible incrementar la velocidad de la aplicación, lo cual nos obliga a hacer que la información que estamos procesando al dibujar esté en memoria y no recurrir a ella desde la base de datos. Para ello se hace una primera carga de la información de la base de datos a la memoria, y ya durante todo el proceso de diseño no se accede al almacenamiento físico salvo para lo más imprescindible y para guardar las modificaciones, incrementando la velocidad del diseñador gráfico notablemente.

Todas estas ventajas en su conjunto generan una de las principales ventajas de esta forma de trabajar, y es que hacen que el uso de diseñador muy intuitivo y sencillo, siendo factible el uso del mismo por cualquier usuario no avanzado de la aplicación.

8.- Metodología

La metodología usada para el diseño de la aplicación ha sido la de la programación funcional y la programación de eventos. Al tener que usar la herramienta Visual Basic 6, y ser esta una herramienta de programación orientada a eventos, nos obliga a usar un sistema de programación algo anticuado ya, que es la clásica programación funcional combinada con la inherente a las aplicaciones visuales que es la programación orienta da a eventos.

Descripción del sistema usado para programar la herramienta de dibujo.

Para el desarrollo del código se ha desarrollado un algoritmo en el que nos hemos basado en un sistema de estados, y en las funciones que proporciona Visual Basic 6.0 con el objeto picture para dibujar. A continuación se explica el algoritmo así como las partes más significativas del código desarrollado para conseguir realizar la tarea de dibujar.

Para el algoritmo de dibujo hemos establecido un sistema de estados. En base a estos estados, establecemos las transiciones entre estados en función de los eventos que se puedan dar en el proceso de dibujo. La idea es poder establecer en todo momento que estamos haciendo respecto al dibujo que estamos tratando, y a que nuevos estados podemos pasar. Los estados establecidos son los siguientes:

- *st_Nada*
- Es el estado inicial. En este estado estamos cuando empezamos con un formato o cargamos uno empezado o cuanod tras lanzar el evento MouseUp no hemos seleccionado nada.
- *stSeleccionandoObjeto*
- En este estado nos encontramos cuando estamos seleccionando un objeto. Se da cuando se lanza el evento MouseDown del objeto Picture y antes de que se dé el evento MouseUp.
- *stObjetoSeleccionado*
- Es el estado que nos indica que hay un objeto seleccionado.
- *stObjetoMover*
- Este estado se establece para controlar el hecho de estar moviendo o desplazando un objeto por el objeto picture.
- *stObjetoRedimensionar*
- Este estado, de forma similar al anterior, controla el que estemos redimensionando un objeto seleccionado, es decir, no lo estamos moviendo sino alternado sus dimensiones. Este estado podría combinarse con el anterior y dejar uno sólo, pero por motivos de

claridad de código, se ha preferido dejarlo como un estado aparte.

- *stNoHayObjeto*
- Este estado se establece cuando tras pulsar el ratón y lanzarse el evento MouseDown, lo hacemos en una zona en la que no hay ningún objeto dibujado. Es necesario para que cuando se lance el evento MouseUp se sepa que no se ha seleccionado nada debido a que no hay nada que seleccionar.

Estos estados se combinan principalmente con los eventos que se lanzan al usar el ratón sobre el objeto Picture. Evidentemente estos eventos son consecuencia del manejo por parte del usuario del ratón sobre el lienzo a la hora de dibujar un objeto o redimensionar el mismo. A continuación se describen los eventos principales y las consecuencias de estos sobre los diferentes estados. En cada evento hay que tener en cuenta qué botón se ha pulsado pues en función de cada botón la aplicación hará tareas distintas.

Para la explicación de esta parte de código usare variables, procedimientos y funciones que detallaré a continuación.

Función Cantidad_Objetos_en_zona_pulsada (X, Y, iObjetoSeleccionado)

Nos indica la cantidad de objetos que hay en la zona pulsada por el ratón. En caso de haber un solo objeto, asignará a la variable iObjetosSeleccionado el valor de ese único objeto y llamamos al procedimiento que lo seleccionará. En caso de haber más de uno, igualmente elegirá el primero que encuentre y se seleccionará igualmente.

Variable iObjetosSeleccionado

Nos dice el valor del objeto seleccionado. Este valor es único para cada objeto, y su valor está comprendida entre -1 y algún valor mayor o igual a 0. En caso de tener valor -1 equivale a que no hay ningún objeto seleccionado. En el texto usaremos la palabra <Ninguno> para este caso por claridad.

Procedimiento SeleccionarObjeto(iObjetoASeleccionar)

Es el procedimiento que se encarga de seleccionar el objeto que le pasamos como parámetro.

Función CoordenadasEnEsquina(X, Y, iObjetoSeleccionado, EsquinaEstoy)

Nos devuelve Verdadero si he hecho click en alguna esquina del objeto que está seleccionado. Esta función nos ayuda a determinar si lo que queremos hacer es redimensionar el objeto, lo cual se hace pulsando en alguna de las esquinas del objeto, o moverlo, para lo que usamos la siguiente función.

Función CoordenadasEnImagen(X, Y, iObjetoSeleccionado)

Nos devuelve Verdadero si hemos pulsado dentro del objeto que está actualmente seleccionado, con lo que sabremos que queremos mover el objeto en lugar de redimensionarlo.

Variable EsquinaEstoy

Nos dice el valor de la esquina en la que hemos pulsado del objeto seleccionado, en caso de hacer click en alguna esquina.

Función CoordenadasNoEnImagen(X, Y, iObjetoSeleccionado)

Nos indica que hemos hecho click en un objeto que no es el seleccionado actualmente. En este caso queremos seleccionar un objeto distinto al que tenemos seleccionado.

Función bFiguraEnSeccionValida(iObjetoSeleccionado, X, Y, iPosInicialMoverX, iPosInicialMoverY)

Esta función se usa en el evento MouseMove, y se utiliza para, en base a las coordenadas iniciales, y las nuevas donde se sitúa el ratón mientras movemos un objeto, decirnos si la posición destino es válida. Si intentamos desplazar por ejemplo un objeto de la cabecera a una posición que está situada en la sección

de detalles, esta función devolverá ‘Falso’ y el algoritmo impedirá mover a esta nueva posición el objeto.

Procedimiento RedibujoFigura(iObjetoSeleccionado, iEsquinaPulsada, X, Y, iPosInicialMoverX, iPosInicialMoverY)

Este procedimiento re-dibuja el objeto seleccionado teniendo en cuenta que lo estamos redimensionando al mover alguna de sus esquinas a una nueva posición.

Evento MouseDown

Este evento es el que se lanza cuando pulsamos cualquier botón del ratón sobre el objeto Picture.

Si hemos pulsado el botón izquierdo del ratón, primero comprobamos en qué estado estamos para ver la acción a realizar y a qué estado podemos transitar.

Situación 1:

<Botón pulsado> = Izquierdo y <Estado actual> = st_Nada

Si Cantidad_Objetos_en_zona_pulsada = 0

<Nuevo estado> = stNoHayObjeto

Si Cantidad_Objetos_en_zona_pulsada >=1

<Nuevo estado> = stSeleccionandoObjeto

SeleccionarObjeto(iObjetoSeleccionado)

Situación 2:

<Botón pulsado> = Izquierdo y <Estado actual> = st_ObjetoSeleccionado

Si CoordenadasEnEsquina(X, Y, iObjetoSeleccionado, EsquinaEstoy)

<Nuevo estado> = stObjetoRedimensionar

Si CoordenadasEnImagen(X, Y, iObjetoSeleccionado)

<Nuevo estado> = stObjetoMover

Si CoordenadasNoEnImagen(X, Y, iObjetoSeleccionado)

Si Cantidad_Objetos_en_zona_pulsada >=1

<Nuevo estado> = stSeleccionandoObjeto

SeleccionarObjeto(iObjetoSeleccionado)

<Nuevo estado> = stObjetoMover

Cualquier otro caso no hacemos nada

En este evento, básicamente lo que hacemos es comprobar donde estamos, qué hay donde hemos pulsado y hacia dónde vamos. Destacamos el hecho de que ya se selecciona algún objeto en este evento. El hecho de seleccionar un objeto provoca que dicho objeto se redibuje seleccionado, con otro color y con cuatro esquinas que lo enmarcan para poder redimensionarlo. Hay que recordar que el evento MouseDown, sólo puede volver a lanzarse tras un evento MouseUp (soltar el botón del ratón), lo cual es clave para el buen funcionamiento de este algoritmo.

Evento MouseMove

Este evento es el que se lanza cuando movemos el ratón sobre el objeto Picture. Este movimiento puede darse con algún botón del ratón pulsado, algo importante a tener en cuenta.

Si tenemos pulsado el botón izquierdo del ratón mientras movemos el ratón, suele querer decir que tenemos un objeto seleccionado y lo estamos moviendo o redimensionando, en función de si pulsamos sobre él o sobre alguna de las esquinas del mismo.

- <Botón pulsado> = Izquierdo

- Si <Estado actual> = st_Nada o <Estado actual> = st_SeleccionandoObjeto

<No se hace nada>

- Si <Estado actual> = stObjetoMover

Si bFiguraEnSeccionValida(iObjetoSeleccionado, Coordenada_X_Raton, Coordenada_Y_Raton, iPosInicialXAntesMoverRaton,

iPosInicialYAntesMoverRaton)

MuevoFiguraA(iObjetoSeleccionado, Coordenada_X_Raton, Coordenada_Y_Raton, iPosInicialXAntesMoverRaton, iPosInicialYAntesMoverRaton)

<Nuevo estado> = stNoHayObjeto

- Si <Estado actual> = stObjetoRedimensionar

RedibujoFigura(iObjetoSeleccionado, iEsquinaPulsada, X, Y, iPosInicialMoverX, iPosInicialMoverY)

En este evento, destacamos que si el estado actual es Nada, cosa que ocurre si hemos pulsado el ratón donde no hay nada, o el estado actual es st_SeleccionandoObjeto. Este caso es raro que se dé, pero por seguridad lo mantenemos, pues cuando hacemos MouseDown en una zona donde haya algún objeto, ya marcaremos el objeto como seleccionado y el estado será que tenemos un objeto seleccionado. Por lo demás, la idea es clara: si tenemos un objeto seleccionado, y el botón izquierdo pulsado, si movemos el ratón sólo podemos estar haciendo dos cosas: desplazando el objeto, o alterando sus dimensiones. En el primer caso redibujamos el objeto en la nueva posición, siempre y cuando ésta nueva posición sea válida como hemos explicado previamente, y en el otro caso redibujamos el objeto con las nuevas dimensiones.

Evento MouseUp

Este evento se lanza al soltar el botón del ratón, y en cierto modo finaliza ciertos estados y procesos de transición, pues terminamos de mover un objeto, o de redimensionarlo. Es importante dejar el estado actual correctamente para que todo funcione bien y podamos lanzar nuevas transiciones a estados digamos estables.

Este evento es importante siempre y cuando sea el botón izquierdo del ratón el que lanza el evento pues es el botón usado para mover y desplazar objetos.

- <Botón pulsado> = Izquierdo

- Si <Estado actual> = st_Nada o <Estado actual> = st_NoHayObjeto

  <No se hace nada>

  <Estado actual> = st_Nada

- Si <Estado actual> = *stSeleccionandoObjeto*

  <No se hace nada>

  <Estado actual> = stObjetoSeleccionado

- Si <Estado actual> = *stObjetoMover o* <Estado actual> = *stObjetoRedimensionar*

  <No se hace nada>

  <Estado actual> = stObjetoSeleccionado

Este evento sirve para en cierto modo confirmar el estado que nos dice que un objeto se ha seleccionado tras soltar el botón del ratón, o que ha finalizado de redimensionarse o de moverse, pasando al estado de 'objeto seleccionado', o a 'Nada' si habíamos lanzado el evento MouseDown (pulsado el botón del ratón) en una zona donde no había ningún objeto.

Se puede observar que al hacer click en una zona donde hay más de un objeto, el algoritmo escoge el primer objeto que encuentra. Esto limitaría notablemente el uso de la aplicación pues lo más habitual es que coincidan muchos objetos en una misma zona, y que queramos concretar con qué objeto exactamente queremos operar para poder seleccionarlo. Para ello, se ha usado el evento MouseDown en conjunción con el botón derecho del ratón:

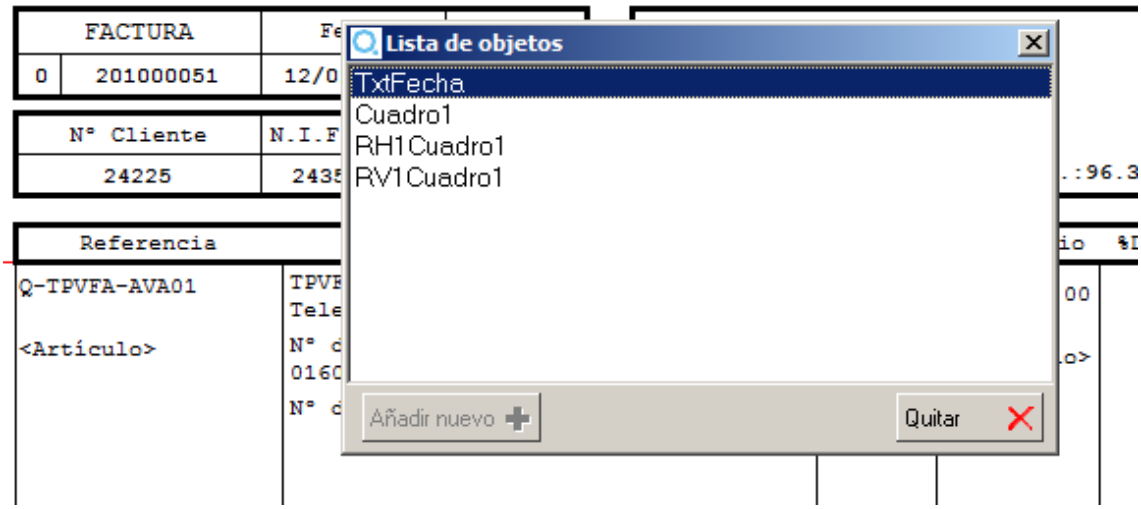
- <Botón pulsado> = Derecho

- Si <Estado actual> = st_Nada o <Estado actual> = stObjetoSeleccionado o <Estado actual> = stObjetoMover

  iNumeroObjetos = ptEnImagenCuantas(X, Y, iObjetoSeleccionado, numFilasCab, numFilasDetalles)

- Si la función ptEnImagenCuantas, que nos dice cuantos objetos hay en la zona pulsada por el ratón, da un valor mayor que cero, se nos abre un menú con la lista de campos que hay para poder seleccionar el que deseemos. Una vez seleccionado uno volvemos al formato con el estado en 'ObjetoSeleccionado'. A continuación se muestra una imagen de ejemplo de la lista de campos de la ue podemos seleccionar el deseado.

Figura 15: Lista de objetos en la zona para seleccionar



- *Función MuevoFiguraA*

Esta función es quizás la más importante debido a la problemática inherente al proceso de dibujar.

El proceso para dibujar y mover un determinado objeto sobre un control Picturebox se basa en la sencilla idea de ir redibujando todo el formato, objeto por objeto, de todos aquellos objetos que no estamos moviendo o redibujando, y después dibujar de forma especial el que estamos moviendo a la nueva posición. Con esto se logra el efecto visual de 'mover' por el control Picturebox un determinado objeto. Esta tarea que es muy sencilla a primera vista, se complica a la hora de llevarla a cabo al desarrollarla con un compilador como Visual Basic 6 que no está orientado a trabajar con elementos gráficos.

En una primera aproximación a la tarea de dibujar un objeto moviéndose

se programó tal y como aquí se describe, es decir cada vez que se lanzaba el evento MouseMove, se redibujaban todos los objetos del formato excepto el objeto seleccionado, y se dejaba para el final este último marcado de forma especial. Esto, si bien funcionaba, lo hacía de una forma totalmente inoperativa debido a la cantidad de procesos que había que realizar en un simple movimiento de ratón. Esto unido al hecho de que el ratón cuando estamos trabajando se mueve casi constantemente, no permitía realizar toda la tarea de dibujado del formato antes del siguiente movimiento de ratón, lo cual provocaba que el programa se saturase quedándose colgado.

Para solventar este problema, se ha decidido usar dos controles Picturebox. El control Picturebox tiene una función que permite copiar el contenido completo de un control Picturebox a otro, y esto se hace de forma muy rápida. Otra utilidad que nos brinda el objeto es la posibilidad de realizar un borrado completo del contenido mediante la instrucción Cls, y que también es casi inmediata. Aprovechando estas dos ventajas, lo que se hace para realizar la tarea de dibujar es que, cuando seleccionamos un objeto haciendo click en él, redibujamos todos los objetos en el Picturebox que está oculto, excepto el que vamos a seleccionar. Antes de hacer este redibujado establecemos la propiedad del Picturebox ‘Autoredraw’ a True. Esta propiedad a True hace que todo lo que se dibuje a continuación, tras ejecutar el borrado (instrucción Cls) no se borrará nada de lo dibujado estando esta propiedad a True. Una vez dibujado todo se copia el contenido al objeto Picture que usamos para visualizar el trabajo.

Tras dibujar todos los objetos que no vamos a mover en este Picturebox, establecemos la propiedad ‘Autoredraw’ a False y después dibujamos el objeto que vamos a mover. Y todo esto en el momento de seleccionar el objeto. A continuación cuando movemos el ratón con el botón izquierdo pulsado, en el evento MouseMove, en lugar de redibujar todos los objetos como se decidió en la primera aproximación al problema, ejecutamos un simple CLS que hace que se borre únicamente el objeto que estamos moviendo debido a la propiedad ‘Autoredraw’, y a continuación dibujamos sólo el objeto que estamos moviendo a la nueva posición.

Con esta forma de implementar el código hacemos que en lugar de tener que redibujar los más de 150 o 200 objetos que puede llegar a tener un formato en cada movimiento de ratón, sólo dibujamos un sólo objeto. La carga de tener que redibujar el formato entero recae de esta forma en la función de selección de nuevo objeto. En este caso, cuando dibujamos los objetos que no se van a mover, al finalizar los copiamos al objeto Picturebox oculto, mediante la posibilidad de hacer este copiado de forma rápida. Este copiado al Picturebox oculto lo hacemos porque si trabajamos directamente sobre el control Picturebox visible, se produce un parpadeo cada vez que movemos el ratón, debido a que la función CLS lo provoca. Para evitarlo, hacemos el trabajo en el Picturebox oculto y copiamos todo el contenido de este al Picturebox visible, con lo que no se produce el efecto 'feo' de parpadeo de todo el formato.

9.- Conclusiones

- La aplicación aquí desarrollada genera una seria de ventajas sobre el software actual de Ambar7. Las principales son las siguientes

  - Incremento de la productividad para el cliente, al poder realizar este trabajo mucho más rápido que antes
  - Mejora de la calidad del software en relación a este apartado de configuración
  - Posibilidad de que los clientes en su mayoría puedan usar esta herramienta, sin necesidad de recurrir a ayuda externa por parte de la empresa suministradora del software
  - Incremento en la productividad del los trabajadores de la empresa suministradora del software al verse reducido el tiempo necesario para ayudar a los clientes finales

10.- Recursos

- Los recursos que sean utilizado para la implementación de esta aplicación se detallan a continuación.

- Compilador Microsoft Visual Basic 6 Service Pack 6
- Motor de base de datos SQL Server 2012 Express Edition
- Motor de base de datos Mysql

11.- Bibliografía

[1] Francesco Balena. Programación avanzada con Microsoft Visual Basic 6.0, Ed. McGraw-Hill

[2] Andrew J. Brust. Stephen Forte. Programación avanzada con Microsoft SQL Server 2005. Ed. McGraw-Hill

[3] William R. Stanek. Microsoft SQL Server 2005. Manual del administrador. Ed. McGraw-Hill

[3] Manual de referencia de MySQL 5.6. http://dev.mysql.com